Project	**IEEE 802.20 Working Group on Mobile Broadband Wireless Access** <http://ieee802.org/20/>
Title	**MBFDD and MBTDD: Proposed Draft Air Interface Specification**
Date Submitted	**2006-01-06**
Source(s)	Jim Tomcik Qualcomm, Incorporated 5775 Morehouse Drive San Diego, CA, 92121 Voice: 858-658-3231 Fax: 858-658-3231 Email: jtomcik@qualcomm.com Radhakrishna Canchi Kyocera Telecom Research Corp. 2480 N. First Street, Suite #280 San Jose, CA 95131 Voice: 408-952-4701 Fax: 408-954-8709 Email: cradhak@ktrc-na.com
Re:	MBWA Call for Proposals
Abstract	This contribution (part of theMBFDD and MBTDD proposal packages for 802.20), contains the proposed Draft Air Interface Specification. The Mobile Broadband Frequency Division Duplex (MBFDD) proposal is a revision of the previously submitted QFDD proposal; the Mobile Broadband Time Division Duplex (MBTDD) proposal is a revision of the previously submitted QTDD proposal, and also a merged proposal including the BEST-WINE proposal previously submitted to 802.20.
Purpose	For consideration of 802.20 in its efforts to adopt FDD and TDD proposals for MBWA.
Notice	
Release	
Patent Policy	

Abstract:
These technical requirements form a compatibility standard for mobile broadband wireless access systems. The requirements ensure that a compliant access terminal can obtain service through any access network conforming to this standard, thus providing a framework for the rapid development of cost-effective, interoperable multivendor mobile broadband wireless access systems. This compatibility standard is targeted for use in a wide variety of licensed frequency bands.

This specification includes provisions for future service additions and expansion of system capabilities. The architecture defined by this specification permits such expansion without the loss of backward compatibility to older access terminals.

Contents

Figures

Tables

Foreword

This foreword is not part of this Standard.

References

[1] FIPS PUB 180-2, Federal Information Processing Standards Publication 180-2[1]

[2] IETF RFC 2409, The Internet Key Exchange (IKE)[2]

[3] Internet Assigned Numbers Authority at http://www.iana.org/

[4] IETF RFC 791, Internet Protocol

[5] IETF RFC 2460, Internet Protocol, Version 6 (IPv6) Specification

[6] ITU-T Recommendation E.212, Identification Plan for Land Mobile Stations, 1988[3]

[7] IETF RFC 3056, Connection of IPv6 Domains via IPv4 Clouds, February 2001

[8] TR45.AHAG, Enhanced Cryptographic Algorithms, Revision B, March 5, 2002[4]

[9] IEEE Std 802.11i/D7.0™, "Part 11: Wireless Medium Access Control (MAC) and Physical layer (PHY) specifications: Specifications for Enhanced Security", October 2003[5,6]

[10] IETF RFC 2104, HMAC: Keyed Hashing for Message Authentication, February 1997

[11] IETF RFC 3095, Robust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed.

[12] IETF RFC 2210, The Use of RSVP with IETF Integrated Services

[13] IETF RFC 2212, Specification of Guaranteed Quality of Service

[14] IETF RFC 2215, General Characterization Parameters for Integrated Service Network Elements

[15] Internet Assigned Numbers Authority RObust Header Compression (ROHC) Profile Identifiers at http://www.iana.org/assignments/rohc-pro-ids

---

[1] FIPS publications are available from the National Technical Information Service (NTIS), U. S. Dept. of Commerce, 5285 Port Royal Road, Springfield, VA 22161 (http://www.ntis.gov/).

[2] IETF publications are available from the Internet Engineering Task Force at http://www.ietf.org/.

[3] ITU-T publications are available from the International Telecommunications Union, Place des Nations, CH-1211, Geneva 20, Switzerland/Suisse (http://www.itu.int/).

[4] TR-45.AHAG publications available from http://ftp.tiaonline.org/TR-45/TR45AHAG/Public/

[5] IEEE and 802 are registered trademarks in the U.S. Patent & Trademark Office, owned by the Institute of Electrical and Electronics Engineers, Incorporated.

[6] IEEE publications are available from the Institute of Electrical and Electronics Engineers, Inc., 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA (http://standards.ieee.org/).

[16] 3GPP2, S.S0055Av2.0, Enhanced Cryptographic Algorithms, Jan 2005. Available at http://www.3gpp2.org/Public_html/specs/S.S0055-A_v2.0_050120.pdf, also published as TIA 946-1

[17] IEEE Std 802.3™-2002, IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications.

[18] IETF RFC 2578, Structure of Management Information Version 2 (SMIv2), April 1999

[19] IETF RFC 2579, Textual Conventions for SMIv2, April 1999

[20] IETF RFC 2580, Conformance Statements for SMIv2, April 1999

[21] IETF RFC 3513, IP Version 6 Addressing Architecture, April 2003

[22] IETF RFC 3748, Extensible Authentication Protocol, June 2004

[23] IETF RFC 4193 Unique Local IPv6 Unicast Addresses, October 2005

[24] 3GPP2, C.R1001-E v1.0, Administration of Parameter Value Assignments for cdma2000™ Spread Spectrum Standards. [7]

[25] ATIS-PP-0700004-2005, High Capacity-Spatial Division Multiple Access (HC-SDMA), September 2005

[7] cdma2000® publications are available from the Third Generation Partnership Project 2 (3GPP2) (http://www.3gpp2.org/).

1 Overview

1.1 Scope of this document

These technical requirements form a compatibility standard for mobile broadband wireless access systems. The requirements ensure that a compliant access terminal can obtain service through any access network conforming to this standard, thus providing a framework for the rapid development of cost-effective, interoperable multivendor mobile broadband wireless access systems. This compatibility standard is targeted for use in a wide variety of licensed frequency bands.

This specification includes provisions for future service additions and expansion of system capabilities. The architecture defined by this specification permits such expansion without the loss of backward compatibility to older access terminals.

1.2 Modes of the specification

This specification has two modes of operation, a Wideband mode and a 625k-MC mode. The Wideband mode is designed to operate for all FDD and TDD bandwidths and is described in Chapters 1 through 12. The 625k-MC mode is designed with 625 KHz carrier bandwidth supporting aggregation of multiple carriers for TDD operation and is described in Chapters 14 through 28.1 and Appendix A.

1.3 Requirements language

Compatibility, as used in connection with this standard, is understood to mean: Any access terminal can obtain service through any access network conforming to this standard. Conversely, all access networks conforming to this standard can service access terminals.

“Shall” and “shall not” identify requirements to be followed strictly to conform to the standard and from which no deviation is permitted. “Should” and “should not” indicate that one of several possibilities is recommended as particularly suitable, without mentioning or excluding others, that a certain course of action is preferred but not necessarily required, or that (in the negative form) a certain possibility or course of action is discouraged but not prohibited. “May” and “need not” indicate a course of action permissible within the limits of the standard. “Can” and “cannot” are used for statements of possibility and capability, whether material, physical, or causal.

1.4 Wideband Mode Overview

1.4.1 Architecture reference model

The architecture reference model is presented in Figure 1. The reference model includes the air interface between the access terminal and the access network. The protocols used over the air interface are defined in this document.

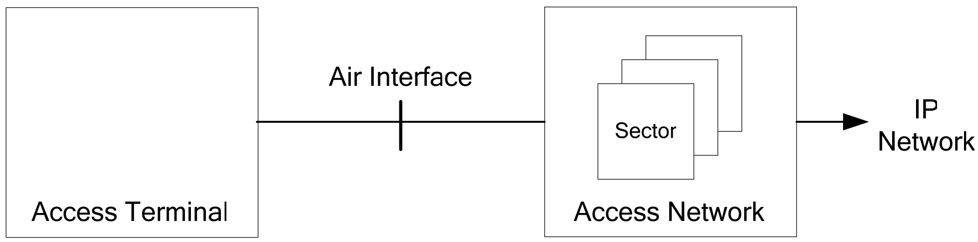

Figure 1 Architecture reference model

The functional units of the reference architecture in Figure 1 are:

Access Network (AN)	The network equipment providing Layer 3 connectivity between an IP network (typically the Internet) and the access terminals.
Access Terminal (AT)	A device providing data connectivity to a user. An access terminal may be connected to a computing device such as a laptop personal computer or it may be a self-contained data device such as a personal digital assistant.
Sector	One set of physical layer channels transmitted between the access network and the access terminals within a given frequency assignment. A sector consists of a reverse link ChannelBand and a forward link ChannelBand.

1.4.2 Protocol architecture

The air interface is layered, with interfaces defined for each layer (and for each protocol within each layer). This architecture allows future modifications to a layer or to a protocol to be isolated.

1.4.2.1 Layers

Figure 2 describes the layering architecture for the air interface. Each layer consists of one or more protocols that perform the layer's functionality.

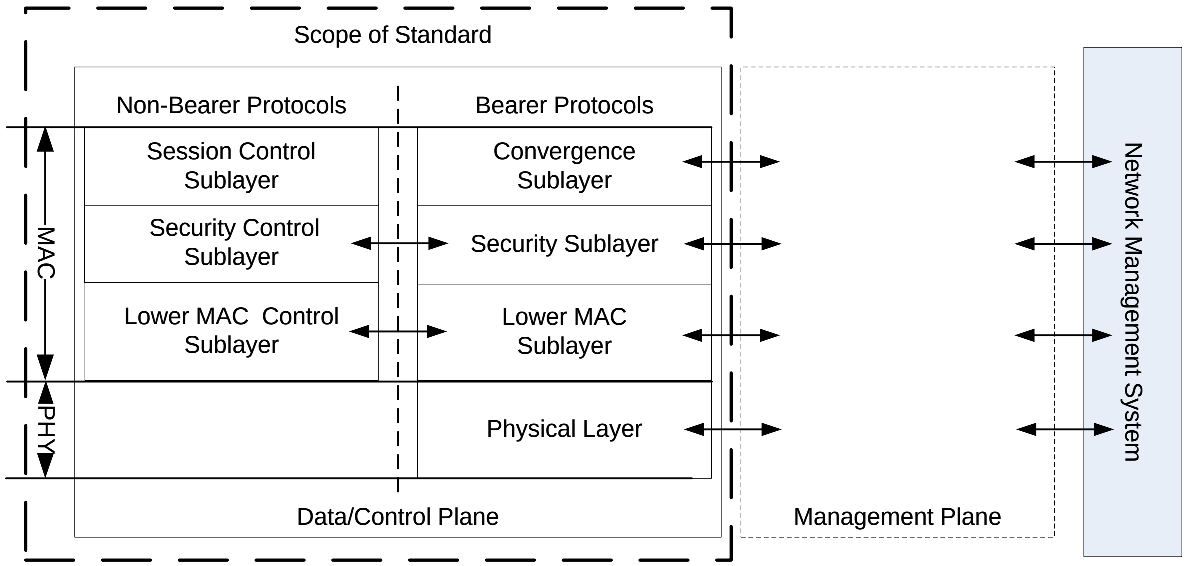

Figure 2 Air interface layering architecture

The protocols and layers specified in Figure 2 are:

Session Control Sublayer
: The Session Control Sublayer provides address management, protocol negotiation, protocol configuration, and state maintenance services. The Session Control Sublayer is a non-bearer layer and, therefore, it does not carry payload on behalf of other layers. The Session Control Sublayer is defined in Chapter 2.

Convergence Sublayer
: The Convergence Sublayer provides protocols and transports used to transport messages and data, and provides multiplexing of distinct transports. For example, it provides the Signaling Transport for transporting air interface protocol messages and the Data Transport for transporting user data. The Convergence Sublayer is defined in Chapter 3.

Security Control Sublayer
: The Security Control Sublayer provides key exchange for use by the Security Sublayer. The Security Control Sublayer is defined in Chapter 4.

Security Sublayer
: The Security Sublayer provides authentication and encryption services. The Security Sublayer is defined in Chapter 5.

Lower MAC Control Sublayer
: The Lower Medium Access Control (MAC) Control Sublayer provides air-link connection establishment and maintenance services. The Lower MAC Control Sublayer is defined in Chapter 6.

Lower MAC Sublayer	The Lower MAC Sublayer defines the procedures used to receive and to transmit over the Physical Layer. The Lower MAC Sublayer is defined in Chapter 7.
Physical Layer	The Physical Layer provides the channel structure, frequency, power output, modulation, and encoding specifications for the Forward and Reverse Channels. The Physical Layer is defined in Chapter 8.

Each layer may contain one or more protocols or transports. Protocols use signaling messages, in-band messages, blocks, or headers to convey information to their peer protocols at the other side of the air-link. When protocols send messages, they use the Signaling Network Protocol (SNP) to transmit these messages. Transports send signaling messages using the Signaling Network Protocol. Blocks are information conveyed to a peer protocol using an encapsulation that is specific to a Physical Layer Channel. For example, the Lower MAC Control Sublayer Overhead Messages Protocol uses the SystemInfo block to carry information to its peer protocol at the access terminal on the forward primary broadcast channel 0 (pBCH0).

1.4.3 Physical layer channels

The hierarchies between the Lower MAC Sublayer Protocols and the Physical Layer Channels for the forward and reverse links are shown in Figure 3 and Figure 4. The following is a brief description of each Physical Layer Channel. A more complete description is provided in Chapter 9. When the context is clear, the complete qualified name is usually omitted (e.g., Quick Paging Channel as opposed to Forward Quick Paging Channel or Data Channel as opposed to Reverse Data Channel).

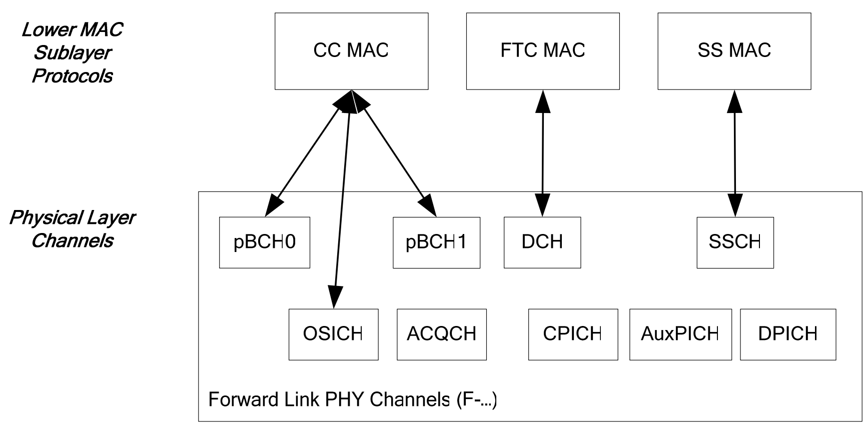

Figure 3 Forward channel structure

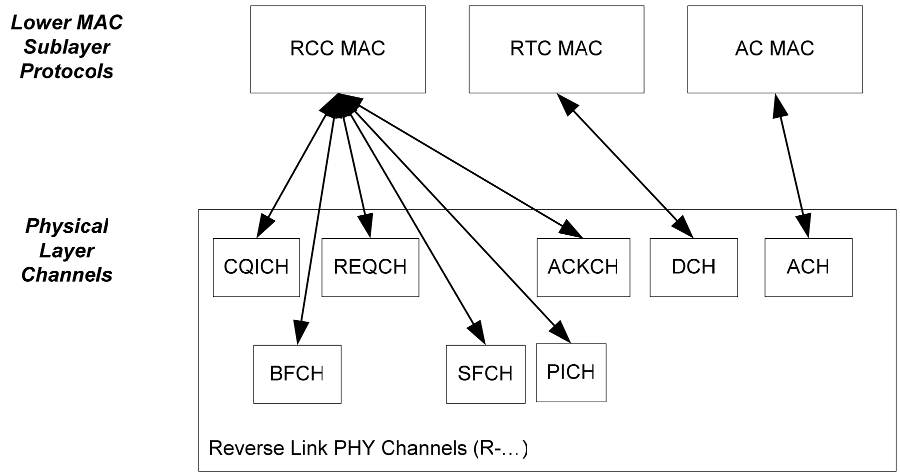

Figure 4 Reverse channel structure

1.4.3.1 Forward physical channels

Forward Acquisition Channel (F-ACQCH)
: Carries an acquisition pilot for an access terminal to use to acquire the system.

Forward Auxiliary Pilot Channel (F-AuxPICH)
: Carries auxiliary pilots for channel estimation from multiple transmit antennas. The Forward Primary Broadcast Channel 1 (F-pBCH1) indicates whether the F-AuxPICH is present.

Forward Common Pilot Channel (F-CPICH)
: Carries the common pilot.

Forward Data Channel (F-DCH)
: Carries information for a specific access terminal. A Forward Data Channel assignment is assigned to an access terminal by a Forward Shared Signaling Channel (F-SSCH) assignment. Also carries broadcast information including pages and sector specific messages.

Forward Dedicated Pilot Channel (F-DPICH)
: Carries the dedicated pilot. This channel is present in BlockHopping mode, which is indicated over the Forward Primary Broadcast Channel 0 (F-pBCH0).

Forward Other Sector Interference Pilot Channel (F-OSICH)
: Carries information about the interference from other sectors to be received by all access terminals.

Forward Primary Broadcast Channel 0 (F-pBCH0)
: Carries information about the system to be received by all access terminals.

Forward Primary Broadcast Channel 1 (F-pBCH1)

Carries information about the sector to be received by all access terminals. Also carries quick pages.

Forward Shared Signaling Channel (F-SSCH)

Carries forward and reverse link data channel assignments, access grants, power control commands, and acknowledgement information for Reverse Data Channel (R-DCH) receptions.

1.4.3.2 Reverse physical channels

Reverse Access Channel (R-ACH)

Used by access terminals to initiate communication with the access network. The Reverse Access Channel is also used by access terminals to obtain timing corrections.

Reverse Acknowledgement Channel (R-ACKCH)

Carries acknowledgement information of a Forward Data Channel (F-DCH) reception.

Reverse Beam Feedback Channel (R-BFCH)

Carries information about the beam index and the quality of the forward link channel.

Reverse Channel Quality Indicator Channel (R-CQICH)

Carries information about the quality of the forward link channel of a sector as received by an access terminal. The Reverse Channel Quality Indicator Channel also carries information about the desired forward link serving sector.

Reverse Data Channel (R-DCH)

Carries information from an access terminal. The Reverse Data Channel is assigned to an access terminal by a Forward Shared Signaling Channel (F-SSCH) assignment.

Reverse Pilot Channel (R-PICH)

Carries the pilot.

Reverse Request Channel (R-REQCH)

Carries information about the buffer level at different quality of service classes for an access terminal. The Reverse Request Channel also carries information about the desired reverse link serving sector.

Reverse Subband Feedback Channel (R-SFCH)

Carries information about the quality of a subband of the forward link channel.

1.4.4 Protocols

1.4.4.1 Interfaces

This standard defines a set of interfaces for communications between protocols in the same entity and between a protocol executing in one entity and the same protocol executing in a peer entity.

In the following the generic term "entity" is used to refer to the access terminal and the access network.

Protocols in this specification have five types of interfaces:

Headers, messages, and blocks
: Used for communications between a protocol executing in one entity and the same protocol executing in a peer entity.

Commands
: Used by a protocol to obtain a service from another protocol within the same entity. For example, *AccessChannelMAC.Deactivate* causes the Access Channel MAC Protocol to abort any access attempt currently in progress.

Indications
: Used by a protocol to convey information regarding the occurrence of an event to another protocol within the same entity. Any protocol can register to receive these indications. For example, the access terminal Control Channel MAC Protocol returns a "Supervision Failed" indication when it is unable to receive the sector parameters message for a certain amount of time. This notification is then used by the Lower MAC Control Sublayer Air Link Management Protocol to close the connection.

Static Public Data
: Used to share information in a controlled way between protocols/transports. Static public data is shared between protocols/transports in the same layer, as well as between protocols/transports in different layers. Static public data is independent of the InUse SessionConfigurationToken and is supported by all subtypes of a protocol. For example the UATI is static public data for a protocol in the Session Control Sublayer. New protocol subtypes may define additional static public data. The additional static public data shall be initialized by the new protocol subtype when it is created.

Dynamic Public Data
: Used to share information in a controlled way between protocols/transports. Dynamic public data is shared between protocols/ transports in the same layer, as well as between protocols/transports in different layers. Dynamic public data is a function of the InUse SessionConfigurationToken and is defined separately for each subtype of a protocol. For example the protocol subtype is always dynamic public data for a protocol.

Commands and indications are written in the form of *Protocol.Command* and *Protocol.Indication*. For example, *AccessChannelMAC.Activate* is a command activating the Access Channel MAC, and *IdleState.ConnectionOpened* is an indication provided by the Lower MAC Control Sublayer Idle State Protocol indicating that the connection is now open. When the context is clear, the *Protocol* part is dropped (e.g., within the Idle State Protocol, *Activate* refers to *IdleState.Activate*).

Commands are always written in the imperative form, since they direct an action. Indications are always written in the past tense since they notify of events that have happened (e.g., *OpenConnection* for a command and *ConnectionOpened* for an indication).

Headers, messages, and blocks are binding on all implementations. Commands, indications, and public data are used as devices to help ensure a clear and precise specification. Access terminals and access networks can be compliant with this specification while choosing a different implementation that exhibits identical behavior.

1.4.4.2 States

When protocols exhibit different behavior as a function of the environment (e.g., if a connection is opened or not, if a session is opened or not, etc.), this behavior is captured in a set of states and events leading to a transition between states.

Unless otherwise specifically mentioned, the state of the access network refers to the state of a protocol engine in the access network as it applies to a particular access terminal. Since the access network communicates with multiple access terminals, multiple independent instantiations of a protocol will exist in the access network, each with its own independent state machine.

Unless otherwise specifically shown, the state transitions due to failure are not shown in the figures for the state transition diagrams.

Typical events leading to a transition from one state to another are the receipt of a message, a command an indication, or the expiration of a timer.

When a protocol is not functional at a particular time (e.g., the Access Channel MAC protocol at the access terminal when the access terminal has an open connection), the protocol is placed in a state called the Inactive state. This state is common for most protocols.

Other common state names are Open, indicating that the session or connection (as applicable to the protocol) is open, and Close, indicating that the session or connection is closed.

If a protocol has a single state other than the Inactive state, that state is usually called the Active state. If a protocol has more than one state other than the Inactive state, all of these states are considered active, and are given individual names (e.g., the Address Management Protocol at the access network has three states: Inactive, Setup, and Open).

1.4.4.3 SessionConfigurationToken

The SessionConfigurationToken is a 16 bit value that defines a complete set of protocol and transport instances that can be used to communicate between the access terminal and the access network.

A SessionConfigurationToken is InUse if the set of protocol and transport instances specified by the SessionConfigurationToken are currently being used to communicate between the access terminal and the access network. Otherwise, a SessionConfigurationToken is Suspended. Only one SessionConfigurationToken shall be InUse at a time.

The Session Configuration Protocol executes its save and commit procedures to swap the InUse SessionConfigurationToken with a Suspended SessionConfigurationToken as shown in Figure 5.

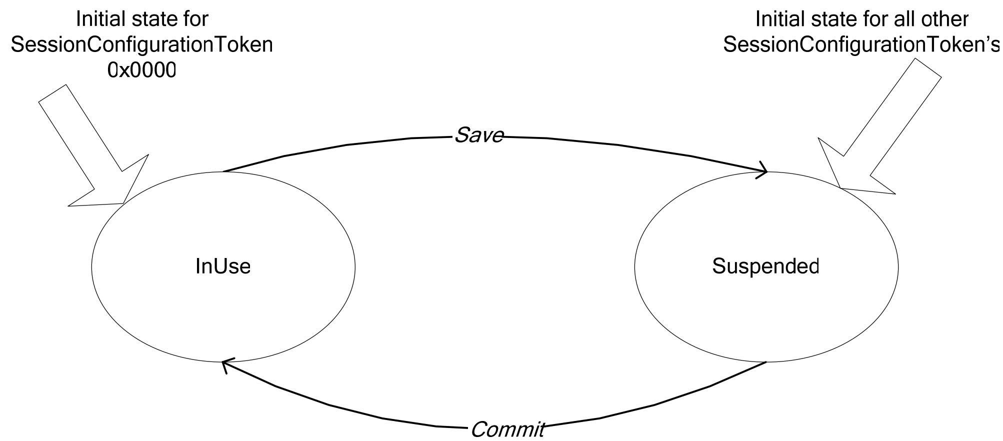

Figure 5 SessionConfigurationToken state diagram

1.4.4.4 InUse and Suspended protocol/transport instances

A protocol or transport instance is InUse if it is currently being used to communicate between the access terminal and the access network. Otherwise, a protocol or transport instance is Suspended. Only one protocol instance of a protocol type shall be InUse at a time. Each transport maps to a Transport defined in the Packet Consolidation Protocol. Only one transport instance corresponding to a Transport shall be InUse at a time. A protocol or transport instance shall correspond to exactly one SessionConfigurationToken.

The Session Configuration Protocol executes its save and commit procedures to swap the InUse protocol and transport instances associated with the current InUse SessionConfigurationToken with the Suspended protocol and transport instances associated with a Suspended SessionConfigurationToken.

Once the access terminal and access network agree upon using a new SessionConfigurationToken, the InUse protocol or transport instances associated with the current InUse SessionConfigurationToken are saved and the Suspended protocol or transport instances associated with the new SessionConfigurationToken are swapped in.

1.4.4.4.1 Protocol initialization and swap

The initialization procedures for a protocol/transport instance are invoked upon creation of the protocol/transport instance. A protocol/transport instance shall be created before it can become an InUse or Suspended protocol/transport instance.

The swap procedures for a Suspended protocol/transport instance are invoked when the Session Configuration Protocol performs its commit procedure to change the InUse SessionConfigurationToken.

If the swap procedure for a Suspended protocol/transport instance sets the state of the InUse protocol instance to a particular initial state, the procedures associated with entering the initial state are executed upon entering the initial state.

1.4.4.5 Procedures and messages

Each protocol/transport specifies procedures, blocks, and messages corresponding to the InUse protocol/transport instances.

1.4.4.6 Common commands

Most protocols support the following two commands:

- *Activate*, which commands the protocol to transition from the Inactive state to some other state.
- *Deactivate*, which commands the protocol to transition to the Inactive state. Some protocols do not transition immediately to the Inactive state, due to requirements on orderly cleanup procedures.

Other common commands are *Open* and *Close*, which command protocols to perform session open/close or connection open/close related functions.

1.4.4.7 Attribute negotiation

The Generic Attribute Update Protocol provides a means to update protocol and transport attributes. The protocol uses an AttributeUpdateRequest message, an AttributeUpdateAccept message, and an AttributeUpdateReject message to negotiate a mutually acceptable configuration. Only the protocol and transport attributes of the InUse protocol and transport instances may be configured using the Generic Attribute Update Protocol.

The Default Session Configuration Protocol defines a ConfigurationRequest message, a ConfigurationAccept message, and a ConfigurationReject message to update protocol and transport attributes for Suspended protocol and transport instances.

Protocols are associated with a Type that denotes the type of the protocol (e.g., Access Channel MAC Protocol) and with a Subtype that denotes a specific instance of a protocol (e.g., the Default Access Channel MAC Protocol).

1.4.4.8 Protocol overview

Figure 6 presents the protocol and transport types defined for each of the layers shown in Figure 2. The following is a brief description of each protocol and transport. A more complete description is provided in the introduction section of each layer.

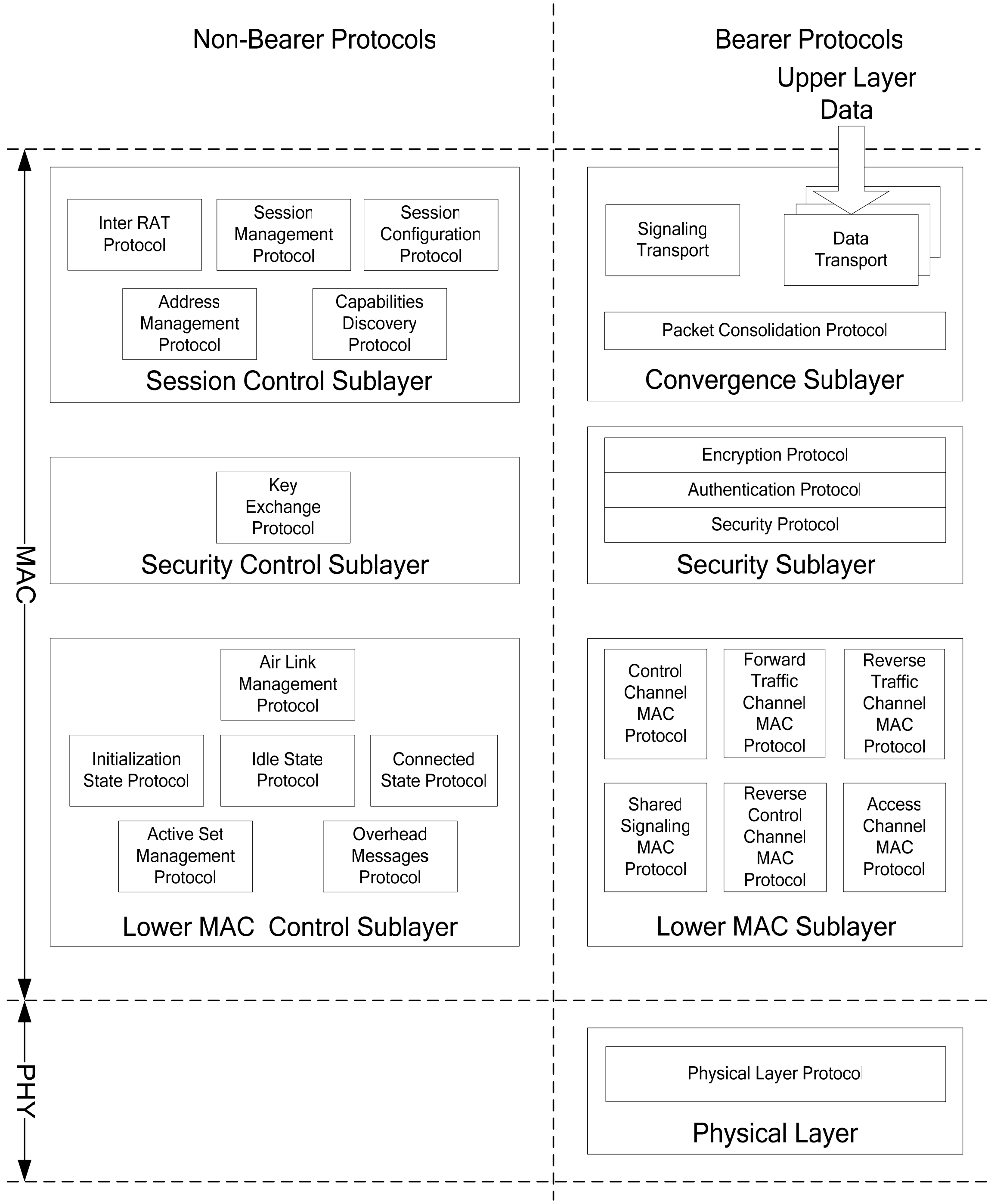

Figure 6 Protocol and transport types

- Session Control Sublayer:
  - Session Management Protocol: Provides means to control the activation and the deactivation of the Address Management Protocol, Capabilities Discovery Protocol and the Session Configuration Protocol. It also provides a session keep-alive mechanism.
  - Address Management Protocol: Provides unicast access terminal identifier (UATI) management.
  - Capabilities Discovery Protocol: Provides means for the access network to discover the capabilities of the access terminal.
  - Session Configuration Protocol: Provides means for negotiation of the SessionConfigurationToken used in the session.
  - Inter RAT Protocol: Provides the means to send messages for other radio access technologies.
- Convergence Sublayer:
  - Signaling Transport: Provides message transmission services, including fragmentation mechanisms, along with reliable and best-effort delivery mechanisms for signaling messages.
  - Data Transport: Provides two route instances for packets for a higher layer packet flow, as well as retransmission and duplicate detection for the packet flow of each route instance. It also provides procedures to enable and disable the Data Transport data flow.
  - Packet Consolidation Protocol: Adds the Packet Consolidation Protocol header to transport packets prior to transmission; and, after reception, removes the Packet Consolidation Protocol header and forwards the transport packets to the correct transport. Provides transmit prioritization and packet encapsulation for the Convergence Sublayer.
- Security Control Sublayer:
  - Key Exchange Protocol: Provides the procedures followed by the access network and the access terminal to exchange security keys for authentication and encryption.
- Security Sublayer:
  - Authentication Protocol: Provides the procedures followed by the access network and the access terminal for authenticating traffic.
  - Encryption Protocol: Provides the procedures followed by the access network and the access terminal for encrypting traffic.
  - Security Protocol: Provides procedures for generating a cryptosync based on the information fetched from the Lower MAC Sublayer that can be used by the Authentication Protocol and the Encryption Protocol.

- Lower MAC Control Sublayer:
  - Air Link Management Protocol: Provides the overall state machine management that an access terminal and an access network follow for the Lower MAC Control Sublayer.
  - Initialization State Protocol: Provides the procedures that an access terminal follows to acquire a network and that an access network follows to support network acquisition.
  - Idle State Protocol: Provides the procedures that an access terminal and an access network follow when a connection is not open.
  - Connected State Protocol: Provides the procedures that an access terminal and an access network follow when a connection is open.
  - Active Set Management Protocol: Provides the means to maintain the active set between the access terminal and the access network.
  - Overhead Messages Protocol: Provides broadcast messages and blocks containing information that is mostly used by Lower MAC Control Sublayer protocols.
- Lower MAC Sublayer:
  - Control Channel MAC Protocol: Provides the procedures followed by the access network to transmit, and by the access terminal to receive, the Control Channels.
  - Access Channel MAC Protocol: Provides the procedures followed by the access terminal to transmit, and by the access network to receive, the Access Channel.
  - Shared Signaling MAC Protocol: Provides the procedures followed by the access network to transmit, and by the access terminal to receive, the physical layer channels controlled by this protocol.
  - Forward Traffic Channel MAC Protocol: Provides the procedures followed by the access network to transmit, and by the access terminal to receive, the Forward Traffic Channel.
  - Reverse Control Channel MAC Protocol: Provides the procedures for the access terminal to transmit, and the access network to receive, the Reverse Control Channels.
  - Reverse Traffic Channel MAC Protocol: Provides the procedures followed by the access terminal to transmit, and by the access network to receive, the Reverse Traffic Channel.
- Physical Layer:
  - Physical Layer Protocol: Provides channel structure, frequency, power output, and modulation specifications for the forward and reverse links.

1.4.5 Default transports

This document defines two default transports that all compliant access terminals and access networks support:

- Default Signaling Transport: Provides the means to carry messages between a protocol/transport in one entity and the same protocol/transport in a peer entity. The Default Signaling Transport consists of a messaging protocol (Signaling Network Protocol) and a link layer protocol that provides message fragmentation, retransmission, and duplicate detection (Signaling Link Protocol).
- Default Data Transport: Consists of a link layer protocol that provides fragmentation, retransmission, and duplicate detection (Radio Link Protocol); a Route Selection Protocol that provides two route instances for a higher layer packet flow; and a Flow Control Protocol that provides flow control of data traffic.

The transports used are negotiated as part of session negotiation.

The air interface can support up to 8 parallel transports. The first transport (Transport0) always carries Signaling. Other transport can be used to carry, for example, the Default Data Transport to support different Quality of Service (QoS) requirements for data or other transports.

1.4.6 Sessions and connections

A session refers to a shared state between the access terminal and the access network. This shared state stores the protocols and protocol configurations that were negotiated and are used for communications between the access terminal and the access network.

Other than to open a session, an access terminal cannot communicate with an access network without having an open session.

A connection is a particular state of the air-link in which the access terminal is assigned a MACID. It also may be assigned a Forward Traffic Channel and/or a Reverse Traffic Channel.

During a single session, the access terminal and the access network can open and close a connection multiple times.

1.4.7 Security

The air interface supports a Security Sublayer, which can be used for authentication and encryption of access terminal traffic transported by the Control Channel, the Access Channel, the Forward Traffic Channel, and the Reverse Traffic Channel.

1.4.8 Physical Layer modes

The Physical Layer consists of two different duplexing modes, two different forward link hopping modes, two different synchronization modes and two different multi-carrier modes. The possible duplexing modes are Time Division Duplexing (TDD) and Frequency Division Duplexing (FDD). The different forward link hopping modes are SymbolRateHopping and BlockHopping. The possible synchronization modes are SemiSynchronous and Asynchronous. The possible multi-carrier modes are MultiCarrierOn and MultiCarrierOff.

1.4.8.1 MIMO support

The air interface supports two modes of operation for the Lower MAC Sublayer and Physical Layer

- Single-input-single-output (SISO) mode, and
- Multiple-input-multiple-output (MIMO) mode.

The MIMO mode is divided into two sub-modes: multiple codeword (MCW) and single codeword (SCW).

1.4.8.1.1 Access terminal requirements

All terminals shall support SISO mode. A MIMO-capable terminal shall support either the MIMO SCW or MCW sub-mode. The MIMO mode requires at least two antennas at the access network and at least two antennas at the access terminal.

1.4.9 Management Information Base

The standard includes the definition of a Management Information Base (MIB) module for managing the MAC and PHY. The objects in this MIB are defined using the mechanisms specified in the Structure of Management Information (SMI). The MIB module specified is compliant to SMIv2 which is described in RFC 2578 [18], RFC 2579 [19] and RFC 2580 [20]. The MIB is defined in Chapter 13.

1.4.10 Definitions

ChannelBand — The set of channels transmitted between the access network and the access terminals within a given frequency assignment. A ChannelBand consists of a Forward Link and a Reverse Link.

Cell — A group of one or more sectors that transmit from a common geographical location.

Dedicated Resource — An access network resource required to provide data service to the access terminal that is granted to the access terminal only when an access terminal has an open connection. Power control and rate control are not considered dedicated resources.

Effective Isotropically Radiated Power (EIRP) — The product of the power supplied to the antenna and the antenna gain in a direction relative to an isotropic antenna.

Effective Radiated Power (ERP) — The product of the power supplied to the antenna and its gain relative to a half-wave dipole in a given direction.

Empty Procedure — A procedure that performs no operations.

Forward Link PHY Frame — The forward link PHY frame consists of $N_{FRAME, F}$ OFDM symbols, see 8.1.11.

Global Positioning System (GPS)
A US government satellite system that provides location and time information to users. See Navstar GPS Space Segment/Navigation User Interfaces ICD-GPS-200 for specifications.

NULL A value which is not in the specified range of the field.

OFDM Symbol An OFDM symbol is comprised of individually modulated subcarriers which carry complex-valued data.

PHY Frame Index An integer value f such that $f = (n \times s) + m$, where n is the number of PHY frames in a superframe, s is the superframe index, and m is the offset of the frame in the current superframe, where $0 \leq m < n$. The current frame index is specific to a sector and link.

Reservation. Air interface resources set up by the access network to carry a higher layer flow. A Reservation is identified by its ReservationLabel. ReservationLabels are bound to Link Flows that carry higher layer flows. A Reservation can be either in the Open or Close state.

Reverse Link PHY Frame
The reverse link PHY frame consists of $N_{FRAME, R}$ OFDM symbols, see 8.1.11.

Subnet Mask (of length n)
A 128-bit value whose binary representation consists of n consecutive ‘1’s followed by 128-n consecutive ‘0’s.

Superframe One of the fundamental units of transmission on the forward and reverse links. On the Forward Link, a superframe consists of $N_{PREAMBLE}$ OFDM symbols followed by $N_{FDD, FL\ PHYFrames}$ Forward Link PHY Frames in FDD mode and $N_{TDD, FL\ PHYFrames}$ Forward Link PHY Frames in TDD mode. On the Reverse Link, a superframe consists of $N_{FDD, RL\ PHY\ Frames}$ RL PHY Frames in FDD mode and $N_{TDD, RL\ PHY\ Frames}$ RL PHY Frames in TDD mode. $N_{PREAMBLE}$ is defined in Chapter 8; $N_{FDD, FL\ PHY\ Frames}$, $N_{TDD, FL\ PHY\ Frames}$, $N_{FDD, RL\ PHY\ Frames}$ and $N_{TDD, RL\ PHY\ Frames}$ are defined in Chapter 7.

Superframe Index An integer value s such that: $s = \lfloor t / x \rfloor$, where t represents System Time in seconds and x represents the time for a superframe in seconds as defined in Chapter 9. Whenever the document refers to the System Time in superframes, it is referring to the value s.

System Time The time reference used by the system measured in seconds. System time is defined in 1.4.14.

Universal Coordinated Time (UTC)
An internationally agreed-upon time scale maintained by the Bureau International de l’Heure (BIH) used as the time reference by nearly all commonly available time and frequency distribution systems.

VCQI Long term averaged CQI value per interlace and subband.

1.4.11 Abbreviations and acronyms

ACK	Acknowledgment
ALMP	Air Link Management Protocol
AMP	Address Management Protocol
AN	Access network
ASMP	Active Set Management Protocol
AT	Access terminal
ATA	Access terminal assignment
ATI	Access terminal identifier
BATI	Broadcast access terminal identifier
BCD	Binary-coded decimal
BE	Best effort
BPSK	Binary phase shift keying
CC	Control channel
CDMA	Code division multiple access
CQI	Channel quality indicator
CRC	Cyclic redundancy check
CSP	Connected State Protocol
EIRP	Effective isotropically radiated power
F-ACQCH	Forward Acquisition Channel
F-AuxPICH	Forward Auxiliary Pilot Channel
F-CPICH	Forward Common Pilot Channel
F-DCH	Forward Data Channel
F-DPICH	Forward Dedicated Pilot Channel
F-pBCH	Forward Primary Broadcast Channel
F-OSICH	Forward Other Sector Interference Channel
F-SSCH	Forward Shared Signaling Channel
FCP	Flow control protocol
FCS	Frame check sequence
FDD	Frequency Division Duplex
FER	Frame error ratio
FFT	Fast Fourier Transform
FIFO	First in first out
FL	Forward link
FLAB	Forward link assignment block
FTC	Forward traffic channel
FWD	Forward
GPS	Global positioning system

H-ARQ	Hybrid automatic retransmission request
HW	Hardware
IFFT	Inverse Fast Fourier Transform
IFT	Inverse Fourier Transform
IP	Internet protocol
LAN	Local area network
LSB	Least significant bit
MAC	Medium access control
MATI	Multicast access terminal identifier
MCC	Mobile Country Code
MCW	Multiple codeword
MIB	Management Information Base
MIC	Message Integrity Check
MIMO	Multiple input multiple output
MNC	Mobile Network Code
MSB	Most significant bit
N_fft	FFT size
N/A	Not applicable
NAK	Negative acknowledgement
OFDM	Orthogonal frequency division multiplexing
OMP	Overhead Messages Protocol
OSICH	Other sector interference channel
PBRI	Pruned Bit-Reversal Interleaver
PCP	Packet Consolidation Protocol
PDU	Protocol data unit
PER	Packet error rate
PF	Packet format
PHY	Physical layer
PMK	Pairwise Master Key
PN	Pseudo noise (code sequence)
QAM	Quadrature amplitude modulation
QoS	Quality of service
QPSK	Quadrature phase shift keying
R-ACH	Reverse Access Channel
R-ACKCH	Reverse Acknowledgement Channel
R-BFCH	Reverse Beam Feedback Channel
R-CQICH	Reverse Channel Quality Indicator Channel
R-DCH	Reverse Data Channel

R-PICH	Reverse Pilot Channel
R-REQCH	Reverse Request Channel
R-SFCH	Reverse Subband Feedback Channel
RD	Reliable delivery
REV	Reverse
RF	Radio frequency
RLAB	Reverse link assignment block
RLP	Radio link protocol
ROHC	Robust header compression
RSP	Route selection protocol
RTC	Reverse traffic channel
RX	Receive or receiver
SCP	Session Configuration Protocol
SCW	Single codeword
SDMA	Space Division Multiple Access
SID	System Identifier
SISO	Single input single output
SLP	Signaling link protocol
SMP	Session Management Protocol
SNP	Signaling network protocol
SS	Shared signaling
SYNC	Synchronization
T2P	Traffic to pilot transmit power ratio
TCP	Transmission control protocol
TDD	Time Division Duplex
TL	Time limited
TX	Transmit or transmitter
UATI	Unicast access terminal identifier
UTC	Universal Temps Coordine (See Universal Coordinated Time)
WLAN	Wireless LAN

1.4.12 Notation

A[i]	The i^{th} element of array A. The first element of the array is A[0].
$<e_1, e_2, \ldots, e_n>$	A *structure* with elements ‘e1’, ‘e2’, …, ‘en’. Two structures $E = <e_1, e_2, \ldots, e_n>$ and $F = <f_1, f_2, \ldots, f_m>$ are equal if ‘m’ is equal to ‘n’ and e_i is equal to f_i for i=1, …n. Given $E = <e_1, e_2, \ldots, e_n>$ and $F = <f_1, f_2, \ldots, f_n>$, the assignment “E = F” denotes the following set of assignments: $e_i = f_i$, for i=1, …n.
S.e	The member of the structure ‘S’ that is identified by ‘e’.
M[i:j]	Bits i^{th} through j^{th} inclusive ($i \geq j$) of the binary representation of variable M. M[0:0] denotes the least significant bit of M.
$\vert$	Concatenation operator. (A $\vert$ B) denotes variable A concatenated with variable B.
$\times$	Indicates multiplication.
$\lfloor x \rfloor$	Indicates the largest integer less than or equal to x: $\lfloor 1.1 \rfloor = 1, \lfloor 1.0 \rfloor = 1$.
$\lceil x \rceil$	Indicates the smallest integer greater than or equal to x: $\lceil 1.1 \rceil = 2, \lceil 2.0 \rceil = 2$.
$\vert x \vert$	Indicates the absolute value of x: $\vert -17 \vert = 17, \vert 17 \vert = 17$.
$\oplus$	Indicates exclusive OR (modulo-2 addition).
min (x, y)	Indicates the minimum of x and y.
max (x, y)	Indicates the maximum of x and y.
x mod y	Indicates the remainder after dividing x by y: $x \bmod y = x - (y \times \lfloor x/y \rfloor)$.

Unless otherwise specified, the format of field values is unsigned binary.

Unless indicated otherwise, this standard presents numbers in decimal form. Binary numbers are distinguished in the text by the use of single quotation marks. Hexadecimal numbers are distinguished by the prefix ‘0x’.

Unless specified otherwise, each field of a packet shall be transmitted in sequence such that the most significant bit (MSB) is transmitted first and the least significant bit (LSB) is transmitted last. The MSB is the left-most bit in the figures in this document. If there are multiple rows in a table, the top-most row is transmitted first. If a table is used to show the sub-fields of a particular field or variable, the top-most row consists of the MSBs of the field. Within a row in a table, the left-most bit is transmitted first. Notations of the form “repetition factor of N” or “repeated N times” mean that a total of N versions of the item are used.

1.4.13 Malfunction detection

The access terminal shall have a malfunction timer that is separate from and independent of all other functions and that runs continuously whenever power is applied to the transmitter of the access terminal. The timer shall be reset and restarted periodically if the access terminal is functioning properly. If the timer expires, the access terminal shall cease transmission. The maximum time allowed for expiration of the timer is two seconds.

1.4.14 System time

System Time counts the number of seconds that have elapsed since the start of System Time on a per sector basis.

In a synchronous access network, all sector air interface transmissions are referenced to a common system-wide timing reference that uses the Global Positioning System (GPS) time, which is traceable to and synchronous with Universal Coordinated Time (UTC). GPS and UTC differ by an integer number of seconds, specifically the number of leap second corrections added to UTC since January 6, 1980. The start of System Time is January 6, 1980 00:00:00 UTC, which coincides with the start of GPS time.

In a synchronous access network, System Time keeps track of leap second corrections to UTC but does not use these corrections for physical adjustments to the System Time clocks.

In an asynchronous access network, System Time need not be traceable and synchronous to a common timing reference.

1.4.15 Revision number

Access terminals and access networks complying with the requirements of this specification shall set their revision number to 0x01.

2 Session Control Sublayer

2.1 Introduction

2.1.1 General overview

The Session Control Sublayer contains protocols used to negotiate a session between the access terminal and the access network. The Session Control Sublayer consists of non-bearer protocols and does not modify transmitted or received packets.

A session is a shared state maintained between the access terminal and the access network, including information such as:

- A Unicast address assigned to the access terminal (UATI).
- The set of protocols and applications used by the access terminal and the access network to communicate over the air-link.
- Configuration settings for these protocols and applications (e.g., authentication keys, parameters for Lower MAC Sublayer Protocols, etc.).

During a single session, the access terminal and the access network can open and close a connection multiple times. Therefore, sessions will be closed rarely and only on occasions, such as when the access terminal leaves the coverage area or during prolonged periods in which the access terminal is unavailable.

The Session Control Sublayer contains the following protocols:

- Session Management Protocol: Provides the means to control the activation of the other Session Control Sublayer protocols. In addition, this protocol ensures that the session is still valid and manages closing of the session.
- Address Management Protocol: Specifies procedures for the initial UATI assignment and maintains the access terminal addresses.
- Session Configuration Protocol: Provides the means to negotiate the *SessionConfigurationToken*'s used during the session.
- Capabilities Discovery Protocol: Provides the means for the access network to discover the capabilities of the access terminal.
- Inter RAT Protocol: Provides the means to send messages for other radio access technologies.

The relationship between the Session Control Sublayer protocols is illustrated in Figure 7.

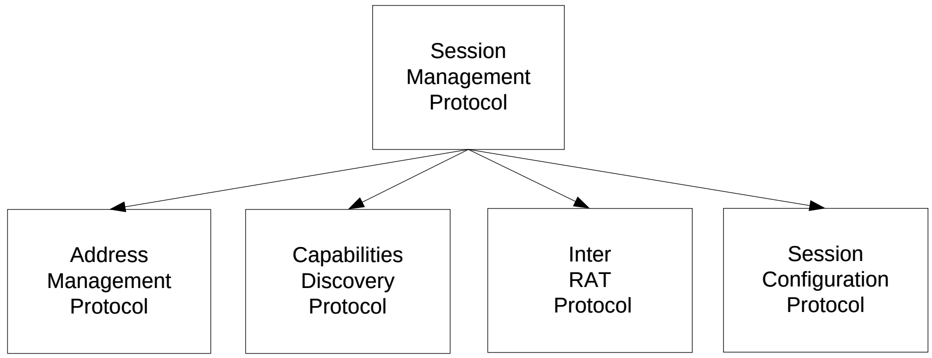

Figure 7 Session Control Sublayer protocols

2.2 Default Session Management Protocol

2.2.1 Overview

The Default Session Management protocol provides the means to control the activation of the Address Management Protocol, the Capabilities Discovery Protocol, Inter RAT Protocol and the Session Configuration Protocol when a session is established. This protocol also periodically ensures that the session is still valid and manages closing the session.

This protocol uses the Signaling Transport to transmit and receive messages.

The actual behavior and message exchange in each state of this protocol are mainly governed by protocols that are activated by the Default Session Management Protocol. These protocols return indications, which trigger the state transitions of this protocol.

This protocol can be in one of four states:

- Inactive State: This state applies only to the access terminal. In this state, there are no communications between the access terminal and the access network.
- AMP Setup State: In this state, the access terminal and access network perform exchanges governed by the Address Management Protocol, and the access network assigns a UATI to the access terminal.
- Open State: In this state, a session is open.
- Close State: This state applies only to the access network. In this state, the access network waits for the close procedure to complete.

Figure 8 provides an overview of the access terminal states and state transitions.

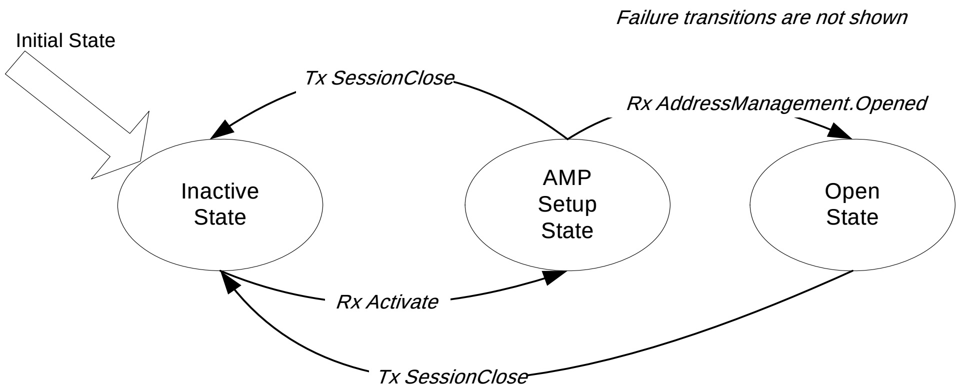

Figure 8 Session Management Protocol state diagram (access terminal)

Figure 9 provides an overview of the access network states and state transitions.

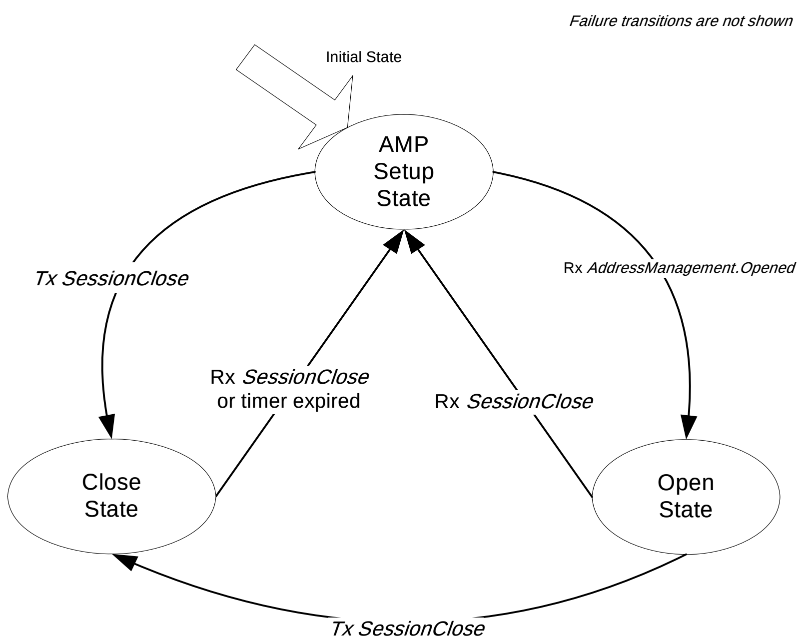

Figure 9 Session Management Protocol state diagram (access network)

2.2.2 Primitives

2.2.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

2.2.2.2 Return indications

This protocol returns the following indications:

- *SessionOpened*
- *SessionClosed*

2.2.3 Public data

2.2.3.1 Static public data

- SessionSeed

2.2.3.2 Dynamic public data

- Subtype for this protocol

2.2.4 Protocol initialization and swap procedures

2.2.4.1 Protocol initialization

Upon creation, the instance of this protocol in the access terminal and access network shall perform the following:

- The value of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol at the access network shall enter the AMP Setup State.
- The protocol at the access terminal shall enter the Inactive State.

2.2.4.2 Protocol swap

Upon swap, the protocol in the access terminal and the access network shall enter the Open State

2.2.5 Procedures

2.2.5.1 Processing the Activate command

If the access terminal receives the *Activate* command in the Inactive State, it shall transition to the AMP Setup State.

If the access terminal receives the *Activate* command in any state other than the Inactive State, the command shall be ignored.

The access network shall ignore the command.

The list of events that causes an *Activate* command to be sent to this protocol is outside the scope of this specification.

2.2.5.2 Processing the Deactivate command

If the access terminal receives a *Deactivate* command in the Inactive State, the command shall be ignored.

If the access terminal receives a *Deactivate* command in any state other than the Inactive State, the access terminal shall perform the following:

- Send a SessionClose message to the access network.
- Perform the procedures for closing the session in 2.2.5.7.
- Transition to the Inactive State.

If the access network receives a *Deactivate* command in the Close State, the command shall be ignored.

If the access network receives a *Deactivate* command in any state other than the Close State, the access network shall send a SessionClose message and transition to the Close State.

The list of events that causes a *Deactivate* command to be sent to this protocol is outside the scope of this specification.

2.2.5.3 Processing the SessionOpen message

If the access network receives the SessionOpen message in any state other than AMP Setup state, it shall ignore the message.

If the access network receives the SessionOpen message in the AMP Setup state, it shall issue an *AddressManagement.Activate* command.

2.2.5.4 Processing the SessionClose message

If the access terminal receives a SessionClose message in the Inactive State, it shall ignore the message.

If the access terminal receives a SessionClose message in any state other than the Inactive State, the access terminal shall perform the following:

- Send a SessionClose message to the access network.
- Perform the procedures for closing the session in 2.2.5.7.
- Transition to the Inactive State.

If the access network receives a SessionClose message in the Close State, the access network shall process it as specified in 2.2.5.11.

If the access network receives a SessionClose message in any state other than the Close State, the access network shall perform the following:

- Perform the procedures for closing the session in 2.2.5.7.
- Transition to the AMP Setup State.

2.2.5.5 Processing failed indications

The access terminal shall ignore an *AddressManagement.Failed* or a *SessionConfiguration.Failed* indication, if the access terminal receives the indication in the Inactive State.

If the access terminal receives an *AddressManagement.Failed* or a *SessionConfiguration.Failed* indication while in any state other than the Inactive State, then the access terminal shall perform the following:

- Send a SessionClose message to the access network.
- Perform the procedures for closing the session in 2.2.5.7.
- Transition to the Inactive State.

If the access network receives an *AddressManagement.Failed* or a *SessionConfiguration.Failed* indication, the access network shall perform the following:

- Send a SessionClose message to the access terminal.
- Transition to the Close State.

2.2.5.6 Processing the ForwardTrafficChannelMAC.SessionLost indication

If the access terminal receives a *ForwardTrafficChannelMAC.SessionLost* indication in the Inactive State, it shall ignore the indication.

If the access terminal receives a *ForwardTrafficChannelMAC.SessionLost* in any state other than the Inactive State, the access terminal shall perform the following:

- Perform the procedures for closing the session in 2.2.5.7.
- Transition to the Inactive State.

2.2.5.7 Procedures for closing the session

The access terminal or access network shall perform the following to close the session:

- Issue an *AirLinkManagement.CloseConnection* command.
- Issue an *AddressManagement.Deactivate* command.
- Issue a *CapabilitiesDiscovery.Deactivate* command.
- Issue a *InterRAT.Deactivate* command.
- Issue a *SessionConfiguration.Deactivate* command.
- Return a *SessionClosed* indication.

2.2.5.8 Inactive state

This state only applies to the access terminal. In this state there are no communications between the access terminal and the access network. The access terminal does not maintain any session-related state and the access network may be unaware of the access terminal's existence within its coverage area when the access terminal's Session Management Protocol is in this state.

Upon entering this state the access terminal shall perform the following:

- Set public data *SessionSeed* to the 32-bit pseudorandom number generated using output of the pseudorandom number generator specified in 10.6.

2.2.5.9 AMP setup state

In this state the Session Management Protocol in the access terminal sends a request to the access network asking for the session to be opened and waits for the Address Management Protocol to respond.

2.2.5.9.1 Access terminal requirements

Upon entering the AMP Setup State, the access terminal shall perform the following:

- Send SessionOpen message to the access network
- Send an *AddressManagement.Activate* command to the Address Management Protocol

If the access terminal receives an *AddressManagement.Opened* indication, it shall perform the following:

- Issue a *CapabilitiesDiscovery.Activate* command.
- Issue a *InterRAT.Activate* command.
- Issue a *SessionConfiguration.Activate* command.
- Return a *SessionOpened* indication.
- Transition to the Open State.

2.2.5.9.2 Access network requirements

Upon entering the AMP Setup State, the access network is waiting for a SessionOpen message.

When the access network receives a SessionOpen message, it shall issue an *AddressManagement.Activate* command to the Address Management Protocol.

If the access network receives an *AddressManagement.Opened* indication, it shall perform the following:

- Issue a *CapabilitiesDiscovery.Activate* command.
- Issue a *InterRAT.Activate* command.
- Issue a *SessionConfiguration.Activate* command.
- Return a *SessionOpened* indication.
- Transition to the Open State.

2.2.5.10 Open state

In the Open State, the access terminal has an assigned UATI and the access terminal and the access network have a session.

The access terminal and the access network shall support the keep-alive mechanism defined in 2.2.5.10.1.

2.2.5.10.1 Keep-alive functions

The access terminal and the access network shall monitor the traffic on the transports, directed to or from the access terminal. If either the access terminal or the access network detects a period of inactivity of at least SessionCloseTimer/$N_{SMPKeepAlive}$ minutes, it may send a KeepAliveRequest message. The recipient of the message shall respond by sending the KeepAliveResponse message. When a KeepAliveResponse message is received, the access terminal shall not send another KeepAliveRequest message for at least SessionCloseTimer/$N_{SMPKeepAlive}$ minutes.

If the access terminal does not detect any traffic from the access network directed to it for a period of at least SessionCloseTimer minutes, it shall perform the following:

- Perform the procedures for closing the session in 2.2.5.7.
- Transition to the Inactive State.

If the access network does not detect any traffic from the access terminal directed to it for a period of at least SessionCloseTimer minutes, it should perform the following:

- Perform the procedures for closing the session in 2.2.5.7.
- Transition to the AMP Setup State.

If the value of SessionCloseTimer is set to zero, the access terminal and the access network shall not send or expect keep-alive messages, and shall disable the transitions occurring as a consequence of not receiving these messages.

2.2.5.11 Close state

The Close State is associated only with the protocol in the access network. In this state, the protocol in the access network waits for a SessionClose message from the access terminal or an expiration of a timer.

The access network shall set the Close State timer upon entering this state. The value of this timer shall be set to $T_{SMPMinClose}$.

When the access network receives a SessionClose message or when the Close State timer expires, the protocol shall:

- Perform the procedures for closing the session in 2.2.5.7.
- Transition to the AMP Setup State.

While in this state, if the access network receives any packet from the access terminal which is addressed by the UATI assigned during this session and contains anything but a SessionClose message, it shall stay in the Close State and perform the following:

- Discard the packet.
- Respond with a SessionClose message.

2.2.6 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes.

2.2.6.1 SessionOpen

The access terminal sends the SessionOpen message to initiate a session with the access network.

Field	Length (bits)
MessageID	8
SessionSeed	32

MessageID — The access terminal shall set this field to 0x00.

SessionSeed — This field shall be set to the value of the public data *SessionSeed* associated with the access terminal's session.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

2.2.6.2 SessionClose

The sender sends the SessionClose message to terminate the session.

Field	Length (bits)
MessageID	8
CloseReason	8
MoreInfoLen	8
MoreInfo	8 × MoreInfoLen

MessageID — The sender shall set this field to 0x01.

CloseReason — The sender shall set this field to the close reason as shown in Table 1.

Table 1 Encoding of CloseReason field

Field Value	Meaning	MoreInfoLen	MoreInfo
0x00	Normal Close	0	N/A
0x01	Close Reply	0	N/A
0x02	Protocol Error	0	N/A
0x03	Protocol Negotiation Error	variable	Zero or more Type followed by Subtype followed by offending attribute records (see 10.3 for attribute record definition).
0x04	Session Configuration Failure	2	*SessionConfigurationToken*
0x05	Session Lost	0	N/A
0x06	Session Unreachable	0	N/A
0x07	All session resources busy	0	N/A
All other values are reserved			

MoreInfoLen Length in octets of the MoreInfo field.

MoreInfo Additional information pertaining to the closure. The format of this field is determined by the particular close reason.

Channels	FTC RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

2.2.6.3 KeepAliveRequest

The sender sends the KeepAliveRequest to verify that the peer is still alive.

Field	Length (bits)
MessageID	8

MessageID The sender shall set this field to 0x02.

Channels	FTC RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

2.2.6.4 KeepAliveResponse

The sender sends the KeepAliveResponse message as an answer to the KeepAliveRequest message.

Field	Length (bits)
MessageID	8

MessageID The sender shall set this field to 0x03.

Channels	FTC RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

2.2.7 Interface to other protocols

2.2.7.1 Commands sent

This protocol issues the following commands:

- *AddressManagement.Activate*
- *CapabilitiesDiscovery.Activate*
- *InterRAT.Activate*
- *SessionConfiguration.Activate*
- *AddressManagement.Deactivate*
- *CapabilitiesDiscovery.Deactivate*
- *InterRAT.Deactivate*
- *SessionConfiguration.Deactivate*
- *AirLinkManagement.CloseConnection*

2.2.7.2 Indications

This protocol registers to receive the following indications:

- *AddressManagement.Failed*
- *SessionConfiguration.Failed*
- *AddressManagement.Opened*
- *ForwardTrafficChannelMAC.SessionLost*

2.2.8 Configuration attributes

The negotiable attributes for this protocol are listed in Table 2. The access terminal shall use as defaults the values in Table 2 that are listed in ***bold italics***.

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Session Management Protocol.

Table 2 Configurable attributes

Attribute ID	Attribute	Values	Meaning
0xff	SessionCloseTimer	***0x0CA8***	Default is 54 hours.
		0x0000 to 0xFFFF	0x0000 means disable keep-alive messages; all other values are in minutes.

2.2.9 Protocol numeric constants

Constant	Meaning	Value
$N_{SMPType}$	Type field for this protocol	Table 9
$N_{SMPDefault}$	Subtype field for this protocol	0x0000
$N_{SMPKeepAlive}$	Maximum number of keep-alive transactions within SessionCloseTimer.	3
$T_{SMPMinClose}$	Minimum recommended timer setting for Close State	300 seconds

2.2.10 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol and the following parameters.

2.2.10.1 SessionSeed parameter

Table 3 Format of the parameter record for the SessionSeed parameter

Field	Length (bits)
ParameterType	8
Length	8
SessionSeed	32

ParameterType This field shall be set to 0x01 for this parameter record.

Length — This field shall be set to the length of this parameter record in units of octets, excluding the Length field.

SessionSeed — This field shall be set to the value of the SessionSeed associated with the access terminal's session.

2.3 Default Address Management Protocol

2.3.1 Overview

The Default Address Management Protocol provides the following functions:

- Initial UATI assignment
- Maintaining the access terminal UATI as the access terminal moves between subnets.

This protocol uses the Signaling Transport to transmit and receive messages.

This protocol operates in one of three states:

- Inactive State: In this state there are no communications between the access terminal and the access network.
- Setup State: In this state the access terminal and the access network perform a UATIAssignment/UATIComplete exchange to assign the access terminal a UATI.
- Open State: In this state the access terminal has been assigned a UATI. The access terminal and access network may also perform a UATIUpdateRequest/UATIAssignment /UATIComplete or a UATIAssignment/UATIComplete exchange respectively, so that the access terminal obtains a new UATI.

The protocol states and the messages and events causing the transition between the states are shown in Figure 10 and Figure 11.

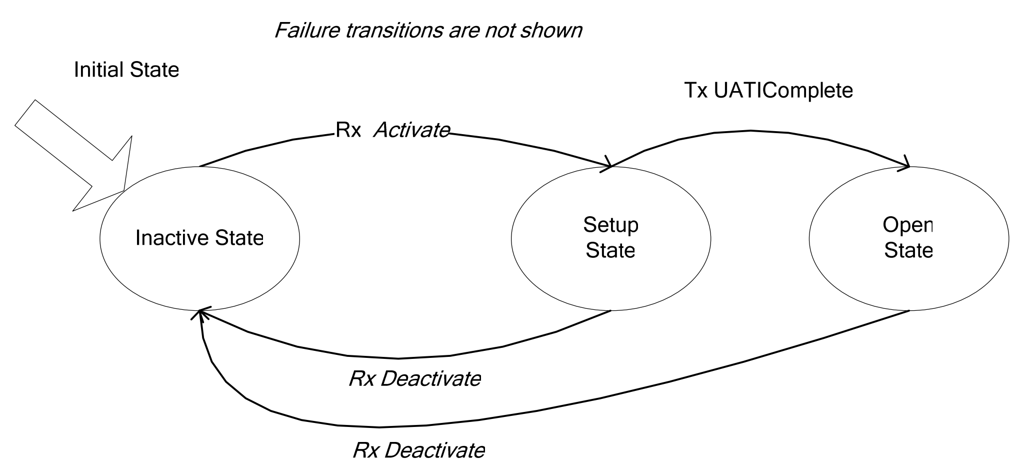

Figure 10 Address Management Protocol state diagram (access terminal)

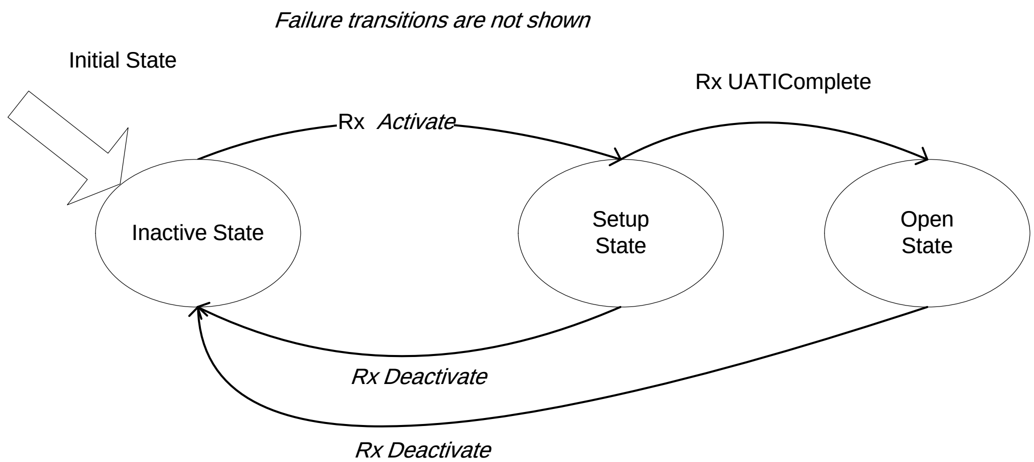

Figure 11 Address Management Protocol state diagram (access network)

2.3.2 Primitives

2.3.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*
- *UpdateUATI*
- *RetrieveHWID*

2.3.2.2 Return indications

This protocol returns the following indications:

- *Opened*
- *UATIReleased*
- *UATIAssigned*
- *Failed*
- *SubnetChanged*

2.3.3 Public data

2.3.3.1 Static public data

- ReceiveATIList
- TransmitUATI

2.3.3.2 Dynamic public data

- Subtype for this protocol

2.3.4 Connection endpoints

The following Connection Endpoints are defined (to be used by the SLP protocol):

- The addresses specified by entries in the ReceiveATIList list whose ATIType is equal to ‘10’ (i.e., UATI) all define the same connection endpoint.
- Each unique <ATI, SectorID> defines a separate connection endpoint. The ATI is an entry in the ReceiveATIList with ATIType equal to ‘00’ (i.e., BATI), and the SectorID is defined in the SectorParameters message in the Lower MAC Control Sublayer.

2.3.5 Protocol initialization and swap procedures

2.3.5.1 Protocol initialization

Upon creation, the protocol in the access terminal and access network shall perform the following:

- The value of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol at the access terminal and the access network shall enter the Inactive State.

2.3.5.2 Protocol swap

Upon swap, the protocol in the access terminal and the access network shall enter the Open State.

2.3.6 Procedures

2.3.6.1 Processing the Activate command

If the protocol receives the *Activate* command in the Inactive State, the protocol shall transition to the Setup State.

If the protocol receives the *Activate* command in any state other than the Inactive State, the command shall be ignored.

2.3.6.2 Processing the Deactivate command

If the protocol receives the *Deactivate* command in the Inactive State, the command shall be ignored.

If the protocol receives the *Deactivate* command in any state other than the Inactive State, the protocol shall transition to the Inactive State and return a *UATIReleased* indication.

2.3.6.3 Processing the UpdateUATI command

If the protocol receives the *UpdateUATI* command in any state other than the Open State, the command shall be ignored.

If the access terminal receives an *UpdateUATI* command in the Open State, it shall send a UATIUpdateRequest message.

If the access network receives an *UpdateUATI* command in the Open State, it may send a UATIAssignment message.

A comprehensive list of events causing the *UpdateUATI* command is beyond the scope of this specification.

2.3.6.4 Processing the RetrieveHWID command

The access terminal shall ignore the *RetrieveHWID* command in all states. The access network shall ignore the *RetrieveHWID* command when it is received in any state other than the Open State.

If the access network receives a *RetrieveHWID* command in the Open State, it may send a HardwareIDRequest message.

2.3.6.5 UATIAssignment message validation

When the access network first sends a UATIAssignment message, it shall set the MessageSequence field of the message to zero. Subsequently, the access network shall increment this field modulo 256 each time it sends a UATIAssignment message.

The access terminal shall initialize a receive pointer for the UATIAssignment message validation, *V(R)*, to 255. When the access terminal receives a UATIAssignment message, it shall validate the message, using the procedure defined in 10.7 (S is equal to 8). The access terminal shall discard the message if it is invalid.

2.3.6.6 Processing HardwareIDRequest message

Upon reception of a HardwareIDRequest message, the access terminal shall respond with a HardwareIDResponse message. The access terminal shall set the HardwareID record of the HardwareIDResponse message to the unique ID that has been assigned to the terminal by the manufacturer.

2.3.6.7 Inactive state

In this state, there are no communications between the access terminal and the access network. The access terminal does not have an assigned UATI, and the access network does not maintain a UATI for the access terminal and may be unaware of the access terminal's existence within its coverage area.

2.3.6.7.1 Access terminal requirements

Upon entering the Inactive State, the access terminal shall perform the following:

- Clear the ReceiveATIList
- Add the following entry to the ReceiveATIList:
  <ATIType = '00', ATI = NULL>.
- Set the TransmitUATI to NULL.
- Set the UATI to NULL.
- Set the UATISubnetMask to NULL.
- Disable the Address timers.

If the access terminal receives an *Activate* command, it shall transition to the Setup State.

2.3.6.7.2 Access network requirements

Upon entering the Inactive State, the access network shall perform the following:

- Set the value of the access terminal’s UATI to NULL.
- Set the value of the access terminal’s UATISubnetMask to NULL.

The access network shall transition to the Setup State if it receives an *Activate* command.

2.3.6.8 Setup state

In this state, the access network assigns a UATI to the access terminal.

2.3.6.8.1 Access terminal requirements

Upon entering the Setup State, the access terminal expects a UATIAssignment message.

If the access terminal does not receive a UATIAssignment message within $T_{ADMPATResponse}$ seconds after entering the Setup state, it shall return a *Failed* indication and transition to the Inactive State.

If the access terminal receives a UATIAssignment message, the access terminal shall validate the message sequence number as defined in 10.7. If the message is valid, it shall perform the following:

- Set its UATI and UATISubnetMask to the UATI and UATISubnetMask fields specified in the message.
- Add the following entry to the ReceiveATIList:
  <ATIType = ‘10’, ATI = UATI[31:0])>.
- Set the TransmitUATI to UATI.
- Return an *Opened* indication.
- Return a *UATIAssigned* indication.
- Send a UATIComplete message.
- Transition to the Open State.

2.3.6.8.2 Access network requirements

In this state the access network shall perform the following:

- Assign a Unicast Access Terminal Identifier (UATI) to the access terminal for the session.
- Send a UATIAssignment message.

When the access network receives the corresponding UATIComplete message with the MessageSequence field that is equal to the MessageSequence field of the UATIAssignment message sent, it shall perform the following:

- Return an *Opened* indication.
- Return a *UATIAssigned* indication.
- Transition to the Open State.

If the access network does not receive the corresponding UATIComplete message in response to the UATIAssignment message, it may re-transmit the UATIAssignment message. If the access network does not receive the UATIComplete message after an implementation-specific number of re-transmissions of the UATIAssignment message, it shall return a *Failed* indication and transition to the Inactive State.

2.3.6.9 Open state

In this state the access terminal has been assigned a UATI.

2.3.6.9.1 Access terminal requirements

If the access terminal receives a *ActiveSetManagement.IdleHO* indication or a *ConnectedState.ConnectionClosed*, and then receives an *OverheadMessages.Updated* indication, and if either of the following two conditions is true, it shall send a UATIUpdateRequest message:

- The UATISubnetMask is not equal to the SubnetMask of the sector in the active set, or
- The result of bitwise logical AND of the UATI and its subnet mask specified by UATISubnetMask is different from the result of bitwise logical AND of SectorID and its subnet mask specified by SubnetMask (where SectorID and SubnetMask correspond to the sector in the active set).

If the access terminal receives an *UpdateUATI* command, it shall process the command as specified in 2.3.6.3.

Upon sending a UATIUpdateRequest message, the access terminal shall start a UATIResponse timer with a timeout value of $T_{ADMPATResponse}$ seconds. The access terminal shall disable this timer if either of the following conditions is true:

- The UATISubnetMask is equal to the SubnetMask of the sector in the active set, and the result of bitwise logical AND of the UATI and its subnet mask specified by UATISubnetMask is the same as the result of bitwise logical AND of SectorID and its subnet mask specified by SubnetMask (where SectorID and SubnetMask correspond to the sector in the active set), or
- The access terminal receives a valid UATIAssignment message.

If the UATIResponse timer expires, the access terminal shall return a *Failed* indication and transition to the Inactive State.

If the access terminal receives a UATIAssignment message, the access terminal shall validate the message sequence number as defined in 10.7. If the message is valid, it shall perform the following:

- Set its UATI and UATISubnetMask to the UATI and UATISubnetMask fields in the message.
- Add the following entry to the ReceiveATIList:
  <ATIType = ‘10’, ATI = UATI[31:0]>.
- Set the TransmitUATI to UATI.
- Return a *UATIAssigned* indication.
- Send a UATIComplete message.

- Reset and start an Address timer with a timeout value of $T_{ADMPAddress}$ for the added entry to the ReceiveATIList.

The access terminal shall perform the following when an Address timer corresponding to an entry in the ReceiveATIList expires:

- Disable the Address timer for that entry.
- Delete all of the entries in the ReceiveATIList that are older than the entry whose Address timer has expired. An entry X in the list is considered older than another entry Y, if the entry X has been added to the list prior to the entry Y.

2.3.6.9.2 Access network requirements

The access network may send a UATIAssignment message at any time in this state. The following are some of the possible triggers for sending a UATIAssignment message:

- Receiving an *ActiveSetManagement.ActiveSetUpdated* indication.
- Receiving an *UpdateUATI* command.
- Receiving a valid UATIUpdateRequest message.

The access network may return a *SubnetChanged* indication and send a UATIAssignment message after reception of a *ActiveSetManagement.ActiveSetUpdated* indication. The triggers for returning a *SubnetChanged* indication after reception of a *ActiveSetManagement.ActiveSetUpdated* indication are outside the scope of this specification.

When the access network sends a UATIAssignment message, it shall perform the following:

- Assign a Unicast Access Terminal Identifier (UATI) to the access terminal for the session and include it in a UATIAssignment message.

When the access network receives a UATIComplete message with the MessageSequence field that is equal to the MessageSequence field of the last UATIAssignment message that it has sent, it shall return a *UATIAssigned* indication.

If the access network does not receive the UATIComplete message in response to the corresponding UATIAssignment message within a certain time interval that is specified by the access network[8], it should re-transmit the UATIAssignment message. If the access network does not receive the UATIComplete message after an implementation-specific number of re-transmissions of the UATIAssignment message, it shall return a *Failed* indication and transition to the Inactive State.

[8] The value of this timeout is determined by the access network, and the specification of the timeout value is outside the scope of this document.

2.3.7 Message formats

2.3.7.1 UATIUpdateRequest

The access terminal sends the UATIUpdateRequest message to request that a UATI is reassigned to it by the access network.

Field	Length (bits)
MessageID	8
MessageSequence	8
TransmitUATI	128

MessageID — The access terminal shall set this field to 0x00.

MessageSequence — The access terminal shall increment this field modulo 256 for each new UATIUpdateRequest message sent. If this is the first UATIUpdateRequest message sent by the access terminal, the access terminal shall set this field to zero.

TransmitUATI — The current value of the TransmitUATI.

Channels	RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

2.3.7.2 UATIAssignment

The access network sends the UATIAssignment message to assign or re-assign a UATI to the access terminal.

Field	Length (bits)
MessageID	8
MessageSequence	8
UATISubnetMask	8
SessionSeed	32
UATI	128

MessageID — The access network shall set this field to 0x01.

MessageSequence — The access network shall increment this field modulo 256 for each new UATIAssignment message sent to this access terminal. If this is the first UATIAssignment message sent to this access terminal, the access network shall set this field to zero.

UATISubnetMask — The access network shall set this field to the number of consecutive 1's in the subnet mask of the subnet to which the assigned UATI belongs.

SessionSeed — This field shall be set to the value of the public data *SessionManagement.SessionSeed* associated with the access terminal's session.

UATI — The access network shall set this field to the UATI that it is assigning to the access terminal.

Channels	FTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

2.3.7.3 UATIComplete

The access terminal sends this message to notify the access network that it has received the UATIAssignment message.

Field	Length (bits)
MessageID	8
MessageSequence	8

MessageID — The access terminal shall set this field to 0x02.

MessageSequence — The access terminal shall set this field to the MessageSequence field of the UATIAssignment message whose receipt this message is acknowledging.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

2.3.7.4 HardwareIDRequest

The access network uses this message to query the access terminal of its Hardware ID information.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The access network shall set this field to 0x03.

TransactionID — The access network shall set this field according to 10.8.

Channels	FTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

2.3.7.5 HardwareIDResponse

The access terminal sends this message in response to the HardwareIDRequest message.

Field	Length (bits)
MessageID	8
TransactionID	8
HardwareIDType	24
HardwareIDLength	8
HardwareIDValue	8 × HardwareIDLength

MessageID — The access terminal shall set this field to 0x04.

TransactionID — The access terminal shall set this field the TransactionID field of the corresponding HardwareIDRequest message.

HardwareIDType — The access terminal shall set this field according to Table 4.

Table 4 HardwareIDType encoding

HardwareIDType field value	Meaning
0x300000	48-bit extended unique identifier (EUI-48)
0x400000	64-bit extended unique identifier (EUI-64)
0x100000	32-bit Electronic Serial Number (ESN)
0x00NNNN	Hardware ID "NNNN" from [3]
0xFFFFFF	Null
All other values	Invalid

HardwareIDLength — If HardwareIDType is not set to 0xFFFFFF, the access terminal shall set this field to the length in octets of the HardwareIDValue field; otherwise the access terminal shall set this field to 0x00.

HardwareIDValue — The access terminal shall set this field to the unique ID (specified by HardwareIDType) that has been assigned to the terminal by the manufacturer.

Channels	RTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

2.3.8 Interface to other protocols

2.3.8.1 Commands

This protocol does not issue any commands.

2.3.8.2 Indications

This protocol registers to receive the following indications:

- *ActiveSetManagement.IdleHO*
- *ActiveSetManagement.ActiveSetUpdated*
- *InitializationState.NetworkAcquired*
- *OverheadMessages.Updated*
- *ConnectedState.ConnectionClosed*

2.3.9 Configuration attributes

No configuration attributes are defined for this protocol.

2.3.10 Protocol numeric constants

Constant	Meaning	Value
$N_{ADMPType}$	Type field for this protocol	Table 9
$N_{ADMPDefault}$	Subtype field for this protocol	0x0000
$T_{ADMPATResponse}$	Time to receive UATIAssignment after sending UATIUpdateRequest	120 seconds
$T_{ADMPAddress}$	The duration of time that the access terminal declares an address match if it receives a message that is addressed using either the old or the new UATI	180 seconds

2.3.11 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol and the following parameters.

2.3.11.1 UATI parameter

Table 5 Format of the parameter record for the UATI parameter

Field	Length (bits)
ParameterType	8
Length	8
MessageSequence	8
UATISubnetLength	8
UATI	128

ParameterType — This field shall be set to 0x01 for this parameter record.

Length — This field shall be set to the length of this parameter record in units of octets excluding the Length field.

MessageSequence — This field shall be set to the MessageSequence field of the last UATIAssignment message that was sent by the source access network.

UATISubnetLength — This field shall be set to the number of consecutive 1's in the subnet mask of the subnet to which the assigned UATI belongs.

UATI — This field shall be set to the UATI that it is assigned to the access terminal.

2.4 Default Session Configuration Protocol

2.4.1 Overview

The Default Session Configuration Protocol provides for the negotiation and configuration of the set of *SessionConfigurationToken*'s used during a session.

This protocol uses the Signaling Transport to transmit and receive messages.

The *SessionConfigurationToken* is a 16-bit value that defines a complete set of protocol and transport instances that can be used to communicate between the access terminal and the access network. A protocol instance consists of a protocol subtype, dynamic public data and attribute values. A transport instance consists of a transport subtype, dynamic public data and attribute values. A transport instance is bound to a Transport in the Packet Consolidation Protocol. A listing of *SessionConfigurationToken*'s, including the subtype, dynamic public data and attribute values for each protocol and transport instance defined by the *SessionConfigurationToken* can be found in 11.5.

A *SessionConfigurationToken* is InUse if the set of protocol and transport instances specified by the *SessionConfigurationToken* are currently being used to communicate between the access terminal and the access network. Otherwise, a *SessionConfigurationToken* is Suspended. Only one *SessionConfigurationToken* shall be InUse at a time.

A protocol or transport instance is InUse if it is currently being used to communicate between the access terminal and the access network. Otherwise, a protocol or transport instance is Suspended. Only one protocol instance of a protocol type shall be InUse at a time[9]. Only one transport instance corresponding to a Transport in the Packet Consolidation Protocol shall be InUse at a time. A protocol or transport instance shall correspond to exactly one *SessionConfigurationToken*.

The Session Configuration Protocol executes its Save and Commit procedures to swap the InUse protocol and transport instances associated with the current InUse *SessionConfigurationToken* with the Suspended protocol and transport instances associated with a Suspended *SessionConfigurationToken*. A state diagram for each *SessionConfigurationToken* is shown in Figure 12.

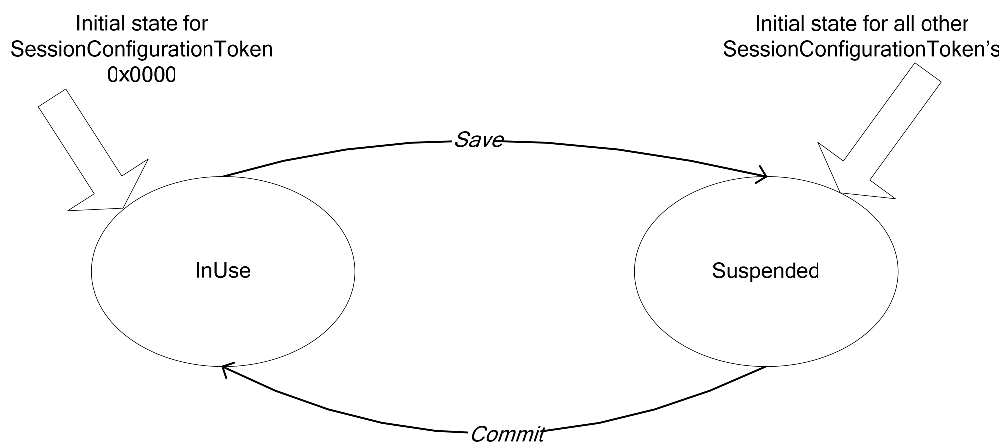

Figure 12 SessionConfigurationToken state diagram

The access network and the access terminal shall use the Generic Attribute Update Protocol in 10.9 to negotiate the configurable attributes of the protocol and transport instances of the InUse *SessionConfigurationToken*. The access network and the access terminal shall not use the Generic Attribute Update Protocol in 10.9 to negotiate the configurable attributes of the protocol and transport instances of a Suspended *SessionConfigurationToken*.

The access network and the access terminal shall use the ConfigurationRequest, ConfigurationAccept and ConfigurationReject messages to negotiate the configurable attributes of the protocol and transport instances of a Suspended *SessionConfigurationToken*. The access network and the access terminal shall not use the ConfigurationRequest, ConfigurationAccept and ConfigurationReject messages to negotiate the configurable attributes of the protocol and transport instances of the InUse *SessionConfigurationToken*.

[9]The Session Configuration Protocol shall have two protocol instances that are temporarily InUse at the same time when a Suspended *SessionConfigurationToken* is swapped with the InUse *SessionConfigurationToken* while the connection is in the Closed state.

A ConfigurationRequest message to update an attribute for a Suspended *SessionConfigurationToken* is defined as legal if the attribute may be updated by an AttributeUpdateRequest message if the Suspended *SessionConfigurationToken* were to become the InUse *SessionConfigurationToken*. Otherwise a ConfigurationRequest message is illegal.

The access network and access terminal shall not send an illegal ConfigurationRequest message.

If the access terminal receives an illegal ConfigurationRequest message, it shall ignore the message. If the access network receives an illegal ConfigurationRequest message, it shall either ignore the message or respond with a ConfigurationReject message.

This protocol operates in one of two states:

- Inactive State: In this state, the protocol waits for an *Activate* command. There are no communications between the access terminal and the access network.
- Active State: In this state the access network may query the access terminal as to which *SessionConfigurationToken*'s are supported and may change the InUse *SessionConfigurationToken*.

The protocol states and the messages and events causing the transition between the states are shown in Figure 13.

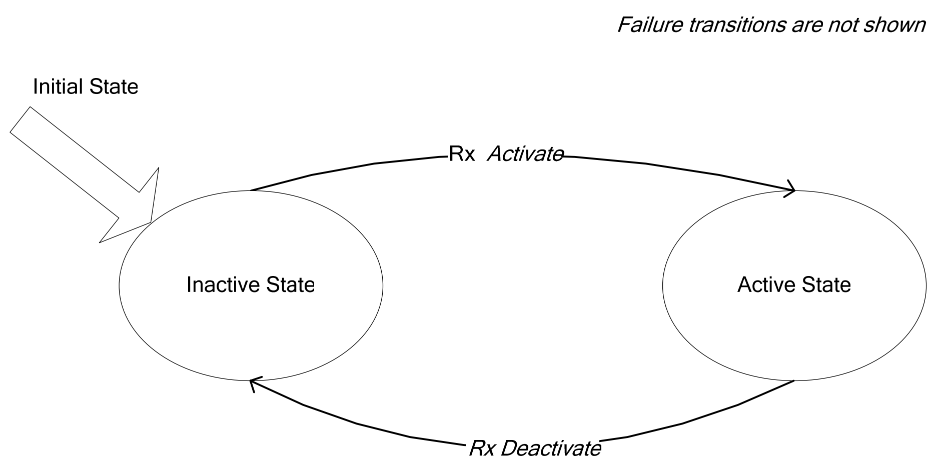

Figure 13 Session Configuration Protocol state diagram

2.4.2 Primitives

2.4.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

2.4.2.2 Return indications

This protocol returns the following indications:

- *Reconfigured*
- *Failed*

2.4.3 Public data

2.4.3.1 Static public data

- *SessionConfigurationToken*
- *ConfigurationLock*

2.4.3.2 Dynamic public data

- Subtype for this protocol

2.4.4 Protocol initialization and swap procedures

2.4.4.1 Protocol initialization

Upon creation, the protocol in the access terminal and access network shall perform the following:

- The protocol at the access terminal and the access network shall enter the Inactive State.
- The access network and the access terminal shall set the ConfigurationLock to UnLocked.
- The access terminal and the access network shall set the *SessionConfigurationToken* to 0x0000.

2.4.4.2 Protocol swap

Upon swap, the protocol in the access terminal and the access network shall perform the following:

- The protocol at the access network and the access terminal shall execute its Commit procedure for the *SessionConfigurationToken* in the static public data.
- The protocol at the access terminal and the access network shall enter the Active State.

2.4.5 Procedures

The access terminal and the access network shall maintain a parameter called ConfigurationLock.

2.4.5.1 Processing the Activate command

If the protocol receives the *Activate* command in the Inactive State, the protocol shall transition to the Active State.

If the protocol receives the *Activate* command in the Active State, the command shall be ignored.

2.4.5.2 Processing the Deactivate command

If the protocol receives the *Deactivate* command in the Inactive State, the command shall be ignored.

If the protocol receives this command in the Active State, it shall transition to the Inactive State.

2.4.5.3 Commit procedure

The access terminal and the access network shall perform the procedures specified in this section when directed to execute its Commit procedure.

The Session Configuration Protocol shall direct all of the protocol instances specified by the *SessionConfigurationToken*, except for the Session Configuration Protocol instance, to perform the following in the order specified:

- Restore the dynamic public data and attributes of the protocol instance.
- The Suspended protocol instance shall become the InUse instance for this protocol type.
- The protocol instance shall perform its Swap Procedure.

The Session Configuration Protocol shall direct all of the transport instances specified by the *SessionConfigurationToken* to perform the following in the order specified:

- Restore the dynamic public data and attributes of the transport instance.
- The Suspended transport instance shall become the InUse transport instance.
- The transport instance shall perform its Swap procedure.

2.4.5.4 Save Procedure

The access terminal and the access network shall perform the procedures specified in this section when directed to execute its Save procedure.

The Session Configuration Protocol shall direct all of the protocol instances specified by the *SessionConfigurationToken*, except for the Session Configuration Protocol instance, to perform the following in the order specified:

- Store the dynamic public data and attributes of the protocol instance.
- The InUse protocol instance shall become Suspended.

The Session Configuration Protocol shall direct all of the transport instances specified by the *SessionConfigurationToken* to perform the following in the order specified:

- Store the dynamic public data and attributes of the transport instance.
- The InUse transport instance shall become Suspended.

2.4.5.5 TokenAssignment message validation

When the access network first sends a TokenAssignment message, it shall set the MessageSequence field of the message to zero. Subsequently, the access network shall increment this field modulo 256 each time it sends a TokenAssignment message.

The access terminal shall initialize a receive pointer for the TokenAssignment message validation, *V(R)*, to 255. When the access terminal receives a TokenAssignment message, it shall validate the message, using the procedure defined in 10.7 (S is equal to 8). The access terminal shall discard the message if it is invalid.

2.4.5.6 Inactive state

Upon entering this state, the protocol shall set the *SessionConfigurationToken* to 0x0000.

If the protocol receives the *Activate* command in the Inactive State, the protocol shall transition to the Active State

2.4.5.7 Active state

2.4.5.7.1 Access terminal requirements

While in this state, the access terminal may send a TokenUpdateRequest message to request the access network to change the value of the *SessionConfigurationToken* if the value of the ConfigurationLock parameter is UnLocked. The access terminal shall not send a TokenUpdateRequest or ConfigurationRequest message if the value of the ConfigurationLock parameter is Locked.

If the access terminal receives a TokenAssignment message requesting to update the value of the *SessionConfigurationToken* in this state, the access terminal shall validate the message sequence number as defined in 10.7. If the message is valid, the access terminal shall perform the following in the order specified:

- If the *SessionConfigurationToken* specified by the TokenAssignment message is the same as the InUse *SessionConfigurationToken*, the Session Configuration Protocol shall send a TokenComplete message.
- Otherwise if the *SessionConfigurationToken* does not specify a Suspended *SessionConfigurationToken*, the access terminal shall return a *Failed* indication and transition to the Inactive state.
- Otherwise, if the *SessionConfigurationToken* specified by the TokenAssignment message is different from the InUse *SessionConfigurationToken* the access terminal shall perform the following:
    - Send a TokenComplete message.
    - If the Air Link Management Protocol is in the Connected State
        - Issue an *AirLinkManagement.CloseConnection* command.
        - Wait to receive a *ConnectedState.ConnectionClosed* indication.
    - Execute the Save procedure for the InUse *SessionConfigurationToken*.
    - Store the dynamic public data and attributes of the InUse Session Configuration Protocol instance.
    - Set the *SessionConfigurationToken* static public data to the value specified in the TokenAssignment message.
    - Restore the dynamic public data and attributes of the Suspended Session Configuration Protocol instance specified by the new *SessionConfigurationToken*.

- Return a *Reconfigured* indication.
- The Session Configuration Protocol instance specified by the new *SessionConfigurationToken* shall execute its Swap procedure and shall become the InUse instance for this protocol type.
- This Session Configuration Protocol instance shall become Suspended.

If the access terminal receives a LockConfiguration message, then the access terminal shall respond with a ConfigurationLockAck message and shall set ConfigurationLock to Locked. If the access terminal receives an UnLockConfiguration message, then the access terminal shall respond with an UnLockConfigurationAck message and shall set ConfigurationLock to UnLocked.

2.4.5.7.2 Access network requirements

While in this state, the access network may send a TokenAssignment message to change the value of the *SessionConfigurationToken* if the value of the ConfigurationLock parameter is UnLocked. The access network shall not send a TokenAssignment or ConfigurationRequest message if the value of the ConfigurationLock parameter is Locked.

Upon receiving a TokenComplete message in response to the TokenAssignment message, the access network shall perform the following:

- If the *SessionConfigurationToken* specified by the TokenAssignment message is different from the InUse *SessionConfigurationToken*, the access network shall perform the following:
  - If the Air Link Management Protocol is in the Connected State
    - Issue an *AirLinkManagement.CloseConnection* command.
    - Wait to receive a *ConnectedState.ConnectionClosed* indication.
  - Execute the Save procedure for the InUse *SessionConfigurationToken*.
  - Store the dynamic public data and attributes of the InUse Session Configuration Protocol instance.
  - Set the *SessionConfigurationToken* static public data to the value specified in the TokenAssignment message.
  - Restore the dynamic public data and attributes of the Suspended Session Configuration Protocol instance specified by the new *SessionConfigurationToken*.
  - Return a *Reconfigured* indication.
  - The Session Configuration Protocol instance specified by the new *SessionConfigurationToken* shall execute its Swap procedure and shall become the InUse instance for this protocol type.
  - This Session Configuration Protocol instance shall become Suspended.

2.4.6 Message formats

2.4.6.1 TokensSupportedRequest

The access network sends the TokensSupportedRequest message to discover the set of *SessionConfigurationToken*'s supported by the access terminal.

Field	Length (bits)
MessageID	8

MessageID The access network shall set this field to 0x00.

Channels	FTC	SLP	Reliable
Addressing	Unicast	**Security**	Required

2.4.6.2 TokensSupportedResponse

The access terminal sends the TokensSupportedResponse message in response to the TokensSupportedRequest message.

Field	Length (bits)
MessageID	8
TokenCount	8
TokenCount occurrences of the following field:	
{	
SessionConfigurationToken	16
}	

MessageID The access terminal shall set this field to 0x01.

TokenCount The access terminal shall set this field to the number of SessionConfigurationToken fields included in this message. The access terminal shall include TokenCount occurrences of the following field with the message.

SessionConfigurationToken
The access terminal shall set this field to a *SessionConfigurationToken* supported by the access terminal.

Channels	RTC	SLP	Reliable
Addressing	Unicast	**Security**	Required

2.4.6.3 TokenUpdateRequest

The access terminal sends the TokenUpdateRequest message to request a new InUse *SessionConfigurationToken* assignment from the access network.

Field	Length (bits)
MessageID	8
MessageSequence	8
TokenCount	8
TokenCount occurrences of the following field:	
{	
SessionConfigurationToken	16
}	

MessageID The access terminal shall set this field to 0x02.

MessageSequence The access terminal shall increment this field modulo 256 for each new TokenUpdateRequest message sent. If this is the first TokenUpdateRequest message sent by the access terminal, the access terminal shall set this field to zero.

TokenCount The access terminal shall set this field to the number of SessionConfigurationToken fields included in this message, where the *SessionConfigurationToken* values are in descending order of preference. The access terminal shall include TokenCount occurrences of the following field with the message.

SessionConfigurationToken
The access terminal shall set this field to a *SessionConfigurationToken* supported by the access terminal.

Channels	RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

2.4.6.4 TokenAssignment

The access network sends the TokenAssignment message to change the InUse *SessionConfigurationToken*.

Field	Length (bits)
MessageID	8
MessageSequence	8
SessionConfigurationToken	16

MessageID The access network shall set this field to 0x03.

MessageSequence The access network shall increment this field modulo 256 for each new TokenAssignment message sent to this access terminal. If this is the first TokenAssignment message sent to this access terminal, the access network shall set this field to zero.

SessionConfigurationToken
The access network shall set this field to the *SessionConfigurationToken* that it is assigning to the access terminal.

Channels	FTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

2.4.6.5 TokenComplete

The access terminal sends the TokenComplete message to notify the access network that it has received the TokenAssignment message.

Field	Length (bits)
MessageID	8
MessageSequence	8

MessageID The access terminal shall set this field to 0x04.

MessageSequence The access terminal shall set this field to the MessageSequence field of the TokenAssignment message whose receipt this message is acknowledging.

Channels	RTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

2.4.6.6 LockConfiguration

The access network sends the LockConfiguration message to set the ConfigurationLock parameter in the access terminal to Locked.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID The access network shall set this field to 0x05.

TransactionID The access network shall set this field according to 10.8.

Channels	FTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

2.4.6.7 LockConfigurationAck

The access terminal sends the LockConfigurationAck message to acknowledge the receipt of a LockConfiguration message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID The access terminal shall set this field to 0x06.

TransactionID The access terminal shall set this field to the TransactionID field of the LockConfiguration message that is being acknowledged.

Channels	RTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

2.4.6.8 UnLockConfiguration

The access network sends the UnLockConfiguration message to set the ConfigurationLock parameter in the access terminal to UnLocked.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID The access network shall set this field to 0x07.

TransactionID The access network shall set this field according to 10.8.

Channels	FTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

2.4.6.9 UnLockConfigurationAck

The access terminal sends the UnLockConfigurationAck message to acknowledge the receipt of an UnLockConfiguration message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The access terminal shall set this field to 0x08.

TransactionID — The access terminal shall set this field to the TransactionID field of the UnLockConfiguration message that is being acknowledged.

Channels	RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

2.4.6.10 ConfigurationRequest

The sender sends an ConfigurationRequest message to offer a set of attribute-values for a given attribute of a protocol or transport instance for a Suspended *SessionConfigurationToken*.

Field	Length (bits)
MessageID	8
TransactionID	8
SessionConfigurationToken	16
ProtocolType	8
One or more instances of the following field: {	
AttributeRecord	Attribute dependent
}	

MessageID — The sender shall set this field to 0x09.

TransactionID — The sender shall set this field according to 10.8.

ProtocolType — The sender shall set this field to the Type value in Table 9 for the protocol or transport associated with the attributes being negotiated.

AttributeRecord — The format of this record is specified in 10.3.

Channels	FTC RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

2.4.6.11 ConfigurationAccept

The sender sends a ConfigurationAccept message in response to a ConfigurationRequest message to accept the offered attribute values.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID The sender shall set this field to 0x0a.

TransactionID The sender shall set this field to the TransactionID field of the ConfigurationRequest message that is being accepted.

Channels	FTC RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

2.4.6.12 ConfigurationReject

The access network sends a ConfigurationReject message in response to a ConfigurationRequest message to reject the offered attribute values.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID The sender shall set this field to 0x0b.

TransactionID The sender shall set this field to the TransactionID field of the ConfigurationRequest message that is being rejected.

Channels	FTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

2.4.7 Interface to other protocols

2.4.7.1 Commands

This protocol issues the following command:

- *AirLinkManagement.CloseConnection*

2.4.7.2 Indications

This protocol registers to receive the following indication:

- *ConnectedState.ConnectionClosed*

2.4.8 Configuration attributes

No configuration attributes are defined for this protocol.

2.4.9 Protocol numeric constants

Constant	Meaning	Value
$N_{SCPType}$	Type field for this protocol	Table 9
$N_{SCPDefault}$	Subtype field for this protocol	0x0000

2.4.10 Session State information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol and the following parameters.

2.4.10.1 ConfigurationLock parameter

Table 6 Format of the parameter record for the ConfigurationLock parameter

Field	Length (bits)
ParameterType	8
Length	8
ConfigurationLock	8

ParameterType — This field shall be set to 0x01 for this parameter record.

Length — This field shall be set to the length of this parameter record in units of octets excluding the Length field.

ConfigurationLock — This field shall be set to 0x00 if the value of the ConfigurationLock is UnLocked and it shall be set to 0x01 if the value of the ConfigurationLock is set to Locked.

2.4.10.2 SessionConfigurationToken parameter

Table 7 Format of the parameter record for the SessionConfigurationToken parameter

Field	Length (bits)
ParameterType	8
Length	8
SessionConfigurationToken	16
MessageSequence	8

ParameterType This field shall be set to 0x02 for this parameter record.

Length This field shall be set to the length of this parameter record in units of octets excluding the Length field.

SessionConfigurationToken
This field shall be set to the value of the InUse *SessionConfigurationToken* assigned to the access terminal.

MessageSequence This field shall be set to the MessageSequence field of the last TokenAssignment message that was sent by the access network.

2.5 Default Capabilities Discovery Protocol

2.5.1 Overview

The Default Capabilities Discovery Protocol allows the access network to discover the capabilities of the access terminal.

This protocol uses the Signaling Transport to transmit and receive messages.

This protocol operates in one of two states:

- Inactive State: In this state, the protocol waits for an *Activate* command. There are no communications between the access terminal and the access network.
- Active State: In this state the access terminal and the access network perform a CapabilitiesRequest/CapabilitesResponse exchange.

The protocol states and the messages and events causing the transition between the states are shown in Figure 14.

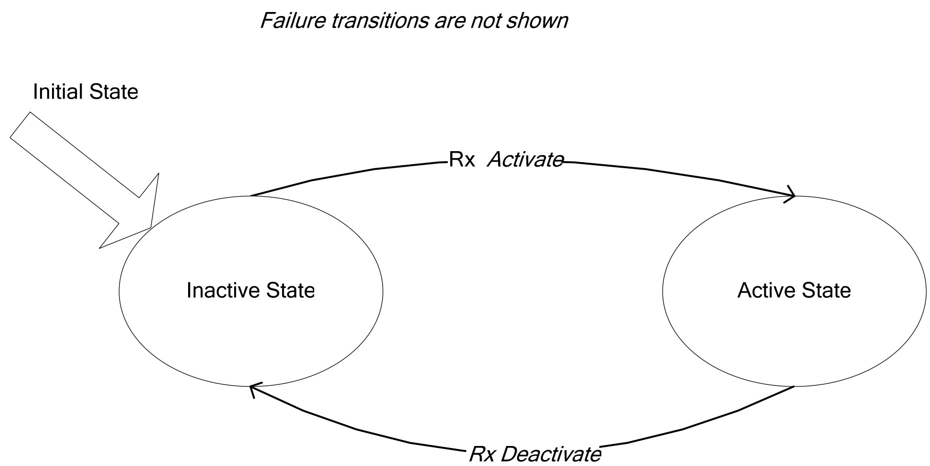

Figure 14 Session Capabilities Discovery Protocol state diagram

2.5.2 Primitives

2.5.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

2.5.2.2 Return indications

This protocol does not return any indications.

2.5.3 Public data

2.5.3.1 Static public data

This protocol does not define any static public data.

2.5.3.2 Dynamic public data

- Subtype for this protocol
- All the attributes listed in 2.5.8.

2.5.4 Protocol initialization and swap procedures

2.5.4.1 Protocol initialization

Upon creation, the protocol in the access terminal and access network shall perform the following:

- The value of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol at the access terminal and the access network shall enter the Inactive State

2.5.4.2 Protocol swap

Upon swap, the protocol in the access terminal and the access network shall enter the Active State.

2.5.5 Procedures

2.5.5.1 Processing the Activate command

If the protocol receives the *Activate* command in the Inactive State, the protocol shall transition to the Active State.

If the protocol receives the *Activate* command in the Active State, the command shall be ignored.

2.5.5.2 Processing the Deactivate command

If the protocol receives the *Deactivate* command in the Inactive State, the command shall be ignored.

If the protocol receives the *Deactivate* command in the Active State, the protocol shall transition to the Inactive State

2.5.5.3 Inactive state

In this state, there are no communications between the access terminal and the access network.

In this state the protocol waits for the *Activate* command. See 2.5.5.1 for processing of the *Activate* command.

2.5.5.4 Active state

In this state the access terminal and the access network perform a CapabilitiesRequest/CapabilitiesResponse exchange.

2.5.6 Message formats

2.5.6.1 CapabilitiesRequest

The access network sends the CapabilitiesRequest message to discover the capabilities of the access terminal.

Field	Length (bits)
MessageID	8
TransactionID	8
AttributeCount	8
AttributeCount occurrences of the following field:	
{	
AttributeID	16
}	

MessageID The access network shall set this field to 0x00.

TransactionID The access network shall set this field according to 10.8.

AttributeCount The access network shall set this field to the number of AttributeID fields included in this message. The sender shall set this field to 0x00 to request the value of all attributes defined in 2.5.8. The access network shall include AttributeCount occurrences of the following field with the message.

AttributeID The access network shall set this field to the AttributeID for which this request is generated.

Channels	FTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

2.5.6.2 CapabilitiesResponse

The access terminal sends the CapabilitiesResponse message in response to the CapabilitiesRequest message.

Field	Length (bits)
MessageID	8
TransactionID	8
AttributeCount	8

AttributeCount occurrences of the following field:

{

Field	Length (bits)
AttributeRecord	Attribute dependent

}

MessageID — The access terminal shall set this field to 0x01.

TransactionID — The access terminal shall set this value to the TransactionID field of the corresponding CapabilitiesResponse message.

AttributeCount — The access terminal shall set this field to the number of AttributeRecord fields included in this message. The access terminal shall include AttributeCount occurrences of the following field with the message.

AttributeRecord — An attribute record containing a single attribute value. The format of the AttributeRecord is given in 10.3. The access terminal shall not include more than one attribute record with the same attribute identifier.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

2.5.7 Interface to other protocols

2.5.7.1 Commands

This protocol does not issue any commands.

2.5.7.2 Indications

This protocol does not register to receive any indications.

2.5.8 Configuration attributes

The access terminal and the access network shall not use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Capabilities Discovery Protocol.

2.5.8.1 Simple attributes

The negotiable simple attributes for this protocol are listed in Table 8. The access terminal shall use as defaults the values in Table 8 that are listed in ***bold italics***.

Table 8 Configurable attributes

Attribute ID	Attribute	Values	Meaning
0x0000	NumRxAntennas	***0x0001***	1 receive antennas supported at the access terminal
		0x0001 to 0x0004	Number of receive antennas supported at the access terminal
		All other values	Reserved
0x0001	MaxPacketFormatFwd	***0x0001***	1 is the maximum Packet Format that can be supported by the access terminal on the forward link.
		0x0000 to 0x000f	Number of the maximum Packet Format that can be supported by the access terminal on the forward link.
		All other values	Reserved
0x0002	MaxPacketFormatRev	***0x0001***	1 is the maximum Packet Format that can be supported by the access terminal on the reverse link.
		0x0000 to 0x0009	Number of the maximum Packet Format that can be supported by the access terminal on the reverse link.
		All other values	Reserved
0x0003	MaxMIMOAssignmentFwd	***0x0000***	The access terminal does not support MIMO mode.
		0x0008 to 0x0800	The maximum number of sub-carriers that can be assigned to the access terminal on the forward link in units of carriers, when in MIMO mode.
		All other values	Reserved

Attribute ID	Attribute	Values	Meaning
0x0004	NumCarriers	***0x0001***	1 is the maximum number of carriers supported by the Access Terminal in multi-carrier mode.
		0x0001-0x0004	Maximum number of carriers supported by the access terminal in multi-carrier mode.
		All other values	Reserved
0x0005	MaxInterlaceAssignmentFwd	***0x0001***	1 is the maximum number of interlaces on which the access terminal can simultaneously receive MAC packets.
		0x0001 to 0x0006 (FDD) 0x0001 to 0x000c (TDD)	The maximum number of interlaces on which the access terminal can simultaneously receive MAC packets.
		All other values	Reserved
0x0006	MaxInterlaceAssignmentRev	***0x0001***	The maximum number of interlaces on which the access terminal can simultaneously transmit MAC packets.
		0x0001 to 0x0006 (FDD) 0x0001 to 0x000c (TDD)	The maximum number of interlaces on which the access terminal can simultaneously transmit MAC packets.
		All other values	Reserved
0x0007	MaxPacketSizeFwd	***0x0001***	Maximum MAC packet size of 1 kbits can be received by the access terminal per interlace on the forward link.
		0x0001 to 0x0190	The maximum packet size that can be received by the access terminal per interlace on the forward link in units of kbits.
		All other values	Reserved

Attribute ID	Attribute	Values	Meaning
0x0008	MaxPacketSizeRev	***0x0001***	A maximum MAC packet size of 1 kbits that can be transmitted by the access terminal per interlace on the reverse link.
		0x0001 to 0x0064	The maximum packet size that can be transmitted by the access terminal per interlace on the reverse link in units of kbits.
		All other values	Reserved
0x0009	SCWLayersSupported	***0x0000***	The access terminal does not support SCW transmission.
		0x0001 to NumRx Antennas	The maximum number of layers that the access terminal can support in MIMO SCW transmission.
		All other values	Reserved
0x000a	MCWLayersSupported	***0x0000***	The access terminal does not support MCW transmission.
		0x0001 to NumRx Antennas	The maximum number of layers that the access terminal can support in MIMO MCW transmission
		All other values	Reserved
0x000b	STTDSupport	***0x0000***	The access terminal does not support STTD transmission.
		0x0001	The access terminal supports STTD transmission.
		All other values	Reserved
0x000c	HalfDuplexSupportRequired	***0x0000***	The access terminal does not require half duplex support from the access network.
		0x0001	The access terminal requires half duplex support from the access network.
		All other values	Reserved
0x000d	MaxPHYSubPacketSize	***0x0000***	The access terminal supports a maximum packet size of 4096 at the Physical Layer.
		0x0001	The access terminal supports a maximum packet size of 8192 at the Physical Layer.
		All other values	Reserved

2.5.8.2 Complex attributes

The following complex attributes and default values are defined (see 10.3 for attribute record definition).

2.5.8.2.1 Support*PP*RAT attribute

PP is the two-digit hexadecimal RAT type except for 0x00 according to 11.4, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
RATSupported	8	0x00
SupportedRATParametersLength	8	0x00
SupportedRATParameters	SupportedRATParametersLength × 8	N/A

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x01*PP*.

RATSupported — The sender shall set this field to 0x00 if the RAT *PP* is not supported. Otherwise, the sender shall set this field to 0x01 if the RAT *PP* is supported. All other values are reserved.

SupportedRATParametersLength — The sender shall set this field to the length of the SupportedRATParameters record in units of octets. If the RATSupported field is set to 0x00, the sender shall set this field to 0x00. If the RATSupported field is set to 0x01, the sender shall set this field to 0x00 for RAT type 0x01 to 0x06.

SupportedRATParameters — If SupportedRATParametersLength is 0x00, the sender shall omit this record.

2.5.8.2.2 SupportedChannelBands attribute

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
ChannelBandCount	8	0x00

ChannelBandCount occurrences of the following field:

{

LowerChannelBandRecord	ChannelBandRecordType Dependent	N/A
UpperChannelBandRecord	ChannelBandRecordType Dependent	N/A

}

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x0200.

ChannelBandCount — The sender shall set this field to the number of ChannelBandRecords associated with this attribute. The sender shall include ChannelBandCount occurrences of the following two fields with the attribute.

LowerChannelBandRecord
The sender shall set this field to the lower channel band record definition according to 10.1 for the range of channel bands supported.

UpperChannelBandRecord
The sender shall set this field to the upper channel band record definition according to 10.1 for the range of channel bands supported.

2.5.9 Protocol numeric constants

Constant	Meaning	Value
$N_{CDPType}$	Type field for this protocol	Table 9
$N_{CDPDefault}$	Subtype field for this protocol	0x0000

2.5.10 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

2.6 Default Inter Radio Access Technology (RAT) Protocol

2.6.1 Overview

The Default Inter RAT Protocol allows the access network and access terminal to send messages for other radio access technologies.

This protocol uses the Signaling Transport to transmit and receive messages.

This protocol operates in one of two states:

- Inactive State: In this state, the protocol waits for an *Activate* command. There are no communications between the access terminal and the access network.
- Active State: In this state the access terminal or the access network may send a InterRATBlob message.

The protocol states and the messages and events causing the transition between the states are shown in Figure 15.

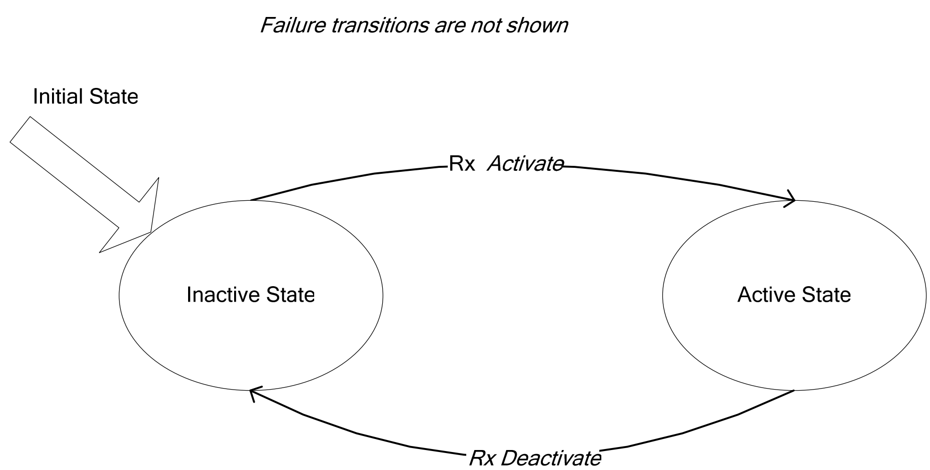

Figure 15 Inter RAT Protocol state diagram

2.6.2 Primitives

2.6.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

2.6.2.2 Return indications

This protocol does not return any indications.

2.6.3 Public data

2.6.3.1 Static public data

This protocol does not define any static public data.

2.6.3.2 Dynamic public data

- Subtype for this protocol

2.6.4 Protocol initialization and swap procedures

2.6.4.1 Protocol initialization

Upon creation, the protocol in the access terminal and access network shall perform the following:

- The protocol at the access terminal and the access network shall enter the Inactive State

2.6.4.2 Protocol swap

Upon swap, the protocol in the access terminal and the access network shall enter the Active State.

2.6.5 Procedures

2.6.5.1 Processing the Activate command

If the protocol receives the *Activate* command in the Inactive State, the protocol shall transition to the Active State.

If the protocol receives the *Activate* command in the Active State, the command shall be ignored.

2.6.5.2 Processing the Deactivate command

If the protocol receives the *Deactivate* command in the Inactive State, the command shall be ignored.

If the protocol receives the *Deactivate* command in the Active State, the protocol shall transition to the Inactive State

2.6.5.3 Inactive state

In this state, there are no communications between the access terminal and the access network.

In this state the protocol waits for the *Activate* command. See 2.6.5.1 for processing of the *Activate* command.

2.6.5.4 Active state

In this state the access terminal or the access network may send an InterRATBlob message.

2.6.6 Message formats

2.6.6.1 InterRATBlob

The access network or access terminal sends this message when it has another radio access technology's message to send.

Field	Length (bits)
MessageID	8
TechnologyType	8
TechnologyBlobLength	8
TechnologyBlob	8 × TechnologyBlobLength

MessageID The sender shall set this field to 0x00.

TechnologyType The sender shall include this field to indicate the type of technology as specified in 11.4.

TechnologyBlobLength The sender shall set this field to the length, in octets, of the TechnologyBlob.

TechnologyBlob The sender shall set this field to the message for the other technology. The interpretation of this field is beyond the scope of this specification.

Channels	RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

2.6.6.2 InterRATIDRequest

The access network uses this message to query the access terminal of its RAT ID information.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID The access network shall set this field to 0x01.

TransactionID The access network shall set this field according to 10.8.

Channels	FTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

2.6.6.3 InterRATIDResponse

The access terminal sends this message in response to the InterRATIDRequest message.

Field	Length (bits)
MessageID	8
TransactionID	8
TechnologyTypeCount	8

TechnologyTypeCount occurrences of the following record:

{

Field	Length (bits)
TechnologyIDType	8
TechnologyIDLength	8
TechnologyIDValue	8 × TechnologyIDLength

}

MessageID — The access terminal shall set this field to 0x02.

TransactionID — The access terminal shall set this field the TransactionID field of the corresponding InterRATIDRequest message.

TechnologyTypeCount — The access terminal shall set this field to the number of TechnologyIDType fields in this message. The access terminal shall include TechnologyTypeCount occurrences of the following three fields with the message.

TechnologyIDType — The access terminal shall include this field to indicate the type of technology as specified in 11.4. The access terminal shall not set this field to 0x00 or 0x01.

TechnologyIDLength — The access terminal shall set this field to the length, in octets, of the TechnologyIDValue.

TechnologyIDValue — The access terminal shall set this field to the unique ID (specified by TechnologyType) that has been assigned to the terminal by the radio access technology as specified in 11.4.

Channels	RTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

2.6.7 Interface to other protocols

2.6.7.1 Commands

This protocol does not issue any commands.

2.6.7.2 Indications

This protocol does not register to receive any indications.

2.6.8 Configuration attributes

This protocol does not have any configurable attributes.

2.6.9 Protocol numeric constants

Constant	Meaning	Value
$N_{IRATPType}$	Type field for this protocol	Table 9
$N_{IRATPDefault}$	Subtype field for this protocol	0x0000

2.6.10 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

3 Convergence Sublayer

3.1 Introduction

3.1.1 General Overview

The Convergence Sublayer contains protocols and transports used to transport messages and data between the access terminal and the access network.

The Convergence Sublayer contains the following protocols and transports:

- Signaling Transport: Provides the means to carry messages between a protocol/transport in one entity and the same protocol/transport in the other entity. The Default Signaling Transport consists of a messaging protocol (Signaling Network Protocol) and a link layer protocol that provides message fragmentation, retransmission, and duplicate detection (Signaling Link Protocol).
- Data Transport: Provides the means to carry upper layer data. The Default Data Transport consists of a link layer protocol that provides fragmentation, retransmission, and duplicate detection (Radio Link Protocol); a Route Selection Protocol that provides two route instances for a higher layer packet flow; and a Flow Control Protocol that provides flow control of data traffic.
- Packet Consolidation Protocol: Provides multiplexing of distinct transports, transmit prioritization and packet encapsulation. The Default Packet Consolidation Protocol provides 8 Transports. Each Transport defined by the Default Packet Consolidation Protocol maps to a data-bearing transport such as the Signaling or Data Transport. The first Transport (Transport 0) always carries Signaling, and the other Transports can be used to carry, for example, the Default Packet Transport to support different Quality of Service (QoS) requirements for data or other transports.

The relationship between the Convergence Sublayer protocols is illustrated in Figure 16.

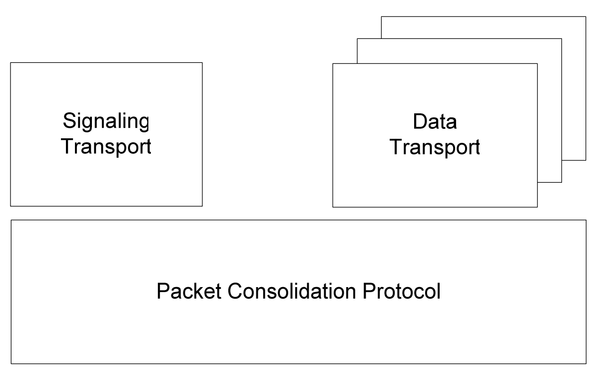

Figure 16 Convergence Sublayer protocols

3.2 Default Signaling Transport

3.2.1 Introduction

3.2.1.1 General overview

The Default Signaling Transport is used to transport messages that manage air interface protocol objects in the access network and access terminal. The Default Signaling Transport encompasses the Signaling Network Protocol (SNP) and the Signaling Link Protocol (SLP). Protocols and transports use SNP to exchange messages.

SNP provides a one octet header that defines the Type of the protocol or transport with which the message is associated. The SNP uses the header to route the message to the appropriate protocol or transport instance.

SLP provides message fragmentation, reliable and best-effort message delivery and duplicate detection for messages that are delivered reliably.

The relationship between SNP and SLP is illustrated in Figure 17.

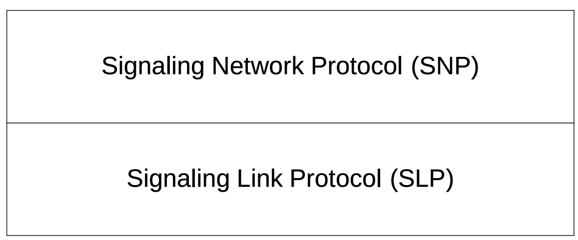

Figure 17 Default signaling transport protocols

3.2.1.2 Public data

3.2.1.2.1 Static public data

This transport does not define any static public data.

3.2.1.2.2 Dynamic public data

- Subtype for this transport

3.2.1.3 Data encapsulation

Figure 18 and Figure 19 illustrate the relationship between a message, SNP packets, SLP packets, and Packet Consolidation Protocol payloads. Figure 18 shows a case where SLP does not fragment the SNP packet. Figure 19 shows a case where the SNP packet is fragmented into more than one SLP payload.

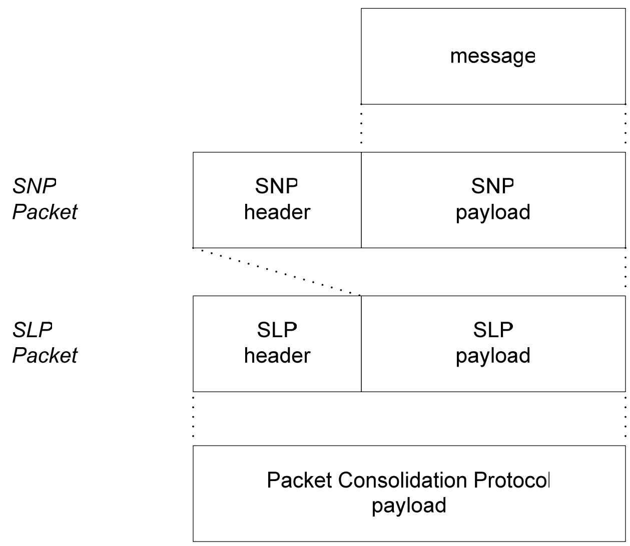

Figure 18 Message encapsulation (non-fragmented)

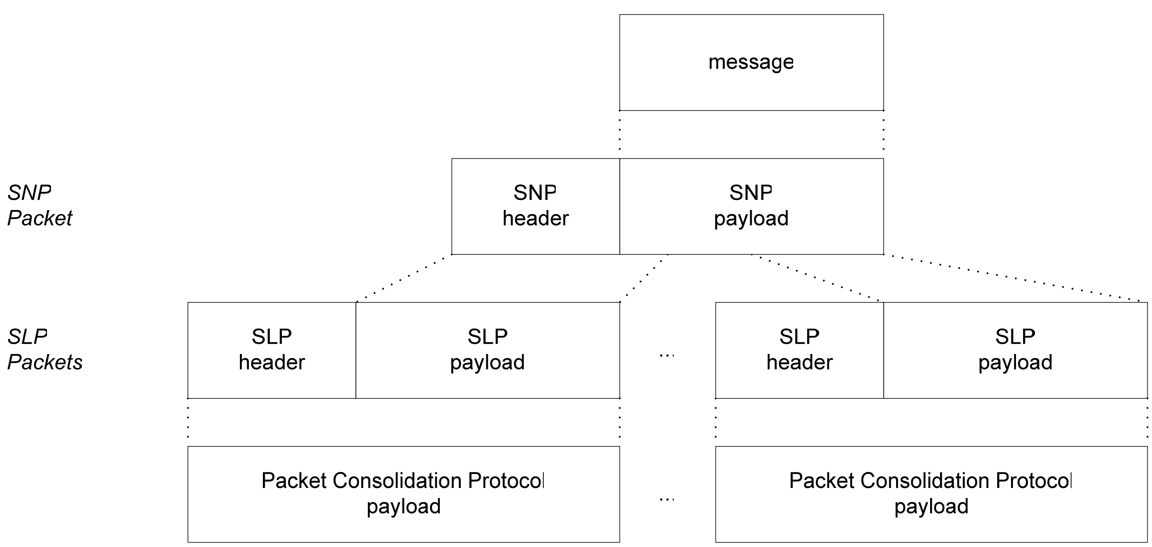

Figure 19 Message encapsulation (fragmented)

3.2.2 Transport initialization and swap procedures

3.2.2.1 Transport initialization

Upon creation, the instance of the Signaling Transport in the access terminal and access network shall set the value of the attributes for this transport to the default values specified for each attribute.

3.2.2.2 Transport swap

This transport defines an empty swap procedure.

3.2.3 General signaling requirements

3.2.3.1 General requirements

The following requirements are common to all protocols and transports that carry messages using SNP and that provide for message extensibility. The access terminal and the access network shall abide by the following rules when generating and processing any signaling message carried by SNP:

- Messages shall be an integer number of octets in length.
- The fields of the message shall be generated in the order specified by the message format definition. Within each field, the most significant bit of the field shall be generated and processed first.
- Message identifiers shall be unambiguous for each protocol Type and for each protocol and transport Subtype for all protocols and transports compatible with the Air Interface, defined by MinimumRevision and above.
- For future revisions, the transmitter shall add new fields only at the end of a message. The transmitter shall not add fields if their addition makes the parsing of previous fields ambiguous for receivers whose protocol revision is equal to or greater than MinimumRevision.
- The receiver shall discard all unrecognized messages.
- The receiver shall discard all unrecognized fields.
- The receiver shall discard a message if any of the fields in the message is set to a value outside of the defined field range, unless the receiver is specifically directed to ignore this field. A field value is outside of the allowed range if a range was specified with the field and the value is not in this range, or the field is set to a value that is defined as invalid. The receiver shall discard a field in a message if the field is set to a reserved value.

3.2.3.2 Message information

Each message definition contains information regarding channels on which the message can be transmitted, whether the message requires SLP reliable or best-effort delivery, and the addressing modes applicable to the message. This information is provided in the form of a table, an example of which is given in Figure 20.

Channels	FTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

Figure 20 Sample message information

The following values are defined:

- Channels: This information field indicates the MAC protocols in the data path on which this message can be transmitted. The sender of the message shall send the message only on the MAC protocol(s) indicated by this information field. Values are:
  - FTC for Forward Traffic Channel MAC,
  - RTC for Reverse Traffic Channel MAC.
- SLP: Signaling Link Protocol requirements. The sender of the message shall send the message only using the SLP in the mode(s) indicated by this information field. Values are:
  - Best Effort: the message is sent once and is subject to erasure, and
  - Reliable: erasures are detected and the message is retransmitted one or more times, if necessary.
- Addressing: Addressing modes for the message. The sender of the message shall send the message only with an address type(s) indicated by this information field. Values are:
  - Broadcast if a broadcast address can be used with this message, and
  - Unicast if a unicast address can be used with this message.
- Security: Security modes for the message. The sender of the message shall send the message only with a security type(s) indicated by this information field. Values are:
  - Required: if SecurityEnabled public data of the Security Protocol is set to ‘1’, then the message shall be sent with IsSecure field of the Lower MAC header set to ‘1’. Any message received when SecurityEnabled public data of the Security Protocol is set to ‘1’ and the IsSecure field of the Lower MAC header is set to ‘0’ shall be discarded, and
  - Optional: the message is always processed.

3.2.4 Signaling Network Protocol

3.2.4.1 Overview

The Signaling Network Protocol (SNP) routes messages to protocols and transports specified by the Type field provided in the SNP header.

The actual protocol indicated by the Type is defined by the InUse SessionConfigurationToken. For example, Type 0x11 is associated with the Session Management Protocol. The specific Session Management Protocol used (and, therefore, the Session Management protocol generating and processing the messages delivered by SNP) is defined by the InUse SessionConfigurationToken.

The Type field forms a single octet header.

The remainder of the message following the SNP header is processed by the protocol specified by the Type.

SNP is a protocol associated with the Default Signaling Transport.

3.2.4.2 Primitives

3.2.4.2.1 Commands

This protocol does not define any commands.

3.2.4.2.2 Return indications

This protocol does not return any indications.

3.2.4.3 Protocol data unit

The protocol data unit for this protocol is an SNP packet. Each SNP packet consists of one message sent by a protocol using SNP.

The protocol constructs an SNP packet by adding the SNP header (see 3.2.4.6) in front of the payload. The structure of the SNP packet is shown in Figure 21.

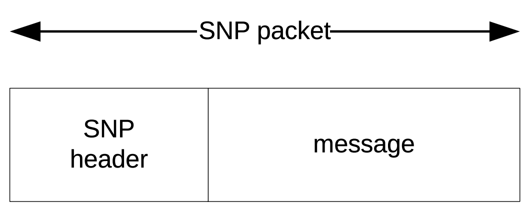

Figure 21 SNP packet structure

3.2.4.4 Procedures

SNP receives messages for transmission from multiple protocols and transports. SNP shall add the SNP header to each message and forward it for transmission to SLP.

SNP receives messages from SLP. SNP shall route these messages to their associated protocols and transports according to the value of the Type field in the SNP header.

If an SNP message is to be transmitted on the Forward Traffic Channel or on the Reverse Traffic Channel, and if a connection is not open, SNP shall issue an *AirLinkManagement.OpenConnection* command. SNP should queue all messages requiring transmission in the Forward Traffic Channel or in the Reverse Traffic Channel until the protocol receives an *IdleState.ConnectionOpened* indication.

3.2.4.5 Type definitions

Type definitions associated with the default protocol stack are presented in Table 9. The constant name and protocol layer are provided for informational purposes.

Table 9 Default protocol stack type values

Type	Protocol	Constant Name
0x00	Physical Layer Protocol	$N_{PHYType}$
0x01	Control Channel MAC Protocol	$N_{CCMPType}$
0x02	Access Channel MAC Protocol	$N_{ACMPType}$
0x03	Forward Traffic Channel MAC Protocol	$N_{FTCMPType}$
0x04	Reverse Traffic Channel MAC Protocol	$N_{RTCMPType}$
0x05	Reverse Control Channel MAC Protocol	$N_{RCCMPType}$
0x06	Shared Signaling MAC Protocol	$N_{SSMPType}$
0x07	Air Link Management Protocol	$N_{ALMPType}$
0x08	Initialization State Protocol	$N_{ISPType}$
0x09	Idle State Protocol	$N_{IDPType}$
0x0a	Connected State Protocol	$N_{CSPType}$
0x0b	Active Set Management Protocol	$N_{ASMPType}$
0x0c	Overhead Messages Protocol	$N_{OMPType}$
0x0d	Authentication Protocol	N_{APType}
0x0e	Encryption Protocol	N_{EPType}
0x0f	Security Protocol	N_{SPType}
0x10	Key Exchange Protocol	$N_{KEPType}$
0x11	Session Management Protocol	$N_{SMPType}$
0x12	Address Management Protocol	$N_{ADMPType}$
0x13	Session Configuration Protocol	$N_{SCPType}$
0x14	Capabilities Discovery Protocol	N_{CDPTyp}
0x15	InterRAT Protocol	$N_{IRATPType}$
0x16	Packet Consolidation Protocol	$N_{PCPType}$
0x17	Transport 0	$N_{TPT0Type}$

Type	Protocol	Constant Name
0x18	Transport 1	$N_{TPT1Type}$
0x19	Transport 2	$N_{TPT2Type}$
0x1a	Transport 3	$N_{TPT3Type}$
0x1b	Transport 4	$N_{TPT4Type}$
0x1c	Transport 5	$N_{TPT5Type}$
0x1d	Transport 6	$N_{TPT6Type}$
0x1e	Transport 7	$N_{TPT7Type}$

3.2.4.6 SNP packet header

The SNP shall place the following header in front of every message that it sends:

Field	Length (bits)
Type	8

Type This field shall be set the Type value in Table 9 for the protocol or Transport associated with the encapsulated message.

3.2.4.7 Message formats

No messages are defined for this protocol.

3.2.4.8 Interface to other protocols

3.2.4.8.1 Commands

This protocol issues the following command:

- *AirLinkManagement.OpenConnection*

3.2.4.8.2 Indications

This protocol registers to receive the following indications:

- *IdleState.ConnectionOpened*

3.2.5 Signaling Link Protocol

3.2.5.1 Overview

The purpose of the Signaling Link Protocol (SLP) is to provide best effort and reliable delivery for SNP packets. SLP provides retransmission and duplicate detection for messages using reliable delivery. SLP provides fragmentation and re-assembly for SNP packets. SLP does not ensure in-order delivery of SNP packets.

The delivery flow variable P takes value “BE” or ‘0’ for best effort delivery, and value “RD” or ‘1’ for reliable delivery. The reliable delivery flow provides two sequence spaces variables Q_{Tx} and Q_{Rx} for transmission and reception of SNP packets respectively. The transmitter toggles the sequence space variable Q_{Tx} between '0' and '1' to indicate a reset. The receiver sequence space variable Q_{Rx} tracks the value of Q_{Tx} and detects when the transmitter has performed a reset. For best effort delivery flow, the SequenceSpace field in the SLP header takes value '0' to indicate no sequence number or packet framing fields are present in the SLP header, and value '1' to indicate the presence of the sequence number and packet framing fields in the SLP header.

SLP is a protocol associated with the Default Signaling Transport.

3.2.5.2 Primitives

3.2.5.2.1 Commands

This protocol does not define any commands.

3.2.5.2.2 Return indications

This protocol does not return any indications.

3.2.5.3 Protocol data unit

The transmission unit of this protocol is an SLP packet.

3.2.5.4 Procedures

Unless explicitly specified, SLP requirements for the access terminal and the access network are identical; and are, therefore, presented in terms of transmitter and receiver.

SLP receives SNP packets for transmission and forms an SLP packet by prepending the SLP packet header defined in 3.2.5.5 to a contiguous subset of the received octets. The policy SLP follows in determining the number of octets to send in an SLP packet is beyond the scope of this specification. It is subject to the following requirements:

- The size of an SLP packet shall not exceed the maximum payload length that can be carried by the Packet Consolidation Protocol given the target channel and current transmission rate on that channel.
- The SLP payload shall contain octets from no more than one SNP packet.

SLP shall construct the SLP payload(s) from an SNP packet. If the SNP packet exceeds the current maximum SLP payload size, then the sender shall fragment the SNP packet. If the sender does not fragment the SNP packet, then the SNP packet is the SLP payload. If the sender does fragment the SNP packet, then each SNP packet fragment is an SLP payload.

SLP makes use of the ResetRxRequest, ResetRxAck, ResetTxIndication, ResetTxAck, and ReceiverStatus messages to perform control related operations.

SLP is an Ack and Nak-based protocol with a sequence space size of $2^{\text{SequenceLength}}$ bytes.

All operations and comparisons performed on SLP packet sequence numbers shall be carried out in unsigned modulo 2^J arithmetic, where J represents the value of SequenceLength. For any SLP octet sequence number X, the sequence numbers in the range $[X+1, X+2^{J-1}-1]$ shall be considered greater than X and the sequence numbers in the range $[X-2^{J-1}, X-1]$ shall be considered smaller than X.

3.2.5.4.1 Initialization and reset

The SLP initialization procedure initializes the SLP variables and data structures in one end of the link. The SLP reset procedure guarantees that SLP state variables on both sides are synchronized. The reset procedure includes initialization.

If the protocol receives an *IdleState.ConnectionOpened* indication then the access terminal and the access network shall perform the initialization procedures defined in 3.2.5.4.1.1.1 and 3.2.5.4.1.1.2 for both the reliable and best effort flows.

The SLP shall set the sequence space variables Q_{Tx} and Q_{Rx} to '0' after reception of an *IdleState.ConnectionOpened* indication. The SLP shall toggle the value of the sequence space variables Q_{Tx} and Q_{Rx} between '0' and '1' for every subsequent reset.

3.2.5.4.1.1 Initialization procedure

3.2.5.4.1.1.1 Initialization procedure for the SLP transmitter

When SLP transmitter performs the initialization procedure it shall:

- Reset the send state variable $V(S)_P$ to zero.
- Clear the retransmission queue.

3.2.5.4.1.1.2 Initialization procedure for the SLP receiver

When SLP receiver performs the initialization procedure it shall:

- Reset the receive state variables $V(R)_P$ and $V(N)_P$ to zero.
- Clear the re-assembly buffer.

3.2.5.4.1.2 Reset procedure

The reset procedure shall only be used to reset the reliable delivery flows.

3.2.5.4.1.2.1 Reset procedure for the initiating side when it is an SLP transmitter

If the side initiating a reset procedure for the reliable delivery flow is an SLP transmitter, then it shall:

- Perform the SLP transmitter initialization procedure defined in 3.2.5.4.1.1.1 for the reliable delivery flow.
- Toggle the value of the sequence space variable Q_{Tx}.
- Send a ResetTxIndication message.

The SLP transmitter shall not reset again until it receives a ReceiverStatus message with a SequenceSpace field equal to the new value of the sequence space variable Q_{Tx} from the SLP receiver, or a ResetTxAck message with a TransactionID field equal to the TransactionID sent in the ResetTxIndication message.

The SLP transmitter shall ignore any received ResetRxRequest messages until it receives a ReceiverStatus message with a SequenceSpace field with the new value of the sequence space variable Q_{Tx} from the SLP receiver, or a ResetTxAck message with a TransactionID field equal to the TransactionID sent in the ResetTxIndication message.

The SLP transmitter may determine that the ResetTxIndication was lost if it does not receive a ReceiverStatus message with a SequenceSpace field equal to the new value of the sequence space variable $Q_{,Tx}$ from the SLP receiver, or a ResetTxAck message, within an implementation-dependent time interval based on $T_{SLPResponse}$ and an estimate of the round-trip delay. If the SLP transmitter determines that the ResetTxIndication was lost, then the SLP transmitter shall send a new ResetTxIndication message.

3.2.5.4.1.2.2 Reset procedure for initiating side when it is an SLP receiver

If the side initiating a reset procedure for the reliable delivery flow is an SLP receiver, then it shall enter the SLP Reset state. Upon entering the SLP Reset state, SLP shall:

- Perform the SLP receiver initialization procedure defined in 3.2.5.4.1.1.2 for the reliable delivery flow.
- Toggle the value of the sequence space variable Q_{Rx}.
- Send a ResetRxRequest message
- Ignore all SLP data octets received while in the SLP Reset state for the reliable delivery flow with SequenceSpace field not equal to the sequence space variable Q_{Rx}.
- If a ResetRxAck message is received with a TransactionID field equal to the TransactionID sent in the ResetRxRequest message, SLP shall leave the Reset state.
- If an SLP data octet is received with a SequenceSpace field equal to the sequence space variable Q_{Rx}, SLP shall leave the SLP Reset state.

If a ResetRxAck is received while not in the SLP Reset state, the message shall be ignored.

The SLP receiver may determine that the ResetRxRequest was lost if it does not leave the SLP Reset state within an implementation-dependent time interval based on $T_{SLPResponse}$ and an estimate of the round-trip delay. If the SLP receiver determines that the ResetRxRequest was lost, then the SLP receiver shall send a new ResetRxRequest message.

3.2.5.4.1.2.3 Reset procedure for the responding side when it is an SLP receiver

If the side responding to a reset procedure for the reliable delivery flow is an SLP receiver, then upon receiving a ResetTxIndication message, SLP shall perform the following procedures:

- If the sequence space variable Q_{Rx} is not equal to the value of the SequenceSpace field in the ResetTxIndication message, then SLP shall:
  - Perform the SLP receiver initialization procedure defined in 3.2.5.4.1.1.2 for the reliable delivery flow.
  - Toggle the value of the sequence space variable Q_{Rx}.
- Respond with a ResetTxAck message.

Upon receiving an SLP data octet for the reliable delivery flow with a SequenceSpace field not equal to the sequence space variable Q_{Rx}, SLP shall:

- Perform the SLP receiver initialization procedure defined in 3.2.5.4.1.1.2 for the reliable delivery flow.
- Toggle the value of the sequence space variable Q_{Rx}.

3.2.5.4.1.2.4 Reset procedure for the responding side when it is a SLP transmitter

If the side responding to a reset procedure for the reliable delivery flow is an SLP transmitter, then upon receiving a ResetRxRequest message, SLP shall perform the following procedures:

- If the sequence space variable Q_{Tx} is not equal to the value of the SequenceSpace field in the ResetRxRequest message, then SLP shall:
  - Perform the SLP transmitter initialization procedure defined in 3.2.5.4.1.1.1 for the reliable delivery flow.
  - Toggle the value of the sequence space variable Q_{Tx}.
- Respond with a ResetRxAck message.

3.2.5.4.1.2.5 SLP Reset message flows

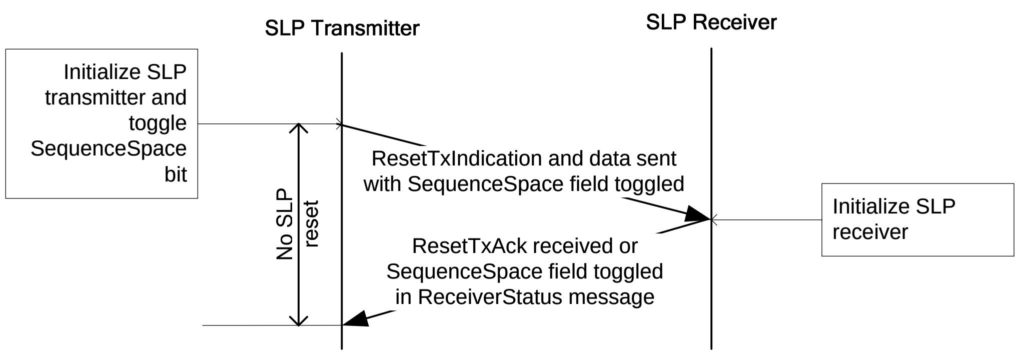

Figure 22 SLP reset procedure initiated by SLP transmitter

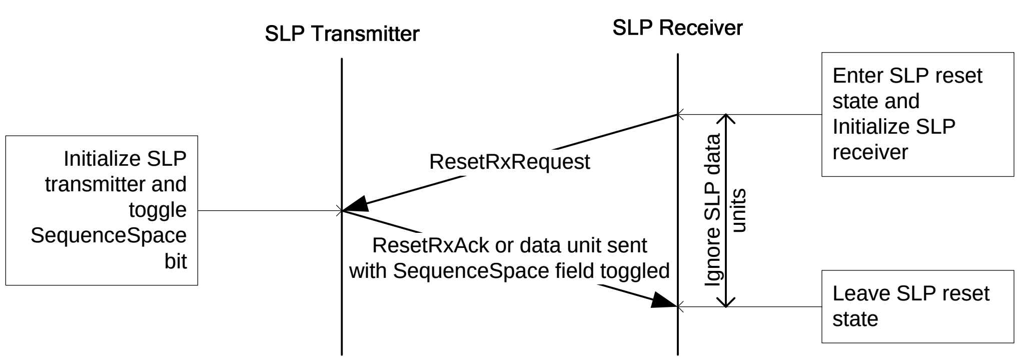

Figure 23 SLP reset procedure initiated by SLP receiver

3.2.5.4.2 SLP transmit procedures

The SLP transmitter shall maintain a SequenceLength-bit variable $V(S)_P$ for all transmitted SLP octets (see Figure 24), where the delivery flow *P* takes on the value "BE" or '0' for best effort delivery and "RD" or '1' for reliable delivery. $V(S)_P$ is the sequence number of the next SLP octet to be sent on delivery flow *P*. The sequence number field (SEQ) in each new SLP packet transmitted shall be set to $V(S)_P$, corresponding to the sequence number of the first octet in the payload. The sequence number of the i^{th} octet in the payload (with the first octet being octet 0) is implicitly given by SEQ+*i*. If the SEQ field is included in the SLP header, then $V(S)_P$ shall be incremented for each octet contained in the SLP payload. If the SEQ field is not included in the SLP header, then $V(S)_P$ shall not be incremented.

If the SLP payload contains the beginning of an SNP packet, then the sender shall set the SLP header First field to '1'; otherwise, the sender shall set the SLP header First field to '0'.

If the SLP payload contains the end of an SNP packet, then the sender shall set the SLP header Last field to '1'; otherwise, the sender shall set the SLP header Last field to '0'.

3.2.5.4.2.1 Best effort delivery transmit procedures

If the SLP payload contains the beginning and end of an SNP packet, the sender shall set the SLP header SequenceSpace field to '0'; otherwise, the sender shall set the SLP header SequenceSpace field to '1'.

3.2.5.4.2.2 Reliable delivery transmit procedures

If a ReceiverStatus message is received with the SequenceSpace field not equal to the value of Q_{Tx}, the message shall be ignored.

If the SLP transmitter is an access terminal, and if a *ReverseTrafficChannelMAC.ReverseTrafficPacketsMissed* indication is received for octets sent with the sequence space not equal to the value of Q_{Tx}, then the indication shall be ignored.

Upon receiving a *ReverseTrafficChannelMAC.ReverseTrafficPacketsMissed* indication, the SLP transmitter in the access terminal shall transmit the requested octets(s) if the requested octets have not been retransmitted $N_{SLPAttempt}$-1 times before.

If the SLP transmitter is an access network, and if a *ForwardTrafficChannelMAC.ForwardTrafficPacketsMissed* indication is received for octets sent with the sequence space not equal to the value of Q_{Tx}, then the indication shall be ignored.

Upon receiving a *ForwardTrafficChannelMAC.ForwardTrafficPacketsMissed* indication, the SLP transmitter in the access network shall transmit the requested octets(s) if the requested octets have not been retransmitted $N_{SLPAttempt}$-1 times before.

Upon receiving a ReceiverStatus message, SLP shall transmit the missing octet(s) (if any) conveyed by the ReceiverStatus message if those octets have not been retransmitted $N_{SLPAttempt}$-1 times before. If the $V(R)_{RD}$ conveyed in the ReceiverStatus message is smaller than $V(S)_{RD} - 1$, then the SLP transmitter may re-transmit one or more of the octets with sequence numbers from $V(R)_{RD}$ to $V(S)_{RD} - 1$, inclusive, if those octets have not been retransmitted $N_{SLPAttempt}$-1 times before.

The SLP transmitter shall meet the following requirements for each octet transmitted:

1. After transmitting an octet, the SLP transmitter shall start a wait Ack timer for time $T_{SLPWaitAck}$.
2. If the SLP transmitter receives a ReceiverStatus message acknowledging the octet before the wait Ack timer expires, the SLP transmitter shall disable the timer.
3. If the timer expires and the octet has not been retransmitted $N_{SLPAttempt}$-1 times before, the SLP transmitter shall retransmit the octets and repeat steps 1 and 2.

If the SLP transmitter is the access network, and the ReceiverStatus record includes any sequence number greater than or equal to $V(S)_{RD}$, SLP shall perform the reset procedures specified in 3.2.5.4.1.2.1 for forward link reliable delivery flow.

If the SLP transmitter is the access terminal, and the ReceiverStatus record includes any sequence number greater than or equal to $V(S)_{RD}$, SLP shall perform the reset procedures specified in 3.2.5.4.1.2.1 for reverse link reliable delivery flow.

If SLP has already transmitted $2^{SequenceLength-1}$ SLP octets, SLP shall transmit an SLP octet with sequence number *n*, only after receiving acknowledgments for the SLP octets transmitted with sequence number $n - 2^{SequenceLength-1}$ and below, or after determining that these SLP octets could not be delivered.

Reliable delivery SLP packets shall be stored in the buffer when they are first transmitted and may be deleted from the buffer, when they are acknowledged or when SLP determines that they could not be delivered.

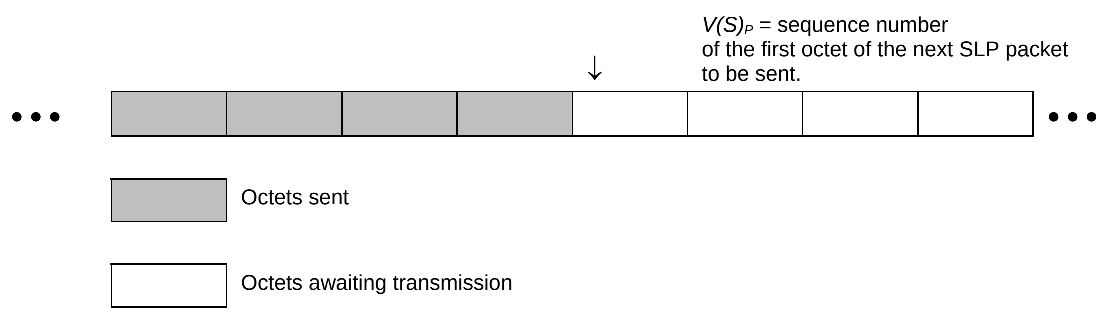

Figure 24 SLP transmit sequence number variable

3.2.5.4.3 SLP receive procedures

The SLP receiver shall maintain an independent re-assembly buffer for each Connection Endpoint as defined by the Address Management Protocol.

The SLP receiver shall maintain two SequenceLength-bit variables for receiving, $V(R)_P$ and $V(N)_P$ (see Figure 25), where *P* is "BE" or '0' for best effort delivery and "RD" or '1' for reliable delivery. $V(R)_P$ contains the sequence number of the next octet expected to arrive. $V(N)_P$ contains the sequence number of the first missing octet, as described below.

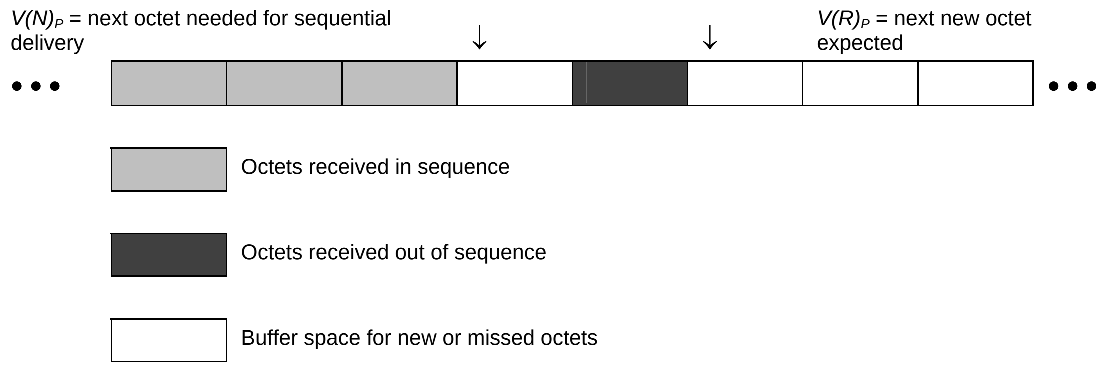

Figure 25 SLP receive sequence number variables

In addition, the SLP receiver shall keep track of the status of each octet in its re-assembly buffer indicating whether the octet was received or not. Use of this status is implied in the following procedures.

3.2.5.4.3.1 Best effort delivery receive procedures

If the SequenceSpace field in the SLP header is '0', SLP shall pass the complete SNP packet to the SNP. Otherwise, in the following, *X* denotes the sequence number of a received octet. For each received octet, SLP shall perform the following procedures:

- If $X < V(R)_{BE}$:
  - SLP shall perform the initialization procedure defined in 3.2.5.4.1.1.2 for the best effort flow.
- SLP shall store the received octet in the re-assembly buffer.
- SLP shall set $V(R)_{BE}$ to X+1.
- SLP shall pass all complete SNP packets in the re-assembly buffer, that have not been passed to the SNP, from the beginning of the re-assembly buffer upward, to the SNP.

The SLP receiver shall meet the following requirements for each SLP packet received on a best effort flow:

- If the SLP packet is not carrying the last segment of a fragmented SNP packet, the SLP receiver shall start a wait next segment timer for time $T_{SLPWaitNextSegment}$. If the wait next segment timer is currently enabled, the SLP receiver shall reset and restart the timer.
- If the SLP packet is carrying the last segment of a fragmented SNP packet, the SLP receiver shall disable the timer.
- If the timer expires and the last segment of a fragmented SNP packet has not been received, the SLP receiver shall perform the initialization procedures defined in 3.2.5.4.1.1.2 for the best effort flow.

3.2.5.4.3.2 Reliable delivery receive procedures

The SLP receiver informs the SLP transmitter of the status of octets in its receive buffer by sending a ReceiverStatus message. The ReceiverStatus message shall convey all missing data from $V(N)_{RD}$ onwards that has not been conveyed in a ReceiverStatus message $N_{SLPAttempt}$-1 times before, and $V(R)_{RD}$. The ReceiverStatus message may convey missing data that has been conveyed in $N_{SLPAttempt}$-1 previous ReceiverStatus messages. The ReceiverStatus message shall not convey status of octets with sequence number less than $V(N)_{RD}$.

In the following, X denotes the sequence number of a received octet. For each received octet, SLP shall perform the following procedures:

- The SLP receiver shall send a ReceiverStatus message for the octet such that the message arrives at the SLP transmitter before the $T_{SLPWaitAck}$ timer expires.
- If $X < V(N)_{RD}$, the octet shall be discarded as a duplicate.
- If $V(N)_{RD} \le X < V(R)_{RD}$, and the octet is not already stored in the re-assembly buffer, then:
    - SLP shall store the received octet in the re-assembly buffer.
    - SLP shall pass all complete SNP packets in the re-assembly buffer, that have not been passed to the SNP, from the beginning of the re-assembly buffer upward, to the SNP.
    - If $X = V(N)_{RD}$, then SLP shall set $V(N)_{RD}$ to (LAST+1) where LAST is the sequence number of the last contiguous octet in the re-assembly buffer.
- If $V(N)_{RD} < X < V(R)_{RD}$, and the octet has already been stored in the re-assembly buffer, then the octet shall be discarded as a duplicate.
- If X = $V(R)_{RD}$, then:
    - SLP shall store the received octet in the re-assembly buffer.
    - SLP shall pass all complete SNP packets in the re-assembly buffer, that have not been passed to the SNP, from the beginning of the re-assembly buffer upward, to the SNP.
    - If $V(R)_{RD} = V(N)_{RD}$, then SLP shall increment $V(N)_{RD}$ and $V(R)_{RD}$.
    - If $V(R)_{RD} \neq V(N)_{RD}$, then SLP shall increment $V(R)_{RD}$.
- If $X > V(R)_{RD}$, then:
    - SLP shall store the octet in the re-assembly buffer.
    - SLP shall pass all complete SNP packets in the re-assembly buffer, that have not been passed to the SNP, from the beginning of the re-assembly buffer upward, to the SNP.
    - SLP shall include a Nak for the missing SLP octets from $V(R)_{RD}$ to X-1, inclusive in the ReceiverStatus message.
    - SLP shall set $V(R)_{RD}$ to X+1.

The SLP receiver shall include all missing octets in each ReceiverStatus message sent. If $N_{SLPAttempt}$-1 Naks have been sent for a missing octet, the SLP shall set $V(N)_{RD}$ to the sequence number of the next missing octet, or to $V(R)_{RD}$ if there are no remaining missing octets. If the SLP receiver determines that a missing octet shall not be retransmitted, the SLP shall set $V(N)_{RD}$ to the sequence number of the next missing octet, or to $V(R)_{RD}$ if there are no remaining missing octets. The SLP may determine that a missing octet shall not be retransmitted based on the arrival time of the first octet received after the missing octet, the number of attempts for each octet $N_{SLPAttempt}$ and the retransmission time $T_{SLPWaitAck}$. Further recovery is the responsibility of the protocol sending the missing SNP packet(s).

3.2.5.5 SLP packet header

The SLP packet header, which precedes the SLP payload, has the following format:

Field	Length (bits)
ReliableDelivery	1
SequenceSpace	1
First	0 or 1
Last	0 or 1
SEQ	0 or SequenceLength
Reserved	0 or 6

ReliableDelivery — Reliable or best effort delivery flag. The sender shall set this flag to ‘1’ for the reliable delivery flow. Otherwise the sender shall set this flag to ‘0’.

SequenceSpace — Sequence space flag for reliable delivery, and sequence space and framing present flag for best effort delivery. If the ReliableDelivery field is set to ‘1’, the sender shall set this flag to the value of the sequence space variable Q_{Tx}. If the ReliableDelivery field is set to ‘0’, the sender shall set this flag to ‘1’ if the First, Last and SEQ fields are included in the SLP packet header. Otherwise, the sender shall set this flag to ‘0’.

First — The sender shall include this field if the ReliableDelivery field is set to ‘1’, or the ReliableDelivery field is set to '0' and the SequenceSpace field is set to 1. Otherwise the sender shall omit this field. If the payload of this SLP packet is the first segment of a SNP packet, then the sender shall set this field to ‘1’. Otherwise, the sender shall set this field to ‘0’.

Last — The sender shall include this field if the ReliableDelivery field is set to ‘1’, or the ReliableDelivery field is set to '0' and the SequenceSpace field is set to 1. Otherwise the sender shall omit this field. If the payload of this SLP packet is the last segment of a SNP packet, then the sender shall set this field to ‘1’. Otherwise, the sender shall set this field to ‘0’.

SEQ — The sender shall include this field if the ReliableDelivery field is set to ‘1’, or the ReliableDelivery field is set to '0' and the SequenceSpace field is set to 1. Otherwise the sender shall omit this field. The sender shall set this field to the SLP sequence number of the first octet in the SLP payload.

Reserved — The sender shall include this field and set it to ‘000000’ if the ReliableDelivery field is set to ‘0’ and SequenceSpace field is set to ‘0’. Otherwise, the sender shall omit this field.

3.2.5.6 Message formats

3.2.5.6.1 ResetRxRequest

The SLP receiver in the access terminal or the access network sends the ResetRxRequest message to reset its peer SLP transmitter.

Field	Length (bits)
MessageID	8
TransactionID	8
Reserved	7
SequenceSpace	1

MessageID — The sender shall set this field to 0x00.

TransactionID — The sender shall set this field according to 10.8.

Reserved — The sender shall set this field to '0000000'. The receiver shall ignore this field.

SequenceSpace — The sender shall set this flag to the value of the sequence space variable Q_{Rx}.

Channels	FTC RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.2.5.6.2 ResetRxAck

The SLP transmitter in the access terminal or the access network sends the ResetRxAck message to complete the SLP reset procedure.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The sender shall set this field to 0x01.

TransactionID — The sender shall set this field to the TransactionID of the associated ResetRxRequest message.

Channels	FTC RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.2.5.6.3 ResetTxIndication

The SLP transmitter in the access terminal or the access network sends the ResetTxIndication message to reset its peer SLP receiver.

Field	Length (bits)
MessageID	8
TransactionID	8
Reserved	7
SequenceSpace	1

MessageID — The sender shall set this field to 0x02.

TransactionID — The sender shall set this field according to 10.8.

Reserved — The sender shall set this field to '0000000'. The receiver shall ignore this field.

SequenceSpace — The sender shall set this flag to the value of the sequence space variable Q_{Tx}.

Channels	FTC RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.2.5.6.4 ResetTxAck

The SLP receiver in the access terminal or the access network sends the ResetTxAck message in response to the ResetTxIndication message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The sender shall set this field to 0x03.

TransactionID — The sender shall set this field to the TransactionID of the associated ResetTxIndication message.

Channels	FTC RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.2.5.6.5 ReceiverStatus

The access terminal and the access network send the ReceiverStatus message to acknowledge the receipt of one or more SLP octets or to request the retransmission of one or more SLP octets for the reliable delivery flow.

Field	Length (bits)
MessageID	8
Reserved0	7
SequenceSpace	1
ReportCount	8
ReportCount occurrences of the following 4 fields:	
{	
Reserved1	4
FirstErasedOctet	SequenceLength
Reserved2	4
WindowLen	SequenceLength
}	
Reserved3	4
VR	SequenceLength

MessageID — The sender shall set this field to 0x04.

Reserved0 — The sender shall set this field to ‘0000000’. The receiver shall ignore this field.

SequenceSpace — The sender shall set this flag to the value of the sequence space variable Q_{Rx}.

ReportCount — The sender shall set this field to the number of Report records included in this message. The sender shall include ReportCount occurrences of the following four fields with the message.

Reserved1 — The sender shall set this field to ‘0000’. The receiver shall ignore this field.

FirstErasedOctet — The sender shall set this field to the sequence number of the first SLP octet erased in a sequence of erased octets.

Reserved2 — The sender shall set this field to ‘0000’. The receiver shall ignore this field.

WindowLen — The sender shall set this field to the length of the erased window in octets.

Reserved3 — The sender shall set this field to ‘0000’. The receiver shall ignore this field.

VR — The sender shall set this field to $V(R)_{RD}$.

Channels	FTC RTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

3.2.5.7 Interface to other protocols

3.2.5.7.1 Commands

This protocol does not issue any commands.

3.2.5.7.2 Indications

This protocol registers to receive the following indications:

- *IdleState.ConnectionOpened*
- *ReverseTrafficChannelMAC.ReverseTrafficPacketsMissed* along with parameters indicating the missing octets.
- *ForwardTrafficChannelMAC.ForwardTrafficPacketsMissed* along with parameters indicating the missing octets.

3.2.5.8 Protocol numeric constants

Constant	Meaning	Value
SequenceLength	Length of the sequence number in the SLP header	20
$N_{SLPRequestLevelRev}$	QoSFlow field is set to '00' for signaling requests in the R-REQCH for a reverse Link SLP packet	'00'
$N_{SLPAttempt}$	Maximum Number of attempts for sending a reliable-delivery SLP packet	3
$T_{SLPWaitAck}$	Retransmission timer for a reliable delivery SLP packet	200 ms
$T_{SLPWaitNextSegment}$	Wait timer for the next segment of a best effort delivery SLP packet	10 seconds
$T_{SLPResponse}$	Time period an SLP receiver has to respond to ResetRxRequest and ResetTxIndication messages.	1 second

3.2.6 Configuration attributes

No configuration attributes are defined for this protocol.

3.2.7 Session state information

The Session State Information record (see 10.10) consists of parameter records.

This transport defines the following parameter records in addition to the configuration attributes for this transport.

3.2.7.1 SignalingLinkState parameter

Table 10 Format of the parameter record for the SignalingLinkState parameter

Field	Length (bits)
ParameterType	8
Length	8
QTxState	1
QRxState	1
Reserved	6

ParameterType This field shall be set to 0x01 for this parameter record.

Length This field shall be set to the length of this parameter record in units of octets excluding the Length field.

QTxState This field shall be set to the value of the sequence state variable Q_{Tx}.

QRxState This field shall be set to the value of the sequence state variable QRx.

Reserved This field shall be set to '000000'. The receiver shall ignore this field.

3.3 Default Data Transport

3.3.1 Introduction

3.3.1.1 General overview

The Default Data Transport provides multiple packet streams that can be used to carry packets between the access terminal and the access network. Each packet stream is called a Link Flow. Each Link Flow provides two routes for transmission and reception of higher layer payloads. These routes are named Route A and Route B and can be carried using a single receiver-transmitter pair. Each route is associated with a transmitter-receiver pair. Figure 26 shows the association between a forward Link Flow and the transmitters and receivers for its two routes. Figure 27 shows the reference architecture for a reverse Link Flow.

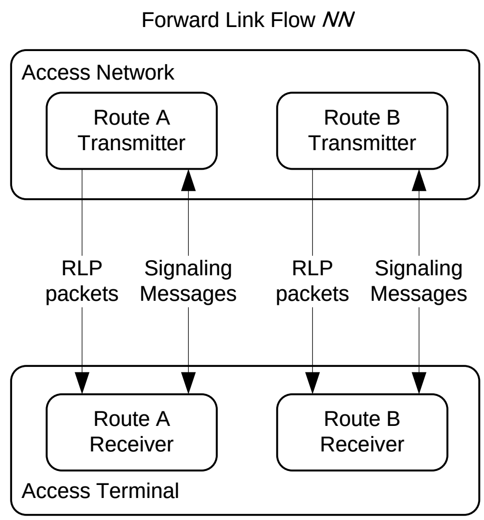

Figure 26 Reference architecture for a forward link flow

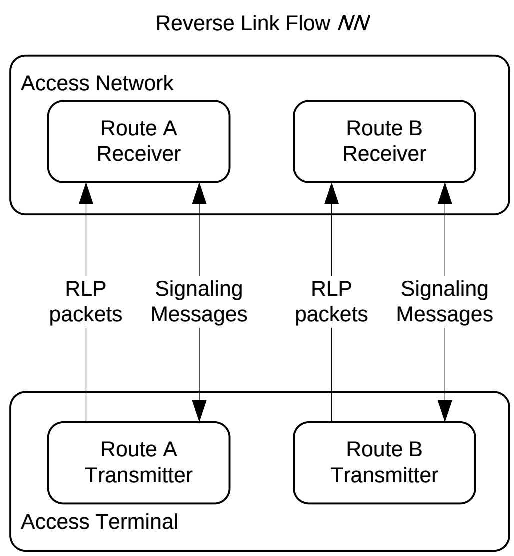

Figure 27 Reference architecture for a reverse link flow

The relationship between the Default Data Transport protocols is illustrated in Figure 28.

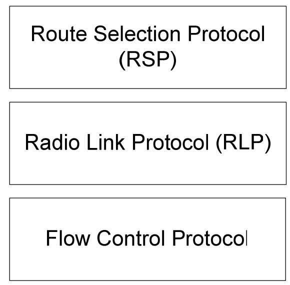

Figure 28 Default data transport protocols

Figure 29 illustrates the relationship for each Link Flow between the Default Data Transport and the higher layer protocols supported by the Default Data Transport. The Flow Protocol and the Route Protocol are referred to as higher layer protocols. The protocols defined in the Default Data Transport are shown shaded. The Route Selection Protocol routes Flow Protocol PDUs to either instance A or instance B of the Route Protocol. Instance A of the Route Protocol is bound to Route A of the Link Flow. Instance B of the Route Protocol is bound to Route B of the Link Flow.

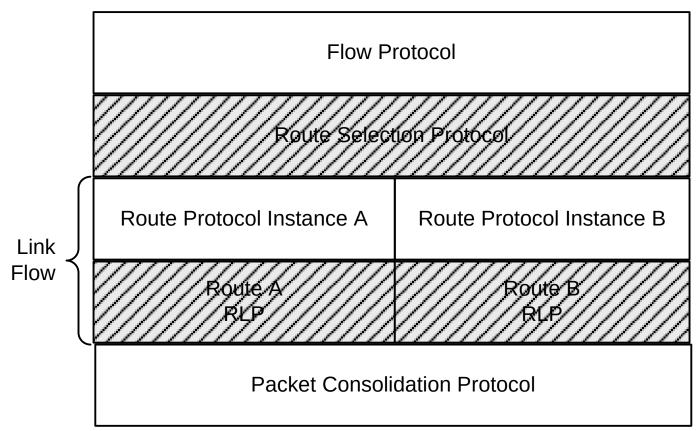

Figure 29 Relationship between default data transport and higher layer protocols

The Default Data Transport provides:

- The Route Selection Protocol, which routes Flow Protocol PDUs over either Route A or Route B of a Link Flow.
- The Radio Link Protocol (RLP), which provides retransmission (if needed) and duplicate detection of higher layer packets transmitted on each route.
- The Flow Control Protocol, which provides flow control for the Default Data Transport.
- The ability to negotiate protocol parameters for all protocols in the Default Data Transport.

3.3.1.2 Public data

3.3.1.2.1 Static public data

This transport does not define any static public data.

3.3.1.2.2 Dynamic public data

- Subtype for this transport
- Flow*NN*RequestLevelRev, where *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive, where hexadecimal digits A through F are specified in upper case letters.

3.3.1.3 Data encapsulation

Figure 30 illustrates the relationship between packets from the Route Protocol, RLP packets, and Packet Consolidation Protocol payload.

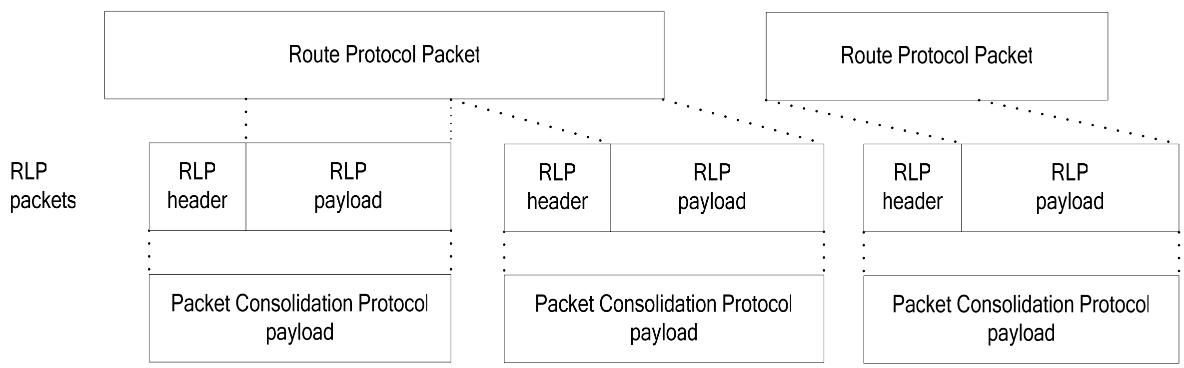

Figure 30 Default data transport encapsulation

The Default Data Transport uses the Signaling Transport to transmit and receive messages.

3.3.2 Transport initialization and swap procedures

3.3.2.1 Transport initialization

Upon creation, the instance of the Data Transport (i.e., corresponding to the Transport defined in the Packet Consolidation Protocol to which this transport is bound) in the access terminal and access network shall perform the following:

- The value of the attributes for this transport instance shall be set to the default values specified for each attribute.
- The Flow Control Protocol associated with the instance of the Data Transport at the access terminal and access network shall enter the Open State[10].

[10] Forward and reverse link Reservations 0xff initialized in the Open state so that data can be sent without having to perform a state transition.

- Forward and reverse link Reservations with ReservationLabel 0xff shall enter the Open state. All other Reservations shall enter the Close state.
- The Route Selection Protocol shall enter the A Open B Draining state.

3.3.2.2 Transport swap

Upon swap, the instance of the Data Transport (i.e., corresponding to the Transport defined in the Packet Consolidation Protocol to which this transport is bound) in the access terminal and access network shall perform the following:

- The Route Selection Protocol shall enter the A Open B Draining state.

3.3.3 Route Selection Protocol

3.3.3.1 Overview

The Route Selection Protocol provides means to select either instance A or instance B of the Route Protocol. The Route Selection Protocol routes Flow Protocol PDUs to the selected instance of the Route Protocol. Instance A of the Route Protocol is bound to Route A of the Link Flow. Instance B of the Route Protocol is bound to Route B of the Link Flow. The Route Selection Protocol is a protocol associated with the Default Data Transport.

3.3.3.2 Primitives

3.3.3.2.1 Commands

This protocol does not define any commands.

3.3.3.2.2 Return indications

This protocol does not return any indications.

3.3.3.3 Protocol data unit

The Route Selection Protocol routes Flow Protocol PDUs to the Route Protocol without modifying them. Hence, the transmission unit of this protocol is the same as a Flow Protocol PDU. The Flow Protocol for a forward Link Flow *NN* is identified by the ProtocolID field of the Flow*NN*FlowProtocolParametersFwd attribute. The Flow Protocol for a reverse Link Flow *NN* is identified by the ProtocolID field of the Flow*NN*FlowProtocolParametersRev attribute.

3.3.3.4 Procedures

3.3.3.4.1 General requirements

If the Flow*NN*SimultaneousDeliveryOnBothRoutesFwd attribute of forward Link Flow *NN* is 0x0000, then forward Link Flow *NN* delivers Flow Protocol PDUs in order. If the Flow*NN*SimultaneousDeliveryOnBothRoutesFwd attribute of forward Link Flow *NN* is 0x0001, then forward Link Flow *NN* may deliver Flow Protocol PDUs out of order.

3.3.3.4.2 Access terminal requirements

The Route Selection Protocol associated with an activated Link Flow can be in one of four states: A Open B Draining, A Open B Activating, A Draining B Open, or A Activating B Open. The Route Selection Protocol instance associated with all activated Link Flows shall be in the same state at any time. When a Link Flow is activated, the Route Selection Protocol shall enter the state that the Route Selection Protocols of other activated Link Flows are in. If no other Link Flows are activated when a Link Flow is activated, then the Route Selection Protocol shall enter the A Open B Draining state. Figure 31 shows the state diagram for the Route Selection Protocol at the access terminal.

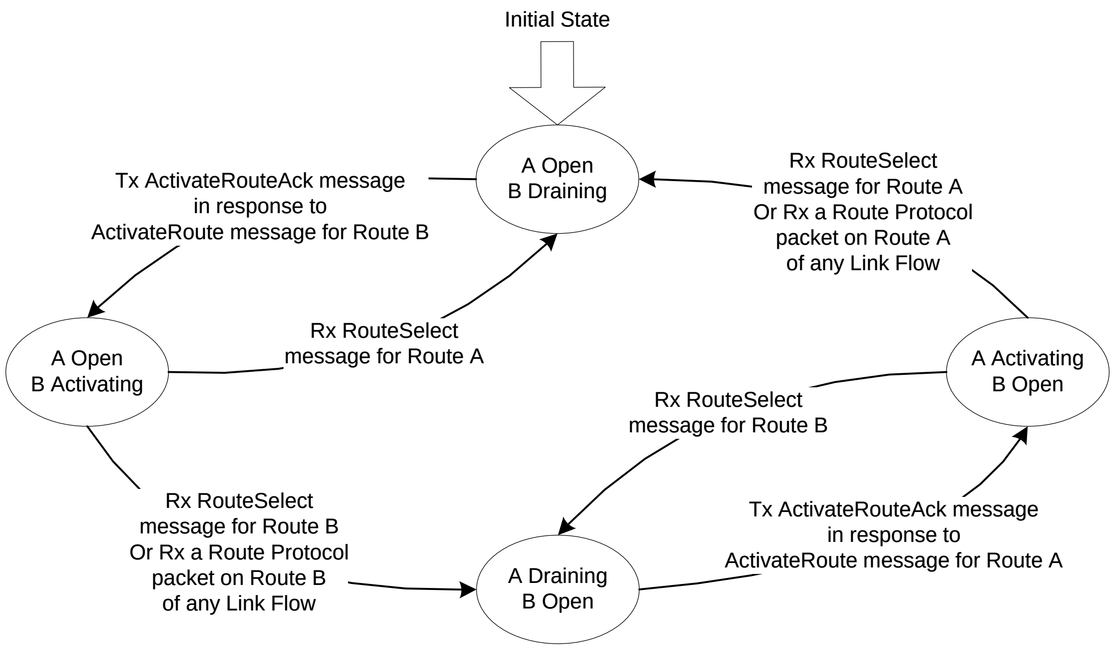

Figure 31 Route selection protocol state diagram (access terminal)

3.3.3.4.2.1 A Open B Draining state

3.3.3.4.2.1.1 State transitions

Upon receiving an ActivateRoute message requesting to activate Route B, the access terminal shall perform the following:

- The Route Selection Protocol shall issue a *RadioLinkProtocol.InitializeRoute* command with Route B as the argument.
- The access terminal shall initialize the Route Protocol bound to Route B.
- After the Radio Link Protocol and the Route Protocol are initialized, the access terminal shall send an ActivateRouteAck message, and shall transition to the A Open B Activating state.

Upon receiving a RouteSelect message for Route A, the access terminal shall respond with a RouteSelectAck message.

3.3.3.4.2.1.2 Transmitter requirements

The access terminal shall route Flow Protocol PDUs to Route A. The access terminal shall not route Flow Protocol PDUs to Route B.

3.3.3.4.2.1.3 Receiver requirements

The access terminal shall pass Flow Protocol PDUs received on Route A to the Flow Protocol.

If the Flow*NN*SimultaneousDeliveryOnBothRoutesFwd attribute for Link Flow *NN* is 0x0001, then the access terminal shall pass Flow Protocol PDUs received on Route B of the Link Flow to the Flow Protocol if the access terminal has not received an ActivateRoute message requesting to activate Route B since the last time it entered this state.

If the Flow*NN*SimultaneousDeliveryOnBothRoutesFwd attribute for Link Flow *NN* is 0x0000, then the access terminal shall pass Flow Protocol PDUs received on Route B of the Link Flow to the Flow Protocol if the access terminal has not passed Flow Protocol PDUs received on Route A of the Link Flow to the Flow Protocol since the last time the access terminal entered this state and if the access terminal has not received an ActivateRoute message requesting to activate Route B since the last time it entered this state. If the Flow*NN*SimultaneousDeliveryOnBothRoutesFwd attribute for Link Flow *NN* is 0x0000, then the access terminal shall discard Flow Protocol PDUs received on Route B of the Link Flow if the access terminal has passed Flow Protocol PDUs received on Route A of the Link Flow to the Flow Protocol since the access terminal entered this state.

3.3.3.4.2.2 A Open B Activating state

3.3.3.4.2.2.1 State transitions

Upon receiving a RouteSelect message requesting to select Route B, the access terminal shall respond with a RouteSelectAck message, and shall transition to the A Draining B Open state.

Upon receiving Flow Protocol PDU on Route B of any Link Flow, the access terminal shall store the Flow Protocol PDU received from Route B for processing in the A Draining B Open state and shall transition to the A Draining B Open state.

Upon receiving a RouteSelect message requesting to select Route A, the access terminal shall respond with a RouteSelectAck message, and shall transition to the A Open B Draining state.

3.3.3.4.2.2.2 Transmitter requirements

The access terminal shall route Flow Protocol PDUs to Route A. The access terminal shall not route Flow Protocol PDUs to Route B.

3.3.3.4.2.2.3 Receiver requirements

The access terminal shall pass Flow Protocol PDUs received on Route A to the Flow Protocol.

3.3.3.4.2.3 A Draining B Open state

3.3.3.4.2.3.1 State transitions

Upon receiving an ActivateRoute message requesting to activate Route A, the access terminal shall perform the following:

- The Route Selection Protocol shall issue a *RadioLinkProtocol.InitializeRoute* command with Route A as the argument.
- The access terminal shall initialize the Route Protocol bound to Route A.
- After the Radio Link Protocol and the Route Protocol are initialized, the access terminal shall send respond with an ActivateRouteAck message, and shall transition to the A Activating B Open state.

Upon receiving a RouteSelect message for Route B, the access terminal shall respond with a RouteSelectAck message.

3.3.3.4.2.3.2 Transmitter requirements

The access terminal shall route Flow Protocol PDUs to Route B. The access terminal shall not route Flow Protocol PDUs to Route A.

3.3.3.4.2.3.3 Receiver requirements

The access terminal shall pass Flow Protocol PDUs received on Route B to the Flow Protocol.

If the Flow*NN*SimultaneousDeliveryOnBothRoutesFwd attribute for Link Flow *NN* is 0x0001, then the access terminal shall pass Flow Protocol PDUs received on Route A of the Link Flow to the Flow Protocol if the access terminal has not received an ActivateRoute message requesting to activate Route A since the last time it entered this state.

If the Flow*NN*SimultaneousDeliveryOnBothRoutesFwd attribute for Link Flow *NN* is 0x0000, then the access terminal shall pass Flow Protocol PDUs received on Route A of the Link Flow to the Flow Protocol if the access terminal has not passed Flow Protocol PDUs received on Route B of the Link Flow to the Flow Protocol since the access terminal entered this state and if the access terminal has not received an ActivateRoute message requesting to activate Route A since the last time it entered this state. If the Flow*NN*SimultaneousDeliveryOnBothRoutesFwd attribute for Link Flow *NN* is 0x0000, then the access terminal shall discard Flow Protocol PDUs received on Route A of the Link Flow if the access terminal has passed Flow Protocol PDUs received on Route B of the Link Flow to the Flow Protocol since the last time the access terminal entered this state.

3.3.3.4.2.4 A Activating B Open state

3.3.3.4.2.4.1 State transitions

Upon receiving a RouteSelect message requesting to select Route A, the access terminal shall respond with a RouteSelectAck message, and shall transition to the A Open B Draining state.

Upon receiving Flow Protocol PDU on Route A of any Link Flow, the access terminal shall store the Flow Protocol PDU received on Route A for processing in the A Open B Draining state and shall transition to the A Open B Draining state.

Upon receiving a RouteSelect message requesting to select Route B, the access terminal shall respond with a RouteSelectAck message, and shall transition to the A Draining B Open state.

3.3.3.4.2.4.2 Transmitter requirements

The access terminal shall route Flow Protocol PDUs to Route B. The access terminal shall not route Flow Protocol PDUs to Route A.

3.3.3.4.2.4.3 Receiver requirements

The access terminal shall pass Flow Protocol PDUs received on Route B to the Flow Protocol.

3.3.3.4.3 Access network requirements

Upon sending an ActivateRoute message requesting to activate Route A, the access network shall issue a *RadioLinkProtocol.InitializeRoute* command with Route A as the argument and initialize the Route Protocol bound to Route A.

Upon sending an ActivateRoute message requesting to activate Route B, the access network shall issue a *RadioLinkProtocol.InitializeRoute* command with Route B as the argument and initialize the Route Protocol bound to Route B.

3.3.3.5 Message formats

3.3.3.5.1 RouteSelect

The access network sends this message to transition the access terminal to the A Open B Draining or the A Draining B Open state.

Field	Length (bits)
MessageID	8
TransactionID	8
Route	1
Reserved	7

MessageID — The access network shall set this field to 0x00.

TransactionID — The access network shall set this field according to 10.8.

Route — The access network shall set this field to ‘0’ to transition the access terminal to the A Open B Draining state. The access network shall set this field to ‘1’ to transition the access terminal to the A Draining B Open state.

Reserved — The access network shall set this field to ‘0000000’. The access terminal shall ignore this field.

Channels	FTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.3.3.5.2 RouteSelectAck

The access terminal sends this message to acknowledge the receipt of a RouteSelect message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID The access terminal shall set this field to 0x01.

TransactionID The access terminal shall set this field to the TransactionID field to the RouteSelect message whose receipt is being acknowledged by this message.

Channels	RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.3.3.5.3 ActivateRoute

The access network sends this message to transition the access terminal to the A Activating B Open state or the A Open B Activating state.

Field	Length (bits)
MessageID	8
TransactionID	8
Route	1
Reserved	7

MessageID The access network shall set this field to 0x02.

TransactionID The access network shall set this field according to 10.8.

Route The access network shall set this field to ‘0’ to transition the access terminal to the A Activating B Open state. The access network shall set this field to ‘1’ to transition the access terminal to the B Activating A Open state.

Reserved The access network shall set this field to ‘0000000’. The access terminal shall ignore this field.

Channels	FTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

3.3.3.5.4 ActivateRouteAck

The access terminal sends this message to acknowledge the receipt of an ActivateRoute message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The access terminal shall set this field to 0x03.

TransactionID — The access terminal shall set this field to the TransactionID field to the ActivateRoute message whose receipt is being acknowledged by this message.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

3.3.3.6 Interface to other protocols

3.3.3.6.1 Commands

This protocol issues the following commands:

- *RadioLink.InitializeRoute* with the argument indicating which Route is to be initialized.

3.3.3.6.2 Indications

This protocol does not register to receive any indications.

3.3.3.7 Protocol numeric constants

This protocol does not define any protocol numeric constants.

3.3.4 Radio Link Protocol

3.3.4.1 Overview

The Radio Link Protocol (RLP) provides one or more packet streams with an acceptably low erasure rate for efficient operation of higher layer protocols (e.g., TCP). When used as part of the Default Data Transport, the protocol carries one or more packet streams from the higher layer. RLP is a protocol associated with the Default Data Transport.

3.3.4.2 Primitives

3.3.4.2.1 Commands

This protocol defines the following commands:

- *InitializeRoute* with argument indicating which Route is to be initialized.

3.3.4.2.2 Return indications

This protocol does not return any indications.

3.3.4.3 Protocol data unit

The transmission unit of this protocol is an RLP packet.

3.3.4.4 Procedures

A forward Link Flow is defined to be activated if the Flow*NN*ActivatedFwd attribute is set to 0x0001, where *NN* is the hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive.

A reverse Link Flow is defined to be activated if the Flow*NN*ActivatedRev attribute is set to 0x0001.

A Link Flow is defined to be deactivated if it is not activated.

Each Route of the Link Flow receives packets for transmission from the corresponding instance of the Route Protocol and forms an RLP packet by prepending the RLP packet header defined in 3.3.4.4.3 with a number of received contiguous octets.

The Route Protocol for a forward Link Flow *NN* is identified by the ProtocolID field of the Flow*NN*RouteProtocolParametersFwd attribute. The Route Protocol for a reverse Link Flow *NN* is identified by the ProtocolID field of the Flow*NN*RouteProtocolParametersRev attribute.

If the Route Protocol is NULL[11], then the transmitter shall set Route Protocol packets to Flow Protocol packets routed along the Route. If the Route Protocol is NULL, then the receiver shall set Flow Protocol packets to Route Protocol packets received on the Route.

If the Flow*NN*OutOfOrderDeliveryToRouteProtocolFwd attribute of forward Link Flow *NN* is 0x0000, then each Route of forward Link Flow *NN* delivers packets of the corresponding instance of the Route Protocol in order. If the Flow*NN*OutOfOrderDeliveryToRouteProtocolFwd attribute of forward Link Flow *NN* is 0x0001, then each Route of forward Link Flow *NN* may deliver packets of the corresponding instance of the Route Protocol out of order.

If the Flow*NN*OutOfOrderDeliveryToRouteProtocolRev attribute of reverse Link Flow *NN* is 0x0000, then each Route of reverse Link Flow *NN* delivers packets of the corresponding instance of the Route Protocol in order. If the Flow*NN*OutOfOrderDeliveryToRouteProtocolRev attribute of reverse Link Flow *NN* is 0x0001, then each Route of reverse Link Flow *NN* may deliver packets of the corresponding instance of the Route Protocol out of order.

The policy RLP follows in determining the number of octets to send in an RLP packet is beyond the scope of this specification. It is subject to the following requirements:

- The size of an RLP packet shall not exceed the maximum payload length that can be carried by a Packet Consolidation Protocol packet given the target channel and current transmission rate on that channel.
- An RLP packet shall contain octets from no more than one Route Protocol packet.

[11] Route Protocol being NULL means that a Route Protocol has not been negotiated.

RLP makes use of the ResetRxRequest, ResetRxAck, ResetTxIndication, ResetTxAck, and ReceiverStatus in-band messages to perform control related operations.

The access terminal shall not initiate negotiation of the ANSupportedQoSProfiles attribute. The access network shall not initiate negotiation of the ATSupportedQoSProfiles attribute.

The access network shall not initiate modification of the Reservation*KK*QoSListFwd or the Reservation*KK*QoSListRev attributes. If the access network receives an AttributeUpdateRequest message requesting to set the Reservation*KK*QoSListFwd or the Reservation*KK*QoSListRev attribute to its default value, then the access network shall respond with an AttributeUpdateAccept message.

The access terminal shall not initiate modification of the Reservation*KK*QoSUsedFwd or the Reservation*KK*QoSUsedRev attributes.

The access terminal uses the AttributeUpdateRequest message with the Reservation*KK*QoSListFwd attributes to add, modify, or remove the QoS for forward Reservation *KK*. The access terminal requests one or more QoSAttributeSets in order of preferences for forward Reservation *KK*. The access terminal uses the AttributeUpdateRequest message with the Reservation*KK*QoSListRev attributes to add, modify, or remove the QoS for reverse Reservation *KK*. The access terminal requests one or more QoSAttributeSets in order of preferences for reverse Reservation *KK*. Each QoSAttributeSet contains a group of detailed QoS parameters.

The access network stores the requested QoSAttributeSets in the AttributeUpdateRequest message for the Reservation*KK*QoSListFwd attribute and grants one. The access network informs the access terminal which QoSAttributeSet it granted by the QoSAttributeSet_ID of the Reservation*KK*QoSUsedFwd attribute in an AttributeUpdateRequest message.

The access network stores the requested QoSAttributeSets in the AttributeUpdateRequest message for the Reservation*KK*QoSListRev attribute and grants one. The access network informs the access terminal which QoSAttributeSet it granted by the QoSAttributeSet_ID of the Reservation*KK*QoSUsedRev attribute in an AttributeUpdateRequest message.

If the access terminal sends a new AttributeUpdateRequest for the Reservation*KK*QoSListFwd attribute, the new requested Reservation*KK*QoSListFwd attribute shall replace the previous requested Reservation*KK*QoSListFwd attribute for forward Reservation *KK*. If the access terminal sends a new AttributeUpdateRequest for the Reservation*KK*QoSListRev attribute, the new requested Reservation*KK*QoSListRev attribute shall replace the previous requested Reservation*KK*QoSListRev attribute for reverse Reservation *KK*. In any new requested Reservation*KK*QoSListFwd or Reservation*KK*QoSListRev attribute for Reservation *KK*, the access terminal shall not re-use values for QoSAttributeSet_ID from the previous two Reservation*KK*QoSListFwd or Reservation*KK*QoSListRev attributes respectively, negotiated for Reservation *KK*.

The access network may change the granted QoSAttributeSet_ID to another QoSAttributeSet_ID in the group of QoSAttributeSet_IDs most recently requested by the access terminal. The access network shall inform the access terminal of the new granted QoSAttributeSet_ID.

The access network shall not initiate modification of the Reservation*KK*PacketFilterFwd or the Reservation*KK*PacketFilterRev attributes. If the access network receives an AttributeUpdateRequest message requesting to set the Reservation*KK*PacketFilterFwd or the Reservation*KK*PacketFilterRev attributes to their default values, then the access network shall respond with an AttributeUpdateAccept message. If the FilterSpecType is set to 0x02 according to Table 13, then the packet filter for Reservation*KK*PacketFilterFwd or Reservation*KK*PacketFilterRev shall match all packets that do not match a packet filter with a lower value FilterPrecedence field.

The access network may send a FlowQoSDetect message to inform the access terminal that it should add a new Reservation or modify the Reservation*KK*QoSListFwd or Reservation*KK*QoSListRev attribute for an existing Reservation *KK*. If the access terminal determines that the FlowQoSDetect message corresponds to a Reservation that it has not already added or modified, the access terminal should send an AttributeUpdateRequest message for the Reservation*KK*QoSListFwd or Reservation*KK*QoSListRev attribute in response to the FlowQoSDetect message.

When forward Link Flow *NN* is activated, the access network and the access terminal shall not update the following attributes:

- Flow*NN*FlowProtocolParametersFwd
- Flow*NN*RouteProtocolParametersFwd
- Flow*NN*SequenceLengthFwd
- Flow*NN*DataUnitFwd
- Flow*NN*SimultaneousDeliveryOnBothRoutesFwd
- Flow*NN*OutOfOrderDeliveryToRouteProtocolFwd

When reverse Link Flow *NN* is activated, the access network and the access terminal shall not update the following attributes:

- Flow*NN*FlowProtocolParametersRev
- Flow*NN*RouteProtocolParametersRev
- Flow*NN*SequenceLengthRev
- Flow*NN*DataUnitRev
- Flow*NN*OutOfOrderDeliveryToRouteProtocolRev

The ProtocolID field of the Flow*NN*FlowProtocolParametersFwd attribute shall be set to a value that is supported by the access terminal as indicated in the ATSupportedFlowProtocolParameters*PP* attribute. The ProtocolID field of the Flow*NN*FlowProtocolParametersRev attribute shall be set to a value that is supported by the access terminal as indicated in the ATSupportedFlowProtocolParameters*PP* attribute. The ProtocolID field of the Flow*NN*RouteProtocolParametersFwd attribute shall be set to a value that is supported by the access terminal as indicated in the ATSupportedRouteProtocolParameters*PP* attribute. The ProtocolID field of the Flow*NN*RouteProtocolParametersRev attribute shall be set to a value that is supported by the access terminal as indicated in the ATSupportedRouteProtocolParameters*PP* attribute.

The fields of the ProtocolParameters record of the Flow*NN*FlowProtocolParametersFwd attribute shall be set to values that are in accordance with those supported by the AT as indicated in the SupportedProtocolsParametersValues record of the ATSupportedFlowProtocolParameters*PP* attribute. The fields of the ProtocolParameters record of the Flow*NN*FlowProtocolParametersRev attribute shall be set to values that are in accordance with those supported by the AT as indicated in the SupportedProtocolsParametersValues record of the ATSupportedFlowProtocolParameters*PP* attribute.

The fields of the ProtocolParameters record of the Flow*NN*RouteProtocolParametersFwd attribute shall be set to values that are in accordance with those supported by the AT as indicated in the SupportedProtocolsParametersValues record of the ATSupportedRouteProtocolParameters*PP* attribute. The fields of the ProtocolParameters record of the Flow*NN*RouteProtocolParametersRev attribute shall be set to values that are in accordance with those supported by the AT as indicated in the SupportedProtocolsParametersValues record of the ATSupportedRouteProtocolParameters*PP* attribute.

If the Flow*NN*DataUnitFwd attribute of forward Link Flow *NN* is 0x0000, then the data unit for the Link Flow shall be octets. Otherwise the data unit for the Link Flow shall be RLP packet payloads. If the Flow*NN*DataUnitRev attribute of reverse Link Flow *NN* is 0x0000, then the data unit for the Link Flow shall be octets. Otherwise the data unit for the Link Flow shall be RLP packet payloads.

3.3.4.4.1 Initialization and reset

The RLP initialization procedure initializes the RLP variables and data structures in one end of the link. The RLP reset procedure guarantees that RLP state variables on both sides are synchronized. The reset procedure includes initialization.

If the protocol receives an *IdleState.ConnectionOpened* indication, then the access terminal and the access network shall perform the initialization procedures defined in 3.3.4.4.1.1.1 and 3.3.4.4.1.1.2 for both routes of all activated Link Flows.

The access network shall perform the initialization procedure defined in 3.3.4.4.1.1.1 for both routes of forward Link Flow *NN* when forward Link Flow *NN* is activated. The access terminal shall perform the initialization procedure defined in 3.3.4.4.1.1.2 for both routes of forward Link Flow *NN* when forward Link Flow *NN* is activated.

The access terminal shall perform the initialization procedure defined in 3.3.4.4.1.1.1 for both routes of reverse Link Flow *NN* when reverse Link Flow *NN* is activated. The access network shall perform the initialization procedure defined in 3.3.4.4.1.1.2 for both routes of reverse Link Flow *NN* when reverse Link Flow *NN* is activated.

Upon receiving an *InitializeRoute* command, the access terminal shall perform the initialization procedures defined in 3.3.4.4.1.1 for the specified Route for all activated Link Flows. Upon receiving an *InitializeRoute* command, the access network shall perform the initialization procedures defined in 3.3.4.4.1.1 for the specified Route of all activated Link Flows.

Each Link Flow provides sequence spaces variables $Q_{NN,Tx}$ and $Q_{NN,Rx}$ at the transmitter and receiver respectively. The transmitter toggles the sequence space variable $Q_{NN,Tx}$ between ‘0’ and ‘1’ to indicate a reset. The receiver sequence space variable $Q_{NN,Rx}$ tracks the value of $Q_{NN,Tx}$ and detects when the transmitter has performed a reset.

The RLP shall set the sequence space variables $Q_{NN,Tx}$ and $Q_{NN,Rx}$ to ‘0’ after reception of an *IdleState.ConnectionOpened* indication. The RLP shall toggle the value of the sequence space variables $Q_{NN,Tx}$ and $Q_{NN,Rx}$ between ‘0’ and ‘1’ for every subsequent reset.

3.3.4.4.1.1 Initialization procedure

3.3.4.4.1.1.1 Initialization procedure for the RLP transmitter

When RLP transmitter performs the initialization procedure it shall:

- Reset the send state variable $V(S)_{NN,P}$ to zero, where *NN* indicates the Link Flow, and *P* indicates the Route which is being initialized.
- Clear the retransmission queue.

3.3.4.4.1.1.2 Initialization procedure for the RLP receiver

When RLP receiver performs the initialization procedure it shall:

- Reset the receive state variables $V(R)_{NN,P}$ and $V(N)_{NN,P}$ to zero.
- Clear the resequencing buffer.

3.3.4.4.1.2 Reset procedure

3.3.4.4.1.2.1 Reset procedure for the initiating side when it is an RLP transmitter

If the side initiating a reset procedure is an RLP transmitter for the Route of the Link Flow (or all Link Flows) being reset, then it shall:

- Perform the RLP transmitter initialization procedure defined in 3.3.4.4.1.1.1 for the Route of the Link Flow being reset.
- Toggle the value of the sequence space variable $Q_{NN,Tx}$ for the Route of the Link Flow being reset.
- Send a ResetTxIndication message.

The RLP transmitter shall not reset again until it receives a ReceiverStatus message for the Route of the Link Flow being reset with a SequenceSpace field equal to the new value of the sequence space variable $Q_{NN,Tx}$ from the RLP receiver, or a ResetTxAck message with a TransactionID field equal to the TransactionID sent in the ResetTxIndication message.

The RLP transmitter shall ignore any received ResetRxRequest messages for the Route of the Link Flow being reset until it receives a ReceiverStatus message with a SequenceSpace field with the new value of the sequence space variable $Q_{NN,Tx}$ from the RLP receiver, or a ResetTxAck message with a TransactionID field equal to the TransactionID sent in the ResetTxIndication message.

The RLP transmitter may determine that the ResetTxIndication was lost if it does not receive a ReceiverStatus message for the Route of the Link Flow being reset with a SequenceSpace field equal to the new value of the sequence space variable $Q_{NN,Tx}$ from the RLP receiver, or a ResetTxAck message, within an implementation-dependent time interval based on $T_{RLPResponse}$ and an estimate of the round-trip delay. If the RLP transmitter determines that the ResetTxIndication was lost, then the RLP transmitter shall send a new ResetTxIndication message.

3.3.4.4.1.2.2 Reset procedure for initiating side when it is an RLP receiver

If the side initiating a reset procedure is an RLP receiver for the Route of the Link Flow being reset, then it shall enter the RLP Reset State. Upon entering the RLP Reset state, RLP shall:

- Perform the RLP receiver initialization procedure defined in 3.3.4.4.1.1.2 for the Route of the Link Flow being reset.
- Toggle the value of the sequence space variable $Q_{NN,Rx}$ for the Route of the Link Flow being reset.
- Send a ResetRxRequest message.
- Ignore all RLP data units received for the Route of the Link Flow being reset while in the RLP Reset state with SequenceSpace field not equal to the sequence space variable $Q_{NN,Rx}$.
- If a ResetRxAck message is received for the Route of the Link Flow being reset with a TransactionID field equal to the TransactionID sent in the ResetRxRequest message, RLP shall leave the RLP reset state.
- If an RLP data unit is received with a SequenceSpace field equal to the sequence space variable $Q_{NN,Rx}$ for the Route of the Link Flow being reset, RLP shall leave the RLP Reset state.

If a ResetRxAck is received for a Route while the Route is not in the RLP Reset state, the message shall be ignored.

The RLP receiver may determine that the ResetRxRequest was lost if it does not leave the RLP Reset state within an implementation-dependent time interval based on $T_{RLPResponse}$ and an estimate of the round-trip delay. If the RLP receiver determines that the ResetRxRequest was lost, then the RLP receiver shall send a new ResetRxRequest message.

3.3.4.4.1.2.3 Reset procedure for the responding side when it is an RLP receiver

If the side responding to a reset procedure is an RLP receiver for the Route of the Link Flow being reset, then upon receiving a ResetTxIndication message, RLP shall perform the following procedures:

- If the sequence space variable $Q_{NN,Rx}$ for the Route of the Link Flow being reset is not equal to the value of the SequenceSpace field in the ResetTxIndication message, then RLP shall:
  - Perform the RLP receiver initialization procedure defined in 3.3.4.4.1.1.2 for the Route of the Link Flow being reset.
  - Toggle the value of the sequence space variable $Q_{NN,Rx}$ for the Route of the Link Flow being reset.
- Respond with a ResetTxAck message.

Upon receiving an RLP data unit for the Route of the Link Flow with a SequenceSpace field not equal to the sequence space variable $Q_{NN,Rx}$, RLP shall:

- Perform the RLP receiver initialization procedure defined in 3.3.4.4.1.1.2 for the Route of the Link Flow being reset.
- Toggle the value of the sequence space variable $Q_{NN,Rx}$ for the Route of the Link Flow being reset.

3.3.4.4.1.2.4 Reset procedure for the responding side when it is an RLP transmitter

If the side responding to a reset procedure is an RLP transmitter for the Route of the Link Flow being reset, then upon receiving a ResetRxRequest message, RLP shall perform the following procedures:

- If the sequence space variable $Q_{NN,Tx}$ is not equal to the value of the SequenceSpace field in the ResetRxRequest message, then RLP shall:
  - Perform the RLP transmitter initialization procedure defined in 3.3.4.4.1.1.1 for the Route of the Link Flow being reset.
  - Toggle the value of the sequence space variable $Q_{NN,Tx}$ for the Route of the Link Flow being reset.
- Respond with a ResetRxAck message.

3.3.4.4.1.2.5 RLP reset message flows

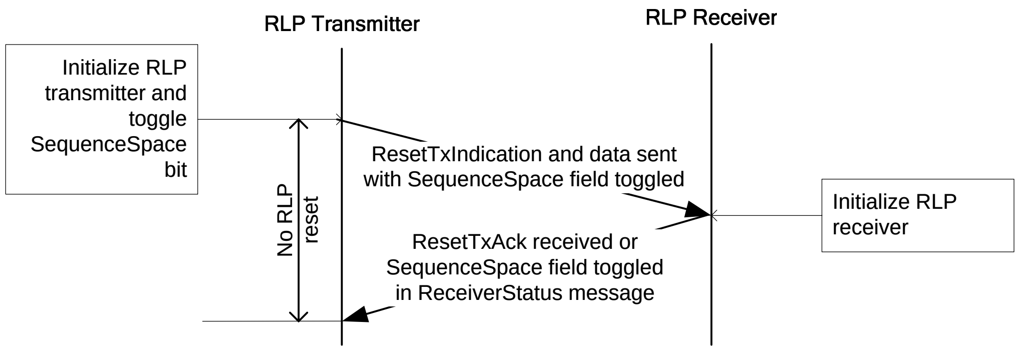

Figure 32 RLP reset procedure initiated by RLP transmitter

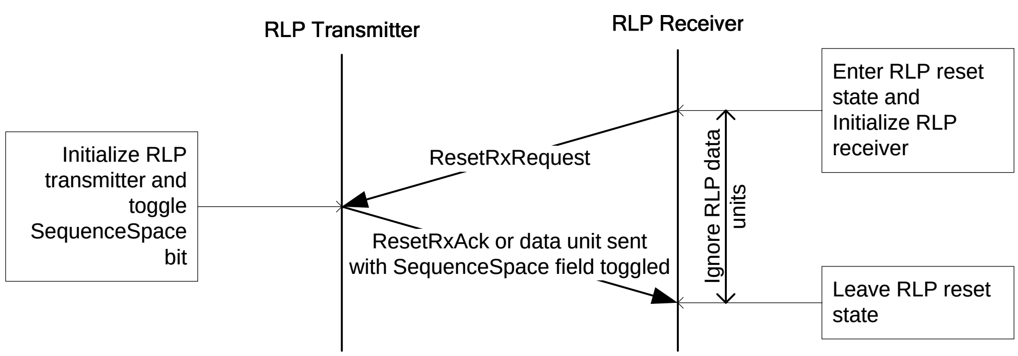

Figure 33 RLP reset procedure initiated by RLP receiver

3.3.4.4.2 Data transfer

RLP is an Ack and/or Nak-based protocol with a sequence space of SequenceLength bits, where SequenceLength is indicated by the Flow*NN*SequenceLengthFwd and Flow*NN*SequenceLengthRev attribute for forward and reverse Link Flow *NN*, respectively.

All operations and comparisons performed on RLP packet sequence numbers shall be carried out in unsigned modulo 2^S arithmetic, where S represents the value of SequenceLength. For any RLP octet sequence number *N*, the sequence numbers in the range $[N+1, N+2^{S-1}-1]$ shall be considered greater than *N* and the sequence numbers in the range $[N-2^{S-1}, N-1]$ shall be considered smaller than *N*.

3.3.4.4.2.1 RLP transmit procedures

The RLP transmitter shall maintain a SequenceLength-bit variable $V(S)_{NN,P}$ for all transmitted RLP data units (see Figure 34), where *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive, and *P* is the Route indicator that takes values of either A or B. $V(S)_{NN,P}$ is the sequence number of the next RLP data unit to be sent on Route *P* of Link Flow *NN*. The sequence number field (SEQ) in each new RLP packet transmitted shall be set to $V(S)_{NN,P}$, corresponding to the sequence number of the first data unit in the packet. If the data unit is octets, then the sequence number of the i^{th} octet in the packet (with the first octet being octet 0) is implicitly given by SEQ+*i*. $V(S)_{NN,P}$ shall be incremented for each data unit contained in the packet.

If an RLP data unit is to be transmitted on the Forward Traffic Channel or on the Reverse Traffic Channel, and if a connection is not open, RLP shall issue an *AirLinkManagement.OpenConnection* command. RLP should queue all data units requiring transmission in the Forward Traffic Channel or in the Reverse Traffic Channel until the protocol receives an *IdleState.ConnectionOpened* indication.

If Flow*NN*SequenceLengthFwd is 0x0000, then the access network will follow the procedures in 3.3.4.4.2.1.1 when transmitting an RLP packet. If Flow*NN*SequenceLengthFwd is not 0x0000, then the access network will follow the procedures in 3.3.4.4.2.1.2 when transmitting an RLP packet.

If Flow*NN*SequenceLengthRev is 0x0000, then the access terminal will follow the procedures in 3.3.4.4.2.1.1 when transmitting an RLP packet. If Flow*NN*SequenceLengthRev is not 0x0000, then the access terminal will follow the procedures in 3.3.4.4.2.1.2 when transmitting an RLP packet.

3.3.4.4.2.1.1 Transmit procedures for flows with SequenceLength of zero

If the Flow*NN*SequenceLengthFwd or the Flow*NN*SequenceLengthRev is 0x0000, the RLP transmitter shall set the First and Last fields of the RLP header to '1'.

If the Flow*NN*SequenceLengthFwd is 0x0000, then the Flow*NN*AckNakEnableFwd shall be set to 0x0000. If the Flow*NN*SequenceLengthFwd is 0x0000, then the Flow*NN*FTCMACNakEnableFwd attribute should be set to 0x0000. If the Flow*NN*SequenceLengthFwd is 0x0000, then the Flow*NN*OutOfOrderDeliveryToRouteProtocolFwd shall be set to 0x0001.

If the Flow*NN*SequenceLengthRev is 0x0000, then the Flow*NN*AckNakEnableRev shall be set to 0x0000. If the Flow*NN*SequenceLengthRev is 0x0000, then the Flow*NN*RTCMACNakEnableRev attribute should be set to 0x0000. If the Flow*NN*SequenceLengthRev is 0x0000, then the Flow*NN*OutOfOrderDeliveryToRouteProtocolRev shall be set to 0x0001.

3.3.4.4.2.1.2 Transmit procedures for flows with non-zero SequenceLength

The RLP transmitter should allow sufficient time before deleting an RLP packet payload transmitted for the first time.

If a ReceiverStatus message is received with the SequenceSpace field not equal to the value of $Q_{NN,Tx}$, the message shall be ignored.

Upon receiving a ReceiverStatus message, RLP shall transmit the missing data unit(s) (if any) conveyed by the ReceiverStatus message if those data units are available and if those data units have not been retransmitted before in response to a ReceiverStatus message. Upon receiving a ReceiverStatus message, RLP may transmit the missing data unit(s) (if any) conveyed by the ReceiverStatus message if those data units are available and if those data units have been retransmitted before in response to a ReceiverStatus message.

If the RLP transmitter is the access network and if Flow*NN*AckNakEnableFwd is not 0x0000 or 0x0001, then the access network may determine that transmitted data units have been lost if it does not receive a ReceiverStatus message acknowledging the receipt of the data units within an implementation-dependent time interval based on the AckTimer and an estimate of the round-trip delay. If the RLP transmitter is the access network and if Flow*NN*AckNakEnableFwd is not 0x0000 or 0x0001 and the access network determines that transmitted data units were lost, then the access network shall re-transmit the data units if they have not been re-transmitted in response to a ReceiverStatus message. If the RLP transmitter is the access network and if Flow*NN*AckNakEnableFwd is not 0x0000 or 0x0001 and the access network determines that transmitted data units were lost, then the access network may re-transmit the data units if they have been re-transmitted in response to a ReceiverStatus message.

If the RLP transmitter is the access terminal and if Flow*NN*AckNakEnableRev is not 0x0000 or 0x0001, then the access terminal may determine that transmitted data units have been lost if it does not receive a ReceiverStatus message acknowledging the receipt of the data units within an implementation-dependent time interval based on the AckTimer and an estimate of the round-trip delay. If the RLP transmitter is the access terminal and if Flow*NN*AckNakEnableRev is not 0x0000 or 0x0001 and the access terminal determines that transmitted data units were lost, then the access terminal shall re-transmit the data units if they have not been re-transmitted in response to a ReceiverStatus message. If the RLP transmitter is the access terminal and if Flow*NN*AckNakEnableRev is not 0x0000 or 0x0001 and the access terminal determines that transmitted data units were lost, then the access terminal may re-transmit the data units if they have been re-transmitted in response to a ReceiverStatus message.

If the RLP transmitter is the access network, and the ReceiverStatus record includes any sequence number greater than or equal to $V(S)_{NN,P}$, RLP shall perform the reset procedures specified in 3.3.4.4.1.2.1 for Route *P* of forward Link Flow *NN*. If the RLP transmitter is the access terminal, and the ReceiverStatus record includes any sequence number greater than or equal to $V(S)_{NN,P}$, RLP shall perform the reset procedures specified in 3.3.4.4.1.2.1 for Route *P* of reverse Link Flow *NN*. If the ReceiverStatus record does not include any sequence number greater than or equal to $V(S)_{NN,P}$ but the requested data units are not available for retransmission, RLP shall ignore the ReceiverStatus record for data units that are not available.

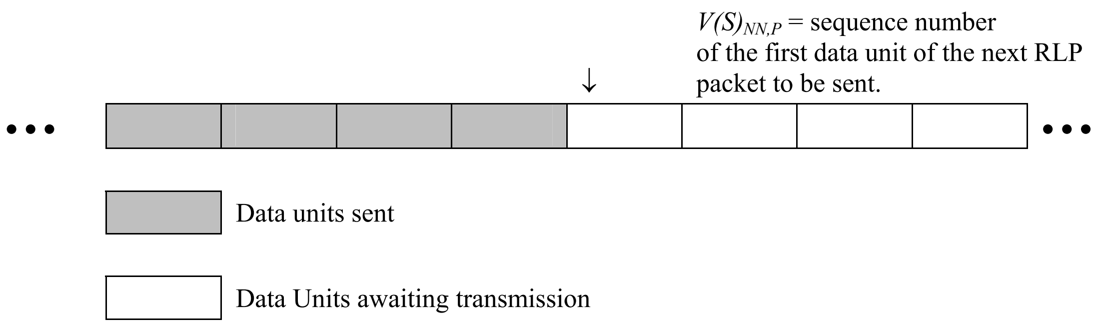

Figure 34 RLP transmit sequence number variable

Upon receiving a *ForwardTrafficChannelMAC.ForwardTrafficPacketsMissed* indication for forward Link Flow *NN*, the RLP transmitter in the access network shall transmit the requested data units(s) if and only if all of the following conditions are satisfied:

- Flow*NN*FTCMACNakEnableFwd attribute is set to 0x0001.
- The requested data units have not been retransmitted before.
- The requested data units are available.
- The sequence space for the data units sent is equal to the value of $Q_{NN,Tx}$.

Upon receiving a *ReverseTrafficChannelMAC.ReverseTrafficPacketsMissed* indication for reverse Link Flow *NN*, the RLP transmitter in the access terminal shall transmit the requested data units(s) if and only if all of the following conditions are satisfied:

- Flow*NN*RTCMACNakEnableRev attribute is set to 0x0001.
- The requested data units have not been retransmitted before.
- The requested data units are available.
- The sequence space for the data units sent is equal to the value of $Q_{NN,Tx}$.

If Flow*NN*AckNakEnableFwd is 0x0001, then the transmitter at the access network for each Route of Link Flow *NN* shall meet the following requirements:

- After transmitting a packet, the RLP transmitter shall start a flush timer for time FlushTimer, where FlushTimer is a parameter of the Flow*NN*TimersFwd attribute.
- If the RLP transmitter sends another packet before the flush timer expires, the RLP transmitter shall reset and restart the timer.
- If the timer expires, the RLP transmitter shall disable the flush timer and the RLP transmitter should send an RLP packet that contains at least the data unit with sequence number $V(S)_{NN,P}$-1.

If Flow*NN*AckNakEnableRev is 0x0001, then the transmitter at the access terminal for each Route of Link Flow *NN* shall meet the following requirements:

- After transmitting a packet, the RLP transmitter shall start a flush timer for time FlushTimer, where FlushTimer is a parameter of the Flow*NN*TimersRev attribute.
- If the RLP transmitter sends another packet before the flush timer expires, the RLP transmitter shall reset and restart the timer.

- If the timer expires, the RLP transmitter shall disable the flush timer and the RLP transmitter should send an RLP packet that contains at least the data unit with sequence number $V(S)_{NN,P}$-1.

The RLP transmitter should not transmit more than $2^{SequenceLength-1}$ first-time data units in any AbortTimer interval, where SequenceLength is the length of the SEQ field in the RLP header for the corresponding Link Flow.

3.3.4.4.2.1.3 Reservation State Maintenance

The ReservationLabel parameter of the Flow*NN*ReservationFwd or Flow*NN*ReservationRev attribute indicates the higher layer flows associated with Link Flow *NN*. Each ReservationLabel shall be associated with no more than one forward Link Flow. Each ReservationLabel shall be associated with no more than one reverse Link Flow.

Each Reservation can be in one of the following two states:

- Close State
- Open State

The transmitter should transmit a higher layer packet using the Link Flow associated with the higher layer flow if the associated Link Flow is activated and if the Reservation is in the Open state. The transmitter should transmit a higher layer packet belonging to a higher layer flow that is not associated with any Link Flow using the Link Flow with ReservationLabel 0xff. The transmitter may transmit a higher layer packet belonging to a higher layer flow identified by a Reservation that is in the Close state using the Link Flow with ReservationLabel 0xff. The transmitter may transmit a higher layer packet belonging to a higher layer flow identified by a Reservation that is bound to a de-activated Link Flow using the Link Flow with ReservationLabel 0xff.

Figure 35 and Figure 36 show the state transition diagram at the access terminal and the access network. State transitions that may be caused by *IdleState.ConnectionOpened*, *ConnectedState.ConnectionClosed*, and *ActiveSetManagement.ConnectionLost* indications are not shown.

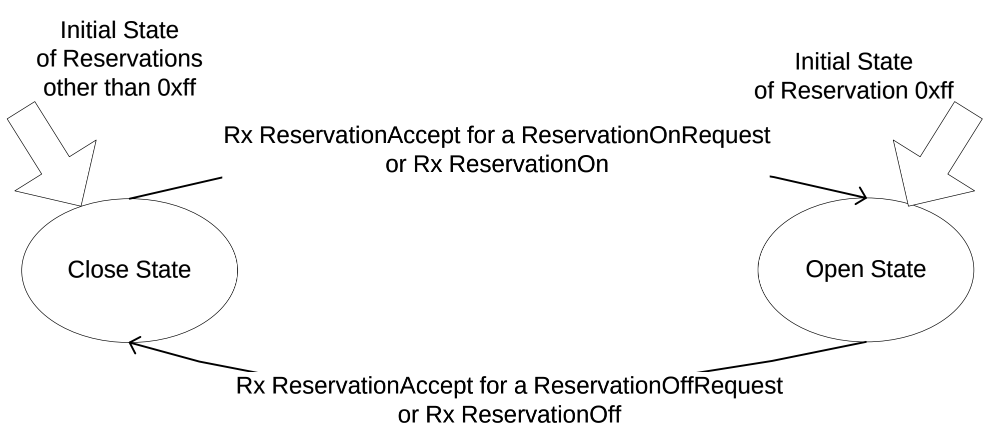

Figure 35 Reservation state diagram (access terminal)

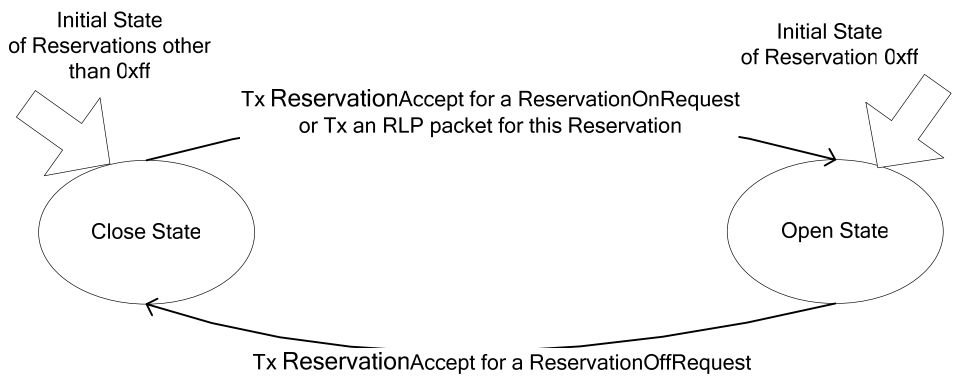

Figure 36 Reservation state diagram (access network)

3.3.4.4.2.1.3.1 State independent requirements

3.3.4.4.2.1.3.1.1 Access terminal requirements

Upon receiving a RevReservationOn message, the access terminal shall:

- Respond with a ReservationAccept message within the time period specified by $T_{RLPResponse}$ of receiving the RevReservationOn message.
- Set the TransactionID field of the ReservationAccept message to that of the RevReservationOn message.

Upon receiving a RevReservationOff message, the access terminal shall:

- Respond with a ReservationAccept message within the time period specified by $T_{RLPResponse}$ of receiving the RevReservationOff message.
- Set the TransactionID field of the ReservationAccept message to that of the RevReservationOff message.

Upon receiving a FwdReservationOn message, the access terminal shall:

- Respond with a FwdReservationAck message within the time period specified by $T_{RLPResponse}$ of reception of the FwdReservationOn message.
- Set the TransactionID field of the FwdReservationAck message to that of the FwdReservationOn message.

Upon receiving a FwdReservationOff message, the access terminal shall

- Respond with a FwdReservationAck message within the time period specified by $T_{RLPResponse}$ of receiving the FwdReservationOff message.
- Set the TransactionID field of the FwdReservationAck message to that of the FwdReservationOff message.

3.3.4.4.2.1.3.1.2 Access network requirements

The access network may re-send a FwdReservationOn message if it does not receive a FwdReservationAck message containing the same TransactionID within the time period specified by $T_{RLPResponse}$ of sending the FwdReservationOn message.

The access network may re-send a FwdReservationOff message if it does not receive a FwdReservationAck message containing the same TransactionID within the time period specified by $T_{RLPResponse}$ of sending the FwdReservationOff message.

The access network may send a RevReservationOn message to transition the state of the reverse link Reservation of the access terminal to the Open state. The access network may re-send a RevReservationOn message if it does not receive a ReservationAccept message containing the same TransactionID within the time period specified by $T_{RLPResponse}$ of sending the RevReservationOn message.

The access network may send a RevReservationOff message to transition the state of the reverse link Reservation of the access terminal to the Close state. The access network may re-send a RevReservationOff message if it does not receive a ReservationAccept message containing the same TransactionID within the time period specified by $T_{RLPResponse}$ of sending the RevReservationOff message.

If the access network receives a ReservationOnRequest message, it shall:

- Send either a ReservationAccept message or a ReservationReject message within the time period specified by $T_{RLPResponse}$ of reception of the ReservationOnRequest message.
- Set the TransactionID field of the ReservationAccept or ReservationReject message to that of the ReservationOnRequest message.

If the access network receives a ReservationOffRequest message, it shall:

- Send a ReservationAccept or a ReservationReject message within the time period specified by $T_{RLPResponse}$ of reception of the ReservationOffRequest message.
- Set the TransactionID field of the ReservationAccept or ReservationReject message to that of the ReservationOffRequest message.

3.3.4.4.2.1.3.2 Close state

3.3.4.4.2.1.3.2.1 Access terminal requirements

The access terminal shall not transmit PDUs from higher layer flows belonging to this Reservation using any Link Flow other than the Link Flow associated with ReservationLabel 0xff.

The access terminal may send a ReservationOnRequest message to request transition of the Reservation to the Open state[12]. The access terminal may re-send a ReservationOnRequest message if it does not receive a corresponding ReservationAccept or ReservationReject message within the time

[12] Note that the ReservationOnRequest message supports requests for multiple Reservations on both the forward and reverse links. This arrangement allows requests for groups of Reservations (e.g., for bidirectional higher layer application flows) to be combined in the same ReservationOnRequest message.

period specified by $T_{RLPResponse}$ of sending the ReservationOnRequest message. If the ReservationOnRequest message contains a Reservation bound to a reverse Link Flow, then the Reservation shall transition to the Open state when the access terminal receives the corresponding ReservationAccept message.

Upon receiving a RevReservationOn message, the access terminal shall transition the Reservation to the Open state. Upon receiving an *IdleState.ConnectionOpened* indication, the access terminal shall transition the Reservations to the Open State whose corresponding Reservation*KK*IdleStateRev attribute is 0x0002, where *KK* is the two-digit hexadecimal ReservationLabel in the range 0x00 to 0xff inclusive.

3.3.4.4.2.1.3.2.2 Access network requirements

If the Reservation entered this state as a result of any condition other than the following conditions, then the access network shall send a FwdReservationOff message upon entering this state:

- The access network transmitted a ReservationAccept message in response to a ReservationOffRequest message requesting to transition the Reservation to the Close state, or
- Reservation*KK*IdleStateFwd attribute of the Reservation is 0x0001 or 0x0002, and the Reservation transitioned to the Close state because the Connection was closed or lost.

Upon sending a ReservationAccept message for a Reservation Label bound to a forward Link Flow in response to a ReservationOnRequest message, the access network shall transition the Reservation to the Open state.

Upon sending a FwdReservationOn message, the access network shall transition the Reservation to the Open state. Upon receiving an *IdleState.ConnectionOpened* indication, the access network shall transition the Reservations to the Open state whose corresponding Reservation*KK*IdleStateFwd attribute is 0x0002, where *KK* is the two-digit hexadecimal ReservationLabel in the range 0x00 to 0xff inclusive.

3.3.4.4.2.1.3.3 Open state

3.3.4.4.2.1.3.3.1 Access terminal requirements

The access terminal may transmit PDUs from higher layer flows belonging to this Reservation using the Link Flow to which the Reservation is bound.

The access terminal may send a ReservationOffRequest message to request the transition of a Reservation to the Close state. The access terminal may re-send a ReservationOffRequest message if it does not receive a ReservationAccept or ReservationReject message within the time period specified by $T_{RLPResponse}$ of sending the ReservationOffRequest message. If the ReservationOffRequest message contains a Reservation bound to a reverse Link Flow, then the access terminal shall transition the Reservation to the Close state when the access terminal receives a ReservationAccept message.

Upon receiving a RevReservationOff message, the access terminal shall transition the Reservation to the Close state.

Upon receiving a *ConnectedState.ConnectionClosed* or *ActiveSetManagement.ConnectionLost* indication, the access terminal shall transition to the Close state Reservations whose corresponding Reservation*KK*IdleStateRev attribute is 0x0001 or 0x0002, where *KK* is the two-digit hexadecimal ReservationLabel.

3.3.4.4.2.1.3.3.2 Access network requirements

The access network may transmit PDUs from higher layer flows belonging to this Reservation using the Link Flow to which the Reservation is bound.

Upon sending a ReservationAccept message for a ReservationLabel bound to a forward Link Flow in response to a ReservationOffRequest message, the access network shall transition the Reservation to the Close state.

Upon receiving a *ConnectedState.ConnectionClosed* or *ActiveSetManagement.ConnectionLost* indication, the access network shall transition to the Close state Reservations whose corresponding Reservation*KK*IdleStateFwd attribute is 0x0001 or 0x0002, where *KK* is the two-digit hexadecimal ReservationLabel.

If, for any *KK*, all of the following conditions are true, the access network shall take action within $T_{Turnaround}$, where $T_{Turnaround}$ is equal to 2 seconds, such that at least one of the following conditions would no longer be true (e.g., by modifying the value of Reservation*KK*QoSUsedFwd or by transitioning forward Reservation *KK* to the Close state):

- Reservation*KK*QoSListFwd is set to a non-default value.
- Forward Reservation *KK* is in the Open state.
- Reservation*KK*QoSUsedFwd is set to the default value, or the QoSAttributeSet_ID field in Reservation*KK*QoSUsedFwd is not equal to the value of any QoSAttributeSet_ID field in the corresponding Reservation*KK*QoSListFwd attribute.

If, for any *KK*, all of the following conditions are true, the access network shall take action within $T_{Turnaround}$, where $T_{Turnaround}$ is equal to 2 seconds, such that at least one of the following conditions would no longer be true (e.g., by modifying the value of Reservation*KK*QoSUsedRev or by transitioning reverse Reservation *KK* to the Close state):

- Reservation*KK*QoSListRev is set to a non-default value.
- Reverse Reservation *KK* is in the Open state.
- Reservation*KK*QoSUsedRev is set to the default value or the QoSAttributeSet_ID field in Reservation*KK*QoSUsedRev is not equal to the value of any QoSAttributeSet_ID field in the corresponding Reservation*KK*QoSListRev attribute.

3.3.4.4.2.2 RLP receive procedures

If SecurityEnabled public data of the Security Protocol is set to ‘1’, then the RLP receiver shall discard any data unit received for which the IsSecure field of the Lower MAC header is set to ‘0’.

If Flow*NN*SequenceLengthFwd is 0x0000, then the access network will follow the procedures in 3.3.4.4.2.2.1 when receiving an RLP packet. If Flow*NN*SequenceLengthFwd is not 0x0000, then the access network will follow the procedures in 3.3.4.4.2.2.2 when receiving an RLP packet.

If Flow*NN*SequenceLengthRev is 0x0000, then the access terminal will follow the procedures in 3.3.4.4.2.2.1 when receiving an RLP packet. If Flow*NN*SequenceLengthRev is not 0x0000, then the access terminal will follow the procedures in 3.3.4.4.2.2.2 when receiving an RLP packet.

3.3.4.4.2.2.1 Receive procedures for flows with a SequenceLength of zero

If the Flow*NN*SequenceLengthFwd or the Flow*NN*SequenceLengthRev is 0x0000, then the RLP receiver shall perform the following:

- If the First and Last fields of the RLP header are '1', the RLP receiver shall pass the complete Route Protocol packet to the Route Protocol layer.
- Otherwise, the RLP receiver shall discard the packet.

3.3.4.4.2.2.2 Receive procedures for flows with a non-zero SequenceLength

The RLP receiver shall maintain two SequenceLength-bit variables for receiving, $V(R)_{NN,P}$ and $V(N)_{NN,P}$ (see Figure 37), where *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive, and *P* is the Route indicator that takes values of either A or B. $V(R)_{NN,P}$ contains the sequence number of the next data unit expected to arrive. $V(N)_{NN,P}$ contains the sequence number of the first missing data unit, as described below.

In addition, the RLP receiver shall keep track of the status of each data unit in its resequencing buffer indicating whether the data unit was received or not. Use of this status is implied in the following procedures. The RLP receiver informs the RLP transmitter of the status of data units in its receive buffer by sending a ReceiverStatus message. The ReceiverStatus message shall convey status of all missing data from $V(N)_{NN,P}$ onwards that has not been conveyed in a previous ReceiverStatus message and $V(R)_{NN,P}$. The ReceiverStatus message may convey status of missing data that has been conveyed in previous ReceiverStatus messages. The ReceiverStatus message shall not convey status of data units with a sequence number less than $V(N)_{NN,P}$.

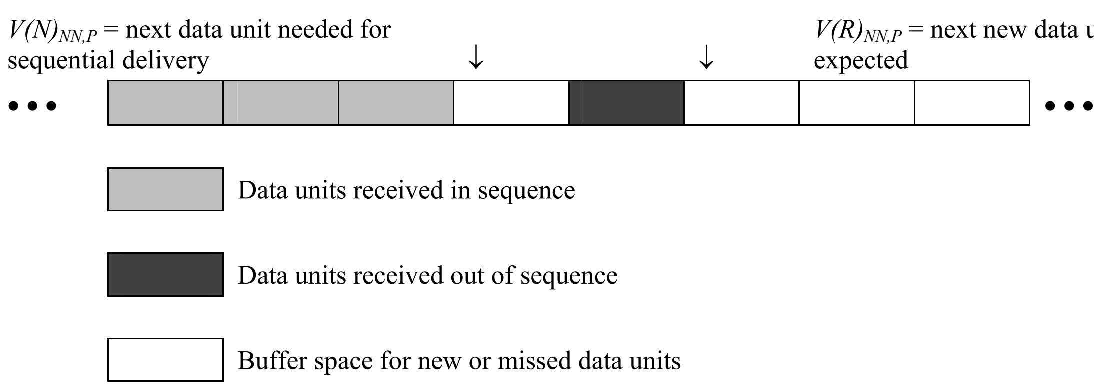

Figure 37 RLP receive sequence number variables

In the following, *X* denotes the sequence number of a received data unit. For each received data unit, RLP shall perform the following procedures in order:

- If the RLP receiver is an access terminal and Flow*NN*AckNakEnableFwd is 0x0002, then RLP shall send a ReceiverStatus message within AckTimer interval of receiving the data unit.

- If the RLP receiver is an access network and Flow*NN*AckNakEnableRev is 0x0002, then RLP shall send a ReceiverStatus message within AckTimer interval of receiving the data unit.
- If the RLP receiver is an access terminal and Flow*NN*AckNakEnableFwd is 0x0003 and the Last field of the RLP header is '0', then RLP shall send a ReceiverStatus message within AckTimer interval of receiving the data unit.
- If the RLP receiver is an access network and Flow*NN*AckNakEnableRev is 0x0003 and the Last field of the RLP header is '0', then RLP shall send a ReceiverStatus message within AckTimer interval of receiving the data unit.
- If the RLP receiver is an access terminal and Flow*NN*AckNakEnableFwd is 0x0003 and the Last field of the RLP header is '1', then RLP shall send a ReceiverStatus message upon receiving the data unit.
- If the RLP receiver is an access network and Flow*NN*AckNakEnableRev is 0x0003 and the Last field of the RLP header is '1', then RLP shall send a ReceiverStatus message upon receiving the data unit.
- If $X < V(N)_{NN,P}$, the data unit shall be discarded as a duplicate.
- If $V(N)_{NN,P} \leq X < V(R)_{NN,P}$, and the data unit is not already stored in the resequencing buffer, then:
  - RLP shall store the received data unit in the resequencing buffer.
  - If $X = V(N)_{NN,P}$ and if in-order delivery of Route Protocol packets is required, then RLP shall pass all contiguous complete Route Protocol packets in the resequencing buffer that have not been passed to the Route Protocol, from the beginning of the resequencing buffer upward to the Route Protocol. RLP shall then set $V(N)_{NN,P}$ to (LAST+1) where LAST is the sequence number of the last contiguous data unit in the resequencing buffer.
  - If $X = V(N)_{NN,P}$ and if in-order delivery of Route Protocol packets is not required, then RLP shall pass all complete Route Protocol packets in the resequencing buffer that have not been passed to the Route Protocol layer, from the beginning of the resequencing buffer upward to the Route Protocol. RLP shall then set $V(N)_{NN,P}$ to (LAST+1) where LAST is the sequence number of the last contiguous data unit in the resequencing buffer.
- If $V(N)_{NN,P} < X < V(R)_{NN,P}$ and the data unit has already been stored in the resequencing buffer, then the data unit shall be discarded as a duplicate.
- If $X = V(R)_{NN,P}$, then:
  - If $V(R)_{NN,P} = V(N)_{NN,P}$, then RLP shall increment $V(N)_{NN,P}$ and $V(R)_{NN,P}$ and shall pass all complete Route Protocol packets in the resequencing buffer that have not been passed to the Route Protocol, from the beginning of the resequencing buffer upward to the Route Protocol.
  - If $V(R)_{NN,P} \neq V(N)_{NN,P}$, then RLP shall increment $V(R)_{NN,P}$ and shall store the data unit in the resequencing buffer. If in-order delivery of Route Protocol packets is not required, then RLP shall pass all complete Route Protocol packets in the resequencing buffer that have not been passed to the Route Protocol, from the beginning of the resequencing buffer upward to the Route Protocol.

- If $X > V(R)_{NN,P}$, then:
  - RLP shall store the data unit in the resequencing buffer.
  - If in-order delivery of Route Protocol packets is not required, then RLP shall pass all complete Route Protocol packets in the resequencing buffer that have not been passed to the Route Protocol, from the beginning of the resequencing buffer upward to the Route Protocol.
  - If the RLP receiver is an access network, then RLP shall set an RLP abort timer to AbortTimer, where AbortTimer is a parameter of the Flow*NN*TimersRev attribute, for each missing RLP data unit from $V(R)_{NN,P}$ to X-1, inclusive. If the RLP receiver is an access terminal, then RLP shall set an RLP abort timer to AbortTimer where AbortTimer is a parameter of the Flow*NN*TimersFwd attribute, for each missing RLP data unit from $V(R)_{NN,P}$ to X-1, inclusive.
  - RLP shall set $V(R)_{NN,P}$ to X+1.
  - If the RLP receiver is an access terminal and if the Flow*NN*AckNakEnableFwd attribute is not 0x0000, then RLP shall send a ReceiverStatus message. If the RLP receiver is an access network and if the Flow*NN*AckNakEnableRev attribute is not 0x0000, then RLP shall send a ReceiverStatus message.

If a missing data unit has not arrived when its RLP abort timer expires and if in-order delivery of Route Protocol packets is required, then RLP shall pass all complete Route Protocol packets that have not been passed to the Route Protocol, from the beginning of the resequencing buffer upward up to the next missing data unit to the Route Protocol. RLP may pass to the Route Protocol partially received packets with an indication of partial packet delivery.

If the RLP receiver is the access network and if Flow*NN*AckNakEnableRev is not 0x0000, then the access network may determine that a ReceiverStatus message or the retransmitted data units have been lost if it does not receive the data units within an implementation-dependent time interval based on an estimate of the round-trip delay and if other packets were received from the access terminal. If the RLP receiver is the access network and if Flow*NN*AckNakEnableRev is not 0x0000 and the access network determines that a ReceiverStatus message or the retransmitted data units were lost, and the abort timer for the retransmitted data units has not expired, then the access network shall transmit a ReceiverStatus message.

If the RLP receiver is the access terminal and if Flow*NN*AckNakEnableFwd is not 0x0000, then the access terminal may determine that a ReceiverStatus message or the retransmitted data units have been lost if it does not receive the data units within an implementation-dependent time interval based on an estimate of the round-trip delay and if other packets were received from the access network. If the RLP receiver is the access terminal and if Flow*NN*AckNakEnableFwd is not 0x0000 and the access terminal determines that a ReceiverStatus message or the retransmitted data units were lost, and the abort timer for the retransmitted data units has not expired, then the access terminal shall transmit a ReceiverStatus message.

RLP shall set $V(N)_{NN,P}$ to the sequence number of the next missing data unit, or to $V(R)_{NN}$ if there are no remaining missing data units. Further recovery is the responsibility of higher layer protocols.

3.3.4.4.3 In-band message transfer

The access network shall send the in-band messages in 3.3.4.6 on forward Link Flow $N_{LinkFlowMax}$-1. The access network shall not send an in-band message on forward Link Flows 0x00 to $N_{LinkFlowMax}$-2.

The access terminal shall send the in-band messages in 3.3.4.6 on reverse link Flow $N_{LinkFlowMax}$-1. The access terminal shall not send an in-band message on reverse Link Flows 0x00 to $N_{LinkFlowMax}$-2.

The access network and access terminal shall not send an in-band message using the Signaling Transport. The access network and access terminal shall discard an in-band message received from the Signaling Transport.

All in-band messages shall apply only to the instance of the Default Data Transport sending and receiving the message. All in-band messages shall apply only to the Route that the message is sent and received on.

3.3.4.4.4 RLP packet header

The RLP packet header, which precedes the RLP payload, has the following format:

Field	Length (bits)
LinkFlowNumber	4
Route	1
SequenceSpace	1
First	1
Last	1
SEQ	0, 8, 16, or 24

LinkFlowNumber — The identifier for this Link Flow.

Route — If this RLP packet is sent on Route A, then the sender shall set this field to ‘0’. Otherwise, the sender shall set this field to ‘1’.

SequenceSpace — The sender shall set this flag to the value of the sequence space variable $Q_{NN,Tx}$.

First — If the payload of this RLP packet is the first segment of a Route Protocol packet, then the sender shall set this field to ‘1’. Otherwise, the sender shall set this field to ‘0’.

Last — If the payload of this RLP packet is the last segment of a Route Protocol packet, then the sender shall set this field to ‘1’. Otherwise, the sender shall set this field to ‘0’.

SEQ — The RLP sequence number of the first data unit in the RLP payload[13]. If this RLP packet is being sent on the forward link, the length of this field is indicated by the Flow*NN*SequenceLengthFwd attribute corresponding to this flow. If this RLP packet is being sent on the reverse link, the length of this field is indicated by the Flow*NN*SequenceLengthRev attribute corresponding to this flow.

3.3.4.5 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes.

3.3.4.5.1 ReservationOnRequest

The access terminal sends this message to request transition of one or more Reservations to the Open State.

Field	Length (bits)
MessageID	8
TransactionID	8
ReservationCount	8
ReservationCount occurrences of the following three fields:	
{	
Reserved	7
Link	1
ReservationLabel	8
}	

MessageID — The access terminal shall set this field to 0x04.

TransactionID — The access terminal shall set this field according to 10.8 for each ReservationOnRequest or ReservationOffRequest message sent.

ReservationCount — The access terminal shall set this field to the number of ReservationLabel fields in this message. The access terminal shall include ReservationCount occurrences of the following three fields with the message.

Reserved — The access terminal shall set this field to '0000000'. The access network shall ignore this field.

Link — If this request is for a forward Reservation, then the access terminal shall set this field to '1'. If this request is for a reverse Reservation, then the access terminal shall set this field to '0'.

[13] When data unit is set to RLP payload, the RLP packet contains one data unit.

ReservationLabel — The access terminal shall set this field to the Reservation for which this request is generated.

Channels	RTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

3.3.4.5.2 ReservationOffRequest

The access terminal sends this message to request transition of one or more Reservations to the Close State.

Field	Length (bits)
MessageID	8
TransactionID	8
ReservationCount	8
ReservationCount occurrences of the following three fields:	
{	
Reserved	7
Link	1
ReservationLabel	8
}	

MessageID — The access terminal shall set this field to 0x05.

TransactionID — The access terminal shall set this field according to 10.8 for each ReservationOnRequest or ReservationOffRequest message sent.

ReservationCount — The access terminal shall set this field to the number of ReservationLabel fields in this message. The sender shall include ReservationCount occurrences of the following three fields with the message.

Reserved — The access terminal shall set this field to ‘0000000’. The access network shall ignore this field.

Link — If this request is for a forward Reservation, then the access terminal shall set this field to ‘1’. If this request is for a reverse Reservation, then the access terminal shall set this field to ‘0’.

ReservationLabel — The access terminal shall set this field to the Reservation for which this request is generated.

Channels	RTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

3.3.4.5.3 ReservationAccept

The access network sends this message to acknowledge reception of and allow the state transition requested by a ReservationOnRequest or ReservationOffRequest message. The access terminal sends this message to acknowledge reception of and accept the state transition requested by a RevReservationOn or RevReservationOff message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID The sender shall set this field to 0x06.

TransactionID The access network shall set this field to the TransactionID field of the ReservationOnRequest or ReservationOffRequest message to which the access network is responding. The access terminal shall set this field to the TransactionID field of the RevReservationOn or RevReservationOff message to which the access terminal is responding.

Channels	FTC RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.3.4.5.4 ReservationReject

The access network sends this message to acknowledge reception of and deny the state transition requested by a ReservationOnRequest or ReservationOffRequest message.

Field	Length (bits)
MessageID	8
TransactionID	8
ReservationCount	8
ReservationCount occurrences of the following three fields:	
{	
Reserved	7
AllowableLink	1
AllowableReservationLabel	8
}	

MessageID The access network shall set this field to 0x07.

TransactionID The access network shall set this field to the TransactionID field of the ReservationOnRequest or ReservationOffRequest message to which the access network is responding.

ReservationCount — The access network shall set this field to the number of ReservationLabel fields in this message. The sender shall include ReservationCount occurrences of the following three fields with the message.

Reserved — The access network shall set this field to ‘0000000’. The access terminal shall ignore this field.

AllowableLink — If the Reservation for which the access network would have allowed the state transition requested in the ReservationOnRequest or ReservationOffRequest message is a forward Reservation, then the access network shall set this field to ‘1’. If the Reservation for which the access network would have allowed the state transition requested in the ReservationOnRequest or ReservationOffRequest message is a reverse Reservation, then the access network shall set this field to ‘0’.

AllowableReservationLabel — The access network shall set this field to the Reservation for which the access network would have allowed the state transition requested in the ReservationOnRequest or ReservationOffRequest message.

Channels	FTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.3.4.5.5 RevReservationOn

The access network sends this message to transition an activated reverse Reservation to the Open state.

Field	Length (bits)
MessageID	8
TransactionID	8
ReservationCount	8
ReservationCount occurrences of the following field:	
{	
ReservationLabel	8
}	

MessageID — The access network shall set this field to 0x08.

TransactionID — The access network shall set this field according to 10.8 for each RevReservationOn or RevReservationOff message sent..

ReservationCount — The access network shall set this field to the number of ReservationLabel fields in this message. The sender shall include ReservationCount occurrences of the following field with the message.

ReservationLabel — The access network shall set this field to the Reservation that is to be transitioned to the Open state.

Channels	FTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

3.3.4.5.6 RevReservationOff

The access network sends this message to transition an activated reverse link Reservation to the Close state.

Field	Length (bits)
MessageID	8
TransactionID	8
ReservationCount	8
ReservationCount occurrences of the following field:	
{	
ReservationLabel	8
}	

MessageID — The access network shall set this field to 0x09.

TransactionID — The access network shall set this field according to 10.8 for each RevReservationOn or RevReservationOff message sent. If this is the first RevReservationOn or RevReservationOff message sent by the access network, then the access network shall set this field to zero.

ReservationCount — The access network shall set this field to the number of Reservation fields in this message. The sender shall include ReservationCount occurrences of the following field with the message.

ReservationLabel — The access network shall set this field to the Reservation which is to be transitioned to the Close state.

Channels	FTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

3.3.4.5.7 FwdReservationOn

The access network sends this message to inform the access terminal when a forward Reservation transitions to the Open state.

Field	Length (bits)
MessageID	8
TransactionID	8
ReservationCount	8
ReservationCount occurrences of the following field:	
{	
ReservationLabel	8
}	

MessageID — The access network shall set this field to 0x0a.

TransactionID — The access network shall set this field according to 10.8 for each FwdReservationOn or FwdReservationOff message sent.

ReservationCount — The access network shall set this field to the number of ReservationLabel fields in this message. The sender shall include ReservationCount occurrences of the following field with the message.

ReservationLabel — The access network shall set this field to the Reservation which is to be transitioned to the Open state.

Channels	FTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.3.4.5.8 FwdReservationOff

The access network sends this message to inform the access terminal when a forward Reservation transitions to the Close state.

Field	Length (bits)
MessageID	8
TransactionID	8
ReservationCount	8
ReservationCount occurrences of the following field:	
{	
ReservationLabel	8
}	

MessageID	The access network shall set this field to 0x0b.
TransactionID	The access network shall set this field according to 10.8 for each FwdReservationOn or FwdReservationOff message sent.
ReservationCount	The access network shall set this field to the number of ReservationLabel fields in this message. The sender shall include ReservationCount occurrences of the following field with the message.
ReservationLabel	The access network shall set this field to the Reservation which is to be transitioned to the Close state.

Channels	FTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.3.4.5.9 FwdReservationAck

The access terminal sends this message to acknowledge reception of the FwdReservationOn or the FwdReservationOff message and to accept the related state transition.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID	The access network shall set this field to 0x0c.
TransactionID	The access terminal shall set this field to the TransactionID field of the FwdReservationOn or FwdReservationOff message to which the access network is responding.

Channels	RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.3.4.5.10 FlowQoSDetect

The access network sends this message to inform the access terminal that it should add a new Reservation or modify the Reservation*KK*QoSListFwd or Reservation*KK*QoSListRev attribute for an existing Reservation *KK*. The access network may include a packet filter specification in the message.

Field	Length (bits)
MessageID	8
Reserved1	6
Link	1
ReservationIncluded	1
ReservationLabel	0 or 8
ReservationPriority	8
QoSAttributeSetCount	8
QoSAttributeSetCount occurrences of the following record:	
{	
QoSAttributeSetLength	8
QoSAttributeSet	QoSAttributeSetLength ×8
}	
FilterSpecCount	8
FilterSpecCount occurrences of the following record:	
{	
FilterSpecType	8
FilterSpecLength	8
FilterSpec	FilterSpecLength × 8
}	

MessageID — The access network shall set this field to 0x0d.

Reserved1 — The access network shall set this field to ‘000000’. The access terminal shall ignore this field.

Link — If this message is for a forward Reservation, then the access network shall set this field to ‘1’. If this message is for a reverse Reservation, then the access network shall set this field to ‘0’.

ReservationIncluded — The access network shall set this field to ‘1’ to modify an existing Reservation. The access network shall set this field to ‘0’ to add a new Reservation.

ReservationLabel — If ReservationIncluded is ‘0’, then the access network shall omit this field. Otherwise, the access network shall set this field to the Reservation for which this message is generated.

ReservationPriority	The access network shall set this field to indicate the priority to be assigned to the reservation. The value 0x00 indicates the highest priority and the value 0xff indicates the lowest priority.
QoSAttributeSetCount	The access network shall set this field to the number of QoSAttributeSets associated with this reservation. Each QoSAttributeSet contains one set of acceptable QoS parameters. If multiple QoS attribute sets are included, the sender shall include the QoS attribute sets in descending order of preference. The sender shall include QoSAttributeSetCount occurrences of the following two fields with the message.
QoSAttributeSetLength	The access network shall set this field to the length, in octets, of the QoSAttributeSet.
QoSAttributeSet	The QoS parameters requested for the reservation. The access network shall set this record as defined in 3.3.6.2.9.1.
FilterSpecCount	The access network shall set this field to the number of FilterSpecs associated with this reservation. The sender shall include FilterSpecCount occurrences of the following three fields with the message.
FilterSpecType	The access network shall set this field to an identifier for the Filter Specification Type according to Table 13.
FilterSpecLength	The access network shall set this field to the length of the FilterSpec field in units of octets.
FilterSpec	If FilterSpecType is 0x01, then the access network shall set this record as defined in 3.3.6.2.13.1. If FilterSpecType is 0x02, then the sender shall set this record as defined in 3.3.6.2.13.2. Otherwise, the sender shall omit this record.

Channels	FTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.3.4.6 In-band message formats

3.3.4.6.1 ResetRxRequest

The RLP receiver in the access terminal or the access network sends the ResetRxRequest message to reset its peer RLP transmitter.

Field	Length (bits)
MessageID	8
TransactionID	8
ResetAllFlows	1
LinkFlowNumber	4
SequenceSpace	1
Reserved	2

MessageID — The sender shall set this field to 0xf0.

TransactionID — The sender shall set this field according to 10.8.

ResetAllFlows — The sender shall set this field to '1' to reset all Link Flows. Otherwise the sender shall set this field to '0'.

LinkFlowNumber — The sender shall set this field to the Link Flow that is reset. The sender shall set this field to '0000' if the ResetAllFlows field is set to '1'. The receiver shall ignore this field if the ResetAllFlows field is set to '1'.

SequenceSpace — The sender shall set this flag to the value of the sequence space variable $Q_{NN,Rx}$. The sender shall set this field to '0' if the ResetAllFlows field is set to '1'. The receiver shall ignore this field if the ResetAllFlows field is set to '1'.

Reserved — The sender shall set this field to '00'. The receiver shall ignore this field.

3.3.4.6.2 ResetRxAck

The RLP transmitter in the access terminal or the access network sends the ResetRxAck message in response to the ResetRxRequest message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The sender shall set this field to 0xf1.

TransactionID — The sender shall set this field to the TransactionID of the associated ResetRxRequest message.

3.3.4.6.3 ResetTxIndication

The RLP transmitter in the access terminal or the access network sends the ResetTxIndication message to reset its peer RLP receiver.

Field	Length (bits)
MessageID	8
TransactionID	8
ResetAllFlows	1
LinkFlowNumber	4
SequenceSpace	1
Reserved	2

MessageID — The sender shall set this field to 0xf2.

TransactionID — The sender shall set this field according to 10.8.

ResetAllFlows — The sender shall set this field to '1' to reset all Link Flows. Otherwise the sender shall set this field to '0'.

LinkFlowNumber — The sender shall set this field to the Link Flow that is reset. The sender shall set this field to '0000' if the ResetAllFlows field is set to '1'. The receiver shall ignore this field if the ResetAllFlows field is set to '1'.

SequenceSpace — The sender shall set this flag to the value of the sequence space variable $Q_{NN,Tx}$. The sender shall set this field to '0' if the ResetAllFlows field is set to '1'. The receiver shall ignore this field if the ResetAllFlows field is set to '1'.

Reserved — The sender shall set this field to '00'. The receiver shall ignore this field.

3.3.4.6.4 ResetTxAck

The RLP receiver in the access terminal or the access network sends the ResetTxAck message in response to the ResetTxIndication message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The sender shall set this field to 0xf3.

TransactionID — The sender shall set this field to the TransactionID of the associated ResetTxIndication message.

3.3.4.6.5 ReceiverStatus

The access terminal and the access network send the ReceiverStatus message to acknowledge the receipt of one or more RLP data units or to request the retransmission of one or more RLP data units.

Field	Length (bits)
MessageID	8
LinkFlowNumber	4
SequenceSpace	1
LatestDataUnit	1
Reserved0	1
TimeStampIncluded	1
Reserved1	0 or 4
TimeStamp	0 or 12
SequenceLength	8
ReportCount	8
ReportCount occurrences of the following record:	
{	
FirstErasedDataUnit	SequenceLength
WindowLen	SequenceLength
}	
VR	SequenceLength

MessageID — The sender shall set this field to 0xf4.

LinkFlowNumber — The sender shall set this field to the Link Flow for which this ReceiverStatus is being sent.

SequenceSpace — The sender shall set this flag to the value of the sequence space variable $Q_{NN,Rx}$.

Reserved0 — The sender shall set this field to ‘0’. The receiver shall ignore this field.

LatestDataUnit — If the latest data unit in the receive buffer is the data unit with sequence number V(R) − 1, then the sender shall set this field to ‘1’. Otherwise, the sender shall set this field to ‘0’.

TimeStampIncluded — If the value of the Flow*NN*AckNakEnableFwd attribute is 0x0002 or 0x0003, then the access terminal shall set this field to ‘1’. Otherwise, the access terminal shall set this field to ‘0’. If the value of the Flow*NN*AckNakEnableRev attribute is 0x0002 or 0x0003, then the access network shall set this field to ‘1’. Otherwise, the access network shall set this field to ‘0’. *NN* is the two-digit hexadecimal Link Flow number.

Reserved1 — If TimeStampIncluded is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to ‘0000’. The receiver shall ignore this field.

TimeStamp	If TimeStampIncluded is '0', then the sender shall omit this field. Otherwise, the sender shall set this field to the 12 least significant bits of the system time, in units of frames, when the latest data unit in the receive buffer was received.
SequenceLength	The sender shall set this field to the length of the sequence number as indicated by the Flow*NN*SequenceLengthFwd or Flow*NN*SequenceLengthRev attribute for forward or reverse Link Flow *NN*, respectively in units of bits.
ReportCount	The sender shall set this field to the number of Report records included in this message. The sender shall include ReportCount occurrences of the following two fields with the message.
FirstErasedDataUnit	The sender shall set this field to the sequence number of the first RLP data unit erased in a sequence of erased data units.
WindowLen	The sender shall set this field to the length of the erased window in data units.
VR	The sender shall set this field to $V(R)_{NN,P.}$

3.3.4.7 Interface to other protocols

3.3.4.7.1 Commands

This protocol issues the following command:

- *AirLinkManagement.OpenConnection*

3.3.4.7.2 Indications

This protocol registers to receive the following indications:

- *IdleState.ConnectionOpened*
- *ReverseTrafficChannelMAC.ReverseTrafficPacketsMissed* along with parameters indicating the Link Flow number and missing octets.
- *ForwardTrafficChannelMAC.ForwardTrafficPacketsMissed* along with parameters indicating the Link Flow number and missing octets.
- *ConnectedState.ConnectionClosed*
- *ActiveSetManagement.ConnectionLost*

3.3.4.8 RLP packet priorities

For a given Link Flow, the sender shall assign higher priority to packets containing retransmitted transport traffic than packets containing only first time transmissions. If Flow*NN*TransmitAbortTimerRev is not set to 0x0000, then the access terminal should transmit a higher layer data unit within Flow*NN*TransmitAbortTimerRev time of the higher layer data unit being received. The access terminal may use the Flow*NN*TransmitAbortTimerRev attribute to determine the priority of reverse RLP packets. If Flow*NN*TransmitAbortTimerFwd is not set to 0x0000, then the access network should transmit a higher layer data unit within Flow*NN*TransmitAbortTimerFwd time of the higher layer data unit being received. The access network may use the Flow*NN*TransmitAbortTimerFwd attribute to determine the priority of forward RLP packets.

3.3.4.9 Protocol numeric constants

Constant	Meaning	Value
$N_{LinkFlowMax}$	Maximum total number of activated and deactivated Link Flows.	16
$T_{RLPResponse}$	Time period within which the access network is to respond to ReservationOnRequest, ReservationOffRequest, ResetTxIndication and ResetRxRequest messages. Time period within which the access terminal is to respond to ResetTxIndication and ResetRxRequest messages.	1 second

3.3.5 Flow Control Protocol

3.3.5.1 Overview

The Flow Control Protocol provides procedures and messages used by the access terminal and the access network to perform flow control for the forward link of the Default Data Transport.

This protocol can be in one of the following states:

- Close State: in this state, the Default Data Transport for the access network does not send any RLP packets.
- Open State: in this state, the Default Data Transport for the access network can send RLP packets.

Figure 38 and Figure 39 show the state transition diagram at the access terminal and the access network.

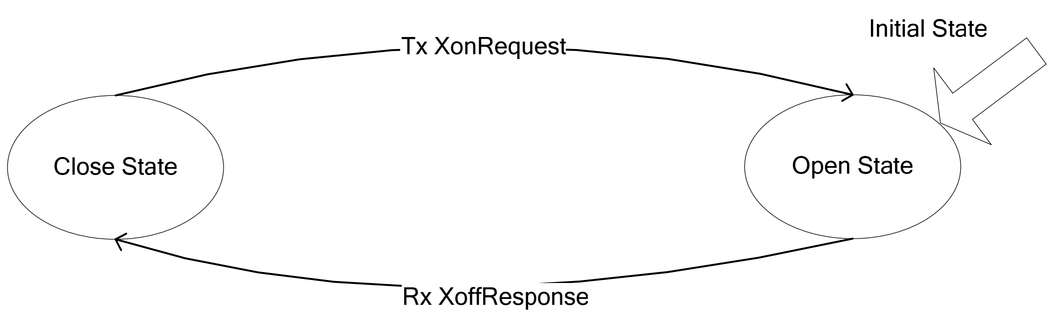

Figure 38 Flow control protocol state diagram (access terminal)

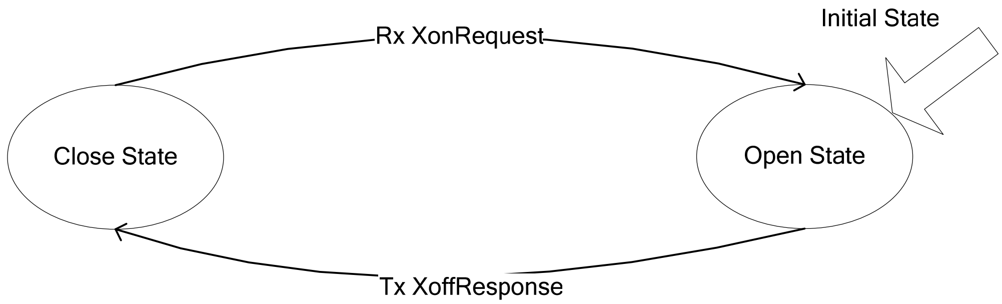

Figure 39 Flow control protocol state diagram (access network)

The flow control protocol is a protocol associated with the Default Data Transport.

3.3.5.2 Primitives

3.3.5.2.1 Commands

This protocol does not define any commands.

3.3.5.2.2 Return indications

This protocol does not return any indications.

3.3.5.3 Protocol data unit

The transmission unit of this protocol is a message. This is a control protocol and, therefore, it does not carry payload on behalf of other layers or protocols.

3.3.5.4 Procedures

All messages for the flow control protocol shall apply only to the instance of the Default Data Transport sending and receiving the message.

The flow control protocol makes use of the XOnRequest, XOnResponse, XOffRequest, and XOffResponse in-band messages as defined in 3.3.4.4.3.

3.3.5.4.1 Transmission and processing of RestartNetworkInterface message

The access network may send a RestartNetworkInterface message to direct the access terminal to restart the interface between the Data Transport and the higher layer.

Upon receiving a RestartNetworkInterface message, the access terminal shall send a RestartNetworkInterfaceAck message within the time period specified by $T_{FCResponse}$, and shall restart the interface between the Data Transport and the higher layer. The access terminal may also restart higher layer protocols.

3.3.5.4.2 Close state

In this state, the access network shall not send any RLP packets. In this state, the access network may send RLP in band messages.

3.3.5.4.2.1 Access terminal requirements

The access terminal shall send an XonRequest message when it is ready to receive RLP packets from the access network.

The access terminal shall transition to the Open State when it sends an XonRequest message.

3.3.5.4.2.2 Access network requirements

If the access network receives an XonRequest message, it shall:

- Send an XonResponse message within the time period specified by $T_{FCResponse}$ after reception of the XonRequest message to acknowledge reception of the message.
- Transition to the Open State.

3.3.5.4.3 Open state

In this state, the access terminal and the access network may send or receive any RLP packets.

3.3.5.4.3.1 Access terminal requirements

The access terminal may re-send an XonRequest message if it does not receive an XonResponse message or an RLP packet (corresponding to this instance of the Default Data Transport) within the time period specified by $T_{FCResponse}$ after sending the XonRequest message.

The access terminal may send an XoffRequest message to request the access network to stop sending RLP packets. The access terminal shall transition to the Close state when it receives an XoffResponse message with a TransactionID field equal to the TransactionID sent in the XoffRequest message.

The access terminal may re-send an XoffRequest message if it does not receive an XoffResponse message within the time period specified by $T_{FCResponse}$ after sending the XoffRequest message.

3.3.5.4.3.2 Access network requirements

If the access network receives an XoffRequest message, it shall

- Send an XoffResponse message within the time period specified by $T_{FCResponse}$ after reception of the XoffRequest message to acknowledge reception of the message.
- Transition to the Close State.

If the access network receives an XonRequest message, it shall send an XonResponse message within the time period specified by $T_{FCResponse}$ after reception of the XonRequest message to acknowledge reception of the message.

3.3.5.5 Message formats

3.3.5.5.1 RestartNetworkInterface

The access network sends this message to request the access terminal to restart the network interface.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID The access network shall set this field to 0x0e.

TransactionID The access network shall set this field according to 10.8.

Channels	FTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.3.5.5.2 RestartNetworkInterfaceAck

The access terminal sends this message to acknowledge reception of a RestartNetworkInterface message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID The access terminal shall set this field to 0x0f.

TransactionID The access terminal shall set this value to the value of the TransactionID field of the corresponding RestartNetworkInterface message.

Channels	RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

3.3.5.6 In-band message formats

3.3.5.6.1 XonRequest

The access terminal sends this message to request transition to the Open State.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The access terminal shall set this field to 0xf5.

TransactionID — The access terminal shall set this field according to 10.8.

3.3.5.6.2 XonResponse

The access network sends this message to acknowledge reception of the XonRequest message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The access network shall set this field to 0xf6.

TransactionID — The access network shall set this field to the value of the TransactionID field of the corresponding XonRequest message.

3.3.5.6.3 XoffRequest

The access terminal sends this message to request transition to the Close State.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The access terminal shall set this field to 0xf7.

TransactionID — The access terminal shall set this field according to 10.8.

3.3.5.6.4 XoffResponse

The access network sends this message to acknowledge reception of the XoffRequest message.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The access network shall set this field to 0xf8.

TransactionID — The access network shall set this field to the value of the TransactionID field of the corresponding XoffRequest message.

3.3.5.7 Interface to other protocols

3.3.5.7.1 Commands

This protocol does not issue any commands.

3.3.5.7.2 Indications

This protocol does not register to receive any indications.

3.3.5.8 Protocol numeric constants

Constant	Meaning	Value
$T_{FCResponse}$	Time period within which the access terminal and access network are to respond to flow control messages.	200 ms

3.3.6 Configuration attributes for the default data transport

The access terminal shall support default values of all attributes.

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Data Transport.

3.3.6.1 Simple attributes

The negotiable simple attribute for this protocol is listed in Table 11. The access terminal and the access network shall use as defaults the values in Table 11 that are listed in ***bold italics***.

Table 11 Configurable values

Attribute ID	Attribute	Values	Meaning
0xffff	MaxAbortTimer	***0x01f4***	Maximum abort timer defaults to 500 ms.
		0x0000 to 0x2710	Maximum abort timer in units of ms.
		All other values	Reserved
0xfe0f	Flow0FAckNakEnableFwd	***0x0000***	RLP receivers associated with forward Link Flow 0x0F do not transmit ReceiverStatus messages.
		All other values	Reserved.
0xfe*NN* *NN* is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*AckNakEnableFwd *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	0x0000	RLP receivers associated with forward Link Flow *NN* do not transmit ReceiverStatus messages.
		0x0001	RLP receivers associated with forward Link Flow *NN* transmit a ReceiverStatus message when missing data units are detected.
		0x0002	RLP receivers associated with forward Link Flow *NN* transmit a ReceiverStatus message when missing data units are detected. RLP receivers associated with forward Link Flow *NN* send ReceiverStatus messages within AckTimer interval of receiving a data unit.

Attribute ID	Attribute	Values	Meaning
		0x0003	RLP receivers associated with forward Link Flow *NN* transmit a ReceiverStatus message when missing data units are detected. RLP receivers associated with forward Link Flow *NN* send ReceiverStatus messages within AckTimer interval of receiving a data unit. The receivers are required to send a ReceiverStatus message immediately upon receiving an RLP packet carrying the last segment of a higher layer packet.
		All other values	Reserved.
0xfd0f	Flow0FAckNakEnableRev	***0x0000***	RLP receivers associated with reverse Link Flow 0x0F do not transmit ReceiverStatus messages.
		All other values	Reserved.
0xfd*NN* *NN* is the two-digit hexadecimal Link Flow number of the reverse Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*AckNakEnableRev *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	0x0000	RLP receivers associated with reverse Link Flow *NN* do not transmit ReceiverStatus messages.
		0x0001	RLP receivers associated with reverse Link Flow *NN* transmit a ReceiverStatus message when missing data units are detected.
		0x0002	RLP receivers associated with reverse Link Flow *NN* transmit a ReceiverStatus message when missing data units are detected. RLP receivers associated with reverse Link Flow *NN* send ReceiverStatus messages within AckTimer interval of receiving a data unit.
		0x0003	RLP receivers associated with reverse Link Flow *NN* transmit a ReceiverStatus message when missing data units are detected. RLP receivers associated with reverse Link Flow *NN* send ReceiverStatus messages within AckTimer interval of receiving a data unit. The receivers are required to send a ReceiverStatus message immediately upon receiving an RLP packet carrying the last segment of a higher layer packet.
		All other values	Reserved.

Attribute ID	Attribute	Values	Meaning
0xfc0f	Flow0FFTCMACNakEnableFwd	***0x0000***	RLP is to ignore *ForwardTrafficChannelMAC.ForwardTrafficPacketsMissed* indication.
		All other values	Reserved.
0xfc*NN* *NN* is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*FTCMACNakEnableFwd *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0001***	RLP is to retransmit data units when a *ForwardTrafficChannelMAC.ForwardTrafficPacketsMissed* indication is received.
		0x0000	RLP is to ignore *ForwardTrafficChannelMAC.ForwardTrafficPacketsMissed* indication.
		All other values	Reserved
0xfb0f	Flow0FRTCMACNakEnableRev	***0x0000***	RLP is to ignore *ReverseTrafficChannelMAC.ReverseTrafficPacketsMissed* indication.
		All other values	Reserved.
0xfb*NN* *NN* is the two-digit hexadecimal Link Flow number of the reverse Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*RTCMACNakEnableRev *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0001***	RLP is to retransmit data units when a *ReverseTrafficChannelMAC.ReverseTrafficPacketsMissed* indication is received.
		0x0000	RLP is to ignore *ReverseTrafficChannelMAC.ReverseTrafficPacketsMissed* indication.
		All other values	Reserved
0xfa*KK* *KK* is the two-digit hexadecimal ReservationLabel.	Reservation*KK*IdleStateFwd *KK* is the two-digit hexadecimal ReservationLabel.	***0x0000***	Reservation does not change states when a Connection is closed.
		0x0001	Reservation transitions to the Close state when a Connection is closed.
		0x0002	Reservation transitions to the Open state when a Connection is opened and transitions to the Close state when a Connection is closed.
		All other values	Reserved
0xf9*KK* *KK* is the two-digit hexadecimal ReservationLabel.	Reservation*KK*IdleStateRev *KK* is the two-digit hexadecimal ReservationLabel.	***0x0000***	Reservation does not change states when a Connection is closed.
		0x0001	Reservation transitions to the Close state when a Connection is closed.
		0x0002	Reservation transitions to the Open state when a Connection is opened and transitions to the Close state when a Connection is closed.
		All other values	Reserved

Attribute ID	Attribute	Values	Meaning
0xf800	Flow00ActivatedFwd	***0x0001***	Forward Link Flow 0x00 is activated.
		0x0000	Forward Link Flow 0x00 is not activated.
		All other values	Reserved
0xf80f	Flow0FActivatedFwd	***0x0001***	Forward Link Flow 0x0F is activated.
		All other values	Reserved
0xf8*NN* *NN* is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x01 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*ActivatedFwd *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0000***	Forward Link Flow *NN* is not activated.
		0x0001	Forward Link Flow *NN* is activated.
		All other values	Reserved
0xf700	Flow00ActivatedRev	***0x0001***	Reverse Link Flow 0x00 is activated.
		0x0000	Reverse Link Flow 0x00 is not activated.
		All other values	Reserved
0xf70f	Flow0FActivatedRev	***0x0001***	Reverse Link Flow 0x0F is activated.
		All other values	Reserved
0xf7*NN* *NN* is the two-digit hexadecimal Link Flow number of the reverse Link Flow in the range 0x01 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*ActivatedRev *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0000***	Reverse Link Flow *NN* is not activated.
		0x0001	Reverse Link Flow *NN* is activated.
		All other values	Reserved
0xf60f	Flow0FSequenceLengthFwd	***0x0001***	Forward Link Flow 0x0F has a 8-bit sequence number.
		All other values	Reserved
0xf6*NN* *NN* is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*SequenceLengthFwd *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	0x0000	Forward Link Flow *NN* has a 0-bit sequence number.
		0x0001	Forward Link Flow *NN* has a 8-bit sequence number.
		0x0002	Forward Link Flow *NN* has a 16-bit sequence number.
		0x0003	Forward Link Flow *NN* has a 24-bit sequence number.
		All other values	Reserved

Attribute ID	Attribute	Values	Meaning
0xf50f	Flow0FSequenceLengthRev	***0x0001***	Reverse Link Flow 0x0F has a 8-bit sequence number.
		All other values	Reserved
0xf5*NN* *NN* is the two-digit hexadecimal Link Flow number of the reverse Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*SequenceLengthRev *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	0x0000	Reverse Link Flow *NN* has a 0-bit sequence number.
		0x0001	Reverse Link Flow *NN* has a 8-bit sequence number.
		0x0002	Reverse Link Flow *NN* has a 16-bit sequence number.
		0x0003	Reverse Link Flow *NN* has a 24-bit sequence number.
		All other values	Reserved
0xf40f	Flow0FDataUnitFwd	***0x0001***	Data unit for forward Link Flow 0x0F is RLP packet payload.
		All other values	Reserved
0xf4*NN* *NN* is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*DataUnitFwd *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0000***	Data unit for forward Link Flow *NN* is octets.
		0x0001	Data unit for forward Link Flow *NN* is RLP packet payload.
		All other values	Reserved.
0xf30f	Flow0FDataUnitRev	***0x0001***	Data unit for reverse Link Flow 0x0F is RLP packet payload.
		All other values	Reserved
0xf3*NN* *NN* is the two-digit hexadecimal Link Flow number of the reverse Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*DataUnitRev *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0000***	Data unit for reverse Link Flow *NN* is octets.
		0x0001	Data unit for reverse Link Flow *NN* is RLP packet payload.
		All other values	Reserved.

Attribute ID	Attribute	Values	Meaning
0xf2*NN* *NN* is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive.	Flow*NN*SimultaneousDelivery OnBothRoutesFwd *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0000***	Forward Link Flow *NN* delivers Flow Protocol payload in-order.
		0x0001	Forward Link Flow *NN* may deliver Flow Protocol payload out-of-order.
		All other values	Reserved.
0xf10f	Flow0FOutOfOrderDeliveryT oRouteProtocolFwd	***0x0001***	Each Route of forward Link Flow 0x0F may deliver Route Protocol payload out-of-order.
		All other values	Reserved
0xf1*NN* *NN* is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*OutOfOrderDeliveryT oRouteProtocolFwd *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0000***	Each Route of forward Link Flow *NN* delivers Route Protocol payload in-order.
		0x0001	Each Route of forward Link Flow *NN* may deliver Route Protocol payload out-of-order.
		All other values	Reserved.
0xf00f	Flow0FOutOfOrderDeliveryT oRouteProtocolRev	***0x0001***	Each Route of reverse Link Flow 0x0F may deliver Route Protocol payload out-of-order.
		All other values	Reserved
0xf0*NN* *NN* is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.	Flow*NN*OutOfOrderDeliveryT oRouteProtocolRev *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0000***	Each Route of reverse Link Flow *NN* delivers Route Protocol payload in-order.
		0x0001	Each Route of reverse Link Flow *NN* may deliver Route Protocol payload out-of-order.
		All other values	Reserved.
0xef*NN* *NN* is the two-digit hexadecimal Link Flow number of the reverse Link Flow in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive.	Flow*NN*RequestLevelRev *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0002***	QoSFlow field is set to ‘10’ for signaling requests in the R-REQCH for the reverse Link Flow *NN*.
		0x0000, 0x0001, 0x0003	Value of the QoSFlow field for signaling requests in the R-REQCH for the reverse Link Flow *NN*.
		All other values	Reserved.

Attribute ID	Attribute	Values	Meaning
0xee*NN* *NN* is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive.	Flow*NN*TransmitAbortTimerFwd *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0000***	Maximum delay for transmission of a higher layer data unit for forward Link Flow *NN* is not specified.
		0x0001 to 0x1388	Maximum delay for transmission of a higher layer data unit for forward Link Flow *NN* in units of ms.
		All other values	Reserved.
0xed*NN* *NN* is the two-digit hexadecimal Link Flow number of the reverse Link Flow in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive.	Flow*NN*TransmitAbortTimerRev *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-1 inclusive, where hexadecimal digits A through F are specified in upper case letters.	***0x0000***	Maximum delay for transmission of a higher layer data unit for reverse Link Flow *NN* is not specified.
		0x0001 to 0x1388	Maximum delay for transmission of a higher layer data unit for reverse Link Flow *NN* in units of ms.
		All other values	Reserved.

3.3.6.2 Complex attributes

The following complex attributes and default values are defined (see 10.3 for attribute record definition).

3.3.6.2.1 Flow*NN*TimersFwd attribute

NN is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x00 to $N_{LinkFlowMax}$-1, inclusive, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
AbortTimer	16	0x01f4
FlushTimer	16	0x012c
AckTimer	16	0x0064

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x01*NN*, where *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-1, inclusive.

AbortTimer — The sender shall set this field to the value of the RLP abort timer for this forward Link Flow in units of ms. The sender shall not set this field to a value greater than MaxAbortTimer.

FlushTimer — The sender shall set this field to the value of the RLP flush timer for this forward Link Flow in units of ms. The value of the RLP flush timer shall be less than or equal to that of the corresponding abort timer.

AckTimer — The sender shall set this field to the value of the RLP Ack timer for this forward Link Flow in units of ms.

3.3.6.2.2 Flow*NN*TimersRev attribute

NN is the two-digit hexadecimal Link Flow number of the reverse Link Flow in the range 0x00 to $N_{LinkFlowMax}$-1, inclusive, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
AbortTimer	16	0x01f4
FlushTimer	16	0x012c
AckTimer	16	0x0064

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x02*NN*, where *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-1, inclusive.

AbortTimer — The sender shall set this field to the value of the RLP abort timer for this reverse Link Flow in units of ms. The sender shall not set this field to a value greater than MaxAbortTimer.

FlushTimer — The sender shall set this field to the value of the RLP flush timer for this reverse Link Flow in units of ms. The value of the RLP flush timer shall be less than or equal to that of the corresponding abort timer.

AckTimer — The sender shall set this field to the value of the RLP Ack timer for this reverse Link Flow in units of ms.

3.3.6.2.3 Flow*NN*ReservationFwd attribute

NN is the two-digit hexadecimal Link Flow number of the forward Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2, inclusive, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default for *NN* = 0x00	Default for *NN* > 0x00
Length	8	N/A	N/A
AttributeID	16	N/A	N/A
ReservationCount	8	0x01	0x00

ReservationCount occurrences of the following field:

{

ReservationLabel	8	0xff	N/A

}

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x03*NN*, where *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.

ReservationCount — The sender shall set this field to the number of reservations associated with this Link Flow. The sender shall include ReservationCount occurrences of the following field with the message.

ReservationLabel — The sender shall set this field to the ReservationLabel of the reservation associated with this Link Flow.

3.3.6.2.4 Flow*NN*ReservationRev attribute

NN is the two-digit hexadecimal Link Flow number of the reverse Link Flow in the range 0x00 to $N_{LinkFlowMax}$-2, inclusive, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default for *NN* = 0x00	Default for *NN* >= 0x00
Length	8	N/A	N/A
AttributeID	16	N/A	N/A
ReservationCount	8	0x01	0x00

ReservationCount occurrences of the following field:

{

ReservationLabel	8	0xff	N/A

}

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x04*NN*, where *NN* is the two-digit hexadecimal Link Flow number in the range 0x00 to $N_{LinkFlowMax}$-2 inclusive.

ReservationCount — The sender shall set this field to the number of reservations associated with this Link Flow. The sender shall include ReservationCount occurrences of the following field with the message.

ReservationLabel — The sender shall set this field to the ReservationLabel of the reservation associated with this Link Flow

3.3.6.2.5 ATSupportedFlowProtocolParameters*PP* attribute

PP is the two-digit hexadecimal ProtocolID number for the Flow Protocol according to Table 14, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
ProtocolSupported	8	0x01 for *PP* = 0x00 and 0x01. 0x00 otherwise.
SupportedProtocolParametersValuesLength	8	0x00
SupportedProtocolParametersValues	SupportedProtocolParametersValuesLength × 8	N/A

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x05*PP*.

ProtocolSupported — The sender shall set this field to 0x00 if the Flow Protocol *PP* is not supported. Otherwise, the sender shall set this field to 0x01 if the Flow Protocol *PP* is supported. All other values are reserved.

SupportedProtocolParametersValuesLength

The sender shall set this field to the length of the SupportedProtocolParametersValues record in units of octets. If the ProtocolSupported field is set to 0x00, the sender shall set this field to 0x00. If the ProtocolSupported field is set to 0x01, the sender shall set this field to 0x00 for Flow Protocol ProtocolID 0x00, 0x01 and 0x03. If the ProtocolSupported field is set to 0x01, the sender shall set this field according to 3.3.6.2.5.1 for Route Protocol ProtocolID 0x02.

SupportedProtocolParametersValues

If ProtocolID is 0x02 and ProtocolSupported is 0x01, then the sender shall set this record as defined in 3.3.6.2.5.1. Otherwise, the sender shall omit this record.

3.3.6.2.5.1 Definition of SupportedProtocolParametersValues record when the Flow Protocol or Route Protocol is ROHC

Field	Length (bits)
MaxSupportedMaxCID	16
LargeCIDSupported	1
MaxSupportedMRRU	16
MaxSupportedDelayedDecompressionDepth	8
TimerBasedCompressionSupported	1
SupportedProfileCount	8
SupportedProfileCount occurrences of the following field:	
{	
SupportedProfile	16
}	
Reserved	0 - 7 (as needed)

MaxSupportedMaxCID
The sender shall set this field to the maximum MAX_CID parameter supported.

LargeCIDSupported The sender shall set this field to ‘0’ if large CID representation is not supported according to [11]. Otherwise, the sender shall set this field to ‘1’ if large CID representation is supported.

MaxSupportedMRRU The sender shall set this field to the MRRU supported by the decompressor according to [11]. Default value is 0x0000 (no segmentation).

MaxSupportedDelayedDecompressionDepth
The sender shall set this field to the maximum supported value of delayed decompression depth at the access terminal ROHC de-compressor.

TimerBasedCompressionSupported
The sender shall set this field to ‘1’ if the compressor at the access terminal supports timer based compression mode. Otherwise, the sender shall set this field to ‘0’.

SupportedProfileCount
The sender shall set this field to the number of ROHC profiles supported. The sender shall include SupportedProfileCount occurrences of the following field with the message.

SupportedProfile The sender shall set this field to the ROHC profile supported by the compressor and decompressor. IANA ROHC profile identifier definitions can be found at [15].

Reserved — The sender shall add reserved bits to make the length of the entire record an integer number of octets. The sender shall set these bits to ‘0’. The receiver shall ignore this field.

3.3.6.2.6 ATSupportedRouteProtocolParameters*PP* attribute

PP is the two-digit hexadecimal ProtocolID number for the Route Protocol according to Table 15, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
ProtocolSupported	8	0x01 for *PP* = 0x00. 0x00 otherwise.
SupportedProtocolParametersValuesLength	8	0x00
SupportedProtocolParametersValues	SupportedProtocolParametersValuesLength × 8	N/A

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x06*PP*.

ProtocolSupported — The sender shall set this field to 0x00 if the Route Protocol *PP* is not supported. Otherwise, the sender shall set this field to 0x01 if the Route Protocol *PP* is supported. All other values are reserved

SupportedProtocolParametersValuesLength — The sender shall set this field to the length of the SupportedProtocolParametersValues record in units of octets. If the ProtocolSupported field is set to 0x00, the sender shall set this field to 0x00. If the ProtocolSupported field is set to 0x01, the sender shall set this field to 0x00 for Flow Protocol ProtocolID 0x00. If the ProtocolSupported field is set to 0x01, the sender shall set this field according to 3.3.6.2.5.1 for Route Protocol ProtocolID 0x02.

SupportedProtocolParametersValues — If ProtocolID is 0x02 and ProtocolSupported is 0x01, then the sender shall set this record as defined in 3.3.6.2.5.1. Otherwise, the sender shall omit this record.

3.3.6.2.7 ATSupportedQoSProfiles attribute

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
QoSProfileCount	8	*0x00*
QoSProfileCount occurrences of the following record:		
{		
FlowProfileID	FlowProfileIDType dependent	N/A
}		

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x0700.

QoSProfileCount — The sender shall set this field to the number of QoS profiles that are included in this message.

FlowProfileID — The sender shall set this field to the profile according to 11.6.

3.3.6.2.8 ANSupportedQoSProfiles attribute

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
QoSProfileCount	8	*0x00*
QoSProfileCount occurrences of the following record:		
{		
FlowProfileID	FlowProfileIDType dependent	N/A
}		

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x0800.

QoSProfileCount — The sender shall set this field to the number of QoS profiles that are included in this message.

FlowProfileID — The sender shall set this field to the profile according to 11.6.

3.3.6.2.9 Reservation*KK*QoSListFwd attribute

KK is the two-digit hexadecimal ReservationLabel, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
ReservationPriority	8	*0xff*
QoSAttributeSetCount	8	*0x00*

QoSAttributeSetCount occurrences of the following record:

{

QoSAttributeSetLength	8	N/A
QoSAttributeSet	QoSAttributeSetLength ×8	N/A

}

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x09*KK*, where *KK* is the two-digit hexadecimal ReservationLabel.

ReservationPriority — The sender shall set this field to indicate the priority to be assigned to the reservation. The value 0x00 indicates the highest priority and the value 0xff indicates the lowest priority.[14]

QoSAttributeSetCount — The sender shall set this field to the number of QoSAttributeSets associated with this reservation. Each QoSAttributeSet contains one set of acceptable QoS parameters. If multiple QoS attribute sets are included, the sender shall include the QoS attribute sets in descending order of preference. The sender shall include QoSAttributeSetCount occurrences of the following two fields with the message.

QoSAttributeSetLength — The sender shall set this field to the length, in octets, of the QoSAttributeSet.

QoSAttributeSet — The QoS parameters requested for the reservation. The sender shall set this record as defined in 3.3.6.2.9.1. If the QoSAttributeSet specifies a FlowProfileID, then the sender shall set this field to a FlowProfileID that is included in the ANSupportedQoSProfiles attribute.

[14] The ReservationPriority field may be used by the access terminal to indicate the relative importance of reservations for purposes such as admission control.

3.3.6.2.9.1 Definition of QoSAttributeSet

Field	Length (bits)
QoSAttributeSet_ID	8
Reserved	7
Verbose	1
FlowProfileID	0 or FlowProfileIDType dependent
Traffic_Class	0 or 8
Peak_Rate	0 or 32
Bucket_Size	0 or 32
Token_Rate	0 or 32
Max_Latency	0 or 16
Max_Packet_Loss_Rate	0 or 8
Packet_Size	0 or 16
Max_Jitter	0 or 16

QoSAttributeSet_ID — The sender shall set this field to an identifier for the QoSAttributeSet. The sender shall not set this field to 0x00.

Reserved — The sender shall set this field to '0000000'. The receiver shall ignore this field.

Verbose — If the ProfileID field is included, the sender shall set this field to '1'. Otherwise, the sender shall set this field to '0'.

FlowProfileID — If Verbose is '1', then the sender shall omit this field. Otherwise, the sender shall set this field to the profile according to 11.6.

Traffic_Class — If Verbose is '0', then the sender shall omit this field. Otherwise, the sender shall include this field to indicate the traffic class as specified in Table 12.

Table 12 Traffic Class

Value	Description
‘000’	Unknown
‘001’	Conversational[15]
‘010’	Streaming[16]
‘011’	Interactive[17]
‘100’	Background[18]
‘101’-‘111’	Reserved

Peak_Rate — This field specifies the peak traffic rate as specified in [12], [13], and [14] in units of bytes per second. If Verbose is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the unsigned value n (range from 1 to 4294967295), where the peak rate = n bytes per second. If the sender doesn’t want to specify the peak rate, the sender shall set this field to zero.

Bucket_Size — This field specifies the token bucket size as specified in [12], [13], and [14] in units of bytes. If Verbose is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this filed to the unsigned value n (range from 1 to 4294967295), where the bucket size = n bytes. If the sender doesn’t want to specify the token bucket size, the sender shall set this field to zero.

Token_Rate — This field specifies the token rate as specified in [12], [13], and [14] in units of bytes per second. If Verbose is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the unsigned value n (range from 1 to 4294967295), where the token rate = n bytes per second. If the sender doesn’t want to specify the token rate, the sender shall set this field to zero.

Max_Latency — The maximum latency in units of milliseconds. The maximum latency specifies the maximum acceptable delay from the time that an octet of user data is submitted to the transmitter until the receiver receives the octet. It is measured between the sender and the receiver. If Verbose is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the unsigned value n (range from 1 to 65535), where the maximum latency = n milliseconds. If the sender doesn’t want to specify the maximum latency, the sender shall set this field to zero.

[15] Conversational traffic class has a low latency, medium error rate service requirement

[16] Streaming traffic class has a high latency, medium error rate service requirement

[17] Interactive traffic class has a low latency, low error rate service requirement

[18] Background traffic class has a high latency, low error rate service requirement

Max_Packet_Loss_Rate
This field indicates the maximum higher layer packet loss rate. If Verbose is '0', then the sender shall omit this field. Otherwise, the sender shall set this field to an unsigned value n (range from 1 to 31), where the maximum packet loss rate = 10^(-n/4). If the sender doesn't want to specify the maximum loss rate, the sender shall set this field to zero. When Max_Packet_loss_Rate is used the Packet_Size shall also be specified.

Packet_Size: This field indicates the median packet size, in units of bytes. If Verbose is '0', then the sender shall omit this field. Otherwise, the sender shall set this filed to the unsigned value n (range from 1 to 65535), where the median packet size = n bytes. If the sender doesn't want to specify the median packet size, the sender shall set this field to zero.

Max_Jitter The maximum jitter in units of milliseconds. The maximum jitter specifies the maximum acceptable latency variation from the time that a packet is received until the time that the next packet is received as measured at the receiver. If Verbose is '0', then the sender shall omit this field. Otherwise, the sender shall set this field to the unsigned value n (range from 1 to 65535), where the maximum jitter = n milliseconds. The sender shall set this field to zero to indicate the traffic flow sensitivity to variation in delay is not specified.

3.3.6.2.10 Reservation*KK*QoSListRev attribute

KK is the two-digit hexadecimal ReservationLabel, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
ReservationPriority	8	***0xff***
QoSAttributeSetCount	8	*0x00*
QoSAttributeSetCount occurrences of the following record:		
{		
QoSAttributeSetLength	8	N/A
QoSAttributeSet	QoSAttributeSetLength ×8	N/A
}		

Length Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID The sender shall set this field to 0x0a*KK*, where *KK* is the two-digit hexadecimal ReservationLabel.

ReservationPriority The sender shall set this field to indicate the priority to be assigned to the reservation. The value 0x00 indicates the highest priority and the value 0xff indicates the lowest priority.

QoSAttributeSetCount The sender shall set this field to the number of QoSAttributeSets associated with this reservation. Each QoSAttributeSet contains one set of acceptable QoS parameters. If multiple QoS attribute sets are included, the sender shall include the QoS attribute sets in descending order of preference. The sender shall include QoSAttributeSetCount occurrences of the following two fields with the message.

QoSAttributeSetLength
The sender shall set this field to the length, in octets, of the QoSAttributeSet.

QoSAttributeSet The QoS parameters requested for the reservation. The sender shall set this record as defined in 3.3.6.2.9.1. If the QoSAttributeSet specifies a FlowProfileID, then the sender shall set this field to a FlowProfileID that is included in the ATSupportedQoSProfiles attribute.

3.3.6.2.11 Reservation*KK*QoSUsedFwd attribute

KK is the two-digit hexadecimal ReservationLabel, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
QoSAttributeSet_ID	8	0x00

Length Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID The sender shall set this field to 0x0b*KK*, where *KK* is the two-digit hexadecimal ReservationLabel.

QoSAttributeSet_ID The sender may set this field to the identifier assigned by the corresponding Reservation*KK*QoSListFwd message of the QoSAttributeSet that has been granted; or the sender may set this field to 0x00 to indicate that requested QoSAttributeSet is invalid.

3.3.6.2.12 Reservation*KK*QoSUsedRev attribute

KK is the two-digit hexadecimal ReservationLabel, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
QoSAttributeSet_ID	8	0x00

Length	Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.
AttributeID	The sender shall set this field to 0x0c*KK*, where *KK* is the two-digit hexadecimal ReservationLabel.
QoSAttributeSet_ID	The sender may set this field to the identifier assigned by the corresponding Reservation*KK*QoSListRev message of the QoSAttributeSet that has been granted; or the sender may set this field to 0x00 to indicate that requested QoSAttributeSet is invalid.

3.3.6.2.13 Reservation*KK*PacketFilterFwd attribute

KK is the two-digit hexadecimal ReservationLabel, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
FilterPrecedence	16	*0xff*
FilterSpecCount	8	*0x00*
FilterSpecCount occurrences of the following record:		
{		
FilterSpecType	8	N/A
FilterSpecLength	8	N/A
FilterSpec	FilterSpecLength × 8	N/A
}		

Length	Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.
AttributeID	The sender shall set this field to 0x0d*KK*, where *KK* is the two-digit hexadecimal ReservationLabel.
FilterPrecedence	The sender shall set this field to indicate the precedence of the packet filter for Reservation*KK*PacketFilterFwd among all packet filters defined by the Reservation*KK*PacketFilterFwd attributes of all active forward Reservations associated with the access terminal. The evaluation precedence index is in the range of 0x0000 to 0xffff. The higher the value of the FilterPrecedence field, the lower the precedence of that packet filter. If a given packet matches more than one of the currently active packet filters, the packet is mapped to the Reservation corresponding to the packet filter of highest precedence. A given precedence level may be used only once per access terminal, except 0xffff which is used as an indication of no precedence.
FilterSpecCount	The sender shall set this field to the number of FilterSpecs associated with this reservation. The sender shall include FilterSpecCount occurrences of the following three fields with the message.

FilterSpecType — The sender shall set this field to an identifier for the Filter Specification Type according to Table 13.

Table 13 FilterSpecType for Packet Filter

Value	FilterSpecType
0x00	IP version 4 [4]
0x01	IP version 6 [5]
0x02	Match all
All other values	Reserved

FilterSpecLength — The sender shall set this field to the length of the FilterSpec field in units of octets.

FilterSpec — If FilterSpecType is 0x00, then the sender shall set this record as defined in 3.3.6.2.13.1. If FilterSpecType is 0x01, then the sender shall set this record as defined in 3.3.6.2.13.2. Otherwise, the sender shall omit this record.

3.3.6.2.13.1 Definition of FilterSpec record for IPv4

Field	Length (bits)
IPv4_Source_Address_Included	1
IPv4_Destination_Address_Included	1
Source_Port_Range_Included	1
Destination_Port_Range_Included	1
Packet_Length_Included	1
Protocol_Type_Included	1
Type_of_Service_Included	1
IPSec_SPI_Included	1
Protocol_Type	0 or 8
IPv4_Source_Address_Prefix_Length	0 or 8
IPv4_Destination_Address_Prefix_Length	0 or 8
IPv4_Source_Address	0 or 32
IPv4_Destination_Address	0 or 32
Source_Port_Lower	0 or 16
Source_Port_Upper	0 or 16
Destination_Port_Lower	0 or 16
Destination_Port_Upper	0 or 16
Packet_Length_Lower	0 or 16
Packet_Length_Upper	0 or 16
IPSec_SPI	0 or 32
Type_of_Service	0 or 8
Type_of_Service_Mask	0 or 8

IPv4_Source_Address_Included
The sender shall set this field to '1' to match the value of the Source Address field in the IP packet. Otherwise, the sender shall set this field to '0'.

IPv4_Destination_Address_Included
The sender shall set this field to '1' to match the value of the Destination Address field in the IP packet. Otherwise, the sender shall set this field to '0'.

Source_Port_Range_Included
The sender shall set this field to '1' to match a range of Source Port numbers in the IP packet. Otherwise, the sender shall set this field to '0'.

Destination_Port_Range_Included
The sender shall set this field to '1' to match a range of Destination Port numbers in the IP packet. Otherwise, the sender shall set this field to '0'.

Packet_Length_Included
The sender shall set this field to '1' to match a range of IP packet lengths. Otherwise, the sender shall set this field to '0'.

Protocol_Type_Included
The sender shall set this field to '1' to match the value of the Protocol field in the IP packet. Otherwise, the sender shall set this field to '0'.

Type_of_Service_Included
The sender shall set this field to '1' to match the value of the Type of Service field in the IP packet. Otherwise, the sender shall set this field to '0'.

IPSec_SPI_Included
The sender shall set this field to '1' to match the value of the IPSec Security Parameter Index (SPI) field in the IP packet. Otherwise, the sender shall set this field to '0'.

Protocol_Type
If Protocol_Type_Included is '0', then the sender shall omit this field. Otherwise, the sender shall set this field to the value of the Protocol field to match in the IP packet. The sender shall set this field in the range from 0x00 to 0xff.

IPv4_Source_Address_Prefix_Length
The IPv4_Source_Address up to the IPv4_Source_Address_Prefix_Length is matched against the Source Address in the IP packet. If IPv4_Source_Address_Included is '0', then the sender shall omit this field. Otherwise, the sender shall set this field in the range from 0x01 to 0x10.

IPv4_Destination_Address_Prefix_Length
The IPv4_Destination_Address up to the IPv4_Destination_Address_Prefix_Length is matched against the Destination Address in the IP packet. If IPv4_Destination_Address_Included is '0', then the sender shall omit this field. Otherwise, the sender shall set this field in the range from 0x01 to 0x10.

IPv4_Source_Address If IPv4_Source_Address_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the value of the Source Address field to match in the IP packet.

IPv4_Destination_Address If IPv4_Destination_Address_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the value of the Destination Address field to match in the IP packet.

Source_Port_Lower If Source_Port_Range_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the lowest value of the Source Port Number to match in the IP packet. The sender shall set this field in the range from 0x0000 to 0xffff.

Source_Port_Upper If Source_Port_Range_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the highest value of the Source Port Number to match in the IP packet. The sender shall set this field in the range from Source_Port_Lower to 0xffff.

Destination_Port_Lower If Destination_Port_Range_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the lowest value of the Destination Port Number to match in the IP packet. The sender shall set this field in the range from 0x0000 to 0xffff.

Destination_Port_Upper If Destination_Port_Range_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the highest value of the Destination Port Number to match in the IP packet. The sender shall set this field in the range from Destination_Port_Lower to 0xffff.

Packet_Length_Lower If Packet_Length_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the shortest packet length IP packet to match. The sender shall set this field in the range from 0x0000 to 0xffff.

Packet_Length_Upper If Packet_Length_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the highest packet length IP packet to match. The sender shall set this field in the range from Packet_Length_Lower to 0xffff.

IPSec_SPI If IPSec_SPI_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the value of the IPSec Security Parameter Index (SPI) to match in the IP packet.

Type_of_Service If Type_of_Service_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the value of the Type of Service field to match in the IP packet.

Type_of_Service_Mask

If Type_of_Service_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the bits of the Type_of_Service field to match against the actual value of the corresponding field in the IP packets. The mask contains ones in the bit positions to be used in the matching operation.

3.3.6.2.13.2 Definition of FilterSpec record for IPv6

Field	Length (bits)
IPv6_Source_Address_Included	1
IPv6_Destination_Address_Included	1
Source_Port_Range_Included	1
Destination_Port_Range_Included	1
Packet_Length_Included	1
Traffic_Class_Included	1
Flow_Label_Included	1
IPSec_SPI_Included	1
IPv6_Source_Address_Prefix_Length	0 or 8
IPv6_Destination_Address_Prefix_Length	0 or 8
IPv6_Source_Address	0 or 128
IPv6_Destination_Address	0 or 128
Source_Port_Lower	0 or 16
Source_Port_Upper	0 or 16
Destination_Port_Lower	0 or 16
Destination_Port_Upper	0 or 16
Packet_Length_Lower	0 or 16
Packet_Length_Upper	0 or 16
IPSec_SPI	0 or 32
Traffic_Class	0 or 8
Traffic_Class_Mask	0 or 8
Reserved	0 or 4
Flow_Label	0 or 20

IPv6_Source_Address_Included

The sender shall set this field to ‘1’ to match the value of the Source Address field in the IP packet. Otherwise, the sender shall set this field to ‘0’.

IPv6_Destination_Address_Included

The sender shall set this field to ‘1’ to match the value of the Destination Address field in the IP packet. Otherwise, the sender shall set this field to ‘0’.

Source_Port_Range_Included
The sender shall set this field to ‘1’ to match a range of Source Port numbers in the IP packet. Otherwise, the sender shall set this field to ‘0’.

Destination_Port_Range_Included
The sender shall set this field to ‘1’ to match a range of Destination Port numbers in the IP packet. Otherwise, the sender shall set this field to ‘0’.

Packet_Length_Included
The sender shall set this field to ‘1’ to match a range of IP packet lengths. Otherwise, the sender shall set this field to ‘0’.

Traffic_Class_Included
The sender shall set this field to ‘1’ to match the value of the Traffic Class field in the IP packet. Otherwise, the sender shall set this field to ‘0’.

Flow_Label_Included The sender shall set this field to ‘1’ to match the value of the Flow Label field in the IP packet. Otherwise, the sender shall set this field to ‘0’.

IPSec_SPI_Included The sender shall set this field to ‘1’ to match the value of the IPSec Security Parameter Index (SPI) field in the IP packet. Otherwise, the sender shall set this field to ‘0’.

IPv6_Source_Address_Prefix_Length
If IPv6_Source_Address_Included is ‘0’, then the sender shall omit this field. Otherwise, the IPv6_Source_Address up to the IPv6_Source_Address_Prefix_Length is matched against the Source Address in the IP packet. The sender shall set this field in the range from 0x01 to 0x80.

IPv6_Destination_Address_Prefix_Length
If IPv6_Destination_Address_Included is ‘0’, then the sender shall omit this field. Otherwise, the IPv6_Destination_Address up to the IPv6_Destination_Address_Prefix_Length is matched against the Destination Address in the IP packet. The sender shall set this field in the range from 0x01 to 0x80.

IPv6_Source_Address If IPv6_ Source_Address_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the value of the Source Address field to match in the IP packet.

IPv6_Destination_Address
If IPv6_ Destination_Address_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the value of the Destination Address field to match in the IP packet.

Source_Port_Lower If Source_Port_Range_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the lowest value of the Source Port Number to match in the IP packet. The sender shall set this field in the range from 0x0000 to 0xffff.

Source_Port_Upper
If Source_Port_Range_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the highest value of the Source Port Number to match in the IP packet. The sender shall set this field in the range from Source_Port_Lower to 0xffff.

Destination_Port_Lower
If Destination_Port_Range_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the lowest value of the Destination Port Number to match in the IP packet. The sender shall set this field in the range from 0x0000 to 0xffff.

Destination_Port_Upper
If Destination_Port_Range_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the highest value of the Destination Port Number to match in the IP packet. The sender shall set this field in the range from Destination_Port_Lower to 0xffff.

Packet_Length_Lower
If Packet_Length_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the shortest packet length IP packet to match. The sender shall set this field in the range from 0x0000 to 0xffff.

Packet_Length_Upper
If Packet_Length_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the highest packet length IP packet to match. The sender shall set this field in the range from Packet_Length_Lower to 0xffff.

IPSec_SPI
If IPSec_SPI_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the value of the IPSec Security Parameter Index (SPI) to match in the IP packet.

Traffic_Class
If Traffic_Class_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the value of the Traffic Class field to match in the IP packet. The sender shall set this field in the range from 0x00 to 0xff.

Traffic_Class_Mask
If Traffic_Class_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the bits of the Traffic_Class field to match against the actual value of the corresponding field in the IP packets. The mask contains ones in the bit positions to be used in the matching operation.

Reserved
If Flow_Label_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to ‘0000’. The receiver shall ignore this field.

Flow_Label
If Flow_Label_Included is ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the value of the Flow Label field to match in the IP packet.

3.3.6.2.14 Reservation*KK*PacketFilterRev attribute

KK is the two-digit hexadecimal ReservationLabel, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
FilterPrecedence	16	*0xff*
FilterSpecCount	8	*0x00*
FilterSpecCount occurrences of the following record:		
{		
FilterSpecType	8	N/A
FilterSpecLength	8	N/A
FilterSpec	FilterSpecLength × 8	N/A
}		

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x0e*KK*, where *KK* is the two-digit hexadecimal ReservationLabel.

FilterPrecedence — The sender shall set this field to indicate the precedence of the packet filter for Reservation*KK*PacketFilterRev among all packet filters defined by the Reservation*KK*PacketFilterRev attributes of all active reverse Reservations associated with the access terminal. The evaluation precedence index is in the range of 0x0000 to 0xffff. The higher the value of the FilterPrecedence field, the lower the precedence of that packet filter. If a given packet matches more than one of the currently active packet filters, the packet is mapped to the Reservation corresponding to the packet filter of highest precedence. A given precedence level may be used only once per access terminal, except 0xffff which is used as an indication of no precedence.

FilterSpecCount — The sender shall set this field to the number of FilterSpecs associated with this reservation. The sender shall include FilterSpecCount occurrences of the following three fields with the message.

FilterSpecType — The sender shall set this field to an identifier for the Filter Specification Type according to Table 13.

FilterSpecLength — The sender shall set this field to the length of the FilterSpec field in units of octets.

FilterSpec — If FilterSpecType is 0x00, then the sender shall set this record as defined in 3.3.6.2.13.1. If FilterSpecType is 0x01, then the sender shall set this record as defined in 3.3.6.2.13.2. Otherwise, the sender shall omit this record.

3.3.6.2.15 Flow*NN*FlowProtocolParametersFwd attribute

NN is the two-digit hexadecimal forward Link Flow identifier, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
ProtocolID	8	***0x01***
ProtocolParametersLength	8	***0x00***
ProtocolParameters	ProtocolParametersLength × 8	N/A

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x0f*NN*, where *NN* is the two-digit hexadecimal forward Link Flow number.

ProtocolID — The sender shall set this field to an identifier for the Flow Protocol according to Table 14.

Table 14 ProtocolID for Flow Protocol

Value	Protocol
0x00	NULL
0x01	Internet Protocol (IP) version 4 [4] and version 6 [5]
0x02	Robust Header Compression (ROHC) [11]
0x03	EAP encapsulation over Layer 2 [22]
0x04	IEEE 802.3 / DIX Ethernet [17]
All other values	Reserved

ProtocolParametersLength — The sender shall set this field to the length of the ProtocolParameters field in units of octets.

ProtocolParameters — If ProtocolID is 0x02, then the sender shall set this record as defined in 3.3.6.2.15.1. Otherwise, the sender shall omit this record.

3.3.6.2.15.1 Definition of ProtocolParameters record when the Flow Protocol or Route Protocol is ROHC

Field	Length (bits)
MaxCID	16
LargeCIDs	1
FeedbackForIncluded	1
FeedbackFor	0 or 5
MRRU	16
DelayedDecompressionDepth	8
ProfileCount	8
ProfileCount occurrences of the following field:	
{	
Profile	16
}	
Reserved	0 – 7 (as needed)

MaxCID — The sender shall set this field to the MAX_CID parameter for this ROHC Channel. The sender shall not set this field to a value greater than MaxSupportedMaxCID.

LargeCIDs — If the LARGE_CIDS parameter for this ROHC Channel is false, then the sender shall set this field to ‘0’. Otherwise, the sender shall set this field to ‘1’. The sender shall not set this field to ‘1’ if LargeCIDSupported is not set to ‘1’.

FeedbackForIncluded — If ROHC feedback associated with another Link flow (ROHC channel) is sent on this Link flow (ROHC channel), then this field shall be set to ‘1’. Otherwise, this field shall be set to ‘0’.

FeedbackFor — If FeedbackForIncluded is set to ‘0’, then the sender shall omit this field. Otherwise, the sender shall set this field to the Link flow number (ROHC channel) to which ROHC feedback sent on this Link flow (ROHC channel) refers.

MRRU — The sender shall set this field to the MRRU parameter for this ROHC channel. The sender shall not set this field to a value larger than MaxSupportedMRRU.

DelayedDecompressionDepth — The sender shall set this field to the maximum number of packets that can be buffered and thus possibly be delayed decompressed by the decompressor according to [11] for this ROHC channel. If the value of this field is 0x00, then delayed decompression shall not be enabled. The sender shall not set this field to a value greater than MaxSupportedDelayedDecompressionDepth.

ProfileCount — The sender shall set this field to the number of ROHC profiles supported by the decompressor. The sender shall include ProfileCount occurrences of the following field with the message.

Profile — The sender shall set this field to the ROHC profile supported by the decompressor according to [11]. The sender shall not set this field to a value that is not included in the list of supported Profiles.

Reserved — The sender shall add reserved bits to make the length of the entire record an integer number of octets. The sender shall set these bits to '0'. The receiver shall ignore this field.

3.3.6.2.16 Flow*NN*FlowProtocolParametersRev attribute

NN is the two-digit hexadecimal forward Link Flow identifier, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
ProtocolID	8	***0x01***
ProtocolParametersLength	8	***0x00***
ProtocolParameters	ProtocolParametersLength × 8	N/A

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x10*NN*, where *NN* is the two-digit hexadecimal forward Link Flow number.

ProtocolID — The sender shall set this field to an identifier for the Flow Protocol according to Table 14.

ProtocolParametersLength — The sender shall set this field to the length of the ProtocolParameters field in units of octets.

ProtocolParameters — If ProtocolID is 0x02, then the sender shall set this record as defined in 3.3.6.2.16.1. Otherwise, the sender shall omit this record.

3.3.6.2.16.1 Definition of ProtocolParameters record when the Flow Protocol or Route Protocol is ROHC

Field	Length (bits)
MaxCID	16
LargeCIDs	1
FeedbackForIncluded	1
FeedbackFor	0 or 5
MRRU	16
TimerBasedCompression	1
ProfileCount	8
ProfileCount occurrences of the following field:	
{	
Profile	16
}	
Reserved	0 – 7 (as needed)

MaxCID — The sender shall set this field to the MAX_CID parameter for this ROHC Channel. The sender shall not set this field to a value greater than MaxSupportedMaxCID.

LargeCIDs — If the LARGE_CIDS parameter for this ROHC Channel is false, then the sender shall set this field to '0'. Otherwise, the sender shall set this field to '1'. The sender shall not set this field to '1' if LargeCIDSupported is not set to '1'.

FeedbackForIncluded — If ROHC feedback associated with another Link flow (ROHC channel) is sent on this Link flow (ROHC channel), then this field shall be set to '1'. Otherwise, this field shall be set to '0'.

FeedbackFor — If FeedbackForIncluded is set to '0', then the sender shall omit this field. Otherwise, the sender shall set this field to the Link flow number (ROHC channel) to which ROHC feedback sent on this Link flow (ROHC channel) refers.

MRRU — The sender shall set this field to the MRRU parameter for this ROHC channel. The sender shall not set this field to a value larger than MaxSupportedMRRU.

TimerBasedCompression

The sender shall set this field to '0' if timer based compression according to [11] is not enabled for this ROHC channel. The sender shall set this field to '1' if timer based compression according to [11] is enabled for this ROHC channel. If TimerBasedCompressionSupported is set to '0', then the sender shall not set this field to '1'.

ProfileCount — The sender shall set this field to the number of ROHC profiles supported by the decompressor. The sender shall include ProfileCount occurrences of the following field with the message.

Profile — The sender shall set this field to the ROHC profile supported by the decompressor according to [11]. The sender shall not set this field to a value that is not included in the list of supported Profiles.

Reserved — The sender shall add reserved bits to make the length of the entire record an integer number of octets. The sender shall set these bits to '0'. The receiver shall ignore this field.

3.3.6.2.17 Flow*NN*RouteProtocolParametersFwd attribute

NN is the two-digit hexadecimal forward Link Flow number, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
ProtocolID	8	***0x00***
ProtocolParametersLength	8	***0x00***
ProtocolParameters	ProtocolParametersLength × 8	N/A

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x11*NN*, where *NN* is the two-digit hexadecimal forward Link Flow number.

ProtocolID — The sender shall set this field to an identifier for the Route Protocol according to Table 15.

Table 15 ProtocolID for Route Protocol

Value	Protocol
0x00	NULL
0x02	The Route Protocol is Robust Header Compression (ROHC) [11]
All other values	Reserved

ProtocolParametersLength
The sender shall set this field to the length of the ProtocolParameters field in units of octets.

ProtocolParameters — If ProtocolID is 0x02, then the sender shall set this record as defined in 3.3.6.2.15.1. Otherwise, the sender shall omit this record.

3.3.6.2.18 Flow*NN*RouteProtocolParametersRev attribute

NN is the two-digit hexadecimal forward Link Flow number, where hexadecimal digits A through F are specified in upper case letters.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	16	N/A
ProtocolID	8	***0x00***
ProtocolParametersLength	8	***0x00***
ProtocolParameters	ProtocolParametersLength × 8	N/A

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x12*NN*, where *NN* is the two-digit hexadecimal forward Link Flow number.

ProtocolID — The sender shall set this field to field to an identifier for the Route Protocol according to Table 15.

ProtocolParametersLength
The sender shall set this field to the length of the ProtocolParameters field in units of octets.

ProtocolParameters — If ProtocolID is 0x02, then the sender shall set this record as defined in 3.3.6.2.16.1. Otherwise, the sender shall omit this record

3.3.7 Session state information

The Session State Information record (see 10.10) consists of parameter records.

This transport defines the following parameter records in addition to the configuration attributes for this transport.

3.3.7.1 FlowControlState parameter

Table 16 Format of the parameter record for the FlowControlState parameter

Field	Length (bits)
ParameterType	8
Length	8
FlowControlState	8

ParameterType — This field shall be set to 0x01 for this parameter record.

Length — This field shall be set to the length of this parameter record in units of octets excluding the Length field.

FlowControlState — This field shall be set to 0x00 if the state of the Flow Control Protocol associated with the access terminal's session is Close. Otherwise, this field shall be set to 0x01. All of the other values for this field are reserved.

3.3.7.2 ReservationState parameter

Table 17 Format of the parameter record for the ReservationState parameter

Field	Length (bits)
ParameterType	8
Length	8
OpenReservationCount	8
OpenReservationCount occurrences of the following record:	
{	
Reserved	7
Link	1
ReservationLabel	8
}	

ParameterType — This field shall be set to 0x02 for this parameter record.

Length — This field shall be set to the length of this parameter record in units of octets excluding the Length field.

OpenReservationCount — This field shall be set to the number of Reservations that are in the Open state. The sender shall include OpenReservationCount occurrences of the following three fields with the message.

Reserved — This field shall be set to '0000000'. The receiver shall ignore this field.

Link — This field shall be set to '1' for a forward link Reservation, and to '0' for a reverse link Reservation.

ReservationLabel This field shall be set to the ReservationLabel.

3.3.7.3 RouteState parameter

Table 18 Format of the parameter record for the RouteState parameter

Field	Length (bits)
ParameterType	8
Length	8
RouteSelectionProtocolState	2
NextRouteSelectTransactionID	8
NextActivateRouteTransactionID	8
Reserved	6

ParameterType This field shall be set to 0x03 for this parameter record.

Length This field shall be set to the length of this parameter record in units of octets excluding the Length field.

RouteSelectionProtocolState
This field shall be set to indicate the state of Route Selection Protocol according to Table 19.

NextRouteSelectTransactionID
This field shall be set to the TransactionID field of the next RouteSelect message that will be sent.

NextActivateRouteTransactionID
This field shall be set to the TransactionID field of the next ActivateRoute message that will be sent.

Reserved This field shall be set to '000000'. The receiver shall ignore this field.

Table 19 RouteSelectionProtocolState encoding

State	Value
A Open B Draining	'00'
A Open B Activating	'01'
A Draining B Open	'10'
A Activating B Open	'11'

3.3.7.4 RadioLinkState parameter

Table 20 The format of the parameter record for the RadioLink*NN*State parameter

Field	Length (bits)
ParameterType	8
Length	8
QTxStateVector	$N_{LinkFlowMax}$
QRxStateVector	$N_{LinkFlowMax}$
Reserved	0 – 7 (as needed)

ParameterType — This field shall be set to 0x04 for this parameter record.

Length — This field shall be set to the length of this parameter record in units of octets excluding the Length field.

QTxStateVector — This field shall be set to the vector of binary values of the sequence state variables $[Q_{00,Tx}, Q_{01,Tx}, \ldots, Q_{LFMax,Tx}]$, where *LFMax* is equal to $N_{LinkFlowMax}$-1.

QRxStateVector — This field shall be set to the vector of binary values of the sequence state variables $[Q_{00,Rx}, Q_{01,Rx}, \ldots, Q_{LFMax,Rx}]$, where *LFMax* is equal to $N_{LinkFlowMax}$-1.

Reserved — The sender shall add reserved bits to make the length of the entire record an integer number of octets. The sender shall set these bits to ‘0’. The receiver shall ignore this field.

3.4 Default Packet Consolidation Protocol

3.4.1 Overview

The Default Packet Consolidation Protocol provides the following functions:

- Multiplexing of transports for one access terminal. Each transport maps to a Transport in the Packet Consolidation Protocol. Transport 0 is always assigned to the Signaling Transport. The other Transports can be assigned to transports with different Quality of Service (QoS) requirements, or other types of transports.
- Provision of configuration messages that map transports to Transports.
- Packet consolidation on the transmit side and packet de-multiplexing on the receive side.
- Prioritization of the transmission of packets.

Table 21 specifies the values of Transport Subtypes for transports defined in this specification.

Table 21 Transport subtypes for transports defined in this specification

Value	Meaning
0x0000	Default Signaling Transport.
0x0001	Default Packet Transport.
0xffff	Transport not used
All other values are reserved.	

The Default Packet Consolidation Protocol provides the ability to multiplex up to 8 transports using the Transport field in the Packet Consolidation Protocol header. Transport 0 is always reserved for a Signaling Transport.

This protocol uses the Generic Attribute Update Protocol in 10.9 to map transports to Transports.

Packet Consolidation Protocol packets contain one or more transport packets. The protocol places the Packet Consolidation Protocol header defined in 3.4.8 in front of each transport packet and enough padding to create a maximum length packet. The header added by this protocol for a consolidated packet is 16 bits in length per transport packet and 8 bits in length for padding.

3.4.2 Data encapsulation

Figure 40 illustrates the relationship between a transport packet, a Packet Consolidation Protocol packet, and a Security Sublayer payload for a Packet Consolidation packet containing two transport packets and padding.

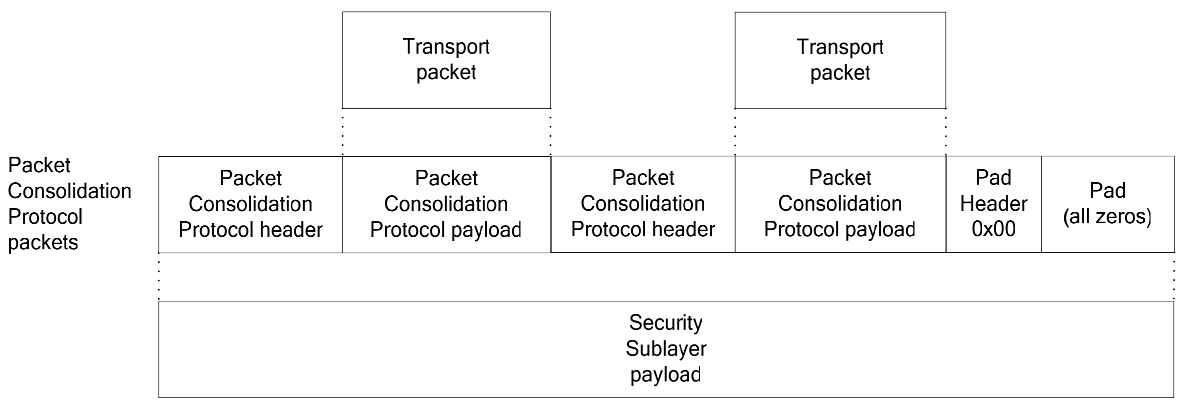

Figure 40 Packet Consolidation Protocol encapsulation

3.4.3 Primitives

3.4.3.1 Commands

This protocol does not define any commands.

3.4.3.2 Return indications

This protocol does not return any indications.

3.4.4 Public data

3.4.4.1 Static public data

This protocol does not define any static public data.

3.4.4.2 Dynamic public data

- Subtype for this protocol

3.4.5 Protocol data unit

The Protocol Data Unit for this protocol is a Packet Consolidation Protocol packet. Packet Consolidation Protocol packets contain transport packets destined to or from the same access terminal address.

3.4.6 Protocol initialization and swap procedures

3.4.6.1 Protocol initialization

Upon creation, the value of the attributes for this protocol instance in the access terminal and access network shall be set to the default values specified for each attribute.

3.4.6.2 Protocol swap

This protocol defines an empty swap procedure.

3.4.7 Procedures

This protocol receives transport packets for transmission from up to 8 different transports. All transmitted packets are forwarded to the Security Sublayer. All Packet Consolidation Protocol packets forwarded to the Security Sublayer shall be octet aligned.

The protocol receives Packet Consolidation Protocol packets from the Security Sublayer and removes the Packet Consolidation Protocol header. The transport packet obtained in this manner is forwarded to the transport indicated by the Transport field of the Packet Consolidation Protocol header.

The maximum size transport packet the protocol can encapsulate depends on the Physical Layer channel on which this packet will be transmitted and on the specific security protocols negotiated.

The access terminal and the access network may use the Generic Attribute Update Protocol messages in 10.9 to map a transport to a Transport that is not already assigned to another Transport.

The access terminal and the access network shall not use the Generic Attribute Update Protocol in 10.9 to map a transport to a Transport that is already assigned to another Transport.

Once the access terminal and the access network agree upon the mapping of a new transport to a Transport, the access terminal and access network shall create an instance of the agreed upon transport and add the instance of the transport to that Transport.

This protocol receives the following information with every transmitted transport packet:

- Destination channel: Forward Unicast Traffic Channel, Forward Broadcast Traffic Channel, or Reverse Traffic Channel.
- Priority number of the transport packet. This field is determined by the Flow*NN*RequestLevelRev public data of the Data Transport and the constant $N_{SLPRequestLevelRev}$ of the Signaling Transport. In this protocol, the use of the priority level is defined only for access terminal transmissions.
- Forced Single Encapsulation: Whether or not the transport packet can be encapsulated with other transport packets in the same Packet Consolidation Protocol packet. (Applicable only on the Forward Data Channel)

3.4.7.1 Destination channels

Associated with a transport packet received by this protocol there shall be a parameter indicating the destination channel on which the packet is to be transmitted.

Associated with a transport packet received by this protocol there may be a parameter indicating a transmission deadline.

3.4.7.2 Priority order

The priority used by the access network to derive Packet Consolidation Protocol packets from transport packets is beyond the scope of this specification.

The priority used by the access terminal to derive Packet Consolidation Protocol packets from transport packets shall follow the following rules.

- Packets with lower priority number, as determined by the Flow*NN*RequestLevelRev public data of the Packet Transport or the constant $N_{SLPRequestLevelRev}$ of the Signaling Transport shall have higher priority for transmission.
- For packets with the same priority number, the packet that was received first by the protocol shall have higher priority for transmission.

Transmission of packets that have higher priority shall take precedence over transmission of packets with lower priority within the constraints imposed by lower layer protocols.

3.4.7.3 Forced single encapsulation

If a Forward Traffic Channel Transport packet is marked as Forced Single Encapsulation, the access network shall encapsulate it without any other transport packets in a Packet Consolidation Protocol packet. The Packet Consolidation Protocol shall also pass an indication down to the physical layer with the Packet Consolidation Protocol packet, instructing the physical layer to ensure that the Physical Layer packet containing this packet does not contain any other Packet Consolidation Protocol packet. Forced Single Encapsulation applies only to the Forward Traffic Channel MAC Layer packets.

Forced Single Encapsulation is used for test services that require a one to one mapping between transport packets and Physical Layer packets.

3.4.7.4 Transmit procedures

The transmitter shall create a Packet Consolidation Protocol packet by adding the Packet Consolidation Protocol header, defined in 3.4.8 in front of every transport packet, concatenating the result and adding enough padding to fill the Security Sublayer payload. The resulting packet length shall not exceed the maximum payload that can be carried on the Physical Layer Channel, given the transmission rate that will be used to transmit the packet and the headers added by the lower layers.

The transmitter shall forward the Packet Consolidation Protocol packet for transmission to the Security Sublayer.

3.4.7.4.1 Pad

When creating a Packet Consolidation Protocol packet, the access network and the access terminal shall add sufficient padding so that the packet fills the Security Sublayer payload and set the padding bits to ‘0’. When receiving a Packet Consolidation Protocol packet, the access network and the access terminal shall ignore the padding bits.

3.4.7.5 Access network procedures

3.4.7.5.1 Control channel

This protocol does not transmit over the Control Channel.

3.4.7.5.2 Broadcast forward traffic channel

All transport packets sent in a Packet Consolidation Protocol packet should be destined to all MAC IDs.

3.4.7.5.3 Unicast forward traffic channel

All transport packets sent in a Packet Consolidation Protocol packet should be destined to one MAC ID.

3.4.8 Packet Consolidation Protocol header

The sender adds the following header in front of every transport packet encapsulated in a Packet Consolidation Protocol packet:

Field	Length (bits)
IsTransport	1
Transport	3
Length	12

IsTransport	This field shall be set to 1.
Transport	The sender shall set this field to the Transport number associated with the transport sending the transport packet following the header.
Length	This field shall be set to the length of the transport packet in octets.

The transport packet shall be at least one byte. The header value 0x00 shall indicate the beginning of the Pad, and the receiver shall ignore a Packet Consolidation Protocol packet beyond the 0x00 header. The Packet Consolidation Protocol packet format is described in Figure 40. In case the transport packets together with the Packet Consolidation Protocol headers fill the entire available payload, the pad and pad header shall be omitted.

3.4.9 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes.

3.4.10 Interface to other protocols

3.4.10.1 Commands

This protocol does not issue any commands.

3.4.10.2 Indications

This protocol does not register to receive any indications.

3.4.11 Configuration attributes

The following complex attribute and default values are defined (see 10.3 for attribute record definition).

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Data Transport.

3.4.11.1 TransportConfiguration attribute

Field	Length (bits)	Default
Length	8	N/A
AttributeID	8	N/A
Transport0	16	0x0000
Transport1	16	0xffff
Transport2	16	0xffff
Transport3	16	0xffff
Transport4	16	0xffff
Transport5	16	0xffff
Transport6	16	0xffff
Transport7	16	0xffff

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x00.

Transport0 — The sender shall set this field to the subtype of the transport used over Transport 0.

Transport1 — The sender shall set this field to the subtype of the transport used over Transport 1.

Transport2 — The sender shall set this field to the subtype of the transport used over Transport 2.

Transport3 — The sender shall set this field to the subtype of the transport used over Transport 3.

Transport4 — The sender shall set this field to the subtype of the transport used over Transport 4.

Transport5 — The sender shall set this field to the subtype of the transport used over Transport 5.

Transport6 — The sender shall set this field to the subtype of the transport used over Transport 6.

Transport7 — The sender shall set this field to the subtype of the transport used over Transport 7.

The sender shall set the Transport*N* fields to one of the non-reserved values for the Transport Subtype as specified in Table 21.

3.4.12 Protocol numeric constants

Constant	Meaning	Value
$N_{PCPType}$	Type field for this protocol.	Table 9
$N_{PCPDefault}$	Subtype field for this protocol	0x0000

3.4.13 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of only the configuration attributes of this protocol.

4 Security Control Sublayer

4.1 Introduction

4.1.1 General overview

The Security Control sublayer provides the following functions:

- **Key Exchange**: Provides the procedures followed by the access network and by the access terminal to exchange security keys for authentication and encryption.

The Security Control Sublayer uses the Key Exchange Protocol to provide these functions.

4.2 Default Key Exchange Protocol

4.2.1 Overview

The Default Key Exchange Protocol provides a method for simultaneous generation of the session key at the access terminal and the access network. The session key is derived from a PairwiseMasterKey (PMK) that is negotiated by higher layer protocols and assumed available at the access terminal and the access network. This protocol supports cases where there may be multiple PairwiseMasterKeys. The procedure for deriving the PairWiseMasterKey is considered to be out of scope for this document.

The session key is used to derive the MIC Key, Authentication Key and Encryption Key. The MIC key is used to verify the four way exchange messages of this protocol. The Authentication Key may be used to authenticate packets (see the Authentication Protocol for details), and the Encryption Key may be used to encrypt packets (see the Encryption Protocol for details).

This protocol also provides methods and messages to change session (security) key after a session has been established.

4.2.2 Primitives

4.2.2.1 Commands

This protocol does not define any commands.

4.2.2.2 Return indications

- *FirstKeyComplete*

4.2.3 Public data

4.2.3.1 Static public data

This protocol does not define any static public data.

4.2.3.2 Dynamic public data

- Subtype for this protocol
- FLAuthKey and its length
- RLAuthKey and its length
- FLEncKey and its length
- RLEncKey and its length
- KeyChangeInitiated

4.2.4 Protocol data unit

The transmission unit of this protocol is a message. This is a control protocol and, therefore, it does not carry payload on behalf of other layers or protocols.

This protocol uses the Signaling Application to transmit and receive messages.

4.2.5 Protocol initialization and swap

4.2.5.1 Protocol initialization

Upon initialization, the value of the attributes for this protocol instance in the access terminal and the access network shall be set to the following default values specified for each attribute.

- Set SKey[i] to zero and its length to 384, for values of i from 0 through 7.
- Set FLAuthKey[i] to zero and its length to 128, for values of i from 0 through 7.
- Set RLAuthKey[i] to zero and its length to 128, for values of i from 0 through 7.
- Set FLEncKey[i] to zero and its length to 128, for values of i from 0 through 7.
- Set RLEncKey[i] to zero and its length to 128, for values of i from 0 through 7.
- Set ATNonce to NULL.
- Set ANNonce to NULL.
- Set LastValidTransactionID to 255.
- Set KeyChangeInitiated to ‘0’.

4.2.5.2 Protocol swap

- Set LastValidTransactionID to 255 upon protocol swap.

4.2.6 Procedures

The Default Key Exchange Protocol uses the KeyRequest, KeyResponse, ANKeyComplete, and ATKeyComplete messages to derive secret session keys, verify that the access terminal and the access network have derived the same session keys, and to exchange security capabilities.

This protocol is able to swap the current session key that is in use with another key that has already been derived from the PMK. This is done using the KeyChange bit included in the MAC header, as well as KeyChangeRequest and KeyChangeAck messages.

4.2.6.1 Access terminal requirements

4.2.6.1.1 Processing the KeyRequest message

Upon receiving the KeyRequest message, the access terminal shall perform the following:

- The access terminal shall declare the message to be valid if the TransactionID of the message does not match the TransactionID of any outstanding KeyRequest message.
- If the KeyRequest message is not valid, then the access terminal shall send an ATKeyComplete message with ResultCode set to 'Transaction ID Invalid', declare failure, and stop performing the rest of the key exchange procedure.
- The access terminal shall identify the PairwiseMasterKey that satisfies PairwiseMasterKeyID = HMAC-SHA256-128 (PairwiseMasterKey, "PMK_Name" | SessionSeed).
  - PairwiseMasterKeyID is a field of the received KeyRequest message, "PMK_Name" is the ASCII encoded value of the string.
  - HMAC-SHA256-128 function is specified in 4.2.6.5.
  - SessionSeed is public data of the Session Management Protocol
  - The notation "|" implies concatenation.
- If the access terminal cannot identify a valid PairwiseMasterKey that satisfies the above Equation, then the access terminal shall declare failure and shall send an ATKeyComplete message with ResultCode set to 0x03, declare failure, and stop performing the rest of the key exchange procedure.
- The access terminal should set ATNonce to PRF(Random number, "Init_Counter", PhyFrameIndex64, 256),
  - Random number is a 256-bit random number. This number may be generated according to the pseudorandom number generator specified in 10.6. If the procedure of 10.6 is used and a physical random number χ is available, a fresh initialization should be used each time the random number is generated.
  - "Init_Counter" is the ASCII encoded value of the string.
  - PhyFrameIndex64 is the 64-bit representation of the PHY Frame Index defined in the Lower MAC Sublayer.
  - PRF function is specified in 4.2.6.4.
- The access terminal shall compute SKey[i] the session key in the following way:

  SKey[i] = PRF(PairwiseMasterKey, "Pairwise_Key_Expansion", SessionSeed|Nonce1|Nonce2, 384).

  - Where i is the SessionKeyIndex field of the corresponding KeyRequest message, Nonce1 = Min(ATNonce, ANNonce), Nonce2 = Max(ATNonce, ANNonce),
  - ANNonce is the ANNonce field of the received KeyRequest message, PairwiseMasterKey is the key associated with the PairwiseMasterKeyID field of the KeyRequest message.
  - "Pairwise_Key_Expansion" is the ASCII encoded value of the string.

- SessionSeed is public data of the Session Management Protocol.
- PRF function is specified in 4.2.6.4.

- The access terminal shall generate MIC Key, Authentication Key and Encryption Key as specified in 4.2.6.3.
- The access terminal shall send a KeyResponse message.

4.2.6.1.2 Processing the ANKeyComplete message

After receiving an ANKeyComplete message with a TransactionID field that matches the TransactionID field of the associated KeyRequest message, the access terminal shall perform the following:

- The access terminal shall generate a MessageIntegrityCode as HMAC-SHA256-128(MICKey[i], *Message*), where *Message* is the received ANKeyComplete message with the MessageIntegrityCode field set to zero, *i* is the SessionKeyIndex field of the corresponding KeyRequest message, and the HMAC-SHA256-128 function is specified in [10].
- The access terminal shall set LastValidTransactionID to the TransactionID field of the ANKeyComplete message and send an ATKeyComplete message with the ResultCode field set to 0x00, unless one of the following conditions holds. In that case, the access terminal shall declare failure and send an ATKeyComplete message with the appropriate ResultCode.
  - If the MessageIntegrityCode computed in the previous step does not match the MessageIntegrityCode field of the ANKeyComplete message In this case, the access terminal shall use the ResultCode corresponding to "Message integrity code failed")
  - Otherwise, if the supported tokens sent by the access network in the ANKeyComplete message include a token that the access terminal supports and prefers to use to the token currently in use (SessionConfigurationToken in the public data of the Session Configuration Protocol). In this case, the access terminal shall use the ResultCode corresponding to "Message integrity code successful, but capabilities verification failed".
- If the access terminal sends a ATKeyComplete message with ResultCode field set to 0x00, and the key exchange was performed for SessionKeyIndex set to the configuration attribute SessionKeyIndexInUse, the access terminal shall generate a *FirstKeyComplete* indication.

4.2.6.1.3 Processing the KeyChangeRequest message

Only the access network is permitted to initiate a key change by sending a KeyChangeRequest message. The access terminal may request initiation of a key change by sending a KeyChangeInitiateRequest message with the appropriate SessionKeyIndexRequested field.

Upon receipt of a KeyChangeRequest message, the access terminal shall verify that KeyChangeInitiated='0'. If not, the access terminal shall abort any key exchange in progress by setting SessionKeyIndexPending to 0xff.

If the access terminal proceeds with the key change process, it shall

- Set the SessionKeyIndexPending to the value received in the KeyChangeRequest message
- Respond with a KeyChangeAck message and set KeyChangeInitiated='1'.
  - In the KeyChangeAck message, the access terminal shall copy TransactionID and KeyIndexPending from the KeyChangeRequest message that caused the generation of this KeyChangeAck message.

4.2.6.1.4 Processing the KeyChange bit

If the access terminal receives a packet with the KeyChange bit in the Forward Traffic Channel MAC header toggled the access terminal shall verify that KeyChangeInitiated is equal to '1'. If not, the access terminal shall abort any key exchange in progress and set SessionKeyIndexPending to 0xff.

If KeyChangeInitiated is equal to '1', the access terminal shall

- Set SessionKeyIndexInUse to the value of SessionKeyIndexPending and set SessionKeyIndexPending to 0xff.
- Update the values of MIC Key, FLAuthKey, RLAuthKey, FLEncKey and RLEncKey following the procedure specified in 4.2.6.3 before the received packet is processed by the Security Sublayer.
- Set KeyChangeInitiated to '0' (completing the key change process).
- Toggle the value of the KeyChange bit in the Reverse Traffic Channel MAC header for subsequent transmissions.

4.2.6.2 Access network requirements

The access network shall initiate the key exchange by sending a KeyRequest message. The access network shall choose a nonce, ANNonce as follows:

- The access network should set ANNonce to
  - PRF(Random number, "Init_Counter", AP SectorID | PhyFrameIndex64, 256).
  - Random number is a 256-bit random number. This number may be generated according to the pseudorandom number generator specified in 10.6. If the procedure of 10.6 is used and a physical random number χ is available, a fresh initialization should be used each time the random number is generated.
  - "Init_Counter" is the ASCII encoded value of the string, and PhyFrameIndex64 is the 64-bit representation of the Phy Frame Index defined in the Lower MAC Sublayer.
  - PRF function is specified in 4.2.6.4.

4.2.6.2.1 Processing the KeyInitiateRequest message

Upon receiving the KeyInitiateRequest message, the access network shall perform the following:

- The access network shall identify the PairwiseMasterKey that satisfies PairwiseMasterKeyID = HMAC-SHA256-128 (PairwiseMasterKey, "PMK_Name" | SessionSeed).
    - PairwiseMasterKeyID is a field of the received KeyRequest message, “PMK_Name” is the ASCII encoded value of the string.
    - SessionSeed is public data of the Session Management Protocol.
    - HMAC-SHA256-128 function is specified in [10].
    - The notation “|” implies concatenation.

If the access network can identify a valid PairwiseMasterKey that satisfies the above equation, then the access network may initiate a session key exchange by sending a KeyRequest message.

4.2.6.2.2 Processing the KeyResponse message

After receiving a KeyResponse message with a TransactionID field that matches the TransactionID field of the associated KeyRequest message, the access network shall perform the following:

- The access network shall compute SKey[i], the session key as follows:

  SKey[i] = PRF(PairwiseMasterKey, “Pairwise_Key_Expansion”, SessionSeed|Nonce1|Nonce2, 384).
    - Where *i* is the SessionKeyIndex field of the corresponding KeyRequest message,
    - “Pairwise_Key_Expansion” is the ASCII encoded value of the string, Nonce1 = Min(ATNonce, ANNonce), Nonce2 = Max(ATNonce, ANNonce).
    - ATNonce is the ATNonce field of the KeyResponse message, and the PRF function is specified in 4.2.6.4.
    - SessionSeed is public data of the Session Management Protocol.
- The access network shall generate MIC Key, Authentication Key and Encryption Key as specified in 4.2.6.3.
- The access network shall generate a MessageIntegrityCode as HMAC-SHA256-128 (MICKey[i], *Message*), where *Message* is the received KeyResponse message with the MessageIntegrityCode field set to zero, *i* is the SessionKeyIndex field of the corresponding KeyRequest message, and the HMAC-SHA256-128 function is specified in [10].
- The access network shall send an ANKeyComplete message and increment LastValidTransactionID unless one of the following conditions holds. In that case, the access network shall declare failure and send an ANKeyComplete message with the appropriate ResultCode.
    - If the MessageIntegrityCode computed in the previous step does not match the MessageIntegrityCode field of KeyResponse message.

- If the supported tokens sent by the access terminal in the KeyResponse message contain a token that the access network supports and prefers to use to the token currently in use (SessionConfigurationToken in the public data of the Session Configuration Protocol).

4.2.6.2.3 Processing the ATKeyComplete message

If the access network receives an ATKeyComplete message with ResultCode field set to a value other than 0x00, the access network shall declare failure and stop performing the rest of the key exchange procedure.

If the access network receives a ATKeyComplete message with ResultCode field set to 0x00, and the key exchange was performed for SessionKeyIndex set to the configuration attribute SessionKeyIndexInUse, the access network shall generate a *FirstKeyComplete* indication.

4.2.6.2.4 Transmitting the KeyChangeRequest message

Upon receipt of KeyChangeInitiateRequest message, the access network may initiate a key change.

The access network shall initiate the key change by sending a KeyChangeRequest message only if no key change request initiated by the access network is in progress, i.e. if the key change timer is inactive and the KeyChangeInitiated bit is set to ‘0’.

Upon sending a KeyChangeRequest message, the access network shall

- start a key change timer for $T_{KEPTimer}$ and abort the key change process by setting SessionKeyIndexPending to 0xff if the timer expires.
- set SessionKeyIndexPending to the proposed SessionKeyIndex.

4.2.6.2.5 Processing the KeyChangeAck message

The access network shall ignore the KeyChangeAck message if KeyChangeInitiated=’1’. Otherwise, upon receipt of a KeyChangeAck message the access network shall

- Set KeyChangeInitiated to ’1’.
- Disable the key change timer.
- Set SessionKeyIndexInUse to the value of SessionKeyIndexPending.
- Toggle the KeyChange bit in the Forward Traffic Channel MAC header for subsequent packet transmissions.
- Update the values of MIC Key, FLAuthKey and FLEncKey following the procedure specified in 4.2.6.3 before the next packet is processed for transmission.

4.2.6.2.6 Processing the KeyChange bit

If KeyChangeInitiated is equal to ‘1’ and the access network receives a packet with the KeyChange bit in the Reverse Traffic Channel MAC header toggled, the access network shall.

- Set SessionKeyIndexPending to 0xff.
- Update the values of MIC Key, RLAuthKey, and RLEncKey following the procedure specified in 4.2.6.3 before the received packet is processed by the Security Sublayer.

- Set KeyChangeInitiated to '0' (completing the key change process).

4.2.6.3 MIC Key, Authentication Key, and Encryption Key generation

The keys used for message integrity code, authentication and encryption are generated from the session key using the procedures specified in this section.

The access network and the access terminal shall compute and store a MIC Key, Authentication Key, and Encryption Key derived from each session key. The keys derived from SKey[*i*] are referred to by the subscript *i*. The Encryption and Authentication Protocols at the access network and the access terminal shall use the Authentication Key and Encryption Key derived from the SKey with index *i* set to the SessionKeyIndexInUse.

The MIC Key, Authentication Key and Encryption Key attributes are computed in the following way.

- The access network and the access terminal shall set the MICKey[*i*] to SKey[*i*][127:0], where *i* is the session key index.
- The access network and the access terminal shall set FLAuthKey[*i*] and RLAuthKey[*i*] to SKey[*i*][255:128], where *i* is the session key index.
- The access network and the access terminal shall set FLEncKey[*i*], and RLEncKey[*i*] to SKey[*i*][383:256], where *i* is the session key index.

4.2.6.4 Pseudorandom function, PRF(*K, A, B, Len*)

A pseudorandom function (PRF) is used in a number of places in this document.

Len shall be no greater than 255*160.

The output of the pseudorandom function is obtained by executing the following pseudo-code:

- R = NULL
- **for** *i* = 0 **to** (*Len*+159)/160 **do**
  - R = R | HMAC-SHA256-160(K, A | Y | B | i),
  - *Y* is a single octet containing the value zero.
  - *i* is a single octet containing the parameter.
  - HMAC-SHA256-160 function is specified in [10].

The output of the PRF function shall be set to the *Len* most significant bits of R.

4.2.6.5 HMAC-SHA256(K, Message)

The HMAC-SHA1 procedure as specified in [10], shall be performed with SHA-256 [1] as the message digest algorithm.

The output of the HMAC-SHA256-128 function shall be set to the 128 Most Significant Bits of the output of the HMAC-SHA1 procedure.

The output of the HMAC-SHA256-160 function shall be set to the 160 Most Significant Bits of the output of the HMAC-SHA1 procedure.

4.2.7 Message format and flows

4.2.7.1 Message flows

4.2.7.1.1 Message flow for Default Key Exchange Protocol

This section describes the message flow for the Default Key Exchange Protocol. Figure 41 shows the message exchanges for the Default Key Exchange Protocol.

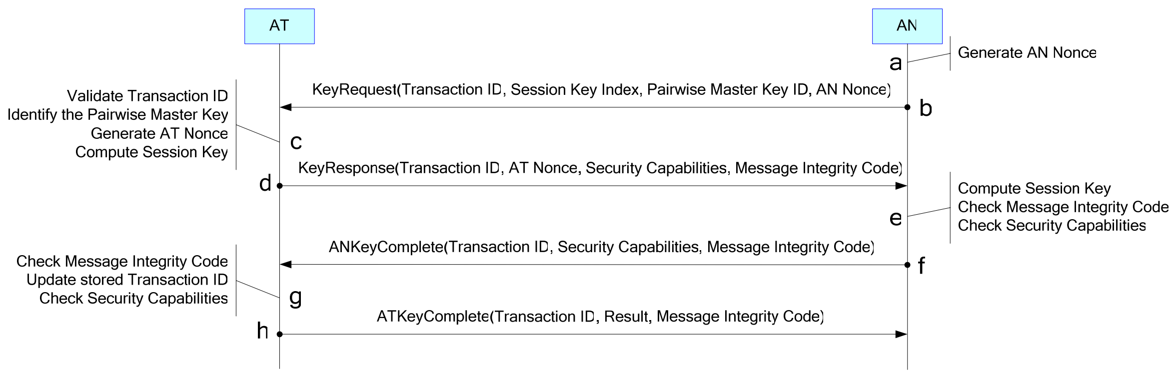

Figure 41 Default Key Exchange Protocol message flow

4.2.7.1.2 Message flow for Security Key Change Protocol

This section describes the message flow for the Key Change Protocol.

Message flow needed to execute key change is shown in Figure 42. Key change may be negotiated by the access terminal or the access network. Solid lines in Figure 42 indicate the messages exchanged between the communicating peers. The value of the KeyChange bit is assumed to be ‘0’. The dotted lines do not indicate messages related to the Key Exchange Protocol. They show the exchange of regular packets after the KeyChange bit is toggled and actual key change is executed.

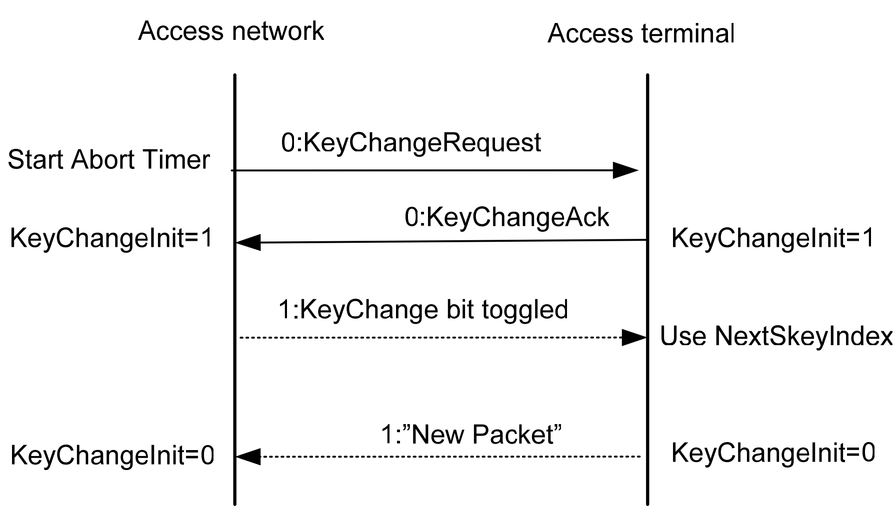

Figure 42 Security Key Change Protocol [??Figure updated above]

4.2.7.2 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes.

4.2.7.2.1 KeyInitiateRequest

The access terminal may send the KeyInitiateRequest message to request the access network to initiate a session key exchange. The access network may or may not initiate a key exchange in response to this message.

Field	Length (bits)
MessageID	8
SessionKeyIndex	8
PairwiseMaskterKeyID	128

MessageID — The access terminal shall set this field to 0x00.

SessionKeyIndex — The access terminal shall set this field to the ID of the SKey for which this key exchange is being initiated. The values 0x08 to 0xff are reserved.

PairwiseMasterKeyID — The access terminal shall set this field to HMAC-SHA256-128 (PMK, "PMK_Name" | SessionSeed), where “PMK_Name” is the ASCII encoded value of the string.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

4.2.7.2.2 KeyRequest

The access network sends the KeyRequest message to initiate the session key exchange.

Field	Length (bits)
MessageID	8
TransactionID	8
SessionKeyIndex	8
PairwiseMaskterKeyID	128
ANNonce	256

MessageID — The access network shall set this field to 0x01.

TransactionID — The access network shall set this field according to 10.8.

SessionKeyIndex — The access network shall set this field to the ID of the SKey for which this key exchange is being initiated. The values 0x08 to 0xff are reserved.

Reserved — The access network shall set this field to ‘00000’. The access terminal shall ignore this field.

PairwiseMasterKeyID — The access network shall set this field to HMAC-SHA256-128 (PMK, "PMK_Name" | SessionSeed), where “PMK_Name” is the ASCII encoded value of the string.

ANNonce — The access network shall set this field to the nonce chosen by the access network.

Channels	FTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

4.2.7.2.3 KeyResponse

The access terminal sends the KeyResponse message in response to the KeyRequest message.

Field	Length (bits)
MessageID	8
TransactionID	8
ATNonce	256
TokenCount	8
TokenCount occurrences of the following field:	
SupportedToken	16
MessageIntegrityCode	128

MessageID — The access terminal shall set this field to 0x02.

TransactionID — The access terminal shall set this field to the value of the TransactionID field of the corresponding KeyRequest message.

ATNonce — The access terminal shall set this field to the nonce chosen by the access terminal.

TokenCount — The access terminal shall set this field to the number of tokens supported by the access terminal.

SupportedToken — The access terminal shall set this field to a token supported by the access terminal.

MessageIntegrityCode — The access terminal shall set this field to HMAC-SHA256-128(MICKey[i], *Message*), where *Message* is set to all fields of this message with this field set to zero, and *i* is SessionKeyIndex field of the corresponding KeyRequest message.

Channels	RTC
Addressing	Unicast

SLP	Reliable
Security	Optional

4.2.7.2.4 ANKeyComplete

The access network sends the ANKeyComplete message in response to the KeyResponse message.

Field	Length (bits)
MessageID	8
TransactionID	8
ResultCode	8
TokenCount	8
TokenCount occurrences of the following field:	
SupportedToken	16
MessageIntegrityCode	128

MessageID — The access network shall set this field to 0x03.

TransactionID — The access network shall set this field to the value of the TransactionID field of the corresponding KeyRequest message.

ResultCode — The access network shall set this field according to Table 22.

Table 22 Definition of result field

Value	Meaning
0x00	Security capabilities verification and message integrity code successful
0x01	Message integrity code failed
0x02	Message integrity code successful, but capabilities verification failed.
0x03	Pairwise MasterKey not found.
0x04	Transaction ID invalid.
All other values	Reserved

TokenCount — The access network sets this field to the number of tokens that the access network supports and includes in this message.

SupportedToken — The access network shall set this field to a token supported by the access network.

MessageIntegrityCode — The access network shall set this field to HMAC-SHA256-128(MICKey[i], Message), where Message is set to all fields of this message with this field set to zero, and *i* is the SessionKeyIndex field of the corresponding KeyRequest message.

Channels	FTC
Addressing	Unicast

SLP	Reliable
Security	Required

4.2.7.2.5 ATKeyComplete

The access terminal sends the ATKeyComplete message in response to the ANKeyComplete message.

Field	Length (bits)
MessageID	8
TransactionID	8
ResultCode	8
MessageIntegrityCode	0 or 128
LastTransactionID	0 or 8

MessageID — The access terminal shall set this field to 0x04.

TransactionID — The access terminal shall set this field to the value of the TransactionID field of the corresponding KeyRequest message.

ResultCode — The access terminal shall set this field according to Table 23.

Table 23 Definition of result field

Value	Meaning
0x00	Security capabilities verification and message integrity code successful
0x01	Message integrity code failed
0x02	Message integrity code successful, but capabilities or token verification failed.
0x03	Pairwise MasterKey not found.
0x04	Transaction ID invalid.
0x05	Key exchange procedures not supported.
All other values	Reserved

MessageIntegrityCode — If the Result field is 0x01 or 0x03, the access terminal shall omit this field. Otherwise, the access terminal shall set this field to HMAC-SHA256-128(MICKey[i], *Message*), where *Message* is set to all fields of this message with this field set to zero, and *i* is the SessionKeyIndex field of the corresponding KeyRequest message.

LastTransactionID — If the MessageIntegrityCode field is set to ‘0x04’, then the access terminal shall set this field to the value of the LastValidTransactionID parameter. Otherwise, the access terminal shall omit this field.

Channels	RTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

4.2.7.2.6 KeyChangeInitiateRequest

The KeyChangeInitiateRequest message is sent by the access terminal to request the access network to initiate a key change.

Field	Length (bits)
MessageID	8
SessionKeyIndexRequested	8

MessageID The sender shall set this field to 0x05.

SessionKeyIndexRequested
The sender shall set this value to the index of requested session key. The values 0x08 to 0xff are reserved.

Channels	RTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

4.2.7.2.7 KeyChangeRequest

The KeyChangeRequest message is sent the by the access network to initiate a key change.

Field	Length (bits)
MessageID	8
TransactionID	8
SessionKeyIndexPending	8

MessageID The sender shall set this field to 0x06.

TransactionID The sender shall set this field according to 10.8.

SessionKeyIndexPending
The sender shall set this value to the index of proposed session key. The values 0x08 to 0xff are reserved.

Channels	FTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

4.2.7.2.8 KeyChangeAck

The KeyChangeAck message is sent by the access terminal to acknowledge the receipt of a KeyChangeRequest message.

Field	Length (bits)
MessageID	8
TransactionID	8
SessionKeyIndexPending	8

MessageID The sender shall set this field to 0x07.

TransactionID The sender shall set this value to the TransactionID field of the corresponding KeyChangeRequest message.

SessionKeyIndexPending
The sender shall set this value to the index of the proposed session key. The values 0x08 to 0xff are reserved.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

4.2.8 Interface to other protocols

4.2.8.1 Commands

This protocol does not issue any commands.

4.2.8.2 Indications

This protocol does not register to receive any indications.

4.2.9 Configuration attributes

The configurable, simple attributes for this protocol are listed in Table 24. The access terminal shall use as defaults the values in Table 24 that are listed in ***bold italics***.

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Key Exchange Protocol. The access terminal or access network shall not use the Generic Attribute Update Protocol in 10.9 to update the SessionKeyIndexInUse attribute.

Table 24 Configurable values

Attribute ID	Attribute	Values	Meaning
0x00	SessionKeyIndexInUse	***0x00***	$SKey_0$ is used
		0x01	$SKey_1$ is used
		0x02	$SKey_2$ is used
		0x03	$SKey_3$ is used
		0x04	$SKey_4$ is used
		0x05	$SKey_5$ is used
		0x06	$SKey_6$ is used
		0x07	$SKey_7$ is used
		0x08 – 0xff	Reserved
0x01	SessionKeyIndexPending	0x00	$SKey_0$ is pending.
		0x01	$SKey_1$ is pending
		0x02	$SKey_2$ is pending
		0x03	$SKey_3$ is pending
		0x04	$SKey_4$ is pending
		0x05	$SKey_5$ is pending
		0x06	$SKey_6$ is pending
		0x07	$SKey_7$ is pending
		0x08 – 0xfe	Reserved
		0xff	Pending Skey is not defined

4.2.10 Protocol numeric constants

Constant	Meaning	Value
$N_{KEPType}$	Type field for this protocol	Table 9
N_{KEPG}	Subtype field for this protocol	0x0001
$T_{KEPTimer}$	Timer duration for response to KeyExchangeRequest message	500 ms

4.2.11 Session state information

The Session State Information record (see 10.10) consists of parameter records.

This protocol defines the following parameter record in addition to the configuration attributes for this protocol.

4.2.11.1 SKey parameter

Table 25 Format of the parameter record for the SKey parameter

Field	Length (bits)
ParameterType	8
Length	8
$SKey_0$Included	1
$SKey_1$Included	1
$SKey_2$Included	1
$SKey_3$Included	1
$SKey_4$Included	1
$SKey_5$Included	1
$SKey_6$Included	1
$SKey_7$Included	1
$SKey_0$	0 or 384
$SKey_1$	0 or 384
$SKey_2$	0 or 384
$SKey_3$	0 or 384
$SKey_4$	0 or 384
$SKey_5$	0 or 384
$SKey_6$	0 or 384
$SKey_7$	0 or 384

ParameterType — This field shall be set to 0x01 for this parameter record.

Length — This field shall be set to the length of this parameter record in units of octets excluding the Length field.

$SKey_0$Included — If $SKey_0$ is zero, then this field shall be set to '0'. Otherwise, this field shall be set to '1'.

$SKey_1$Included — If $SKey_1$ is zero, then this field shall be set to '0'. Otherwise, this field shall be set to '1'.

$SKey_2$Included — If $SKey_2$ is zero, then this field shall be set to '0'. Otherwise, this field shall be set to '1'.

$SKey_3$Included	If $SKey_3$ is zero, then this field shall be set to ‘0’. Otherwise, this field shall be set to ‘1’.
$SKey_4$Included	If $SKey_4$ is zero, then this field shall be set to ‘0’. Otherwise, this field shall be set to ‘1’.
$SKey_5$Included	If $SKey_5$ is zero, then this field shall be set to ‘0’. Otherwise, this field shall be set to ‘1’.
$SKey_6$Included	If $SKey_6$ is zero, then this field shall be set to ‘0’. Otherwise, this field shall be set to ‘1’.
$SKey_7$Included	If $SKey_7$ is zero, then this field shall be set to ‘0’. Otherwise, this field shall be set to ‘1’.
$SKey_0$	If $SKey_0$Included is ‘0’, then this field shall be omitted. Otherwise, this field shall be set to the value of the session key with key index 0x00.
$SKey_1$	If $SKey_1$Included is ‘0’, then this field shall be omitted. Otherwise, this field shall be set to the value of the session key with key index 0x01.
$SKey_2$	If $SKey_2$Included is ‘0’, then this field shall be omitted. Otherwise, this field shall be set to the value of the session key with key index 0x02.
$SKey_3$	If $SKey_3$Included is ‘0’, then this field shall be omitted. Otherwise, this field shall be set to the value of the session key with key index 0x03.
$SKey_4$	If $SKey_4$Included is ‘0’, then this field shall be omitted. Otherwise, this field shall be set to the value of the session key with key index 0x04.
$SKey_5$	If $SKey_5$Included is ‘0’, then this field shall be omitted. Otherwise, this field shall be set to the value of the session key with key index 0x05.
$SKey_6$	If $SKey_6$Included is ‘0’, then this field shall be omitted. Otherwise, this field shall be set to the value of the session key with key index 0x06.
$SKey_7$	If $SKey_7$Included is ‘0’, then this field shall be omitted. Otherwise, this field shall be set to the value of the session key with key index 0x07.

4.2.11.2 Nonce parameter

Table 26 Format of the parameter record for the Nonce parameter

Field	Length (bits)
ParameterType	8
Length	8
NULLATNonce	1
NULLANNonce	1
Reserved	6
ATNonce	0 or 256
ANNonce	0 or 256

ParameterType — This field shall be set to 0x02 for this parameter record.

Length — This field shall be set to the length of this parameter record in units of octets excluding the Length field.

NULLATNonce — If ATNonce is NULL, then this field shall be set to ‘1’. Otherwise, this field shall be set to ‘0’.

NULLANNonce — If ANNonce is NULL, then this field shall be set to ‘1’. Otherwise, this field shall be set to ‘0’.

Reserved — This field shall be set to ‘000000’. The receiver shall ignore this field.

ATNonce — If NULLATNonce is ‘1’, then this field shall be omitted. Otherwise, this field shall be set to the value of the ATNonce.

ANNonce — If NULLANNonce is ‘1’, then this field shall be omitted. This field shall be set to the value of the ANNonce.

4.2.11.3 LastValidTransactionID parameter

Table 27 Format of the parameter record for the LastValidTransactionID parameter

Field	Length (bits)
ParameterType	8
Length	8
LastValidTransactionID	8

ParameterType — This field shall be set to 0x03 for this parameter record.

Length — This field shall be set to the length of this parameter record in units of octets, excluding the Length field.

LastValidTransactionID — This field shall be set to the value of the LastValidTransactionID parameter.

4.2.11.4 PMK parameter

Table 28 Format of the parameter record for the PMK parameter

Field	Length (bits)
ParameterType	8
Length	8
PMKCount	8
PMKCount occurrences of the following two fields:	
PMKLength	8
PMK	PMKLength × 8

ParameterType — This field shall be set to 0x04 for this parameter record.

Length — This field shall be set to the length of this parameter record in units of octets excluding the Length field.

PMKCount — This field shall be set to the number of occurrences of the PMK field in this parameter record.

PMKLength — This field shall be set to the length of the PMK field in units of octets.

PMK — This field shall be set to a PairwiseMasterKey.

5 Security Sublayer

5.1 Introduction

5.1.1 General overview

The Security Sublayer provides the following functions:

- **Cryptosync Generation:** Provides a cryptosync for use by the Authentication and Encryption protocols in the Security Sublayer
- **Authentication**: Provides the procedures followed by the access network and the access terminal for authenticating traffic.
- **Encryption**: Provides the procedures followed by the access network and the access terminal for encrypting traffic.

The Security Sublayer uses the Authentication Protocol, Encryption Protocol, and Security Protocol to provide these functions. In particular, the Security Protocol provides the cryptosync needed by the authentication and encryption protocols, the Authentication Protocol provides authentication, and the Encryption Protocol provides encryption. Figure 43 shows the protocols within the Security Sublayer.

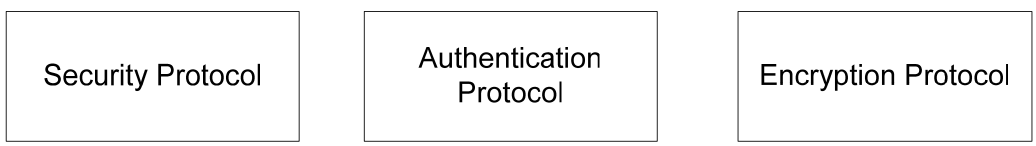

Figure 43 Security Sublayer protocols

5.2 Packet encapsulation for the protocol instances

In the transmit direction, the Security Sublayer receives a Convergence Sublayer Packet, accompanied by a IsSecure field. The Security Sublayer processes this packet and delivers a Lower MAC Payload to the Lower MAC Sublayer, accompanied by the IsSecure field.

In the receive direction, the Security Sublayer receives a Lower MAC Sublayer Packet, accompanied by a IsSecure field. The Security Sublayer processes this packet and delivers a Convergence Sublayer Packet to the Convergence Sublayer, accompanied by a IsSecure field.

Packet encapsulation for the Security Sublayer operates in a different way for the secure and unsecure packets, as described in next.

5.2.1 Packet encapsulation with IsSecure set

When the IsSecure field is set to ‘1’, Figure 44 illustrates the relationship between a Convergence Sublayer packet, an Encryption Protocol packet, an Authentication Protocol packet, a Security Sublayer packet, and the Lower MAC Sublayer payload. The order of Authentication and Encryption is such that it can avoid unnecessary decryption when authentication fails.

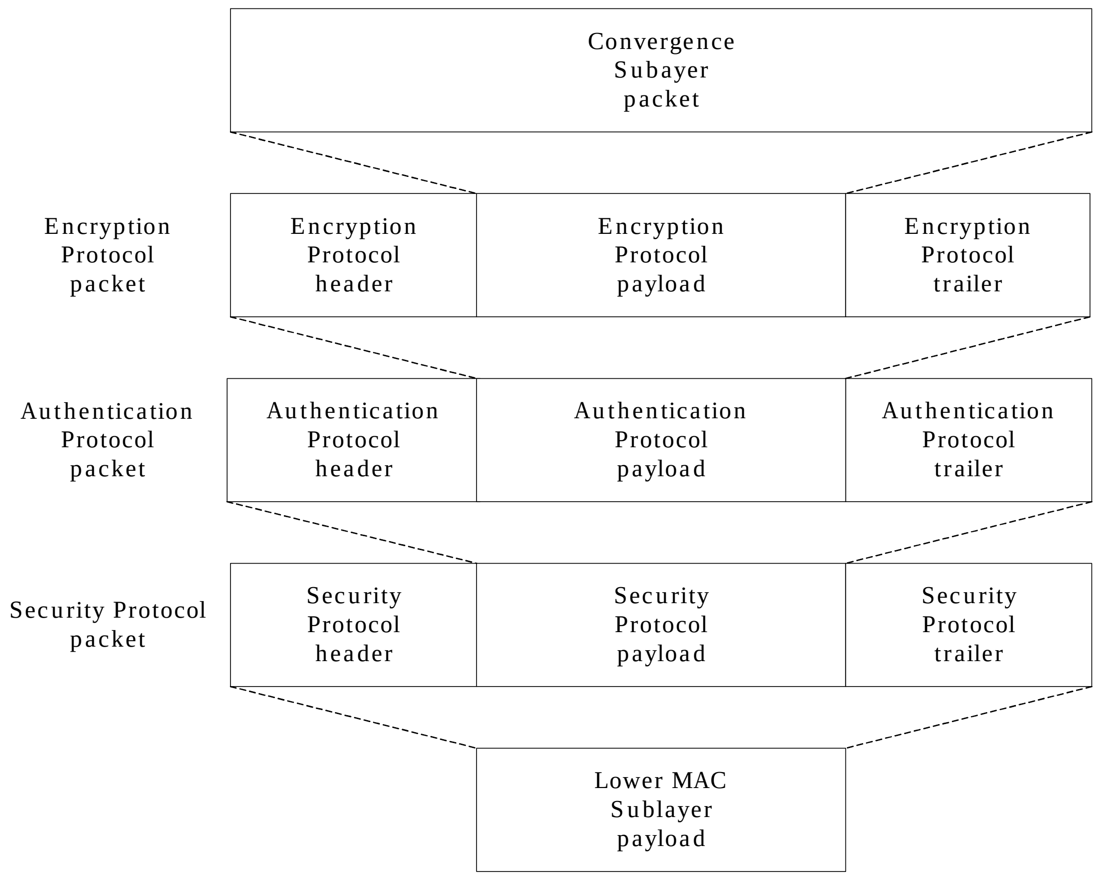

Figure 44 Security Sublayer data encapsulation for IsSecure=1

The Security Sublayer headers or trailers may or may not be present (or equivalently, have a size of zero) if the SessionConfigurationToken specifies the Default Security Protocol or if the configured Security Protocols do not require a header or trailer.

The Encryption Protocol may add a trailer to hide the actual length of the plaintext or padding to be used by the encryption algorithm. The Encryption Protocol Header may contain variables such as an initialization vector (IV) to be used by the Encryption Protocol.

The Authentication Protocol header or trailer may contain the Message Authentication Code that is used to authenticate the portion of the Authentication Protocol Packet that is authenticated.

The Security Protocol header or trailer may contain variables needed by the authentication and encryption protocols (e.g., cryptosync, time-stamp, etc.).

5.2.2 Packet encapsulation with IsSecure not set

If the IsSecure field is set to zero, the relation between a Convergence Sublayer Packet and a Lower MAC Sublayer payload is as shown in Figure 45. The packet does not pass through the Security Protocol, Authentication Protocol and Encryption Protocol.

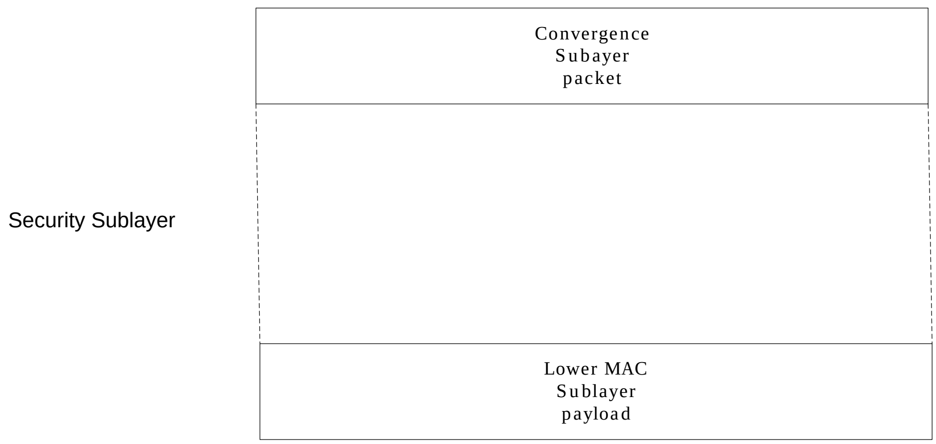

Figure 45 Security Sublayer data encapsulation for IsSecure=0

5.2.3 Security Sublayer data transmit operation overview

When a Convergence Sublayer Packet with IsSecure=0 is delivered to the Security Sublayer, the Security Sublayer sets the Lower MAC Sublayer payload to the Convergence Sublayer Packet.

When a Convergence Sublayer packet with IsSecure=1 is delivered to the Security Sublayer, the following steps are performed by the protocols in the Security Sublayer in the order specified in the following:

- The Security Protocol generates a cryptosync for the channel for which the Convergence Sublayer packet is destined. This value is called TheCryptosync for referencing it in the following steps.
- The Convergence Sublayer packet and TheCryptosync are delivered to the Encryption Protocol.
- The Encryption Protocol uses TheCryptosync, the encryption key, and other parameters specified by the Encryption Protocol (if any) to encrypt the Convergence Sublayer packet and construct the Encryption Protocol packet.
- The Encryption Protocol delivers the Encryption Protocol packet and TheCryptosync to the Authentication Protocol.
- The Authentication Protocol uses TheCryptosync, authentication key, and other parameters specified by the Authentication Protocol to construct the Authentication Protocol packet.
- The Authentication Protocol delivers the Security Sublayer packet to the Security Protocol.
- The Security Protocol delivers the Security Sublayer packet and other specified parameters (if any) to the Lower MAC Sublayer.

5.2.4 Security Sublayer data receive operation overview

When a Lower MAC Sublayer payload with IsSecure=0 is delivered to the Security Sublayer, the Security Sublayer sets the Convergence Sublayer Packet to the Lower MAC Sublayer payload.

When the Security Sublayer receives a Lower MAC Sublayer Packet payload with IsSecure=1, the following steps are performed by the protocols in the Security Sublayer in the order specified below:

- The Security Protocol constructs the cryptosync using information from the Lower MAC Sublayer and the Security Sublayer Protocol header and trailer (if any). For the purpose for referencing this value of cryptosync in the following steps, denote this value as TheCryptosync.
- The Security Protocol removes the Security Protocol header and trailer (if any) and delivers TheCryptosync and the Security Protocol payload to the Authentication Protocol.
- The Authentication Protocol uses TheCryptosync, authentication key, Authentication Protocol payload, Authentication Protocol header and trailer, and other parameters specified by the Authentication Protocol (if any) to verify the authentication signature. If the authentication signature passes, the Authentication Protocol delivers the Authentication Protocol payload to the Encryption Protocol; otherwise, the Authentication Protocol Packet is discarded.
- The Encryption Protocol uses TheCryptosync and the encryption key to decrypt the Encryption Protocol packet. The decrypted payload is then delivered to the Convergence Sublayer.

5.3 Default Encryption Protocol

5.3.1 Overview

The Default Encryption Protocol does not alter the Convergence Sublayer packet payload and does not add an Encryption Protocol header or trailer. It transfers packets between the Authentication Protocol and the Security Protocol.

5.3.2 Primitives

5.3.2.1 Commands

This protocol does not define any commands.

5.3.2.2 Return indications

This protocol does not return any indications.

5.3.3 Public data

5.3.3.1 Static public data

This protocol does not define any static public data.

5.3.3.2 Dynamic public data

- Subtype for this protocol

5.3.4 Protocol data unit

The protocol data unit for this protocol is an Encryption Protocol Packet.

5.3.5 Protocol initialization and swap

5.3.5.1 Protocol initialization

Upon initialization, the value of the attributes for this protocol instance in the access terminal and the access network shall be set to the default values specified for each attribute.

5.3.5.2 Protocol swap

This protocol defines an empty swap procedure.

5.3.6 Procedures

On the transmit side, this protocol shall receive a Convergence Sublayer packet, and it shall forward the packet to Authentication Protocol.

On the receive side, this protocol shall receive an Authentication Protocol packet, and it shall forward the packet to Convergence Sublayer.

5.3.7 Default Encryption Protocol header and trailer

The Default Encryption Protocol does not add a header or a trailer.

5.3.8 Message formats

No messages are defined for this protocol.

5.3.9 Interface to other protocols

5.3.9.1 Commands

This protocol does not issue any commands.

5.3.9.2 Indications

This protocol does not register to receive any indications.

5.3.10 Configuration attributes

No configuration attributes are defined for this protocol.

5.3.11 Protocol numeric constants

Constant	Meaning	Value
N_{EPType}	Type field for this protocol	Table 9
$N_{EPDefault}$	Subtype field for this protocol	0x0000

5.3.12 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

5.4 Default Security Protocol

5.4.1 Overview

The Default Security protocol performs the following tasks:

- Procedures to enter a secure mode of operation, where the access network and access terminal may secure all unicast air interface packets.
- On the transmission side,
  - This protocol generates the cryptosync based on information provided by the Lower MAC Sublayer and makes the cryptosync publicly available. The cryptosync may be used by the negotiated Authentication Protocol and Encryption Protocol.
  - This protocol transfers packets from the Authentication Protocol to the Lower MAC Sublayer.
- On the receiving side,
  - This protocol generates the cryptosync based on information provided by the Lower MAC Sublayer and makes the cryptosync publicly available. The cryptosync may be used by the negotiated Authentication Protocol and Encryption Protocol.
  - This protocol transfers packets from the Lower MAC Sublayer to the Authentication Protocol

5.4.2 Primitives

5.4.2.1 Commands

This protocol does not define any commands.

5.4.2.2 Return indications

This protocol does not return any indications.

5.4.3 Public data

5.4.3.1 Static public data

This protocol does not define any static public data.

5.4.3.2 Dynamic public data

- Subtype for this protocol
- Cryptosync for Security Sublayer packets associated with the FL

- Cryptosync for Security Sublayer packets associated with the RL
- SecurityEnabled

5.4.4 Protocol data unit

The protocol data unit for this protocol is a Security Sublayer packet.

5.4.5 Protocol initialization and swap

5.4.5.1 Protocol Initialization

Upon initialization, the value of the attributes for this protocol instance in the access terminal and the access network shall be set to the default values specified for each attribute.

5.4.5.2 Protocol swap

This protocol defines an empty swap procedure.

5.4.6 Procedures

5.4.6.1 Secure State Procedures

This protocol shall set the SecurityEnabled public data field to the SecurityEnabled configuration attribute.

This protocol shall be said to be in a secure mode of operation (SecurityEnabled mode) if the SecurityEnabled public data is set to 1.

In the SecurityEnabled mode at the access terminal

- This protocol shall set the IsSecure bit to '1' on all transmitted packets

In the SecurityEnabled mode at the access network

- This protocol shall set the IsSecure bit to '1' on all unicast packets transmitted to the access terminal.

5.4.6.2 Generation of the Cryptosync

The Security Protocol shall compute the Cryptosync for the channel on which the Security Sublayer packet is to be sent, or on the channel on which the Security Sublayer packet is received as shown in Table 29.

Table 29 Subfield of the Cryptosync

Subfield	Length (bits)
MACID	12
PilotPN	12
ConnectCount	16
CryptoAttribute	16
PhyFrameIndex	40

MACID	This field shall be set to the MACID of the sending sector in the FL and target sector in the RL with zero padding on the MSB side if needed. A MACID larger than 12 bits is not supported.
PilotPN	This field shall be set to the PilotPN of the sending sector in the FL or, target sector in the RL with zero padding on the MSB side if needed.
ConnectCount	This field shall be set to the current value of ConnectCount, as defined in the public data of the Idle State Protocol.
CryptoAttribute	This field is encoded as specified in Table 30.

Table 30 Encoding of the CryptoAttribute Field

Bit location	Name	Meaning
0 (LSB)	ISFL	IsFL='1' implies FL; IsFL='0' implies RL
1	ISSticky	IsSticky='1' implies channel assignment is sticky; IsSticky='0' implies assignment is non-sticky.
Others	Reserved	Not defined

PHY Frame Index	This field shall be set to the value of the PHY Frame Index as defined in the Overview chapter ("Definitions" section), with bits of zero padding on the MSB side if necessary. The PHY frame index shall be measured at the beginning of the packet transmission, with respect to the sector that is receiving or transmitting the packet.

5.4.6.3 Transmit procedures

When this protocol receives an Authentication Protocol packet from the Authentication Protocol,

- It shall construct a Security Sublayer packet by adding a Security Protocol header and trailer (if any)
- Compute the Cryptosync associated with the Security Sublayer packet as shown in Table 30.
- Deliver the packet for transmission to the Lower MAC Sublayer

5.4.6.4 Receive procedures

When this protocol receives a Security Sublayer packet from the Lower MAC Sublayer, the protocol shall

- Construct an Authentication Protocol packet by removing the Security Protocol header and trailer (if any),
- Compute the Cryptosync associated with the Lower MAC Sublayer packet as shown in Table 30.
- Deliver the Authentication Protocol packet together with the computed value of the cryptosync to the Authentication Protocol.

5.4.7 Header and trailer

The Default Security Protocol does not add a header or a trailer.

5.4.8 Message formats

No messages are defined for this protocol.

5.4.9 Interface to other protocols

5.4.9.1 Commands

This protocol does not issue any commands.

5.4.9.2 Indications

This protocol registers to receive the following indications

- *KeyExchange.FirstKeyComplete*

5.4.10 Configuration attributes

This protocol defines the following configuration attributes.

The access terminal and the access network shall not use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to this protocol. The SecurityEnabled attribute shall be defined by the SessionConfigurationToken of the Session Configuration Protocol.

Table 31 Configurable values

Attribute ID	Attribute	Values	Meaning
0x00	SecurityEnabled	***0x00***	SecurityEnabled mode is off
		0x01	SecurityEnabled mode is on.
		0x02 – 0xff	Reserved

5.4.11 Protocol numeric constants

Constant	Meaning	Value
N_{SPType}	Type field for this protocol	Table 9
N_{SPG}	Subtype field for this protocol	0x0000
$T_{SPSecurityConfirmWait}$	Duration the access network waits before retransmitting the EnableSecurityAssignment.	1 s
$T_{SPSecurityResponseWait}$	Duration the initiator waits for a EnableSecurityResponse message	1 s

5.4.12 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

5.5 Default Authentication Protocol

5.5.1 Overview

The Default Authentication Protocol provides a method for authentication of packets by applying (on the transmit side) and checking (on the receive side) the HMAC-SHA256 message authentication function to *message bits* that are composed of the Authentication Protocol payload, CryptoSync, together with FLAuthKey or RLAuthKey, as appropriate. The HMAC-SHA256 function is defined in RFC 2104 [10] with SHA-256 [1] as the message digest algorithm.

5.5.2 Primitives

5.5.2.1 Commands

This protocol does not define any commands.

5.5.2.2 Return indications

- *Failed*

5.5.3 Public data

5.5.3.1 Static public data

This protocol does not define any static public data.

5.5.3.2 Dynamic public data

- Subtype for this protocol

5.5.4 Protocol data unit

The protocol data unit for this protocol is an Authentication Protocol packet.

5.5.5 Protocol initialization and swap

5.5.5.1 Protocol initialization

Upon initialization, the value of the attributes for this protocol instance in the access terminal and the access network shall be set to the default values specified for each attribute.

5.5.5.2 Protocol swap

This protocol defines an empty swap procedure.

5.5.6 Procedures

On the transmit side: When this protocol receives an Encryption Protocol Packet, it shall add the Authentication Protocol Header defined in 5.5.6.2 in front of the Encryption Protocol Packet and shall forward the newly generated Authentication Protocol Packet to the Security Protocol.

On the receive side: When this protocol receives a Security Sublayer packet from the Security Protocol, it shall check the message authentication code in the Authentication Protocol Header. If the message authentication code passes, this protocol removes the Authentication Protocol Header and shall forward the newly generated Authentication Protocol Packet to the Encryption Protocol.

5.5.6.1 Access terminal requirements

5.5.6.1.1 Transmit Procedures

Upon reception of an Encryption Protocol packet destined for transmission the access terminal shall compute the packet authentication code (PAC) as follows:

- The access terminal shall construct the AuthKey as follows:
  - If the Key Exchange Protocol does not define RLAuthKey as public data, this protocol shall discard the packet.
  - Otherwise, this protocol shall perform the following:
    - If the length of RLAuthKey is equal to the length of AuthKey, then AuthKey shall be RLAuthKey.
    - Otherwise, if the length of RLAuthKey is greater than the length of AuthKey, then AuthKey shall be the $N_{APAuthKeyLength}$ least significant bits of RLAuthKey.
    - Otherwise, if the length of RLAuthKey is less than the length of AuthKey this protocol shall discard the packet.
- The access terminal shall construct the cryptosync as described by the Security Protocol.
- The access terminal shall construct the *message bits* for computing the PAC as shown in Table 32.

Table 32 Message bits for AT PAC computation

Field	Length(bits)
Authentication Protocol Payload	Variable
Cryptosync	96

- The access terminal shall compute the *message digest* as HMAC-SHA256(AuthKey, *message bits*). The PAC field shall be set to the $N_{APMessageAuthCodeLength}$ least significant bits of the *message digest*.

5.5.6.1.2 Receive Procedures

Upon reception of an Authentication Protocol packet, the access terminal shall compute and verify the Lower MAC Sublayer packet authentication code (PAC) given in the authentication protocol header as follows:

- The access terminal shall construct the AuthKey as follows:
  - If the Key Exchange Protocol does not define FLAuthKey as public data, this protocol shall discard the packet.
  - Otherwise, the access terminal shall perform the following:
    - If the length of FLAuthKey is equal to the length of AuthKey, then AuthKey shall be FLAuthKey.
    - Otherwise, if the length of FLAuthKey is greater than the length of AuthKey, then AuthKey shall be the $N_{APAuthKeyLength}$ least significant bits of FLAuthKey.
    - Otherwise, if the length of FLAuthKey is less than the length of AuthKey, this protocol shall discard the packet.
  - The access terminal shall use the cryptosync provided by the Security Protocol.
- The access terminal shall construct the message bits for computing the PAC as shown in Table 33.
- The access terminal shall compute the *message digest* as HMAC-SHA256(AuthKey, *message bits*). The PAC field shall be set to the $N_{APMessageAuthCodeLength}$ least significant bits of the *message digest*.

If the PAC computed in the previous step matches the PAC field in the Protocol Header, then the Protocol shall deliver the Authentication Sublayer Payload to the Encryption Protocol. Otherwise, the Protocol shall issue a *Failed* indication and shall discard the Security Sublayer packet.

5.5.6.2 Access network requirements

5.5.6.2.1 Transmit Procedures

Upon reception of an Encryption Protocol packet destined for transmission the access network shall compute the packet authentication code (PAC) as follows:

- The access terminal shall construct AuthKey as follows:
  - If the Key Exchange Protocol does not define FLAuthKey as public data, the access terminal shall discard the packet.
  - Otherwise, the access network shall perform the following:
    - If the length of FLAuthKey is equal to the length of AuthKey, then AuthKey shall be FLAuthKey.
    - Otherwise, if the length of FLAuthKey is greater than the length of AuthKey, then AuthKey shall be the $N_{APAuthKeyLength}$ least significant bits of FLAuthKey.
    - Otherwise, if the length of FLAuthKey is less than the length of AuthKey, then the access network shall discard the packet.

- The access network shall construct the *message bits* for computing the PAC as shown in Table 33.

Table 33 Message bits for AN PAC computation

Field	Length(bits)
Authentication Protocol Payload	Variable
Cryptosync	96

- The access network shall compute the *message digest* as HMAC-SHA256(AuthKey, *message bits*). The PAC field shall be set to the $N_{APMessageAuthCodeLength}$ least significant bits of the *message digest.*

5.5.6.2.2 Receive Procedures

Upon reception of an Authentication Protocol packet the access network shall compute and verify the Lower MAC Sublayer Packet Authentication Code (PAC) given in the authentication protocol header as follows:

- The access network shall construct the AuthKey as follows:
  - If the Key Exchange Protocol does not define RLAuthKey as public data, the access network shall discard the packet.
  - Otherwise, the access network shall perform the following:
    - If the length of RLAuthKey is equal to the length of AuthKey, then AuthKey shall be RLAuthKey.
    - Otherwise, if the length of RLAuthKey is greater than the length of AuthKey, then AuthKey shall be the $N_{APAuthKeyLength}$ least significant bits of RLAuthKey.
    - Otherwise, if the length of RLAuthKey is less than the length of AuthKey, then the access network shall discard the packet.
  - The access network shall use the cryptosync provided by the Security Protocol.
- The access network shall construct the *message bits* for computing PAC as shown in Table 32.
- The access network shall compute the *message digest* as HMAC-SHA256(AuthKey, *message bits*). The PAC shall be set to the $N_{APMessageAuthCodeLength}$ least significant bits of the *message digest*.

If the PAC computed in the previous step matches the PAC field in the Protocol Header, then the Protocol shall deliver the Authentication Protocol Payload to the Encryption Protocol. Otherwise, the Protocol shall issue a *Failed* indication and shall discard the Security Sublayer packet.

5.5.7 Header and trailer

5.5.7.1 Header

The Default Authentication Protocol header is defined as follows:

Field	Length (bits)
PAC	0 or $N_{APMessageAuthCodeLength}$

PAC Packet Authentication Code. This field shall be computed as specified in 5.5.6.1. This field shall be included if UATIInfoIncluded bit associated with the packet is ‘1’ or the configuration attribute AuthenticationMode is equal to ‘1’.

5.5.7.2 Trailer

The Default Authentication Protocol does not add a trailer.

5.5.8 Message formats

No messages are defined for this protocol.

5.5.9 Interface to other protocols

5.5.9.1 Commands

This protocol does not issue any commands.

5.5.9.2 Indications

This protocol does not register to receive any indications.

5.5.10 Configuration attributes

This protocol defines the following configuration attributes.

The access terminal and the access network shall not use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to this protocol. The AuthenticationMode attribute shall be defined by the SessionConfigurationToken of the Session Configuration Protocol.

Table 34 Configurable values

Attribute ID	Attribute	Values	Meaning
0x00	AuthenticationMode	***0x00***	Only packets with UATIInfoIncluded=1 and IsSecure=1 in the Lower MAC header are authenticated.
		0x01	All packets with IsSecure=1 in the Lower MAC header are authenticated
		0x02 – 0xff	Reserved

5.5.11 Protocol numeric constants

Constant	Meaning	Value
N_{APType}	Type field for this protocol	Table 9 (0x06)
N_{APG}	Subtype field for this protocol	0x0000
$N_{APMessageAuthCodeLength}$	Number of bits in the message authentication code	0x0060
$N_{APAuthKeyLength}$	Length of the authentication key	0x00A0

5.5.12 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

5.6 Generic Encryption Protocol

5.6.1 Overview

The Generic Encryption Protocol uses the AES (or, Rijndael) procedures defined in [16] in order to encrypt the Convergence Sublayer packets and decrypt the Authentication Protocol packets.

5.6.2 Primitives

5.6.2.1 Commands

This protocol does not define any commands.

5.6.2.2 Return indications

This protocol does not return any indications.

5.6.3 Public data

5.6.3.1 Static public data

This protocol does not define any static public data.

5.6.3.2 Dynamic public data

- Subtype for this protocol

5.6.4 Protocol data unit

The protocol data unit for this protocol is an Encryption Protocol Packet.

5.6.5 Protocol initialization and swap

5.6.5.1 Protocol initialization

Upon initialization, the value of the attributes for this protocol instance in the access terminal and the access network shall be set to the default values specified for each attribute.

5.6.5.2 Protocol swap

This protocol defines an empty swap procedure.

5.6.6 Procedures

On the transmit side: When this protocol receives a Convergence Sublayer packet and the cryptosync from the Security Protocol, it shall follow the transmit procedures specified in 5.6.6.3.

On the receive side: When this protocol receives a Security Sublayer packet and the cryptosync from the Security Protocol, it shall follow the receive procedures specified in 5.6.6.4.

5.6.6.1 Constructing the encryption key for the FL

The Generic Encryption Protocol shall construct the encryption keys for the FL as follows:

- the protocol shall construct the encryption key for the Forward Link Channels, FLEncryptionKey, as follows:
  - If the Key Exchange Protocol does not define FLEncKey as public data, this protocol shall discard all packets.
  - Otherwise, this protocol shall perform the following:
    - If the length of FLEncKey is equal to $N_{EPEncKeyLength}$, then FLEncryptionKey shall be set to FLEncKey.
    - Otherwise, if the length of FLEncKey is greater than $N_{EPEncKeyLength}$, then FLEncryptionKey shall be the $N_{EPEncKeyLength}$ most significant bits of FLEncKey.
    - Otherwise, if the length of FLEncKey is less than $N_{EPEncKeyLength}$, this protocol shall discard all packets.

5.6.6.2 Constructing the encryption key for the RL

The Generic Encryption Protocol shall construct the encryption keys for the RL as follows:

- The protocol shall construct the encryption key for the Reverse Link Channels, RLEncryptionKey, as follows:
  - If the Key Exchange Protocol does not define RLEncKey as public data, this protocol shall discard all packets.
  - Otherwise, the protocol shall perform the following:
    - If the length of RLEncKey is equal to $N_{EPEncKeyLength}$, then RLEncryptionKey shall be set to RLEncKey.
    - Otherwise, if the length of RLEncKey is greater than $N_{EPEncKeyLength}$, then RLEncryptionKey shall be the $N_{EPEncKeyLength}$ most significant bits of RLEncKey.
    - Otherwise, if the length of RLEncKey is less than $N_{EPEncKeyLength}$, this protocol shall discard all packets.

5.6.6.3 Transmit procedures

The protocol shall construct the Encryption Protocol packet from the Convergence Sublayer packet that is destined for FL or RL by performing the following for each of the channels:

- The protocol shall call the ESP_AES procedure specified in [16] with its inputs set as follows:
  - Set *key* to the EncryptionKey for the channel under consideration (e.g., FLEncryptionKey).
  - Set *fresh* to the value of the cryptosync provided by the Security Protocol.
  - Set *freshsize* to 96.
  - Set *buf* to the address of the beginning of the memory space that contains the Convergence Sublayer packet.
  - Set *bit_offset* to zero.
  - Set *bit_count* to the length of the Convergence Sublayer Packet in bits.
- After the ESP_AES procedure is returned, the protocol shall set the Encryption Protocol packet to the output of the ESP_AES procedure that starts at the memory space specified by *buf* and is of the same size as the Convergence Sublayer packet.

5.6.6.4 Receive procedures

If the Encryption Protocol packet is received on the FL or RL, then the receiver shall construct the Convergence Sublayer packet from the Encryption Protocol packet by performing the following for each of the channels:

- The protocol shall call the ESP_AES procedure specified in [16] with its inputs set as follows:
  - Set *key* to the EncryptionKey for the channel under consideration (e.g., FLEncryptionKey).
  - Set *fresh* to the value of the cryptosync provided by the Security Protocol.

- Set *freshsize* to 96.
- Set *buf* to the address of the beginning of the memory space that contains the Encryption Protocol packet.
- Set *bit_offset* to zero.
- Set *bit_count* to the length of the Encryption Protocol Packet in bits.

- After the ESP_AES procedure is returned, the protocol shall set the Convergence Sublayer packet to the output of the ESP_AES procedure which starts at the memory space specified by *buf* and is of the same size as the Encryption Protocol packet.

5.6.7 Generic Encryption Protocol header and trailer

The Generic Encryption Protocol does not add a header or a trailer.

5.6.8 Message formats

No messages are defined for this protocol.

5.6.9 Interface to other protocols

5.6.9.1 Commands

This protocol does not issue any commands.

5.6.9.2 Indications

This protocol does not register to receive any indications.

5.6.10 Configuration attributes

No configuration attributes are defined for this protocol.

5.6.11 Protocol numeric constants

Constant	Meaning	Value
N_{EPType}	Type field for this protocol	**Table 9**
N_{EPG}	Subtype field for this protocol	0x0001
$N_{EPEncKeyLength}$	Length of the encryption key	0x00A0

5.6.12 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

6 Lower MAC Control Sublayer

6.1 Introduction

6.1.1 General overview

The Lower MAC Control sublayer controls the state of the air-link by managing the states of individual Lower MAC sublayer protocols, and by providing individual MAC Sublayer protocols with operating parameters. The protocols in this sublayer are control protocols, and do not carry data on the behalf of other protocols. The protocols in this sublayer use the Signaling Transport to transmit and receive messages (with the exception of the Overhead Messages Protocol that sends some information blocks using the Lower MAC Sublayer).

The access terminal and the access network maintain a connection whose state dictates the form in which communications between these entities can take place. The connection can be either of the following three states:

- *Closed Connection and no assigned MAC ID*: The access terminal is not assigned any dedicated air-link resources. Communications between the access terminal and the access network are conducted over the R-ACH, F-pBCH0, F-pBCH1, F-SSCH, and F-DCH physical layer channels.
- *Closed Connection and assigned MAC ID:* This is an intermediate state between a closed connection and an open connection. The access terminal is assigned a MAC ID and associated resources on the forward and reverse links. These resources are used by the access terminal to request a connection, and by the access network to indicate grant or rejection of a connection, possibly based on the identity of the access terminal. This state corresponds to the BindUATI state of the Idle State Protocol.
- *Open Connection and assigned MAC ID*: Opening of the connection indicates that the access network has granted dedicated resources on the forward and reverse links based on the identity of the access terminal. This state corresponds to the Open State of the Connected State Protocol.

The Lower MAC Control Sublayer provides the following connection-related functions:

- Manages initial acquisition of the network.
- Manages opening and closing of connections.
- Maintains an approximate access terminal location in either connection states.
- Manages the radio link between the access terminal and the access network when a connection is open.
- Performs supervision at the access terminal both when the connection is open and when it is closed.

The Lower MAC Control Sublayer performs these functions through the following protocols:

- *Air Link Management Protocol*: This protocol maintains the overall connection state in the access terminal and the access network. The protocol can be in one of three states, corresponding to whether the access terminal has yet to acquire the network (Initialization State), has acquired the network but the connection is closed (Idle State), or has an open connection with the access network (Connected State). This protocol activates one of the following three protocols as a function of its current state.
- *Initialization State Protocol*: This protocol performs the actions associated with acquiring an access network.
- *Idle State Protocol*: This protocol performs the actions associated with an access terminal that has acquired the network, but does not have an open connection. Mainly, these are keeping track of the access terminal's approximate location in support of efficient Paging (using the Active Set Management protocol), the procedures leading to the opening of a connection, and support of access terminal power conservation.
- *Connected State Protocol*: This protocol performs the actions associated with an access terminal that has an open connection. These actions primarily include managing the radio link between the access terminal and the access network, including the management of tune away and selected interlace operation, and the procedures leading to the close of the connection.

In addition to the above protocols, which deal with the state of the connection, the Lower MAC Control sublayer also contains the following protocols:

- *Active Set Management Protocol*: This protocol performs the actions associated with keeping track of an access terminal's location and maintaining the radio link between the access terminal and the access network.
- *Overhead Messages Protocol*: This protocol broadcasts and receives essential parameters over the Control Channel MAC and the Forward Traffic Channel MAC. This protocol also performs supervision on the parameters, and generates SupervisionFailed indications when overhead parameters are not current.

6.1.2 Data encapsulation

This sublayer does not encapsulate data on the behalf of other layers or protocols.

6.2 Default Air Link Management Protocol

6.2.1 Overview

The Default Air Link Management Protocol provides the following functions:

- General state machine and state-transition rules to be followed by an access terminal and an access network for the Lower MAC Control Sublayer.
- Activation and deactivation of Lower MAC Control Sublayer protocols applicable to each protocol state.

- Responding to supervision failures indications from other Protocols, and associated state transitions of Lower MAC Sublayer Protocols and Lower MAC Control Sublayer Protocols.
- Mechanism through which the access network can redirect the access terminal to another network.

The actual behavior and message exchange in each state is mainly governed by protocols that are activated by the Default Air Link Management Protocol. These protocols return indications which trigger the state transitions of this protocol. These protocols also share data with each other in a controlled fashion, by making that data public.

This protocol can be in one of three states:

- *Initialization State*: In this state the access terminal acquires an access network. The protocol activates the Initialization State Protocol to execute the procedures relevant to this state. The access network does not support this state.
- *Idle State*: In this state the connection is closed. The protocol activates the Idle State Protocol to execute the procedures relevant to this state.
- *Connected State*: In this state the connection is open. The protocol activates the Connected State Protocol to execute the procedures relevant to this state.

Figure 46 provides an overview of the access terminal states and state transitions. All transitions are caused by indications returned from protocols activated by the Default Air Link Management Protocol.

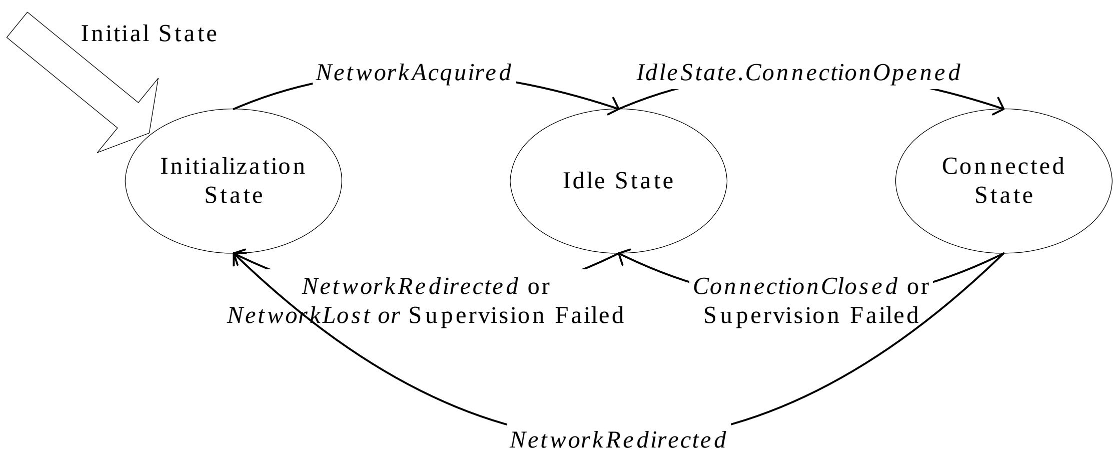

Figure 46 Air Link Management Protocol state diagram (access terminal)

Figure 47 provides an overview of the access network states and state transitions.

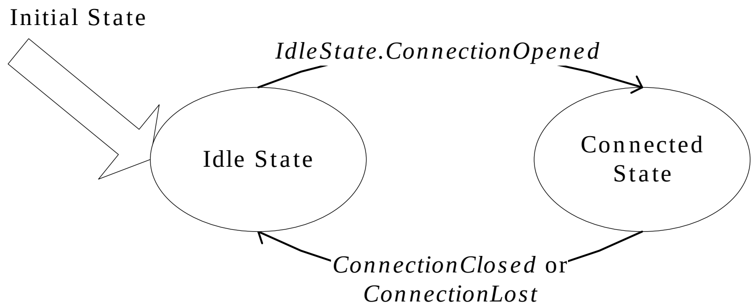

Figure 47 Air Link Management Protocol state diagram (access network)

Table 35 provides a summary of the Lower MAC Control Sublayer and Lower MAC Sublayer protocols that are active in each state.

Table 35 Active protocols per Air Link Management Protocol state

Initialization State[19]	Idle State	Connected State
Overhead Messages Protocol	Overhead Messages Protocol	Overhead Messages Protocol
Initialization State Protocol	Idle State Protocol	Connected State Protocol
Control Channel MAC Protocol[20]	Active Set Management Protocol	Active Set Management Protocol
	Control Channel MAC Protocol	Control Channel MAC Protocol
	Shared Signaling MAC Protocol	Shared Signaling MAC Protocol
	Forward Traffic Channel MAC Protocol	Forward Traffic Channel MAC Protocol
	Access Channel MAC Protocol[21]	Access Channel MAC Protocol[22]
	Reverse Traffic Channel MAC Protocol[23]	Reverse Traffic Channel MAC Protocol
	Reverse Control Channel MAC Protocol[24]	Reverse Control Channel MAC Protocol

[19] Applicable only to access terminal

[20] Activated by the Initialization State Protocol

[21] Used by access terminal only during connection setup

[22] Used by access terminal only during handoff to an asynchronous sector

[23] Used by access terminal only during connection setup

[24] Used by access terminal only during connection setup

6.2.2 Primitives

6.2.2.1 Commands

This protocol defines the following commands:

- *OpenConnection*
- *CloseConnection*

6.2.2.2 Return indications

This protocol does not return any indications.

6.2.3 Public data

6.2.3.1 Static public data

This protocol does not define any static public data.

6.2.3.2 Dynamic public data

- Subtype for this protocol

6.2.4 Protocol initialization and swap procedures

6.2.4.1 Protocol initialization

Upon initialization at the access terminal,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Initialization State.

Upon initialization at the access network,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Idle State.

6.2.4.2 Protocol swap

Upon swap at the access terminal,

- The protocol shall enter the Initialization State.

Upon swap at the access network,

- The protocol shall enter the Idle State.

Upon creation of the InUse instance of this protocol, the access network shall have a single InUse instance of this protocol operating in the Initialization State at the access network, serving all access terminals.

6.2.5 Procedures

6.2.5.1 Command processing

6.2.5.1.1 OpenConnection

If the protocol receives the *OpenConnection* command in the Initialization State, the access terminal shall queue the command and execute it when the access terminal enters the Idle State.

The access network shall ignore the command in the Initialization State.

If the protocol receives this command in the Idle State:

- Access terminal shall issue an *IdleState.OpenConnection* command.
- Access network shall issue an *IdleState.OpenConnection* command.

If the protocol receives this command in the Connected State the command shall be ignored.

6.2.5.1.2 CloseConnection

If the protocol receives the *CloseConnection* command in the Connected State:

- Access terminal shall issue a *ConnectedState.CloseConnection* command.
- Access network shall issue a *ConnectedState.CloseConnection* command.

If the protocol receives this command in any other state it shall be ignored.

6.2.5.2 Initialization state

This state is not applicable to the access network. In the Initialization State the access terminal has no information about the serving access network. In this state the access terminal selects a serving access network and obtains time synchronization from the access network.

6.2.5.2.1 Access terminal requirements

The access terminal shall enter the Initialization State when the Default Air Link Management Protocol is instantiated. This may happen on events such as network redirection and initial power-on. A comprehensive list of events causing the Default Air Link Management Protocol to enter the Initialization State is beyond the scope of this specification.

The access terminal shall issue an *InitializationState.Activate* command upon entering this state. If the access terminal entered this state because the protocol received a Redirect message and a ChannelBand Record was received with the message, the access terminal shall provide the ChannelBand Record with the command.

If the protocol receives an *InitializationState.NetworkAcquired* indication the access terminal shall issue an *InitializationState.Deactivate* command[25] and transition to the Idle State.

[25] Some of the *Deactivate* commands issued by this protocol are superfluous (because the commanded protocol already put itself in the Inactive State) but are specified here for completeness.

6.2.5.2.2 Access network requirements

This state is not defined for the access network.

6.2.5.3 Idle state

In this state the access terminal has acquired the access network but does not have an open connection with the access network.

6.2.5.3.1 Access terminal requirements

The access terminal shall issue the following commands upon entering this state:

- *IdleState.Activate*
- *ActiveSetManagement.Activate*

If the access terminal had a queued *OpenConnection* command, it shall issue an *IdleState.OpenConnection* command.

If the protocol receives an *IdleState.ConnectionOpened* indication, the access terminal shall:

- Issue a *IdleState.Deactivate* command
- Transition to the Connected State

If the protocol receives an *IdleState.ConnectionFailed*, a *ForwardTrafficChannelMAC.SupervisionFailed*, or a *ReverseTrafficChannelMAC.SupervisionFailed* indication, the access terminal shall:

- Issue a *IdleState.Close* command
- Issue a *ActiveSetManagement.Close* command
- Issue a *ReverseTrafficChannelMAC.Deactivate* command
- Issue a *ReverseControlChannelMAC.Deactivate* command

If the protocol receives a Redirect message, a *ActiveSetManagement.NetworkLost*, an *OverheadMessages.SupervisionFailed*, or a *ControlChannelMAC.SupervisionFailed* indication, the access terminal shall:

- Issue a *ActiveSetManagement.Deactivate* command
- Issue a *ReverseTrafficChannelMAC.Deactivate* command
- Issue a *ReverseControlChannelMAC.Deactivate* command
- Issue a *ForwardTrafficChannelMAC.Deactivate* command
- Issue an *OverheadMessages.Deactivate* command
- Issue a *ControlChannelMAC.Deactivate* command
- Issue a *IdleState.Deactivate* command
- Issue a *AccessChannelMAC.Deactivate* command
- Transition to the Initialization State

6.2.5.3.2 Access network requirements

The access network shall issue the following commands upon entering this state:

- *IdleState.Activate*
- *ActiveSetManagement.Activate*

If the protocol receives an *IdleState.ConnectionFailed* indication, or a *ReverseTrafficChannelMAC.SupervisionFailed* indication, the access network shall:

- Issue an *IdleState.Close* command
- Issue a *ReverseTrafficChannel.Deactivate* command
- Issue a *ReverseControlChannelMAC.Deactivate* command
- Issue a *ActiveSetManagement.Close* command
- Issue a *IdleState.Deactivate* command

If the protocol receives an *IdleState.ConnectionOpened* indication, the access network shall:

- Issue a *IdleState.Deactivate* command
- Transition to the Connected State

The access network may send the access terminal a Redirect message to redirect it from the current serving network and optionally, provide it with information directing it to another network. If the access network sends a Redirect message it shall:

- Issue a *ReverseTrafficChannel.Deactivate* command
- Issue a *ReverseControlChannel.Deactivate* command
- Issue a *ActiveSetManagement.Deactivate* command
- Issue a *IdleState.Deactivate* command.
- Transition to the Idle State

6.2.5.4 Connected state

In the Connected State, the access terminal and the access network have an open connection.

6.2.5.4.1 Access terminal requirements

6.2.5.4.1.1 General requirements

The access terminal shall issue the following commands upon entering this state:

- *ConnectedState.Activate*
- *ActiveSetManagement.Open*

If the protocol receives a *ConnectedState.ConnectionClosed*, an *OverheadMessages.SupervisionFailed*, a *ControlChannelMAC.SupervisionFailed*, a *ActiveSetManagement.AssignmentRejected*, or a *ForwardTrafficChannelMAC.SupervisionFailed* indication, the access terminal shall:

- Issue a *ActiveSetManagement.Close* command
- Perform the cleanup procedure defined in 6.2.5.4.1.2
- Transition to the Idle State

If the protocol receives a Redirect message, the access terminal shall:

- Issue a *ActiveSetManagement.Deactivate* command
- Issue an *OverheadMessages.Deactivate* command
- *ForwardTrafficChannelMAC.Deactivate*
- *SharedSignalingMAC.Deactivate*
- Perform the cleanup procedure defined in 6.2.5.4.1.2
- Transition to the Initialization State

6.2.5.4.1.2 Connected state cleanup procedures

The access terminal shall issue the following commands when it exits this state:

- *ReverseTrafficChannel.Deactivate*
- *ReverseControlChannelMAC.Deactivate*
- *ActiveSetManagement.Close*
- *ConnectedState.Deactivate*

6.2.5.4.2 Access network requirements

6.2.5.4.2.1 General requirements

The access network shall issue the following commands upon entering this state:

- *ConnectedState.Activate*
- *ActiveSetManagement.Open*

If the protocol receives a *ConnectedState.ConnectionClosed*, or *ActiveSetManagement.ConnectionLost* indication, the access network shall:

- Issue a *ActiveSetManagement.Close* command
- Perform the cleanup procedures defined in 6.2.5.4.2.2
- Transition to the Idle State

The access network may send the access terminal a Redirect message to redirect it from the current serving network and optionally, provide it with information directing it to another network. If the access network sends a Redirect message it shall:

- Issue a *ActiveSetManagement.Deactivate* command
- Perform the cleanup procedures defined in 6.2.5.4.2.2
- Transition to the Idle State

6.2.5.4.2.2 Connected state cleanup procedures

The access network shall issue the following commands when it exits this state:

- *ReverseTrafficChannel.Deactivate*
- *ReverseControlChannel.Deactivate*
- *ActiveSetManagement.Close*
- *ConnectedState.Deactivate*

6.2.6 Message formats

6.2.6.1 Redirect

The access network sends the Redirect message to redirect the access terminal(s) away from the current network; and, optionally, the access network provides it with information directing it to one of a set of different networks.

Field	Length (bits)
MessageID	8
StayAwayDuration	16
NumChannelBands	8
RedirectReason	8
NumChannelBands instances of the following two fields	
ChannelBandRecord	ChannelBandRecord Type Dependent
TechnologyType	8

MessageID — The access network shall set this field to 0x00.

StayAwayDuration — The access network shall set this field to the duration, in units of seconds, for which the access terminal shall not make an access attempt at the sector sending this message.

RedirectReason — The sender shall set this field to reflect the redirect reason, as shown in Table 36.

Table 36 Encoding of the RedirectReason field

Field value	Description
0x00	Reserved
0x01	Network Busy
0x02	Authentication or billing failure
0x03	Desired QoS unavailable
0x04	No route to host
0x05	Network Maintenance
0xff	General Failure
All other values are reserved	

NumChannelBands — The access network shall set this field to the number of ChannelBand records it is including in this message.

ChannelBandRecord — This field shall be set to the channel band that the access terminal should reacquire. The channel band shall be specified using the standard ChannelBand Record definition, see 10.1.

TechnologyType — This field shall be set to the type of technology, and shall be interpreted as defined in Table 107. This field shall not take the value 0x00.

Channels	FTC	**SLP**	Best Effort
Addressing	Broadcast Unicast	**Security**	Required

6.2.7 Interface to other protocols

6.2.7.1 Commands

This protocol issues the following commands:

- *InitializationState.Activate*
- *InitializationState.Deactivate*
- *IdleState.Activate*
- *IdleState.Deactivate*
- *IdleState.Close*
- *IdleState.OpenConnection*
- *ConnectedState.Activate*
- *ConnectedState.Deactivate*
- *ConnectedState.CloseConnection*
- *ActiveSetManagement.Activate*
- *ActiveSetManagement.Deactivate*

- *ActiveSetManagement.Close*
- *ActiveSetManagement.Open*
- *OverheadMessages.Deactivate*
- *ControlChannelMAC.Deactivate*
- *AccessChannelMAC.Deactivate*
- *ReverseTrafficChannelMAC.Deactivate*
- *ReverseControlChannelMAC.Deactivate*
- *SharedSignalingMAC.Deactivate*

6.2.7.2 Indications

This protocol registers to receive the following indications:

- *InitializationState.NetworkAcquired*
- *IdleState.ConnectionOpened*
- *IdleState.ConnectionFailed*
- *ConnectedState.ConnectionClosed*
- *ActiveSetManagement.ConnectionLost*
- *ActiveSetManagement.NetworkLost*
- *ActiveSetAssignment.AssignmentRejected*
- *OverheadMessages.SupervisionFailed*
- *ControlChannelMAC.SupervisionFailed*
- *ReverseTrafficChannelMAC.SupervisionFailed*
- *ForwardTrafficChannelMAC.SupervisionFailed*

6.2.8 Configuration attributes

No configuration attributes are defined for this protocol.

6.2.9 Protocol numeric constants

Constant	Meaning	Value
$N_{ALMPType}$	Type field for this protocol	Table 9
$N_{ALMPDefault}$	Subtype field for this protocol	0x0000

6.2.10 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

6.3 Default Idle State Protocol

6.3.1 Overview

The Default Idle State Protocol provides the procedures and messages used by the access terminal and the access network when the access terminal has acquired a network and a connection is not open.

This protocol operates in one of the following four states:

- *Inactive State*: In this state the protocol waits for an *Activate* command.
- *Sleep State*: In this state the access terminal may shut down part of its subsystems to conserve power. The access terminal does not monitor any Forward Channel, and the access network is not allowed to transmit unicast packets to it.
- *Monitor State*: In this state the access terminal listens for Pages and QuickPages and if necessary, updates the parameters received from the Overhead Messages Protocol. The access network may transmit unicast packets to the access terminal in this state.
- *Access State*: In this state the access terminal sends access preambles to the access network and receives an access grant from the network. This state is not defined for the access network.
- *BindUATI State*: In this state the access terminal sends an ATIdentifier (UATI or SessionSeed) to the access network and waits for an acknowledgement in the form of a *ForwardTrafficChanneMAC.UATIReceived* indication or a ConnectionOpenResponse message. In this state, the access network sends a packet with the MAC header field UATIInfoIncluded set to '1' to the access terminal and may send a ConnectionGrant message to the access terminal.

Protocol states and events causing the transition between the states are shown in Figure 48 and Figure 49.

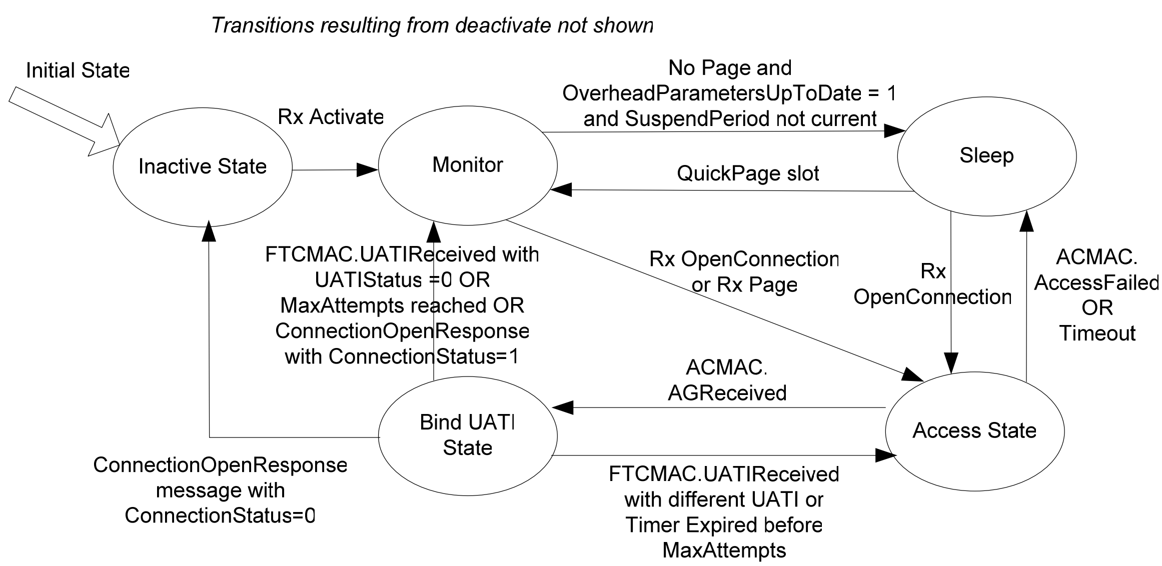

Figure 48 Default Idle State Protocol (access terminal)

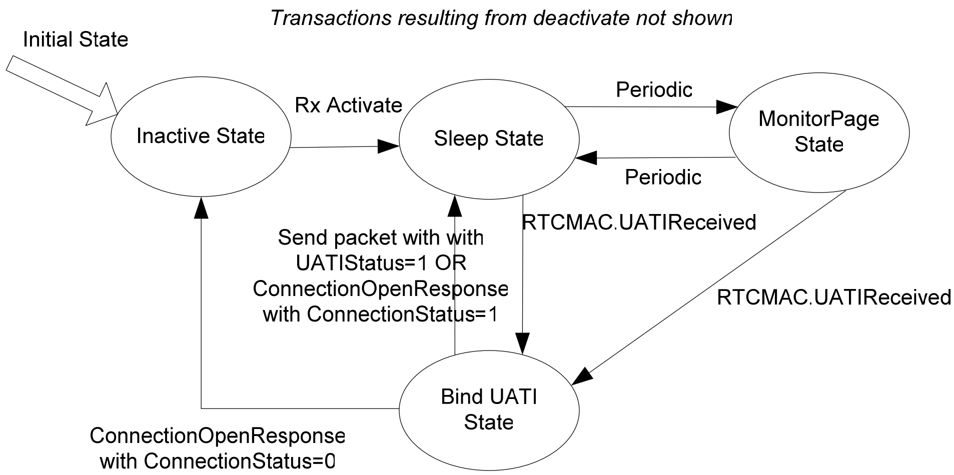

Figure 49 Default Idle State Protocol state (access network)

This protocol supports periodic network monitoring by the access terminal, allowing for significant power savings. The following access terminal operation modes are supported:

- Continuous operation, in which the access terminal continuously monitors the Control Channel.
- Suspended mode operation, in which the access terminal monitors the Control Channel continuously for a period of time and then proceeds to operate in the slotted mode. Suspended mode follows operation in the Air Link Management Protocol Connected State and allows for quick network-initiated reconnection.
- Slotted mode operation, in which the access terminal monitors only selected superframes and sleeps in between. The slotted mode supports staggered operation, where the time interval between the superframes monitored by the terminal increases with time. For the first WakeCount1 sleep instances, the sleep period may be Period1 superframes, and for the next WakeCount2-WakeCount1 sleep instances, the sleep period may be Period2 superframes, and for subsequent sleep instances, the sleep period may be Period3 superframes.
- This protocol supports connection set up: this procedure is always performed at the initiative of the access terminal.[26] It consists of the following steps:
  - Access terminal sending an access preamble on R-ACH.
  - Access network sends an access grant on F-SSCH.
  - Access terminal sends a packet on the R-DCH. This packet contains the UATI or SessionSeed in the MAC header and ConnectionOpenRequest message in the payload.
  - Access network sends a response packet on the F-DCH. The response packet contains the UATI or SessionSeed in the MAC header and ConnectionOpenResponse message in the payload.

[26] The access network may transmit a Page message to the access terminal directing it to initiate the procedure.

6.3.2 Primitives

6.3.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*
- *OpenConnection*
- *Close*

6.3.2.2 Return indications

This protocol returns the following indications:

- *ConnectionOpened*
- *ConnectionFailed*
- *RegistrationRadiusUpdated*

6.3.3 Public data

6.3.3.1 Static public data

- PageTimes array
- ConnectCount
- RQuickPage

6.3.3.2 Dynamic public data

- Subtype for this protocol

6.3.4 Protocol initialization and swap procedures

6.3.4.1 Protocol initialization

Upon initialization at the access terminal and the access network:

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Inactive State.
- The protocol shall set ConnectCount to zero.

6.3.4.2 Protocol swap

Upon swap at the access terminal and access network:

- The protocol shall enter the Inactive State.

6.3.5 Procedures

6.3.5.1 Command processing

6.3.5.1.1 Activate

When the protocol at the access terminal or the access network receives an *Activate* command in the Inactive State:

- If the multi-carrier mode is set to MultiCarrierOn, the access terminal shall select its operating carrier as follows. If PreferredPagingCarrierEnabled is set to '1', the access terminal and the access network shall set *C*=PreferredPagingCarrier. Otherwise, the access terminal and access network shall calculate *C* as the result of applying the hash function (see 10.4) using the following parameters:
  - Key = SessionSeed
  - Decorrelate = 6 × SessionSeed[11:0]
  - N = NumCarriers

  where SessionSeed is given as public data of the Session Management Protocol. The access terminal shall set its carrier to CarrierID = mod(C,NumCarriers), where NumCarriers is the number of carriers in the public data of the Overhead Messages Protocol. The access network should use this CarrierID to communicate with the access terminal.

When the protocol at the access terminal receives an *Activate* command in the Inactive State:

- If the access terminal entered the Idle State upon sending a ConnectionClose message with CloseReason set to "Deregistration Request", the access terminal shall transition to the Sleep State
- Otherwise, the access terminal shall transition to the Monitor State.

When the protocol at the access network receives an *Activate* command in the Inactive State:

- The access network shall transition to the Sleep State.[27]

If the protocol receives this command in any other state the command shall be ignored.

6.3.5.1.2 Deactivate

When the protocol receives a *Deactivate* command in the Inactive State it shall be ignored.

When the protocol receives this command in any other state:

- The access terminal shall transition to the Inactive State.
- The access network shall transition to the Inactive State.

[27] Since the transitions happen asynchronously, this requirement guarantees that the access network will not transmit unicast packets to the access terminal over the Control Channel when the access terminal is not monitoring the channel.

6.3.5.1.3 OpenConnection

When the protocol receives an *OpenConnection* command in the Inactive State, the Access State or the BindUATI State, the command shall be ignored.

When the protocol receives this command in the Sleep State:

- The access terminal shall transition to the Access State.
- The access network shall queue the command and execute it when it is in the Monitor State.

When the protocol receives this command in the Monitor State:

- The access terminal shall transition to the Access State.
- The access network shall send a QuickPage and a Page.

6.3.5.1.4 Close

When the protocol receives a *Close* command in the Inactive State it shall be ignored.

When the protocol receives a *Close* command in any other state:

- The access terminal shall transition to the Monitor State.
- The access network shall transition to the Sleep State.

6.3.5.2 General overview of paging

Paging is implemented using the F-pBCH1 and F-DCH physical layer channels. The F-pBCH1 channel occurs at the beginning of a superframe (in superframes with even superframe numbers), and contains a QuickPaging block. This QuickPaging block may have one of two possible formats.

1. QuickPage block with full ATI results in a one step page.
2. QuickPage block with LSBs of the ATI results in a two step page.

In case the QuickPaging block has the full ATI of the access terminal, a *ControlChannelMAC.PageReceived* indication is generated, completing a one step page process.

In case the QuickPaging block has the LSBs of the ATI of the access terminal, a *ControlChannelMAC.QuickPageReceived* indication is generated, and a QuickPage is considered to be received.

Upon receipt of a QuickPage, the access terminal monitors the F-DCH for the full ATI, and if the ATI is detected, a *ForwardTrafficChannelMAC.PageReceived* indication is generated, completing a two-step page process.

6.3.5.3 General overview of sleep cycle

This section is for informative purposes, and describes which superframes may be used for sending pages by the access network, and receiving pages by the access terminal.

The Default Idle State Protocol allows the implementation of staggered mode sleep, where the terminal sleep period increases with time, from a sleep period of Period1 superframes for the first WakeCount1 sleep periods to Period2 superframes for the next WakeCount2 sleep periods, and finally to Period3 superframes for the remaining sleep periods. Staggered sleep presents the design problem that the access terminal and access network may be unable to synchronize timing for staggering, and therefore may increase the sleep period at different times. This may result in missed pages, where the access network is in monitor state (because it is using a smaller sleep period) while the access terminal is in sleep state (because it is using a larger sleep period). To prevent missed pages, this protocol makes the following design choices:

- The sleep times Period2 and Period3 shall be some multiple of a power of two of Period1. For example, if Period1 is 6 superframes, Period2 may be 24 superframes and Period3 may be 96 superframes. Such setting of the Periods guarantees that the access network is in the monitor state only when the access terminal is also in the monitor state, as shown below.
- The access network initializes with a staggered sleep cycle at a time that is conservative.

Staggered sleep operation is discussed in more detail below. In case the access terminal has advertised a suspend period in a ConnectionClose message, and the access network has received this message, the access network can synchronize slotted mode operation with the access terminal, i.e., the access network and access terminal can enter the monitor state in the same superframe. This synchronous operation is shown in Figure 50.

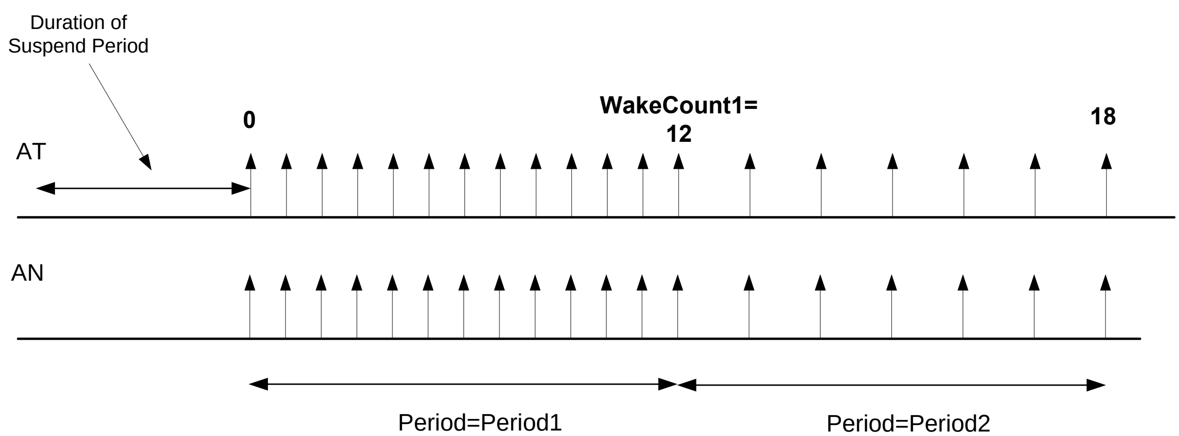

Figure 50 Slotted mode operation when access terminal and access network are synchronized

In case the access network has not received a suspend period from the access terminal, the access network may begin slotted operation at a time that is different from the time the access terminal enters slotted mode. The access network begins to count pages starting at time EarliestFirstPage, where the access network has determined that EarliestFirstPageTime is the earliest time the access terminal could have entered slotted mode.

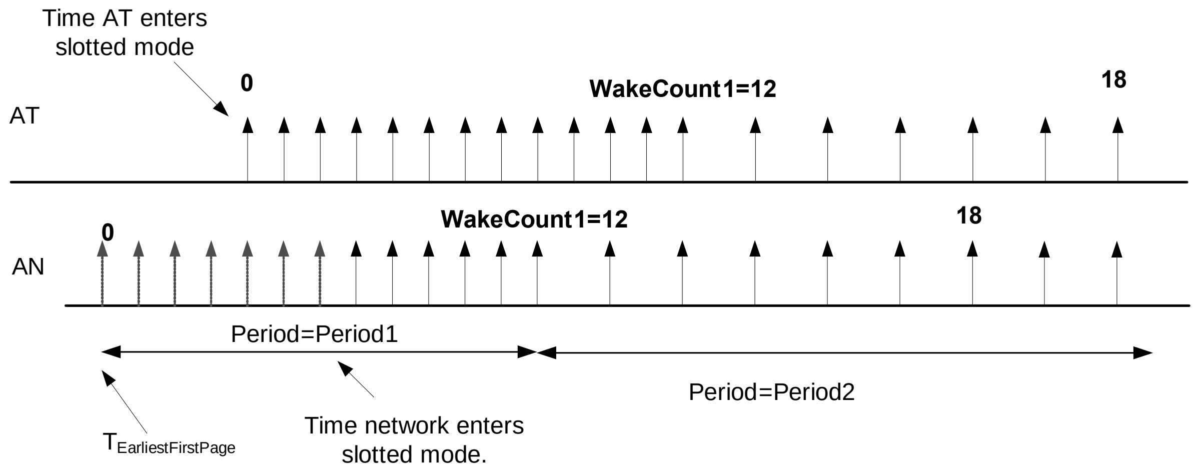

Figure 51 Slotted mode operation when access terminal and access network are not synchronized

6.3.5.4 General procedures

6.3.5.4.1 Paging cycle offset

R (the paging cycle offset) shall be obtained as follows. Since the paging cycle occurs only once every two superframes, the following procedure is designed to return even values.

- If PreferredPageOffsetEnabled is equal to '0', then *R* is the result of applying the hash function (see 10.4) using the following parameters, and multiplying the result by 2:
  - Key = SessionSeed
  - Decorrelate = 6 × SessionSeed[11:0]
  - N = Max(Period1, Period2, Period3, 1)

  where SessionSeed is given as public data of the Session Management Protocol.
- If PreferredPageOffsetEnabled is equal to '1', then *R* is set to twice the PreferredPageOffset.

6.3.5.4.2 RQuickPage calculation

The parameter RQuickPage is used by the Lower MAC Sublayer to determine if a quick page is received, and RQuickPage shall be decided as follows.

- If PreferredQuickPageEnabled is equal to '0', then *RQuickPage* is the result of applying the hash function (see 10.4) using the following parameters:
  - Key = SessionSeed
  - Decorrelate = 6 × SessionSeed[23:12]
  - N = 2^N_{QP_BLK}

  where SessionSeed is given as public data of the Session Management Protocol and N_{QP_BLK} is a numeric constant of the Control Channel MAC Protocol.

If PreferredQuickPageEnabled is equal to ‘1’, then RQuickPage is set to PreferredRQuickPage.

6.3.5.4.3 Paging period calculation

The access network and the access terminal shall compute Period*i* according to Table 37.

Table 37 Computation of Period*i* from SlotCycle*i*

SlotCycle*i*	Period*i*
0x00 to 0x1f	SlotCycleBase * $2^{SlotCycle\textit{i}}$ superframes
0x20 to 0xff	Reserved

6.3.5.4.4 Procedure for page time calculation

Given a value for StartTime (an argument to this procedure), this procedure shall return an array PageTimes[j].

The following formula shall be called the QuickPage formula, and shall be used to determine the PageTimes array:

$$[T+R] \bmod \text{Period} = 0,$$

where R is defined in 6.3.5.4, and values for Period are given in the steps below.

The PageTimes array shall be calculated by the following procedure, using a temporary variable x. This procedure follows the general description in 6.3.5.2.

1. Set x as the minimum value of T that is greater than StartTime and satisfies the QuickPage formula with Period=Period1.
2. For 0≤j<WakeCount1, set PageTimes[j] = x + j*Period1.
3. Set x as the minimum value of T that is greater than PageTimes[WakeCount1-1] and satisfies the QuickPage formula with Period=Period2.
4. For WakeCount1≤j<WakeCount2, set PageTimes[j] = x + (j-WakeCount1)*Period2.
5. Set x as the minimum value of T that is greater than PageTimes[WakeCount2-1] and satisfies the QuickPage formula with Period=Period2.
6. For WakeCount2≤j, set PageTimes[j] = x + (j-WakeCount2)*Period3.

The access network transmits pages for the access terminal in superframe index T, where T takes values PageTimes[j], j=0,1,2, … . The access network may also transmit the QuickPages for the access terminal in superframes that occur FastRepageInterval superframes after values in the PageTimes array.

6.3.5.5 Inactive state

When the protocol is in the Inactive State it waits for an *Activate* command and at the access terminal, sets internal variable NumAccessAttempts to zero.

6.3.5.6 Sleep state

When the access terminal is in the Sleep State it may stop monitoring the access network by issuing the following commands:

- *ControlChannelMAC.Deactivate*
- *SharedSignalingMAC.Deactivate*
- *ForwardTrafficChannelMAC.Deactivate*
- *OverheadMessages.Deactivate*

The access terminal may shut down processing resources to reduce power consumption.

6.3.5.6.1 Access terminal requirements

If the access terminal has a queued *OpenConnection* command, it shall transition to the Access State.

If the access terminal entered the Idle State as upon sending a ConnectionClose message with CloseReason set to “Deregistration Request”, this specification does not specify rules for entering the Monitor State. Otherwise, the access terminal shall transition to the monitor state for any one of the following reasons:

- *Update of overhead messages:* To determine the time of transition to the monitor state, the access terminal may rely on the ExtendedChannelInfoExpiryTime, that is public data of the Overhead Messages Protocol. The exact algorithm used by the access terminal to update overhead messages is beyond the scope of this specification.
- *To receive pages*: The access terminal shall transition to the monitor state in superframes specified by the PageTimes array.
- *To receive fast repages:* If there was a paging error on the last page read by the access terminal (say in superframe T), and T is an entry in the PageTimes array, then the access terminal shall transition to the monitor state in superframe T+FastRepageInterval.

6.3.5.6.2 Access network requirements

Upon entering the Sleep State from the Inactive State, the access network shall:

- Invoke the procedure in 6.3.5.4.4 to determine the PageTimes array. While invoking this procedure, the access network shall set StartTime such that the access terminal does not miss any pages.

In order to set StartTime above, the access network may use the following procedure. If the access network has received a suspend period from the access terminal, it may set StartTime to the superframe number that contains the last part of the suspend period. If the access network did not receive a suspend period, it may set StartTime to the superframe number where the last packet was received from the access terminal.

If the access network entered the Idle State as upon receiving a ConnectionClose message with CloseReason set to “Deregistration Request”, the access network shall not enter the Monitor State.

Otherwise, the access network shall enter the monitor state at one of the following two times:

- The smallest entry in the PageTimes array that is greater than the current superframe number (for routine pages).
- The sum of FastRepageInterval and the largest entry in the PageTimes array that is less than or equal to the current superframe number (for fast repages).

The setting of which one of the two times above is selected is beyond the scope of this specification, and may depend on the implementation of the fast repage mechanism at the access network.

When the access network is in the Sleep State, it is prohibited from sending unicast packets to the access terminal.

If the access network receives a *ReverseTrafficChannelMAC.UATIReceived* indication in the Sleep State, it shall transition to the BindUATI State.

6.3.5.7 Monitor state

The access terminal shall enter the Monitor State either to receive a Page, QuickPage or other messages from the access network.

When the access network is in the Monitor State, it may send unicast packets to the access terminal.

6.3.5.7.1 Access terminal requirements

Upon entering the Monitor State, the access terminal shall:

- Issue *ControlChannelMAC.Activate*
- Issue *ForwardTrafficChannelMAC.Activate*
- Issue *SharedSignalingMAC.Activate*
- Issue *OverheadMessages.Activate*
- Set internal variable NumAccessAttempts to zero

Upon entering the Monitor State from the Inactive State, or if the BindUATI state has been entered since the last visit to the Monitor State, the access terminal shall:

- Invoke the procedure specified in 6.3.5.4.4 to determine the PageTimes array. While invoking this procedure, the access terminal shall set StartTime to the superframe number when the access terminal entered the monitor state.

The access terminal shall comply with the following requirements when in the Monitor State:

- If the current superframe number is in the PageTimes array the Idle State Protocol shall
    - Determine if there is a paging error in the current superframe, where the paging error event is defined in the Control Channel MAC.

The access terminal shall transition to the Access State if any of the following conditions are met:

- The access terminal receives a *ForwardTrafficChannelMAC.PageReceived* indication
- The access terminal receives a *ControlChannelMAC.PageReceived* indication
- The access terminal receives a PageUATI message where the UATI field matches the UATI public data field of the Address Management Protocol
- The access terminal has a queued *OpenConnection* command

The access terminal may transition to the Sleep State if all of the following conditions are met:

- OverheadParametersUpToDate=1
- Transitioning to sleep state will not cause the access terminal to miss a page in the current superframe.
- Access terminal has not advertised a suspend period that is current (see 6.4.5.2.4.1). The suspend period is current if the time advertised in the associated ConnectionClose message is greater than the current system time.[28]

6.3.5.7.2 Access network requirements

The access network shall comply with the following requirements in the Monitor State:

- If the access network receives a *ReverseTrafficChannelMAC.UATIReceived* indication, it shall transition to the BindUATI State. This requirement shall take precedence over other requirements applicable to this state.
- If the access network has a queued *OpenConnection* command, the access network shall
  - Send the access terminal a Page. (Procedures for sending a Page are defined in the Control Channel MAC and the Forward Traffic Channel MAC. The Forward Traffic Channel MAC is used to send the page only if the page does not fit in the Control Channel MAC due to resource limitations.) If a page is sent over the Forward Traffic Channel MAC, the page packet shall begin transmission in PHY Frame index 1 through 7 of the superframe after the superfame where the QuickPage was sent (where the first PHY Frame of the superframe has index 0).
  - If the access terminal has sleep period greater than one superframe, and the page is sent over the Forward Traffic Channel MAC, then the access network shall send the page in the superframe after the superframe where the QuickPage was sent.
  - After the page is sent, transition to the Sleep State
- If the access network does not have a queued *OpenConnection* command, the access terminal shall
  - Transition to the Sleep State.

[28] The access terminal monitors the Control Channel continuously during a suspend period thus avoiding the delay in opening access-network-initiated connections due to the sleep period.

6.3.5.8 Access state

The access terminal and the access network use the Access State to provide a MAC ID to the access terminal.

6.3.5.8.1 Access terminal requirements

The access terminal shall comply with the following requirements.

- Upon entering the Access State the access terminal shall:
  - Issue a *ControlChannelMAC.Activate* command
  - Issue a *SharedSignalingMAC.Activate* command
  - Issue a *AccessChannelMAC.Activate* command
  - Issue a *ForwardTrafficChannelMAC.Activate* command
  - Issue a *OverheadMessages.Activate* command
  - Issue a *AccessChannelMAC.AttemptAccess* command
  - Issue a *ActiveSetManagement.SendPilotReport* command
  - Set a state timer for $T_{IDPATSetup}$ seconds
  - If NumAccessAttempts is '0', generate a ConnectionOpenRequest message
- If the state timer expires or the protocol receives an *AccessChannelMAC.AccessFailed* indication, the access terminal shall set internal variable NumAccessAttempts to 0, return a *ConnectionFailed* indication and transition to the Sleep State.
- If the access terminal receives a *AccessChannelMAC.AccessGrantReceived* indication, it shall:
  - Increment public data ConnectCount by 1
  - Issue a *ReverseTrafficChannelMAC.Activate* command
  - Issue a *ReverseControlChannelMAC.Activate* command
  - Transition to the BindUATI State

6.3.5.8.2 Access network requirements

The Access State is not applicable to the Idle State Protocol at the Access Network.

6.3.5.9 BindUATI state

6.3.5.9.1 Access terminal requirements

The access terminal shall comply with the following requirements.

- Upon entering the BindUATI State the access terminal shall:
  - Start a state timer for $T_{IDSTABind}$ seconds.
  - Increment internal variable NumAccessAttempts by 1.
  - Transmit a ConnectionOpenRequest message with the RegistrationRadiusFlag field set to the public data RegistrationRadiusFlag of the Active Set Management Protocol.

- If the state timer expires, and NumAccessAttempts is less than MaxAccessAttempts, the access terminal shall transition to the Access State.
- If the state timer expires and NumAccessAttempts is greater than or equal to MaxAccessAttempts the access terminal shall execute the cleanup procedures given in 6.3.5.9.1.3.

6.3.5.9.1.1 Processing the ForwardTrafficChannelMAC.UATIReceived indication

The access terminal shall process this indication according to the following rules.

The access terminal shall declare the indication valid if the ATIdentifier that accompanies the indication matches the ATIdentifier last transmitted by the access terminal.

- If the indication is not valid, the access terminal shall perform the following:
  - If NumAccessAttempts is less than MaxAccessAttempts, the access terminal shall transition to the Access State, otherwise
  - The access terminal shall execute the cleanup procedures given in 6.3.5.9.1.3.
- If the access terminal receives a valid indication with UATIStatus equal to 0x1, it shall execute the cleanup procedures given in 6.3.5.9.1.3.
- If the access terminal receives a valid indication with UATIStatus equal to 0x0 it shall reset the state timer.

6.3.5.9.1.2 Processing the ConnectionOpenResponse message

On receiving a ConnectionOpenResponse message and a *ForwardTrafficChannelMAC.UATIReceived* indication in the same packet, the access terminal shall declare the ConnectionOpenResponse message to be invalid if the UATIStatus field with the indication is set to 0x2.

Otherwise, it shall declare the ConnectionOpenResponse message to be valid.

On receiving a valid ConnectionOpenResponse message with ConnectionStatus set to 0x0, the access terminal shall:

- Return a *ConnectionOpened* indication
- Set NumAccessAttempts to 0
- Transition to the Inactive State

On receiving a valid ConnectionOpenResponse message with ConnectionStatus set to 0x1, the access terminal shall:

- Set NumAccessAttempts to 0.
- Generate a *RegistrationRadiusFlagUpdated* indication accompanied by the RegistrationRadius field of the ConnectionOpenResponse message.
- Transition to the Monitor State.

6.3.5.9.1.3 Cleanup procedures for the BindUATI state

The access terminal shall do the following when this procedure is invoked:

- Set NumAccessAttempts to 0
- Issue a *ReverseTrafficChannelMAC.Deactivate* command
- Issue a *ReverseControlChannelMAC.Deactivate* command
- Issue a *AccessChannelMAC.Deactivate* command
- Return a *ConnectionFailed* indication
- Transition to the Sleep State.

6.3.5.9.2 Access network requirements

Upon entering the BindUATI State, the access network shall:

- Send a packet with the header field UATIInfoIncluded set to '1'. The setting of UATIStatus in the MAC header is beyond the scope of this specification.
- If the access network sets UATIStatus to 1 (does not accept the connection), it shall:
  - Return a *ConnectionFailed* indication
  - Issue a *ReverseTrafficChannelMAC.Deactivate* command
  - Issue a *ReverseControlChannelMAC.Deactivate* command
  - Transition to the Sleep State.
- If the access network accepts the connection request (sends a ConnectionOpenResponse message with ConnectionStatus set to 0), the Idle State Protocol shall return a *ConnectionOpened* Indication and transition to the Inactive State. The access network should set the RegistrationRadiusFlag field of the ConnectionOpenResponse message to '0' only if the RegistrationRadiusFlag field of the ConnectionOpenRequest message was set to 0.

Note that activation of the Reverse Traffic Channel MAC and the Reverse Control Channel MAC at the access network is performed by the Access Channel MAC Protocol in the Lower MAC Sublayer.

6.3.6 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes.

6.3.6.1 ConnectionOpenRequest

The access terminal sends the ConnectionRequest message to the access network to request the opening of a connection.

Field	Length (bits)
MessageID	8
ConnectRequestReason	2
RegistrationRadiusFlag	1
Reserved	5

MessageID The access network shall set this field to 0x00.

ConnectRequestReason
The access terminal shall set this field according to Table 38.

Table 38 Encoding of the ConnectRequestReason field

Field value	Description
0x0	Response to page
1	Registration Attempt
2	Terminal initiated data transfer
All other values are reserved	

RegistrationRadiusFlag
The access terminal shall set this field based on RegistrationRadiusFlag that is public data of the Active Set Management Protocol.

Reserved This field shall be ignored by the receiver.

Channels	RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

6.3.6.2 ConnectionOpenResponse

The access network sends the ConnectionOpenResponse message to the access terminal in response to a ConnectionOpenRequest message.

Field	Length (bits)
MessageID	8
ConnectionStatus	1
RegistrationRadiusFlag	1
Reserved	6

MessageID The access network shall set this field to 0x01.

ConnectionStatus The access terminal shall set this field according to Table 39.

Table 39 Encoding of the ConnectionStatus field

Field value	Description
0	Connection opened
1	Registration successful, connection closed
All other values are reserved	

RegistrationRadiusFlag
The setting of this field at the access network is beyond the scope of this specification.

Reserved This field shall be ignored by the receiver.

Channels	FTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

6.3.6.3 PageUATI

The access network may send the PageUATI message to direct the access terminal to request a connection.

Field	Length (bits)
MessageID	8
UATI	128

MessageID The access network shall set this field to 0x02.

UATI The access network shall set this field to the UATI of the access terminal, where the UATI is public data of the Address Management Protocol.

Channels	FTC		**SLP**	Best Effort
Addressing	Broadcast Unicast		**Security**	Required

6.3.6.4 PreferredChannelRequest

The access terminal sends this message to request a connection on a channel that is different from the channel where the access preamble was sent. Sending the PreferredChannelRequest message during connection initiation is optional.

Field	Length (bits)
MessageID	8
MessageSequence	8
PreferredChannelBandCount	8
PreferredChannelBandCount occurrences of the following field:	
PreferredChannelBand	ChannelBandRecordType Dependent

MessageID The access terminal shall set this field to 0x03.

MessageSequence The access terminal shall increment this field modulo 256 for each new PreferredChannelRequest message sent. If this is the first PreferredChannelRequest message sent by the access terminal, the access terminal shall set this field to zero.

PreferredChannelBandCount
The access terminal shall set this field to the number of occurrences of the PreferredChannelBand field in this message.

PreferredChannelBand The access terminal shall set this field to the ChannelBand record specification for the channel on which the access terminal prefers to be assigned a Traffic Channel.

Channels	RTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

6.3.7 Interface to other protocols

6.3.7.1 Commands

This protocol issues the following commands:

- *ActiveSetManagement.SendPilotReport*
- *OverheadMessages.Activate*
- *OverheadMessages.Deactivate*
- *ControlChannelMAC.Activate*
- *ControlChannelMAC.Deactivate*
- *ForwardTrafficChannelMAC.Activate*

- *ReverseTrafficChannelMAC.Activate*
- *ReverseControlChannelMAC.Activate*
- *AccessChannelMAC.AttemptAccess*
- *SharedSignalingMAC.Activate*
- *SharedSignalingMAC.Deactivate*

6.3.7.2 Indications

This protocol registers to receive the following indications:

- *AccessChannelMAC.AccessGrantReceived*
- *AccessChannelMAC.AccessFailed*
- *ReverseTrafficChannelMAC.UATIReceived*
- *ForwardTrafficChannelMAC.UATIReceived*
- *ControlChannelMAC.PageReceived*
- *ControlChannelMAC.QuickPageReceived*
- *ForwardTrafficChannelMAC.PageReceived*

6.3.8 Configuration attributes

The following complex attributes and default values are defined (see 10.3 for attribute record definition).

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Idle State Protocol.

6.3.8.1 Preferred paging attribute

Field	Length (bits)	Default
Length	8	N/A
AttributeID	8	N/A
PreferredPageOffsetEnabled	1	‘0’
PreferredPageOffset	0 or 15	N/A
PreferredQuickPageEnabled	1	0
PreferredRQuickPage	0 or 15	N/A
PreferredPagingCarrierEnabled	1	0
PreferredPagingCarrier	7	N/A
Reserved	7 or 0	N/A

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — This field shall be set to 0x00.

PreferredPageOffsetEnabled
This field shall be set to '1' if PreferredPageOffset field is included in this attribute; otherwise, the sender shall set this field to '0'.

PreferredPageOffset
If PreferredPageOffsetEnabled is set to '1', this field shall specify the superframe in which the access terminal transitions out of the Sleep State (see 6.3.5.6) in order to monitor the Control Channel. This field shall be omitted if PreferredPageOffsetEnabled is set to '0'.

PreferredQuickPageEnabled
This field shall be set to '1' if the PreferredQuickPageCycle field is included in this attribute; otherwise, this field shall be set field to '0'.

PreferredRQuickPage PreferredQuickPageEnabled is set to '1', this field shall be set to specify the response of the access terminal to a QuickPage packet (see 6.3.5.4.2). This field shall be omitted this field if PreferredQuickPageEnabled is set to '0'.

Reserved The length of this field shall be such that the attribute value is octet-aligned. This field shall be set to zero.

6.3.8.2 SlottedMode attribute

Field	Length (bits)	Default
Length	8	N/A
AttributeID	8	N/A
SlotCycleBase	8	0x2
SlotCycle1	8	0x9
SlotCycle2	8	0x9
SlotCycle3	8	0x9
WakeCount1	8	0x0
WakeCount2	8	0x0
Reserved	0	N/A

Length Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID The sender shall set this field to 0x01.

SlotCycleBase The sender shall set this field to the SlotCycleBase that is used in calculating Period*j*. SlotCycleBase shall take only even values.

SlotCycle1 The sender shall set this field to SlotCycle1.

SlotCycle2 The sender shall set this field to SlotCycle2. SlotCycle2 shall be greater than or equal to SlotCycle1.

SlotCycle3 — The sender shall set this field to SlotCycle3. SlotCycle3 shall be greater than or equal to SlotCycle2.

WakeCount1 — The sender shall set this field to WakeCount1.

WakeCount2 — The sender shall set this field to WakeCount2. WakeCount2 shall be greater or equal to than WakeCount1.

Reserved — The sender shall set this field to ‘0000’. The receiver shall ignore this field.

6.3.8.3 FastRepage attribute

Field	Length (bits)	Default
Length	8	N/A
AttributeID	8	N/A
FastRepageEnabled	8	0
FastRepageInterval	16	0

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The sender shall set this field to 0x03.

FastRepageEnabled — The sender shall set this field to 0x01 if FastRepage is enabled

FastRepageInterval — This sender shall set this field to zero if FastRepageEnabled is not equal to 0x01. Otherwise, the sender shall set this field to the interval at which the access network pages the access terminal when the access network receives no response to the page. The unit for this field shall be superframes. This field shall not take odd values. A fast repage is performed only once for each missed page.

6.3.8.4 MaxAccessAttempts attribute

Field	Length (bits)	Default
Length	8	N/A
AttributeID	8	N/A
MaxAccessAttempts	8	0x03

MaxAccessAttempts — The sender shall set this field to the maximum number of visits to the Access State before the access attempt is stopped.

6.3.9 Protocol numeric constants

Constant	Meaning	Value	Comments
$N_{IDPType}$	Type field for this protocol	**Table 9**	
$N_{IDPDefault}$	Subtype field for this protocol	0x0000	
$T_{IDSTABind}$	Maximum access terminal time in the BindUATI State	2.5 seconds	

6.3.10 Session state information

The Session State Information record (see 10.10) consists of the PageTimes array and the parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

6.4 Default Connected State Protocol

6.4.1 Overview

The Default Connected State Protocol provides procedures and messages used by the access terminal and the access network while a connection is open.

This protocol can be in one of three states:

- *Inactive State*: In this state the protocol waits for an *Activate* command.
- *Open State*: In this state the access terminal can use the Reverse Traffic Channel and the access network can use the Forward Traffic Channel and Control Channel to send application traffic to each other.
- *Close State*: This state is associated only with the access network. In this state the access network waits for connection resources to be safely released.

Figure 52 and Figure 53 show the state transition diagrams at the access terminal and the access network respectively.

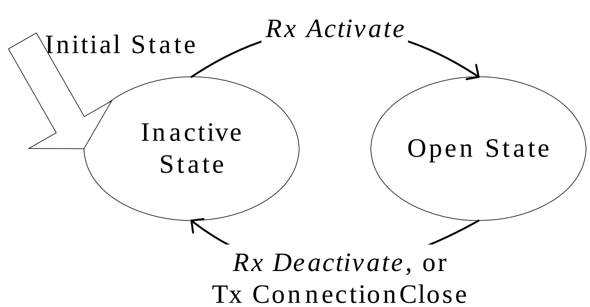

Figure 52 Default Connected State Protocol state diagram (access terminal)

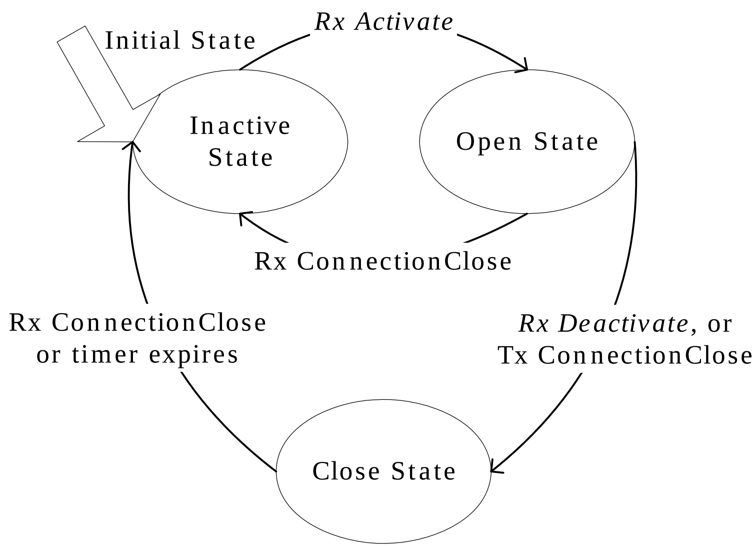

Figure 53 Default Connected State Protocol state diagram (access network)

6.4.2 Primitives

6.4.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*
- *CloseConnection*[29]

6.4.2.2 Return indications

This protocol returns the following indications:

- *ConnectionClosed*
- *RegistrationRadiusUpdated*
- *TunedAway*
- *TunedBack*

6.4.3 Public data

6.4.3.1 Static public data

This protocol does not define any static public data

[29] The *CloseConnection* command performs the same function as the *Deactivate* command and is provided for clarity in the specification.

6.4.3.2 Dynamic public data

- Subtype for this protocol
- RLImplicitDeassignEnabled (configuration attribute)
- FLImplicitDeassignEnabled (configuration attribute)
- TuneAwayStatus
- SelectedInterlaceMode
- SelectedInterlaceAssignment

6.4.4 Protocol initialization and swap procedures

6.4.4.1 Protocol initialization

Upon initialization at the access terminal and the access network:

- The value for each attribute for this protocol instance shall be set to the default value for that attribute.
- The protocol shall enter the Inactive State.

6.4.4.2 Protocol swap

Upon swap at the access terminal and access network:

- The protocol shall enter the Inactive State.

6.4.5 Procedures

6.4.5.1 Command processing

6.4.5.1.1 Activate

When the protocol receives an *Activate* command in the Inactive State:

- The access terminal shall transition to the Open State.
- The access network shall transition to the Open State.

When the protocol receives this command in any other state it shall be ignored.

6.4.5.1.2 Deactivate

When the protocol receives a *Deactivate* command in the Inactive State it shall return a *ConnectionClosed* indication.

When the protocol receives a *Deactivate* command in the Close State the command shall be ignored.

When the protocol receives this command in the Open State:

- Access terminal shall send a ConnectionClose message to the access network and perform the cleanup procedures defined in 6.4.5.2.4.2.
- Access network shall send a ConnectionClose message to the access terminal and transition to the Close State.

6.4.5.1.3 CloseConnection

The access terminal and the access network shall process the *CloseConnection* command following the same procedures used for the *Deactivate* command, see 6.4.5.1.2.

6.4.5.2 Open state

In the Open State, the access terminal and the access network maintain a connection and can use it to exchange application traffic on the Reverse Traffic Channel, Forward Traffic Channel, and Control Channel.

6.4.5.2.1 TuneAway procedures

Tune away defines a repetitive set of time periods during which the access terminal and access network do not exchange any transmission. TuneAway is used by the access terminal and access network to allow the access terminal to measure the availability of other ChannelBands or other technologies. Additionally, TuneAway is used by the access terminal when the some overhead parameters are not up to date.

The beginning and end of tune away is determined by the TuneAway attribute and the TuneAwayRequest and TuneAwayResponse messages. Further, the access terminal and access network may operate on multiple tune away schedules. Each tune away schedule is specified by a separate TuneAway attribute, but may share the same TuneAwayRequest and TuneAwayResponse messages.

The tune away operation is controlled through a variable TuneAwayStatus, that is public data of the protocol.

If the TuneAwayStatus is set to ‘1’ at the access terminal:

- The access terminal may stop monitoring the forward channels and shall stop transmitting on the reverse channels.

If the TuneAwayStatus is set to ‘1’ at the access network:

- The access network may stop monitoring the reverse channels and shall stop transmitting to this access terminal on the forward channels.

6.4.5.2.1.1 TuneAway time calculations

The following formulas determine the beginning and end of tune away periods for a tune away schedule as a function of the TuneAway attribute and the TuneAwayRequest message. In the following calculations, the current TuneAway attribute is used, and the TuneAwayRequest message with MessageSequence matching the last received TuneAwayResponse message is used.

Consider tune away schedule N and consider a given serving sector. Let t_0 be the system time at the beginning of StartSuperframeNumber*N*. Then, tuneaway period number n for a sector with given SectorOffset begins at

$$\text{TuneAwayTime}N_n = t_0 + \text{StartSuperframeOffset}N + \text{SectorOffset} + (n-1)*\text{TuneAwayPeriodicity}N$$

and ends at

$$\text{TuneBackTime}N_n = \text{TuneAwayTime}N_n + \text{TuneAwayDuration}N$$

Since a TuneAwayTime and a TuneBackTime may be misaligned with PHY Frame boundaries, actual tune away and tune back operations obey the following parameters.

TuneAwayFrameN_n shall be set to the PHY Frame number of the reverse link PHY Frame that contains time instance TuneAwayTimeN_n. In case TuneAwayTimeN_n lies on a boundary of PHY Frames j and j+1, TuneAwayFrameN_n shall be set to j.

TuneBackFrameN_n shall be set to the PHY Frame number of the reverse link PHY Frame that contains time instance TuneBackTimeN_n. In case TuneBackTimeN_n lies on a PHY Frame boundary of PHY Frames j and j+1, TuneBackFrameN_n shall be set to j+1.

For each tune away schedule, the access terminal and access network may be in one of three states:

- *Disabled State:* When a tune away schedule is in this state, the access terminal does not tune away for that tune away schedule.
- *Camped State:* When a tune away schedule is in this state, it does not require the access terminal to tune away.
- *TunedAway State:* When a tune away schedule is in this state, it requires the access terminal to tune away.

In addition to the above per schedule state, the access terminal shall also perform State transitions for a tune away schedule.

Access Terminal

The access terminal shall enter the disabled state for a tune away schedule N if

- No valid TuneAwayResponse message has been received
- The access terminal sends a TuneAwayRequest message with the TuneAwayEnabled*N* field set to ‘0’.

The access terminal shall enter the Camped state for a tune away schedule N if

- If the access terminal receives a TuneAwayResponse message with the TuneAwayEnabled*N* field set to ‘1’.
- At the beginning of PHY Frame number TuneBackFrameN_n

The access terminal shall enter the TunedAway state for a tune away schedule N

- At the end of PHY Frame number TuneAwayFrameN_n - 1

Access Network

The access network shall enter the disabled state for a tune away schedule N if

- No TuneAwayResponse message for tune away schedule N has been sent
- The access network sends a TuneAwayResponse message with the TuneAwayEnabled*N* field set to ‘0’.

The access network shall enter the Camped state for a tune away schedule N if

- If the access network sends a TuneAwayResponse message with the TuneAwayEnabled*N* field set to ‘1’.
- At the beginning of PHY Frame number TuneBackFrameN_n

The access network shall enter the TunedAway state for a tune away schedule N

- At the end of PHY Frame number TuneAwayFrameN_n - 1

6.4.5.2.1.2 Procedures for setting TuneAwayStatus

The access terminal shall set the public data TuneAwayStatus as follows:

- If any of the tune away schedules is in TuneAway state, this protocol shall set TuneAwayStatus to ‘1’.
- If QuickChannelInfoUptoDate is ‘0’ or ExtendedChannelInfoUptoDate is ‘0’ in the public data of the Overhead Messages Protocol, the access terminal shall set TuneAwayStatus to ‘1’.
- Otherwise, the access terminal shall set TuneAwayStatus to ‘0’.

The access network shall set the public data TuneAwayStatus as follows:

- If any of the tune away schedules is in TuneAway state, this protocol shall set TuneAwayStatus to ‘1’.
- Otherwise, the access terminal shall set TuneAwayStatus to ‘0’.

This protocol shall generate the following indications:

- If TuneAwayStatus changes from ‘1’ to ‘0’, this protocol shall generate a TunedBack indication.
- If TuneAwayStatus changes from ‘0’ to ‘1’, this protocol shall generate a TunedAway indication.

6.4.5.2.2 SelectedInterlace operation procedures

The access terminal and access network may operate in one of two modes: SelectedInterlaceOn or SelectedInterlaceOff.

- *SelectedInterlaceOn* mode: In this mode, the access network sends certain SSCH blocks to the access terminal only on a set of interlaces called the SelectedInterlaceSet. Details may be found in the Lower MAC Sublayer.
- *SelectedInterlaceOff* mode: In this mode, no restrictions are placed on the access network and access terminal.

6.4.5.2.2.1 State transitions for selected interlace operation

The access network shall enter the SelectedInterlaceOn mode after

- Sending a SelectedInterlaceAck message with SelectedInterlaceEnabled set to ‘1’.

The access terminal shall enter the SelectedInterlaceOn mode after

- Receiving a SelectedInterlaceAck message with SelectedInterlaceEnabled set to ‘1’.

The access terminal shall enter the SelectedInterlaceOff mode after

- Receiving a SelectedInterlacesAssignment message with SelectedInterlacesEnabled equal to ‘0’.
- When the desired serving sector is not the same as the serving sector

The access network shall enter the SelectedInterlaceOff mode after

- Receiving a SelectedInterlaceAck message with the SelectedInterlacesEnabled field equal to ‘0’
- The serving sector for the access terminal changes

If the access terminal receives a SelectedInterlaceAssignment message with SelectedInterlacesEnabled equal to ‘0’, the access terminal shall respond with a SelectedInterlaceAck message.

To change the selected interlace assignment to an access terminal, the access network should first disable selected interlace mode, and then send a SelectedInterlaceAssignment message.

6.4.5.2.2.2 Procedures in selected interlace states

On entering the SelectedInterlaceOn state

- The access terminal and access network shall set public data SelectedInterlaceMode to ‘1’.
- Place the most recent SelectedInterlaceAssignment message in the public data.

On entering the SelectedInterlaceOff state

- The access terminal and access network shall set public data SelectedInterlaceMode to ‘0’.

6.4.5.2.3 Channel measurement procedures

The access network may obtain channel measurement reports from the access terminal by sending a ChannelMeasurementReportRequest message.

If an access terminal receives a ChannelMeasurementReportRequest message, the access terminal may respond with a ChannelMeasurementReport message. Channel measurements are based on the F-CPICH.

6.4.5.2.4 Access terminal requirements

6.4.5.2.4.1 General requirements

The access terminal shall comply with the following requirements when in the Open State:

- The access terminal shall receive the Control Channel and the Forward Traffic Channel.
- The access terminal may request a MIMO mode on the Forward Traffic Channel by sending a MIMORequest message.
- The access terminal shall monitor the overhead messages as specified in the Overhead Messages Protocol (see 6.5.5.4.2).
- If the access terminal receives a ConnectionClose message, it shall generate a *RegistrationRadiusUpdated* indication accompanied by the RegistrationRadiusFlag contained in the message.
- If the access terminal receives a ConnectionClose message, it shall send a ConnectionClose message with CloseReason set to "Close Reply" and execute the cleanup procedures defined in 6.4.5.2.4.2.
- If the access terminal sends a ConnectionClose message, it may advertise, as part of the ConnectionClose message, that it shall be monitoring the Control Channel continuously, until a certain time following the closure of the connection. This period is called a suspend period, and can be used by the access network to accelerate the process of sending a unicast packet (and specifically, a Page message or ActiveSetAssignment message) to the access terminal. The suspend period shall be said to be current from the time the access terminal sends the ConnectionClose message to the time given in the SuspendTime field of the ConnectionClose message.

6.4.5.2.4.2 Cleanup procedures

If the access terminal executes cleanup procedures it shall:

- Return a *ConnectionClosed* indication.
- Transition to the Inactive State.

6.4.5.2.5 Access network requirements

6.4.5.2.5.1 General requirements

The access network shall comply with the following requirements when in the Open State:

- Access network shall receive the Reverse Traffic Channel and may transmit on the Forward Traffic Channel.
- If access network receives a ConnectionClose message, it shall consider the connection closed, and it should execute the cleanup procedures defined in 6.4.5.2.5.2.
- If access network requires closing the connection, it shall transmit a ConnectionClose message, and transition to the Close State.

6.4.5.2.5.2 Cleanup procedures

When the access network performs cleanup procedures it shall:

- Return a *ConnectionClosed* indication.
- Transition to the Inactive State.

6.4.5.3 Close state

The Close State is associated only with the access network. In this state the access network waits for a replying ConnectionClose message from the access terminal or for the expiration of the "CSP Close Timer" defined below.

Upon entering this state, the access network shall set a "CSP Close Timer" for $T_{CSPClose}$ seconds. If the access network receives a ConnectionClose message in this state, or if the timer expires, it shall execute the cleanup procedures defined in 6.4.5.2.5.2, it may close all connection-related resources assigned to the access terminal, and it should transition to the Inactive State.

6.4.6 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes.

6.4.6.1 ConnectionClose

The access terminal and the access network send the ConnectionClose message to close the connection.

Field	Length (bits)
MessageID	8
CloseReason	3
SuspendEnable	1
SuspendTime	0 or 34
RegistrationRadiusFlag	1
Reserved	variable

MessageID — The sender shall set this field to 0x00.

CloseReason — The sender shall set this field to reflect the close reason, as shown in Table 40.

Table 40 Encoding of the CloseReason field

Field value	Description
'000'	Normal Close; Reason Unspecified
'001'	Close Reply
'010'	Connection Error
'011'	Deregistration Request
'100'	Normal close requested by access terminal because the connection was opened for registration.
All other values are reserved	

SuspendEnable — The access terminal shall set this field to '1' if it will enable a suspend period following the close of the connection. The access terminal shall set this field to '0' if the CloseReason field is set to "Deregistration Request". The access network shall set this field to '0'.

SuspendTime — Suspend period end time. This field is included only if the SuspendEnable field is set to '1'. The access terminal shall set this field to the absolute system time of the end of its suspend period in units of superframes.

RegistrationRadiusFlag — This field shall be set by the access terminal to RegistrationRadiusFlag that is public data of the Active Set Management Protocol.

Reserved — The length of this field shall be such that the entire message is octet-aligned. The sender shall set this field to zero. The receiver shall ignore this field.

Channels	FTC RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

6.4.6.2 MIMORequest

This message shall be sent by the access terminal to the access network to indicate its MIMO capabilities.

Field	Length (bits)
MessageID	8
SupportedMIMOMode	2
Reserved	6

MessageID — The access terminal shall set this field to 0x01.

SupportedMIMOMode
The access terminal shall set this field to indicate the MIMO modes it supports, as shown in Table 41.

Table 41 Encoding of the SupportedMIMOMode field

Field value	Description
'00'	MIMO not supported
'01'	Space Time Transmit Diversity (STTD)
'10'	Single Code Word (SCW)
'11'	Multiple Code Word (MCW)

Reserved — The access terminal shall set this field to all zeros.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

6.4.6.3 SelectedInterlaceRequest

This message shall be sent by the access terminal to request a selected interlace mode with a particular sector.

Field	Length (bits)
MessageID	8
PilotPN	12
InterlacesRequested	4

MessageID — This field shall be set to 0x02

PilotPN — This field shall be set to the PilotPN of the sector to which this message is directed. The access network shall ignore this message if the PilotPN does not match the PilotPN of the sector that received the message.

InterlacesRequested — The access terminal shall set this field to indicate a requested number of interlaces requested.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

6.4.6.4 SelectedInterlaceAssignment

This message shall be sent by the access network to assign a selected interlace mode to an access terminal.

Field	Length (bits)
MessageID	8
PilotPN	12
SelectedInterlacesEnabled	1
NumAssignedInterlaces	4
NumAssignedInterlaces instances of the following field	
InterlaceID	3
Reserved	Variable

MessageID — This field shall be set to 0x03.

PilotPN — This field shall be set to the PilotPN of the sector that sent this message.

SelectedInterlacesEnabled — If this field is set to '1' the access terminal shall operate in SelectedInterlace mode. If this field is set to '0' the access terminal shall not operate in SelectedInterlace mode.

NumAssignedInterlaces — The access network shall set this field to the number of assigned interlaces, or to 0 if the SelectedInterlacesAssigned field is set to '0'.

InterlaceID — This field shall be set to an interlace assigned to the access terminal for SelectedInterlace operation.

Reserved — The length of this field shall be such that the entire message is octet-aligned. The sender shall set this field to zero. The receiver shall ignore this field.

Channels	FTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

6.4.6.5 SelectedInterlaceAck

This message shall be sent by the access terminal to acknowledge transition to SelectedInterlacesOff state.

Field	Length (bits)
MessageID	8
PilotPN	12
SelectedInterlaceEnabled	1
Reserved	3

MessageID — This field shall be set to 0x04.

PilotPN — This field shall be set to the PilotPN of the sector to which this message is directed. The access network shall ignore this message if the PilotPN does not match the PilotPN of the sector that received the message.

SelectedInterlaceEnabled — The access terminal shall set this field to ‘1’ if it has selected interlace mode enabled, and to ‘0’ otherwise.

Reserved — The sender shall set this field to zero. The receiver shall ignore this field.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

6.4.6.6 TuneAwayRequest

This message shall be sent by the access terminal to control tune away operations.

Field	Length (bits)
MessageID	8
MessageSequence	16
TuneAwayEnabledMap	$N_{CSPTuneAwayMaxSched}$
NumPilots	3
NumPilots instances of the following field	
ActiveSetIndex	3
SectorOffset	24
Reserved	Variable

MessageID	This field shall be set to 0x05.
MessageSequence	The access terminal shall set this field to the sequence number of this message. The sequence number of this message is 1 more than the sequence number of the last TuneAwayRequest message (modulo 65536) sent by this access terminal. If this is the first TuneAway message sent by the access terminal, it shall set this field to 0x00.
TuneAwayEnabledMap	Bit position *N* of this field shall be set to TuneAwayEnabled*N*. TuneAwayEnabled*N* shall be set to '1' if the terminal will tune away at periodic intervals corresponding to tune away schedule *N*. TuneAwayEnabled*N* shall be set to '0' if the terminal will not tune away corresponding to tune away schedule *N*.
NumPilots	This field shall be set to the number of pilots included in the message.
ActiveSetIndex	This field shall be used to identify Active Set members, as indexed in the ActiveSetAssignment message of the Active Set Management Protocol.
SectorOffset	This field shall be set to the time, in units of 1 microsecond, that the terminal adds to the StartSuperframeOffset attribute when this Active Set member is the serving sector. The access terminal should determine this field based on the timing offset it measures between different sectors.
Reserved	The length of this field shall be such that the entire message is octet-aligned. The sender shall set this field to zero.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

6.4.6.7 TuneAwayResponse

This message shall be sent by the access network to control tune away operations.

Field	Length (bits)
MessageID	8
MessageSequence	16
TuneAwayEnabledMap	$N_{CSPTuneAwayMaxSched}$
Reserved	Variable

MessageID	This field shall be set to 0x06.
MessageSequence	The access network shall set this field to the last received TuneAwayRequest message sent to this access terminal.

TuneAwayEnabledMap

Bit position *N* of this field shall be set to TuneAwayEnabled*N*. TuneAwayEnabled*N* shall be set to '1' if the access network will tune away at periodic intervals corresponding to tune away schedule *N*. TuneAwayEnabled*N* shall be set to '0' if the access network will not tune away corresponding to tune away schedule *N*.

Reserved — The length of this field shall be such that the entire message is octet-aligned. The sender shall set this field to zero. The receiver shall ignore this field.

Channels	FTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

6.4.6.8 ChannelMeasurementReportRequest

The access network sends this message to request a ChannelMeasurementReport from one or more access terminals.

Field	Length (bits)
MessageID	8
PilotPN	12
CarrierID	2
StartPHYFrame	40
NumChannels	3
MeasurementsPerMessage	8
NumMeasurementsRequested	8
Reserved	4

MessageID — This field shall be set to 0x07.

PilotPN — This field shall be set to the PilotPN of the sector requesting the measurement report.

CarrierID — This field shall be set to the carrier on which the measurements are requested.

StartPHYFrame — This field shall be set to the PHY Frame number of the PHYFrame where access terminals are required to begin measurements.

NumChannels — This field shall be set to the number of channels to be measured by the access terminal. Each measured channel corresponds to a different transmit antenna at the sector being measured.

MeasurementsPerMessage

This field shall determine the number of measurements (in terms of PHY Frames measured) to be included in one ChannelMeasurementReport message.

NumMeasurementsRequested
This field shall determine the total number of measurements to be made by the access terminal.

Reserved This field shall be set to zero. The receiver shall ignore this field.

Channels	FTC		**SLP**	Reliable
Addressing	Broadcast Unicast		**Security**	Required

6.4.6.9 ChannelMeasurementReport

The access terminal sends this message to report channel measurements.

Field	Length (bits)
MessageID	8
PilotPN	12
CarrierID	2
StartPHYFrameNumber	40
MeasurementInterval	8
NumMeasurements	8
NumMeasurements instances of the following record{	
NumChannels	3
NumChannels instances of the following record{	
NumTaps	3
NumTaps instances of the following record{	
TapOffset	5
RealGain	8
ImagGain	8
}}}	
Reserved	Variable

MessageID This field shall be set to 0x08.

PilotPN This field shall be set to the PilotPN of the sector for which the measurement was performed.

CarrierID This field shall be set to the carrier on which the measurements are performed.

StartPHYFrameNumber
This field shall be set to the PHY Frame number of the PHY Frame where access terminal made the first measurement reported in this message.

MeasurementInterval	This field shall determine the number of PHY Frames between measurements made by the access terminal.
NumMeasurements	This field shall be set to the number of measurements included in this message. Each measurement corresponds to a different PHY Frame.
NumChannels	This field determines the number of channels measured by the access terminal. The access terminal shall set this field to equal to the NumChannels field in the received ChannelMeasurementReportRequest message.
NumTaps	This field shall be set to the number of taps being reported.
TapOffset	This field shall be set to a offset for which the channel was measured.
RealGain	This field shall be set to the real component of the measured channel gain on the corresponding TapOffset.
ImagGain	This field shall be set to the imaginary component of the measured channel gain on the corresponding TapOffset.
Reserved	The length of this field shall be such that the entire message is octet-aligned. The sender shall set this field to zero. The receiver shall ignore this field.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

6.4.7 Interface to other protocols

6.4.7.1 Commands

This protocol does not issue any commands.

6.4.7.2 Indications

This protocol does not register to receive any indications.

6.4.8 Configuration attributes

The following configuration attributes are defined for this protocol.

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Connected State Protocol.

6.4.8.1 Simple attributes

The negotiable simple attribute for this protocol is listed in Table 42. The access terminal and the access network shall use as defaults the values in Table 42 that are listed in ***bold italics***.

Table 42 Configurable values

Attribute ID	Attribute	Values	Meaning
0x00	RLImplicitDeassignEnabled	***0x00***	Reverse link assignments are expired at the beginning of tune away
		0x01	Reverse link assignments are not expired at the beginning of tune away
		0x02-0xff	Reserved
0x01	FLImplicitDeassignEnabled	***0x00***	Forward link assignments are expired at the beginning of tune away
		0x01	Forward link assignments are not expired at the beginning of tune away
		0x02-0xff	Reserved

6.4.8.2 Complex attributes

The following complex attributes and default values are defined (see 10.3 for attribute record definition).

6.4.8.3 TuneAwaySchedule*N* attribute

N takes values from 0 through $N_{CSPTuneAwayMaxSched}$ -1.

This complex attribute shall determine the periodicity, duration and offset of tuneaways that the access terminal may perform. Such tuneaways may be used for handoff candidate search or alternate technology page reception.

Field	Length (bits)	Default
Length	8	N/A
AttributeID	8	N/A
StartSuperframeNumber	34	0
StartSuperframeOffset	16	0
TuneAwayDuration	24	0
TuneAwayPeriodicity	24	0x989680
Reserved	6	0

Length — Length of the complex attribute in octets. The sender shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — This field shall set this field to 0x0(*N+2*), where N takes values from 0 through $N_{CSPTuneAwayMaxSched}$ -1.

StartSuperframeNumber — To compute the tuneaway cycles, it shall be assumed that the first tuneaway occurred in this superframe.

StartSuperframeOffset — This field is a measure of time in units of 1 micro second. To compute the tuneaway cycles, it shall be assumed that the first tuneaway begins StartSuperframeOffset time after the beginning of superframe number StartSuperframeNumber.

TuneAwayDuration — This field determines the duration of the tune away in units of 1 micro second.

TuneAwayPeriod — This field determines the time between the start of successive tuneaways in units of 1 microsecond.

Reserved — This field shall be set to all zeros.

6.4.9 Protocol numeric constants

Constant	Meaning	Value	Comments
$N_{CSPType}$	Type field for this protocol	Table 9	
$N_{CSPDefault}$	Subtype field for this protocol	0x0000	
$N_{CSPTuneAwayMaxSched}$	Maximum number of tune away schedules	0x04	
$T_{CSPClose}$	Access network timer waiting for a responding ConnectionClose message	1.5 seconds	

6.4.10 Session state information

The Session State Information record (see 10.10) consists of the parameter records.

This protocol defines the following parameter record in addition to the configuration attributes for this protocol.

6.4.10.1 ConnectedState parameter

The following parameter shall be included in the Session State Information record only if the Session State Information is being transferred while the connection is open.

Table 43 The Format of the parameter record for the ActiveSetManagement parameter

Field	Length (bits)
ParameterType	8
Length	8
SelectedInterlaceAssignmentMessageLength	8
SelectedInterlaceAssignmentMessage	Variable
TuneAwayResponseMessageLength	8
TuneAwayResponseMessage	Variable

ParameterType This field shall be set to 0x01 for this parameter record.

Length This field shall be set to the length of this parameter record in units of octets excluding the Length field.

SelectedInterlaceAssignmentMessageLength
This field shall be set to the length of the last SelectedInterlaceAssignment message that was sent by the source access network.

SelectedInterlaceAssignmentMessage
Last SelectedInterlaceAssignment message that was sent by the source access network.

TuneAwayResponseMessageLength
This field shall be set to the length of the last TuneAwayResponse message that was sent by the source access network.

TuneAwayResponseMessage
Last TuneAwayResponse message that was sent by the source access network

6.5 Overhead Messages Protocol

6.5.1 Overview

The Overhead Messages Protocol is responsible for the transmission, reception and supervision of the SystemInfo block, the QuickChannelInfo block, the ExtendedChannelInfo message and the SectorParameters message. The SystemInfo and QuickChannelInfo blocks are broadcast by the access network directly over the Control Channel MAC Protocol. The ExtendedChannelInfo and SectorParameters messages are broadcast using the Signaling Transport.

This protocol can be in one of two states:

- *Inactive State*: In this state, the protocol waits for an *Activate* command. This state corresponds only to the access terminal and occurs when the access terminal has not acquired an access network or is not required to receive overhead messages.
- *Active State*: In this state the access network transmits and the access terminal receives overhead messages.

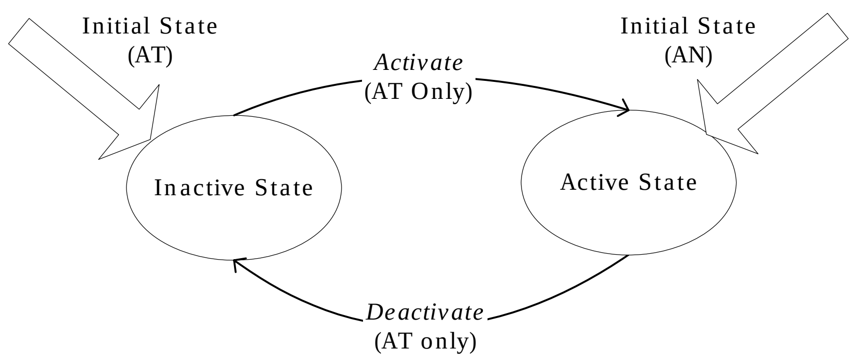

Figure 54 Overhead Messages Protocol state diagram

6.5.2 Primitives

6.5.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

6.5.2.2 Return indications

This protocol returns the following indications:

- *SupervisionFailed*
- *QuickChannelInfoUpdated*
- *ExtendedChannelInfoUpdated*

- *SectorParametersUpdated*
- *OverheadMessagesUpdated*

6.5.3 Public data

6.5.3.1 Static public data

This protocol defines the following static public data:

- An OverheadParameterList that shall contain for each PilotPN in the active set, the following entries. When multi-carrier mode is set to MultiCarrierOn in the public data of the Physical Layer Protocol, each of the following entries shall be maintained independently for each carrier, indexed by CarrierID. Other protocols may refer to a field of a block or message as FieldName(CarrierID). For example, FLChannelTreeIndex(2) refers to the FLChannelTreeIndex on carrier 2.
  - Received SystemInfo block,
  - Received QuickChannelInfo block with associated QuickChannelInfoExpiryTime,
  - Received ExtendedChannelInfo message with associated ExtendedChannelInfoExpiryTime,
- Currently valid SectorParameters messages indexed by the sector's PilotPN
- QuickChannelInfoUpToDate
- OverheadParametersUpToDate
- ExtendedChannelInfoUpToDate
- SectorParametersUpToDate

6.5.3.2 Dynamic public data

- Subtype for this protocol

6.5.4 Protocol initialization and swap procedures

6.5.4.1 Protocol initialization

Upon initialization at the access terminal:

- The value for each attribute for this protocol instance shall be set to the default value for that attribute.
- The protocol shall enter the Inactive State

Upon initialization at the access network:

- The value for each attribute for this protocol instance shall be set to the default value for that attribute.
- The protocol shall enter the Active State

6.5.4.2 Protocol swap

Upon swap at the access terminal the protocol shall enter the Inactive State.

Upon swap at the access network the protocol shall enter the Active State.

6.5.5 Procedures

6.5.5.1 Extensibility requirements

Further revisions of the access network may add new overhead messages.

The access terminal shall discard overhead messages with a MessageID field it does not recognize.

Further revisions of the access network may add new fields to existing overhead messages. These fields shall be added to the end of the message, prior to the Reserved field, if such a field is defined.

The access terminal shall ignore fields it does not recognize.

6.5.5.2 Command processing

The access network shall ignore all commands.

6.5.5.2.1 Activate

If this protocol receives an *Activate* command in the Inactive State:

- The access terminal shall transition to the Active State.
- The access network shall ignore it.

If this protocol receives the command in the Active State, it shall be ignored.

6.5.5.2.2 Deactivate

If this protocol receives a *Deactivate* command in the Inactive State, it shall be ignored.

If this protocol receives the command in the Active State:

- Access terminal shall transition to the Inactive State.
- Access network shall ignore it.

6.5.5.3 Inactive state

This state corresponds only to the access terminal and occurs when the access terminal has not acquired an access network or is not required to receive overhead messages. In this state, the protocol waits for an *Activate* command.

6.5.5.4 Active state

6.5.5.4.1 Access network requirements

If the access network is ready to provide service, it shall broadcast the SystemInfo block, QuickChannelInfo block, ExtendedChannelInfo message and SectorParameters message as specified below. The SystemInfo block, QuickChannelInfo block, ExtendedChannelInfo message and the SectorParameters message shall be public data of the Overhead Messages Protocol.

6.5.5.4.1.1 Procedure for transmission of the SystemInfo block

The SystemInfo block shall be transmitted every N_{pBCH0_Period} superframes. The SystemInfo block shall be carried by the Control Channel MAC Protocol over the pBCH0 physical channel, and shall not pass through the Signaling Transport. N_{pBCH0_Period} is defined in the Physical Layer Protocol. If the multi-carrier mode is MultiCarrierOn, the SystemInfo block shall be transmitted on each carrier, and all contents of the SystemInfo block except the CarrierID, FLReservedInterlaces and NumFLReservedSubbands shall be identical for all carriers. The multi-carrier mode is public data of the physical layer protocol.

6.5.5.4.1.2 Procedure for transmission of the QuickChannelInfo block

The QuickChannelInfo block shall be transmitted in every superframe with an odd superframe index. The QuickChannelInfo block shall be carried by the Control Channel MAC Protocol, over the pBCH1 physical channel, and shall not pass through the Signaling Transport.. The information carried in the QuickChannelInfo block transmitted in superframe m=2k+1 describes the structure of

- All PHY Frames except the first PHY Frame of superframe 2k+1, and
- All PHY Frames of superframe 2k+2
- The first PHY Frame in superframe 2k+3.

The access network should change the contents of the QuickChannelInfo block in accordance with the QuickChannelInfoValidity field of the block.

If the multi-carrier mode is MultiCarrierOn, the QuickChannelInfo block shall be transmitted on each carrier. The multi-carrier mode is public data of the physical layer protocol.

6.5.5.4.1.3 Procedure for transmission of the ExtendedChannelInfo message

The ExtendedChannelInfo message shall be broadcast over the Forward Traffic Channel MAC. The message shall begin transmission in superframes with superframe numbers divisible by $N_{OMPExtendedChannelInfo}$. The ExtendedChannelInfo message may be delivered in one superframe, or in a set of consecutive superframes. If transmission of an ExtendedChannelInfo message begins in superframe n and spans k superframe,

- The ExtendedChannelInfo message should describe the structure of superframes n+k through n+k+ValidityPeriod, where ValidityPeriod is a field of the ExtendedChannelInfo message.
- The structure of superframes n through n+k-1 shall be described by the last ExtendedChannelInfo message transmitted before superframe n.

If the multi-carrier mode is MultiCarrierOn, the ExtendedChannelInfo message shall be transmitted on each carrier. The multi-carrier mode is public data of the physical layer protocol.

6.5.5.4.1.4 Procedure for transmission of the SectorParameters message

The access network should send a SectorParameters message over the Forward Traffic Channel MAC in superframe numbers that are divisible by $N_{OMPSectorParameters}$. The access network shall set the SectorSignature field of the ExtendedChannelInfo message to the SectorSignature field of the next SectorParameters message.

If the multi-carrier mode is MultiCarrierOn, the SectorParameters message shall be transmitted on each carrier. The multi-carrier mode is public data of the physical layer protocol.

6.5.5.4.2 Access terminal requirements

Upon entering the Active State, the access terminal shall invoke the procedure for determining if the OverheadMessages are up-to-date, as specified in 6.5.5.4.2.5, and the procedure for generating TriggerCode based pilot reports, as specified in 6.5.5.4.2.6,. When in the Active State, the access terminal shall perform supervision on the QuickChannelInfo, ExtendedChannelInfo and the SectorParameters messages as specified in 6.5.5.4.3.1, 6.5.5.4.3.2 and 6.5.5.4.3.3, respectively.

If the access terminal receives a *ActiveSetManagement.IdleHO* indication or if it receives a *ConnectedState.ConnectionClosed* indication, the access terminal shall invoke the procedures for determining if the OverheadMessages are up-to-date, as specified in 6.5.5.4.2.5.

When the access terminal receives a ExtendedChannelInfo message from a sector, it shall perform the procedures in 6.5.5.4.2.1.

When the access terminal receives a SectorParameters message from a sector, it shall perform the procedures in 6.5.5.4.2.4.

6.5.5.4.2.1 Procedure for processing SystemInfo block

The access terminal shall place the received SystemInfo block, indexed by PilotPN and CarrierID, in the public data.

6.5.5.4.2.2 Procedure for processing the QuickChannelInfo block and EncapsulatedQuickChannelInfo message

The access terminal shall place the received QuickChannelInfo block, indexed by PilotPN and CarrierID in the public data.

When the access terminal receives a QuickChannelInfo block from a sector, it shall perform the following:

- If the QuickChannelInfo block is received in superframe n, or if a EncapsulatedQuickChannelInfo block is received with the SuperframeNumber field set to n, then at the first end of the first PHY Frame (frame 0) of superframe n, the access terminal shall:
  - If the received QuickChannelInfo block differs from the stored block in the public data in any field except the QuickChannelInfoValidity field, the access terminal shall generate a *QuickChannelInfoUpdated* indication.
  - Store the block, indexed by PilotPN and CarrierID, in the public data
  - If the QuickChannelInfoValidity field is set to m, the access terminal shall set QuickChannelInfoExpiryTime to the end of the first PHY Frame of superframe $2 \times 4^m \lceil n/(2 \cdot 4^m) \rceil + 1$.

6.5.5.4.2.3 Procedure for processing the ExtendedChannelInfo message

When the access terminal receives a ExtendedChannelInfo message from a sector, it shall perform the following:

- The access terminal shall determine the superframe number n when the access network started ExtendedChannelInfo transmission, and the superframe number n+k-1 when the access network ended (or will end) ExtendedChannelInfo message transmission. For example, if transmission of the ExtendedChannelInfo message spans superframes 16 and 17, then n=16 and k=2.
- At the beginning of superframe n+k, the access terminal shall perform the following operations:
  - If the received ExtendedChannelInfo message differs from the ExtendedChannelInfo message in the public data (for the same PilotPN and CarrierID) in any fields except the SystemTime, SectorParametersSignature or ValidityPeriod, the access terminal shall generate an *ExtendedChannelInfoUpdated* indication.
  - Store the ExtendedChannelInfo message, indexed by PilotPN and CarrierID in the public data.
  - Set the ExtendedChannelInfoExpiryTime for the message in the public data to n+ValidityPeriod
- When the access terminal adds a ExtendedChannelInfo message to the public data, it shall process the stored SectorParameters messages according to the following rules:
  - If the public data contains a SectorParameters message with the same PilotPN as the sector that transmitted the ExtendedChannelInfo message, the access terminal shall compare the SectorParametersSignature in the ExtendedChannelInfo message with the SectorParametersSignature in the stored SectorParameters message. If the signatures do not match, the access terminal shall purge the SectorParameters message from the public data

6.5.5.4.2.4 Procedure for processing the SectorParameters message

When the access terminal receives a SectorParameters message, it shall perform the following:

- If the public data contains a SectorParameters message with the same SectorID as the received message, the access terminal shall compare the SectorParametersSignature of the received message with the SectorParametersSignature in the stored SectorParameters message. If the signatures do not match, the access terminal shall:
  - Replace the SectorParameters message in the public data with the received SectorParameters message.
  - If the sector is a member of the Active Set, return a *SectorParametersUpdated* and *OverheadMessagesUpdated* indication.
- If the public data does not contain a SectorParameters message with the SectorID of the received message, the access terminal shall:
  - Add the received SectorParameters message to the public data.
  - If the sector is a member of the Active Set, return a *SectorParametersUpdated* and *OverheadMessagesUpdated* indication.
- If necessary, the access terminal may delete old SectorParameters messages corresponding to sectors not in the Active Set.

6.5.5.4.2.5 Procedure for checking if parameters are up-to-date

When this set of procedures is invoked, the access terminal determines QuickChannelInfoUpToDate, ExtendedChannelInfoUpToDate and SectorParametersUpToDate as follows:

QuickChannelInfoUpToDate shall be set to ‘1’ if all of the following conditions are satisfied for the following members of the Active Set: RLSS, FLSS, DRLSS, DFLSS (as indicated by the public data of the Reverse Control Channel MAC Protocol). This field shall be set to ‘0’ otherwise.

- A QuickChannelInfo block for this member of the Active Set is available in the public data.
- The QuickChannelInfo block that is currently in the public data of the protocol has a validity time that is greater than or equal to the current time.

ExtendedChannelInfoUpToDate shall be set to ‘1’ if all of the following conditions are satisfied for the following members of the Active Set: RLSS, FLSS, DRLSS, DFLSS (as indicated by the public data of the Reverse Control Channel MAC Protocol). This field shall be set to ‘0’ otherwise.

- An ExtendedChannelInfo message for this member of the Active Set is available in the public data.
- The ExtendedChannelInfo message that is currently in the public data of the protocol has a validity time that is greater than or equal to the current time.

SectorParametersUpToDate shall be set to ‘1’ if all of the following conditions are satisfied for all members of the Active Set, and shall be set to ‘0’ otherwise.

- A SectorParameters message with the same PilotPN as this member of the Active Set is available in the public data.
- An ExtendedChannelInfo block for this member of the Active Set is available in the public data.
- The SectorParametersSignature in the last received ExtendedChannelInfo block is the same as the SectorParametersSignature in the stored SectorParameters message.

OverheadParametersUpToDate shall be set to the logical “and” of SectorParametersUpToDate and ExtendedChannelInfoUpToDate. The OverheadParametersUpToDate field is used by the Idle State Protocol.

The access terminal uses tune away procedures of the ConnectedState protocol when QuickChannelInfoUpToDate or ExtendedChannelInfoUpToDate is set to one.

6.5.5.4.2.6 Procedure for ZoneCode-based registration

The access terminal shall store a list of RegistrationZoneCodes associated with subnets visited by the access terminal for future comparisons and for future use. This list is called the RegistrationZoneCodeList. Each entry in the RegistrationZoneCodeList shall include the subnet and the RegistrationZoneCode. Other protocols may cache information keyed by (Subnet, RegistrationZoneCode) pairs. If other protocols cache information keyed by (Subnet, RegistrationZoneCode) pairs, then these protocols shall delete such information when the (Subnet, RegistrationZoneCode) pair is deleted from the RegistrationZoneCodeList.

The access terminal shall be capable of storing exactly $N_{OMPMinZoneSignatureListSize}$ entries in the RegistrationZoneCodeList. If the (Subnet, RegistrationZoneCode) pair from the SectorParameters message from some sector in the Active Set is not included in the RegistrationZoneCodeList, then the access terminal shall add the entry to the RegistrationZoneCodeList. The access terminal shall generate a *ActiveSetManagement.SendPilotReport* command when it adds an entry to the RegistrationZoneCodeList. If there are more entries in the RegistrationZoneCodeList than the supported size of the RegistrationZoneCodeList, the access terminal shall delete the oldest entries first. The access terminal shall delete an entry from the RegistrationZoneCodeList when the entry has stayed in the RegistrationZoneCodeList for 2^(RegistrationZoneMaxAge + 3) × 1.28 seconds.

6.5.5.4.3 Supervision procedures

6.5.5.4.3.1 Supervision of QuickChannelInfo block

The access terminal shall use the following procedure to supervise the QuickChannelInfo block:

- The access terminal shall set a QuickChannelInfo supervision timer for $T_{OMPQCISupervision}$.
- If QuickChannelInfoUpToDate becomes ‘1’ while the timer is active, the access terminal shall disable the timer. If QuickChannelInfoUpToDate becomes ‘0’ while the timer is inactive, the access terminal shall start the timer.
- If the timer expires, the access terminal shall return a *SupervisionFailed* indication and disable the timer.

Delayed reception of the QuickChannelInfo block may also cause a supervision failure at the Lower MAC Sublayer.

6.5.5.4.3.2 Supervision of ExtendedChannelInfo message

The access terminal shall use the following procedure to supervise the ExtendedChannelInfo message:

- The access terminal shall set a ExtendedChannelInfo supervision timer for $T_{OMPECISupervision}$.
- If ExtendedChannelInfoUpToDate becomes '1' while the timer is active, the access terminal shall disable the timer. If ExtendedChannelInfoUpToDate becomes '0' while the timer is inactive, the access terminal shall start the timer.
- If the timer expires, the access terminal shall return a *SupervisionFailed* indication and disable the timer.

6.5.5.4.3.3 Supervision of SectorParameters message

Upon entering the Active State, the access terminal shall start the following procedure to supervise the SectorParameters message:

- The access terminal shall set a SectorParameters supervision timer for $T_{OMPSPSupervision}$.
- If SectorParametersUpToDate becomes '1' while the timer is active, the access terminal shall disable the timer. If SectorParametersUpToDate becomes '0' while the timer is inactive, the access terminal shall start the timer.
- If the timer expires, the access terminal shall return a *SupervisionFailed* indication and disable the timer.

6.5.6 Message and block formats

In the interpretation of these messages, the symbol 'n' is used to denote the value of a bit field. For example, the field CPLength is assigned two bits, and the parameter CPLength takes values N_{FFT}*(1+n)/16, where n is the 2 bit field that takes the value 0, 1, 2 or 3.

6.5.6.1 SystemInfo block

The SystemInfo block shall be transmitted directly by the Control Channel MAC Protocol over the pBCH0 channel, and shall not pass through other sublayers. The SystemInfo block shall have the following format.

Field	Length (bits)
MaximumRevision	4
MinimumRevision	4
CarrierID	2
NumCarriers	2
SystemTimeLSB	12
CPLength	2
NumGuardSubcarriers	3
BlockHoppingEnabled	1
N_FLBurst	2
N_RLBurst	2
FLReservedInterlaces	4
NumFLReservedSubbands	4

MinimumRevision — This field shall be set to the minimum revision number that the sector can support.

MaximumRevision — This field shall be set to the maximum revision number that the sector can support.

CarrierID — This field shall be set to the CarrierID of the carrier this block is transmitted on.

NumCarriers — This field shall determine the number of carriers available at this sector. This parameter shall take the value (n+1).

SystemTimeLSB — This field shall be set to the twelve lower bits of the superframe number at the time the SystemInfo block starts transmission.

CPLength — This field shall determine the cyclic prefix length in units of chips. This parameter shall take the value N_{FFT}*(1+n)/16, where n is equal to the 2 bit field that takes the value 0, 1, 2 or 3.

NumGuardSubcarriers — This parameter shall take the value $N_{GUARD,PR}$ - 32*n. Here $N_{GUARD,PR}$ is a numeric constant of the Physical Layer Protocol.

BlockHoppingEnabled — This field shall be set to ‘1’ if block hopping is enabled. This field shall be set to ‘0’ if symbol rate hopping is enabled.

N_FLBurst This field shall determine the number of forward link PHY Frames that comprise a forward link burst in TDD mode. This parameter shall take the value (n+1). This field shall be ignored in FDD mode.

N_RLBurst This field shall determine the number of reverse link PHY Frames that comprise a reverse link burst in TDD mode. This parameter shall take the value (n+1). This field shall be ignored in FDD mode.

FLReservedInterlaces This field shall determine what interlaces contain reserved bandwidth on the forward link.

Table 44 Interpretation of FLReservedInterlaces

Value	Interpretation: Reserved FL bandwidth on the following interlaces
0000	None
0001	0
0010	0, 1
0011	0, 1, 2
0100	0, 1, 2, 3
0101	0, 1, 2, 3, 4
0110	0, 1, 2, 3, 4, 5
0111	0, 1, 2, 3, 4, 5, 6
1000	0, 1, 2, 3, 4, 5, 6, 7
1001	0, 1, 2, 3, 4, 5, 6, 7, 8
1010	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
1011	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
1100	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
1101	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12
1110	0, 3
1111	0, 6

NumFLReservedSubbands

The interpretation of this field is used by the Physical Layer to govern FL PHY Frame Modulation.

6.5.6.2 QuickChannelInfo block

The QuickChannelInfo block shall be transmitted directly by the Control Channel MAC Protocol over the pBCH1 channel, and shall not pass through other sublayers. The QuickChannelInfo block shall have the following format:

Field	Length (bits)
QuickChannelInfoValidity	3
FLFirstRestrictedSetSubband	$2 + \log_2(N_{CARRIER\_SIZE}/512)$
FLNumRestrictedSetSubbands	2
FLChannelTreeIndex	4
FLSectorHopSeed	4
FLIntraCellCommonHopping	1
FLDPISectorOffset	2
FLDPISectorScramble	1
FLNumSDMADimensions	2
FLNumSubbands	1
FLDiversityHoppingMode	1
NumPilots	1
EffectiveNumAntennas	3
NumCommonPilotTransmitAntennas	1
EnableCommonPilotStaggering	1
EnableAuxPilotStaggering	1
SSCHNumHopports	3
SSCHNumBlocks	3
SSCHModulationSymbolsPerBlock	2

QuickChannelInfoValidity

If this field takes the value m and the current superframe number is n, the access network should keep all fields of the QuickChannelInfo block (except the QuickChannelInfoValidity field) unchanged from superframes m till superframe $2 \cdot 4^m \lceil n/(2 \cdot 4^m) \rceil$.

FLFirstRestrictedSetSubband

This field shall be set to the index of the first restricted subband on the forward link.

FLNumRestrictedSetSubbands

This field shall be set to the number of restricted subbands on the forward link. This field shall be set to 0 if no subbands are restricted. Otherwise, subbands FLFirstRestrictedSetSubband through FLFistRestrictedSetSubband+FLNumRestrictedSetSubbands-1 shall be considered to be restricted subbands, with possible rollover at subband zero.

FLChannelTreeIndex — This field shall be used by the Lower MAC Sublayer.

FLSectorHopSeed — This field shall be used by the PHY Layer to determine the hopping pattern.

FLIntraCellCommonHopping — This field shall be used by the PHY Layer to determine the hopping pattern.

FLDPISectorOffset — This field shall be set to the relative offset of pilots as defined in the F-DPICH section in the Physical Layer.

FLDPISectorScramble — This field shall determine the scrambling of pilots as defined in the sector and cell specific scrambling sections in the Physical Layer.

FLNumSDMADimensions — This field shall determine the number of spatial dimensions on the forward link. This parameter shall take the value n+1.

FLNumSubbands — This field shall determine the number of subbands on the forward link. If n=0, this parameter shall be set to $N_{CARRIER\_SIZE}/128$ and if n=1, this parameter shall be set to $N_{CARRIER\_SIZE}/256$.

FLDiversityHoppingMode — This field shall be used by the Physical Layer to determine the hop pattern for the sector. This field shall be set to '1' if DiversityHoppingMode is On, and to '0' if DiversityHoppingMode is Off.

NumPilots — This field shall determine the nominal number of pilots in F-CPICH as being $N_{CARRIER\_SIZE}/16$ or $N_{CARRIER\_SIZE}/8$, depending on whether the field is set to '0' or '1', respectively.

EffectiveNumAntennas — This field shall determine the effective number of antennas, and this parameter shall take the value n+1. The access network shall set EffectiveNumAntennas to four or below when the BlockHoppingEnabled field of the SystemInfo block is set to '0'.

NumCommonPilotTransmitAntennas — This field shall determine the number of common pilot transmit antennas, and this parameter shall take the value n+1.

EnableCommonPilotStaggering — This field is set to '1' if common pilot staggering is enabled. This field is set to '0' if common pilot staggering is not enabled.

EnableAuxPilotStaggering — This field is set to '1' if auxiliary pilot staggering is enabled. This field is set to '0' if auxiliary pilot staggering is not enabled.

SSCHNumHopports — This field shall determine the number of hop-ports allocated to F-SSCH. This field shall be interpreted as follows:

Table 45 Interpretation of SSCHNumHopports

Value	Interpretation when MultiCarrierOn	Interpretation when MultiCarrierOff
000	48	48 x $N_{CARRIER\_SIZE}$/512
001	64	64 x $N_{CARRIER\_SIZE}$/512
010	80	80 x $N_{CARRIER\_SIZE}$/512
011	96	96 x $N_{CARRIER\_SIZE}$/512
100	128	128 x $N_{CARRIER\_SIZE}$/512
101	160	160 x $N_{CARRIER\_SIZE}$/512
110	208	208 x $N_{CARRIER\_SIZE}$/512
111	256	256 x $N_{CARRIER\_SIZE}$/512

SSCHNumBlocks — This field shall determine the number of blocks carried by the F-SSCH. This parameter shall take the value 2*(n+1).

SSCHModulationSymbolsPerBlock

This field shall determine the number of modulation symbols for each block carried by the F-SSCH.

Table 46 Interpretation of SSCHModulationSymbolsPerBlock

Value	Interpretation
00	45
01	60
10	90
11	180

6.5.6.3 ExtendedChannelInfo

The ExtendedChannelInfo message provides the configuration for the Physical Layer and Lower MAC Sublayer operation. The message consists of several groups, and a group consists of fields, as defined in the following:

Field	Length (bits)
MessageID	8
ValidityPeriod	16
SectorInformation Group	See 6.5.6.3.1
PowerControl Group	See 6.5.6.3.2
AccessParameters Group	See 6.5.6.3.3

MessageID This field shall be set to 0x00.

SectorInformation Group
This field is defined in 6.5.6.3.1.

ValidityPeriod This field determines the time till when the parameters in the ExtendedChannelInfo message are valid . This parameter shall take the value $(1+n)*N_{OMPExtendedChannelInfo}$ superframes.

PowerControl Group This field is defined in 6.5.6.3.2.

AccessParameters Group
This field is defined in 6.5.6.3.3.

Channels	FTC		**SLP**	Best Effort
Addressing	Broadcast	Unicast	**Security**	Optional

6.5.6.3.1 SectorInformation group

The SectorInformation group shall consist of the following fields.

Table 47 SectorInformation group

Field	Length (bits)
PilotPN	12
CarrierID	2
SystemTime	34
SectorParametersSignature	16
RLChannelTreeIndex	4
RLSectorHopSeed	4
RLIntraCellCommonHopping	1
BFCHBeamCodeBookIndex	4
RLDPISectorOffset	2
RLDPISectorScramble	1
RLNumSDMADimensions	2
RLRestrictedSetBitmap	16
RLNumSubbands	1
RLDiversityHoppingMode	1
NumRLControlSubbands	3
RACKBandwidthFactor	2
HalfDuplexModeSupported	1
ReverseLinkSilenceDuration	4
ReverseLinkSilencePeriod	4
TransmitPower	6
CommonPilotPower	4
AuxPilotPower	4
PreamblePilotPower	4

PilotPN — This field shall be set to the PilotPN of the sector this message refers to. The sector this message refers to may be different from the sector transmitting this message.

CarrierID — This field shall be set to the CarrierID field transmitted on the SystemInfo block on this carrier.

SystemTime — This field shall be set to the time at the sector this message refers to at the beginning of the superframe in which the ExtendedChannelInfo block started transmission.

SectorParametersSignature
The access network shall set this field to the SectorParametersSignature of the next SectorParameters message to be transmitted by the access network.

RLChannelTreeIndex This field shall be used by the Lower MAC Sublayer .

RLSectorHopSeed This field shall be used by the PHY Layer to determine the hopping pattern.

RLIntraCellCommonHopping
This field shall be used by the PHY Layer to determine the hopping pattern.

BFCHBeamCodeBookIndex
This field shall refer to the code book index, the code book comprising of transmit weights for SDMA and precoding.

RLDPISectorOffset This field shall be set to the relative offset of pilots as defined in the R-DPICH section in the Physical Layer.

RLDPISectorScramble This field shall be set to the scrambling of pilots as defined in the sector and cell specific scrambling sections of the Physical Layer.

RLNumSDMADimensions
This field shall determine the number of spatial dimensions on the reverse link. This parameter shall take the value n+1.

RLRestrictedSetBitmap
Bit position *j* in this bit field shall be set to 1 if subband j is restricted on the reverse link.

RLNumSubbands This field shall determine the number of subbands on the reverse link. If n=0, this parameter shall take the value $N_{CARRIER\_SIZE}/128$ and if n=1, this parameter shall take the value $N_{CARRIER\_SIZE}/256$.

RLDiversityHoppingMode
This field shall be used by the Physical Layer to determine the hop pattern for the sector. This field shall be set to ‘1’ if DiversityHoppingMode is on, and to ‘0’ if DiversityHoppingMode is off.

NumRLControlSubbands
This field shall determine the number of control subbands on the reverse link and this parameter shall take the value (n+1).

RACKBandwidthFactor
This field shall determine the bandwidth reduction on the R-ACKCH. This parameter shall take the value n+1.

HalfDuplexModeSupported

This field shall be set to '1' if the access network supports half duplex terminals, and shall be set to '0' otherwise. If half-duplex terminals are supported, the access network should assign MAC IDs and channel assignments in a manner that enables half-duplex terminal operation. A half-duplex access terminal is not required to monitor forward link transmissions on a PHY Frame where it is scheduled to make a reverse link transmission.

ReverseLinkSilenceDuration

The access network shall set this field to specify the duration of the Reverse Link Silence Interval. This parameter shall take the value 2^n PHY Frames. In a region with asynchronous sectors, the access network should set this field to a value larger than the timing offset between sectors.

ReverseLinkSilencePeriod

The access network shall set this field to specify the periodicity of occurrence the Reverse Link Silence Interval. This parameter shall take the value

$$\text{ReverseLinkSilencePeriod} = (1+n)*144000$$

The Reverse Link Silence Interval is defined as the time interval of duration ReverseLinkSilenceDuration RL PHY Frames that starts at superframe index m that satisfies the following equation:

$$m \bmod (\text{ReverseLinkSilencePeriod}) = 0$$

TransmitPower — This field shall be set to the transmit power of the sector in units of dBm.

CommonPilotPower — The field shall be determine the power spectral density of the F-CPICH during the FL PHY frame relative to the F-ACQCH. This parameter shall take the value (-4 + n*0.5) dB.

AuxPilotPower — The field shall determine the power spectral density of the F-AuxPICH relative to the F-ACQCH. This parameter shall take the value (-4 + n*0.5) dB.

PreamblePilotPower — The field shall determine the power spectral density of the F-CPICH during the superframe preamble relative to the F-ACQCH. This parameter shall take the value (-4 + n*0.5) dB.

6.5.6.3.2 PowerControl group

Field	Length (bits)
MACIDRange	3
FLPCReportInterval	4
RLCtrlPCMode	1
FastOSIEnabled	1
CtrlAccessOffset	3
BFCHPowerOffset	4
SFCHPowerOffset	4
PICHPowerOffset	4
REQChannelGain0	4
REQChannelGain1	4
REQChannelGain2	4
REQChannelGain3	4
ErasureGain0	4
ErasureGain1	4
ErasureGain2	4
ErasureGain3	4

MACIDRange — This field shall be set to indicate the range of assigned MACID values in the sector. For example, a MACIDRange of 63 indicates that the sector has not assigned MACID values 64 and above. The field shall be interpreted as follows.

Table 48 Interpretation of MACIDRange

Value	Interpretation
000	63
001	127
010	255
011	511
100	1023
101	2047
110 to 111	Reserved

FLPCReportInterval — This field shall determine the periodicity at which power control commands are sent to the access terminal. This parameter shall take the value (n+1) PHY Frames.

RLCtrlPCMode — This field shall determine the closed loop power control mode of the sector, with values '0' corresponding to 'ErasureBased' and '1' corresponding to 'UpDown'.

FastOSIEnabled — This field shall be set to '1' if the F-SSCH transmitted by this sector contains a Fast OSI Segment. This field shall be set to '0' if the F-SSCH transmitted by this sector does not contain a Fast OSI Segment..

CtrlAccessOffset — This field determines the initial gain of the R-CQICH over the R-ACH. The value of this parameter shall be in units of dB in 2's complement notation.

BFCHPowerOffset — This field shall determine the power offset of the R-BFCH relative to the R-CQICH. This parameter shall be in units of dB in 2's complement notation.

SFCHPowerOffset — This field shall determine the power offset of the R-SFCH relative to the R-CQICH. This parameter shall be in units of dB in 2's complement notation.

PICHPowerOffset — This field is determines the gain of the R-PICH over the R-CQICH. This parameter shall be in units of dB in 2's complement notation.

REQChannelGain*j* — This field determines the gain of the R-REQCH over the R-CQICH. This parameter shall be in units of dB in 2's complement notation.

ErasureGain*j* — This field determines the transmit power of erasure sequences for different assignment sizes, and this parameter shall take the value n-4 dB.

6.5.6.3.3 AccessParameters group

Field	Length (bits)
AccessCycleDuration	2
AccessSequencePartition	5
MaxProbesPerSequence	4
ProbeRampUpStepSize	4
RDCHInitialPacketFormat	6
PilotThreshold1	3
PilotThreshold2	3
OpenLoopAdjust	8
$N_{ACMPClass}$ values of the following field	
AccessRetryPersistance	3

AccessCycleDuration — This field shall be determine the duration of the access cycle in units of Control Segment Periods (as defined by the Physical Layer). The AccessCycleDuration parameter shall be set according to the value of the field as follows.

Table 49 Interpretation of AccessCycleDuration

Value	Interpretation in units of Control Segment Periods
00	1
01	2
10	3
11	4

AccessSequencePartition
This field shall indicate the partition of the access sequence space to allow the access terminal to signal pilot power and buffer status information with the access sequence. The interpretation of this field is in the Access Channel MAC Protocol.

MaxProbesPerSequence
This field shall determine the maximum number of probe sequences that can be part of one access sequence. This parameter shall take the value n+1.

ProbeRampUpStepSize This field shall determine the power ramp up used for probes within a probe sequence. This parameter shall take the value 0.5*(1+n) dB.

RDCHInitialPacketFormat
This field shall be set to the packet format that is used on the first transmission the access terminal makes on the R-DCH after getting an access grant.

PilotThreshold1 This field shall determine PilotThreshold1 used by the Access Channel MAC Protocol. This parameter shall take the value -2n dB.

PilotThreshold2 This field shall determine PilotThreshold2 used by the Access Channel MAC Protocol. This parameter shall take the value -2n dB.

AccessRetryPersistance
This field shall determine the persistence probability for determining access sequence backoff. If this field is set to n, the access terminal shall use 2^{-n} as the retry persistence.

OpenLoopAdjust This field shall determine the nominal power to be used by access terminal in the open loop power estimate. The value of this field shall be 70+n dB.

6.5.6.4 SectorParameters

The SectorParameters message is used to convey sector specific information to the access terminals.

Field	Length (bits)
MessageID	8
CountryCode	12
SectorID	128
SubnetMask	8
SectorParametersSignature	16
ChannelBand	ChannelBandRecord Type Dependent
Latitude	22
Longitude	23
RegistrationRadius	11
LeapSeconds	8
LocalTimeOffset	11
RegistrationZoneCodeIncluded	1
RegistrationZoneCode	12
RegistrationZoneMaxAge	4
GloballySynchronous	1
SynchronousWithNextPilot	1
ChannelBandCount	5
ChannelBandCount occurrences of the following record{	
ChannelBand	ChannelBandRecord Type Dependent
NeighborCount	5
NeighborCount occurrences of the following record{	
NeighborPilotPN	12
NeighborCarrierID	2
TransmitPower	6
GloballySynchronous	1
SynchronousWithNextPilot	1
}}	
NumOtherTechnologies	4
NumOtherTechnologies occurrences of the following fields	
TechnologyType	8
TechnologyNeighborListLength	8
TechnologyNeighborList	TechnologyNeighbor ListLength x 8
Reserved	Variable

MessageID	The access network shall set this field to 0x01.
CountryCode	The access network shall set this field to the three-digit BCD (binary coded decimal) encoded representation of the Mobile Country Code (as specified in [6]) associated with this sector.
SectorID	Sector Address Identifier. The access network shall set the value of the SectorID according to the rules specified in 10.11. The access terminal shall not assume anything about the format of the SectorID. The SectorID shall uniquely identify a sector.
SubnetMask	Sector Subnet identifier. The access network shall set this field to the number of consecutive 1's in the subnet mask of the subnet to which this sector belongs.
SectorParametersSignature	SectorParameters message signature. The access network shall change this field if the contents of the SectorParameters message changes.
ChannelBand	ChannelBand record specification for each channel. See 10.1 for the ChannelBand record format.
Latitude	The latitude of the sector. The access network shall set this field to this sector's latitude in units of 0.25 second, expressed as a two's complement signed number with positive numbers signifying North latitudes. The access network shall set this field to a value in the range -1296000 to 1296000 inclusive (corresponding to a range of -90° to +90°).
Longitude	The longitude of the sector. The access network shall set this field to this sector's longitude in units of 0.25 second, expressed as a two's complement signed number with positive numbers signifying East longitude. The access network shall set this field to a value in the range -2592000 to 2592000 inclusive (corresponding to a range of -180° to +180°).
RegistrationRadius	If access terminals are to perform distance based registration, the access network shall set this field to the non-zero "distance" beyond which the access terminal is to send a new PilotReport message (see Default Active Set Management Protocol, 6.6). If access terminals are not to perform distance based registration, the access network shall set this field to 0.
LeapSeconds	The number of leap seconds that have occurred since the start of system time.
LocalTimeOffset	The access network shall set this field to the offset of the local time from System Time. This value will be in units of minutes, expressed as a two's complement signed number.
RegistrationZoneIncluded	The access network shall set this field to '1' if the RegistrationZoneCode and RegistrationZoneMaxAge are included in the message.

RegistrationZoneCode — The access network shall include this field if RegistrationZoneIncluded is set to ‘1’.

RegistrationZoneMaxAge — The access network shall include this field if RegistrationZoneIncluded is set to ‘1’.

SynchronousSystem — The access network shall set this field to ‘1’ if all sectors in the deployment are synchronous. The access network shall set this field to ‘0’ otherwise.

ChannelBandCount — The access network shall set this field to the number of neighbor channels available to the access terminal on this sector.

ChannelBand — ChannelBand record specification for each channel. See 10.1 for the ChannelBand record format. If the optional parameters of this ChannelBand record are different from the parameters for the sector transmitting this message, the access network shall include the optional parameters in the ChannelBand record.

NeighborCount — The access network shall set this field to the number of records specifying neighboring sectors information included in this message for this channel.

NeighborPilotPN — The access network shall set this field to the PilotPN of a neighboring sector that the access terminal should add to its Neighbor Set.

NeighborCarrierID — The access network shall set this field to the CarrierID of the neighboring pilot.

TransmitPower — This field shall be set to the transmit power of the sector in units of dBm.

GloballySynchronous — The access network shall set this field according to the following rule: If any two sectors have the GloballySynchronous field set to '1', then the two sectors are synchronous with each other.

SynchronousWithNextPilot — The access network shall set this field to ‘1’ if this sector is synchronous with the next sector listed in the message. The access network shall set this field to ‘0’ if this sector is the last sector listed in the message, or if this sector is not synchronous with the next sector listed in the message. Rules for determining if two sectors are synchronous are given in the synchronization and timing section of the physical layer, 9.4.2.

NumOtherTechnologies — This field shall be set to the number of other technologies included in the message. Other technology neighbors are included in this message to assist the access terminal in inter-technology handoff.

TechnologyType — This field shall be set to the type of technology, and shall be interpreted as defined in Table 107.

TechnologyNeighborListLength
This field shall be set the length, in bytes, of the neighbor list information for the other technology.

TechnologyNeighborList
This field shall be set to the neighbor list information for the other technology. The interpretation of this field is beyond the scope of this specification.

Reserved — The number of bits in this field is equal to the number needed to make the message length an integer number of octets. The access network shall set this field to zero. The access terminal shall ignore this field.

Channels	FTC
Addressing	Broadcast Unicast

SLP	Best Effort
Security	Optional

6.5.6.5 EncapsulatedQuickChannelInfo

This message may be used to deliver the QuickChannelInfo block over the Forward Traffic Channel MAC. This message may also be used to deliver the QuickChannelInfo block of one sector over the Forward Traffic Channel of another sector.

Field	Length (bits)
MessageID	8
SuperframeNumber	34
PilotPN	12
SectorIDIncluded	1
SectorID	0 or 128
CarrierID	2
QuickChannelInfoLength	8
QuickChannelInfoBlock	Variable
Reserved	Variable

MessageID — This field shall be set to 0x02.

SuperframeNumber — This field shall be set to the superframe number at the sector this message refers to, when that sector transmitted the QuickChannelBlock included in this message over the Control Channel MAC.

PilotPN — This field shall be set to the PilotPN of the sector this message refers to. The sector this message refers to may be different from the sector transmitting this message.

SectorIDIncluded	This field shall be set to '1' if the SectorID of the sector this message refers to is included in the message. The access network shall set this field to '1' if the PilotPN is not sufficient to uniquely identify the sector this message refers to.
SectorID	This field shall be included if SectorIDIncluded is equal to '1', and shall be set to the SectorID of the sector this message refers to.
CarrierID	This field shall be set to the CarrierID field transmitted on the SystemInfo block on this carrier.
QuickChannelInfoLength	This field shall be set to the length, in bits, of the following QuickChannelInfoBlock.
QuickChannelInfoBlock	This field shall be set to the QuickChannelBlock.
Reserved	The number of bits in this field is equal to the number needed to make the message length an integer number of octets. The access network shall set this field to zero. The access terminal shall ignore this field.

Channels	FTC	**SLP**	Best Effort
Addressing	Broadcast Unicast	**Security**	Optional

6.5.7 Interface to other protocols

6.5.7.1 Commands

This protocol sends no commands.

6.5.7.2 Indications

This protocol registers to receive the following indications:

- *ActiveSetManagement.IdleHO*
- *ConnectedState.ConnectionClosed*

6.5.8 Configuration attributes

No configuration attributes are defined for this protocol.

6.5.9 Protocol numeric constants

Constant	Meaning	Value
$N_{OMPType}$	Type field for this protocol	Table 9
$N_{OMPDefault}$	Subtype field for this protocol	0x0000
$T_{OMPQCISupervision}$	QuickChannelInfo supervision timer	0.5 s
$T_{OMPECISupervision}$	ExtendedChannelInfo supervision timer	1 s
$T_{OMPSPSupervision}$	SectorParameters supervision timer	4 s
$N_{OMPExtendedChannelInfo}$	The recommended time between two consecutive ExtendedChannelInfo message transmissions	16 superframes
$N_{OMPSectorParameters}$	The recommended time between two consecutive SectorParameters message transmissions	64 superframes
$N_{OMPMinZoneSignatureListSize}$	Minimum number of entries supported by the access terminal in the RegistrationZoneCodeList	8

6.5.10 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

6.6 Default Active Set Management Protocol

6.6.1 Overview

The Default Active Set Management Protocol provides the procedures and messages used by the access terminal and the access network to keep track of the access terminal's approximate location and to maintain the radio link as the access terminal moves between the coverage areas of different sectors.

This protocol can be in one of three states:

- *Inactive State*: In this state, the protocol waits for an *Activate* command.
- *Idle State*: This state corresponds to the Air Link Management Protocol Idle State. In this state, the access terminal autonomously maintains the Active Set. PilotReport messages from the access terminal to the access network are generated based on the distance between the access terminal's current serving sector and the serving sector at the time the access terminal last sent an update. The generation of such PilotReport messages causes transition from the Idle State to the Connected State.
- *Connected State*: In this state, the access network dictates the access terminal's Active Set. PilotReport messages from the access terminal to the access network are based on changing radio link conditions.

Transitions between states are driven by commands received from other Lower MAC Control sublayer protocols and the transmission and reception of the ActiveSetAssignment message.

The protocol states, messages and commands causing the transition between the states are shown in Figure 55.

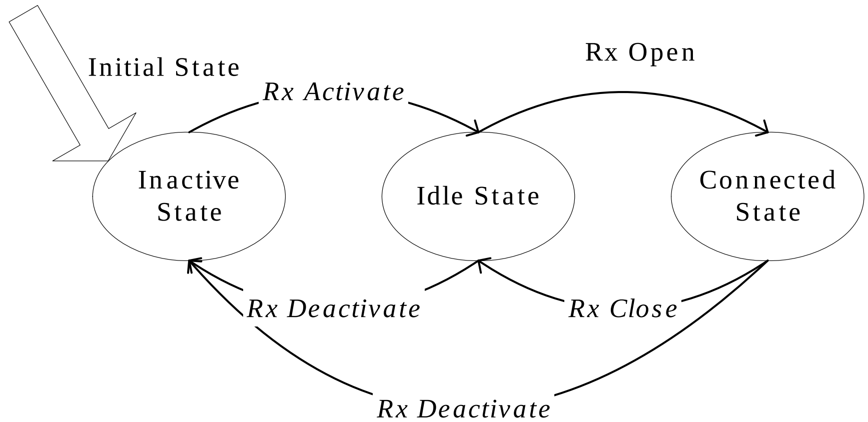

Figure 55 Default Active Set Management Protocol state diagram

This protocol uses parameters that are provided as public data by the Overhead Messages Protocol, configured attributes, or protocol constants.

Table 50 lists all of the protocol parameters obtained from the public data of the Overhead Messages Protocol.

Table 50 Active Set Management Protocol parameters that are public data of the Overhead Messages Protocol

Parameter	Comment
Latitude	Latitude of sector in units of 0.25 second.
Longitude	Longitude of sector in units of 0.25 second.
RegistrationRadius	Distance between the serving sector and the sector in which location was last reported that triggers a new report. If this field is set to zero, then distance triggered reporting is disabled.
NumNeighbors	Number of neighbors specified in the message.
NeighborPN	PilotPN of each neighbor.
NeighborChannelBandIncluded	Set to '1' if a ChannelBand record is included for the neighbor.
NeighborChannelBand	Neighbor ChannelBand record specifying network type and frequency.

6.6.2 Primitives

6.6.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*
- *Open*
- *Close*
- *SendPilotReport*

6.6.2.2 Return indications

This protocol returns the following indications:

- *ConnectionLost* (access network only)
- *NetworkLost*
- *IdleHO*
- *ActiveSetUpdated*
- *AssignmentRejected*

6.6.3 Public data

6.6.3.1 Static public data

This protocol defines the following static public data:

- Active Set
- Current ActiveSetAssignment message
- PilotIncrement specified in the PilotSearch Configuration Attribute
- A list of measured PilotStrengths, and RxPowers indexed by PilotPN
- RegistrationRadiusFlag

6.6.3.2 Dynamic public data

- Subtype for this protocol

6.6.4 Protocol initialization and swap procedures

6.6.4.1 Protocol initialization

Upon initialization at the access terminal and the access network

- The value for each attribute for this protocol instance shall be set to the default value for that attribute.
- The protocol shall enter the inactive state.

6.6.4.2 Protocol swap

- The protocol shall enter the Inactive State.

6.6.5 Procedures

6.6.5.1 Command processing

6.6.5.1.1 Activate

If the protocol receives an *Activate* command in the Inactive State, the access network shall transition to the Idle State.

If the protocol at the access terminal receives an *Activate* command in the Inactive State, the protocol shall transition to the Idle State, and generate a PilotReport message.

If this command is received in any other state, it shall be ignored.

6.6.5.1.2 Deactivate

If the protocol receives a *Deactivate* command in the Inactive State, it shall be ignored.

If the protocol receives this command in any other state, the access terminal and the access network shall transition to the Inactive State.

6.6.5.1.3 Open

If the protocol receives an *Open* command in the Idle State,

- The access terminal shall transition to the Connected State.
- The access network shall transition to the Connected State.

If this command is received in any other state, it shall be ignored.

6.6.5.1.4 Close

If the protocol receives a *Close* command in the Connected State, the access terminal and the access network shall transition to the Idle State.

If this command is received in any other state, it shall be ignored.

6.6.5.2 Processing the RegistrationRadiusUpdated indications

The RegistrationRadiusFlag determines the Idle State registration procedure at the access terminal. When the flag value is 1 registration is performed when the access terminal travels a distance more than RegistrationRadius distance. When flag value is 0 is registration is performed when the access terminal moves to a sector with a different latitude and longitude. The value of the flag at the access terminal is set based on ConnectionOpenResponse and ConnectionClose messages received by the access terminal.

Upon receiving a *IdleState.RegistrationRadiusUpdated* indication or a *ConnectedState.RegistrationRadiusUpdated* indication, the Active Set Management Protocol at the access terminal shall set the RegistrationRadiusFlag public data to the RegistrationRadiusFlag value provided with the indication.

6.6.5.3 Pilots and pilot sets

The access terminal estimates the strength of the Forward Channel transmitted by each sector in its neighborhood. This estimate is based on measuring the strength of the Forward Pilot Channel (specified by the pilot's PilotPN and the pilot's ChannelBand), henceforth referred to as the pilot.

When this protocol is in the Connected State, the access terminal uses pilot strengths to decide when to generate PilotReport messages.

When this protocol is in the Idle State, the access terminal uses pilot strengths to decide which sector's Control Channel it monitors.

The following pilot sets are defined to support the Active Set Management process:[30]

- *Active Set*: The set of pilots associated with the sectors currently serving the access terminal. When a connection is open, a sector is considered to be serving an access terminal when there is a MAC ID assigned to the access terminal in that sector. When a connection is not open, a sector is considered to be serving the access terminal when the access terminal is monitoring that sector's control channel for page reception and distance based registration. Parameters for Active Set member sectors in connected state are sent as part of the ActiveSetAssignment message, and placed in the public data of the Active Set Management Protocol.
- *Candidate Set*: The pilots that are not in the Active Set, but are received by the access terminal with sufficient strength to indicate that the sectors transmitting them are good candidates for inclusion in the Active Set.
- *Neighbor Set*: The set of pilots that are not in either one of the two previous sets, but are likely candidates for inclusion in the Active Set.
- *Remaining Set*: The set of all possible pilots on the current channel assignment, excluding the pilots that are in any of the three previous sets.

At any given instant a pilot in the current ChannelBand is a member of exactly one set.

The access terminal maintains all four sets. The access network maintains only the Active Set.

The access terminal complies with the following rules when searching for pilots, estimating the strength of a given pilot, and moving pilots between sets.

[30] In this context, a pilot identifies a sector.

6.6.5.3.1 Pilot search

The access terminal shall continually search for pilots in the Connected State and whenever it is monitoring any channel in the Idle State. The access terminal shall search for pilots in all pilot sets.

The access terminal should use the same search priority for pilots in the Active Set and Candidate Set. In descending order of search rate, the access terminal shall search, most often, the pilots in the Active Set and Candidate Set, then shall search the pilots in the Neighbor Set, and lastly shall search the pilots in the Remaining Set.

6.6.5.3.2 Pilot strength measurement

The access terminal shall measure the strength of every pilot it searches. The strength estimate formed by the access terminal shall be computed as the ratio of received pilot energy per chip, E_c, to total received spectral density, I_0 (signal and noise). The access terminal shall place the measured ratios in its public data.

In addition, the access terminal shall also measure the received power (RxPower in dBm) from each acquisition pilot at the antenna input and place the result in the public data. If the access terminal has more than one receive antenna, the power shall be averaged across all antennas. If a pilot is not being measured, the RxPower shall be assumed to be negative infinity. The access terminal should update the measured power every superframe.

6.6.5.3.3 Pilot drop timer maintenance

For each pilot, the access terminal shall maintain a pilot drop timer.

If DynamicThresholds is equal to ‘0’, the access terminal shall perform the following:

- The access terminal shall start a pilot drop timer for each pilot in the Candidate Set or the Active Set whenever the strength becomes less than the value specified by PilotDrop. The access terminal shall consider the timer to be expired after the time specified by PilotDropTimer.
- The access terminal shall reset and disable the timer whenever the strength of the pilot becomes greater than the value specified by PilotDrop.

If DynamicThresholds is equal to ‘1’, the access terminal shall perform the following:

- The access terminal shall start a pilot drop timer for each pilot in the Candidate Set whenever the strength of the pilot becomes less than the value specified by PilotDrop The access terminal shall consider the timer value to be expired after the time specified by PilotDropTimer. The access terminal shall reset and disable the timer if the strength of the pilot becomes greater than the value specified by PilotDrop.

- For each pilot in the Active Set, the access terminal shall sort pilots in the Active Set in order of increasing strengths, i.e., $PS_1 < PS_2 < PS_3 < ... < PS_{N_A}$, where N_A is the number of the pilots in the Active Set. The access terminal shall start the timer whenever the strength PS_i satisfies the following inequality:

$$10 \times \log_{10} PS_i < \max\left(\frac{SoftSlope}{8} \times 10 \times \log_{10} PS_{N_A} + \frac{DropIntercept}{2}, -\frac{PilotDrop}{2}\right)$$
$$i = 1, 2, ..., N_A - 1$$

The access terminal shall reset and disable the timer whenever the above inequality is not satisfied for the corresponding pilot.

Sections 6.6.5.3.5 and 6.6.5.7.3 specify the actions the access terminal takes when the pilot drop timer expires.

6.6.5.3.4 Active set management

The access terminal shall support a maximum Active Set size of $N_{ASMPActive}$ pilots.

Rules for maintaining the Active Set are specific to each protocol state (see 6.6.5.6.1 and 6.6.5.7.1).

6.6.5.3.5 Candidate set management

The access terminal shall support a maximum Candidate Set size of $N_{ASMPCandidate}$ pilots.

The access terminal shall add a pilot to the Candidate Set if one of the following conditions is met:

- Pilot is not already in the Active Set or Candidate Set and the strength of the pilot exceeds the value specified by PilotAdd.
- Pilot is deleted from the Active Set, its pilot drop timer has expired, DynamicThresholds is equal to '1', and the pilot strength is above the threshold specified by PilotDrop.
- Pilot is deleted from the Active Set but its pilot drop timer has not expired.

The access terminal shall delete a pilot from the Candidate Set if one of the following conditions is met:

- Pilot is added to the Active Set.
- Pilot's drop timer has expired.
- Pilot is added to the Candidate Set; and, as a consequence, the size of the Candidate Set exceeds $N_{ASMPCandidate}$. In this case, the access terminal shall delete the weakest pilot in the set. Pilot A is considered weaker than pilot B:
  - If pilot A has an active drop timer but pilot B does not.
  - If both pilots have an active drop timer and pilot A's drop timer is closer to expiration than pilot B's.
  - If neither of the pilots has an active drop timer and pilot A's strength is less than pilot B's.

6.6.5.3.6 Neighbor set management

The access terminal shall support a minimum Neighbor Set size of $N_{ASMPNeighbor}$ pilots.

- The access terminal shall maintain a counter, AGE, for each pilot in the Neighbor Set as follows.

The access terminal shall perform the following in the order specified:

- If a pilot is added to the Active Set or Candidate Set, it shall be deleted from the Neighbor Set.
- If a pilot is deleted from the Active Set, but not added to the Candidate Set, then it shall be added to the Neighbor Set with the AGE of 0.
- If a pilot is deleted from the Candidate Set, but not added to the Active Set, then it shall be added to the Neighbor Set with the AGE of 0.
- If the size of the Neighbor Set is greater than the maximum Neighbor Set supported by the access terminal, the access terminal shall delete enough pilots from the Neighbor Set such that the size of the Neighbor Set is the maximum size supported by the access terminal. The pilots with higher AGE are deleted first[31].
- If the access terminal receives an *OverheadMessages.SectorParametersUpdated* indication, then:
  - The access terminal shall increment the AGE for every pilot in the Neighbor Set.
  - For each pilot in the neighbor list given as public data by the Overhead Messages Protocol that is a member of the Neighbor Set, the access terminal shall perform the following:
    - The access terminal shall set the AGE of this neighbor list pilot to the minimum of its current AGE and NeighborMaxAge.
  - For each pilot in the neighbor list given as public data by the Overhead Messages Protocol (in the order specified in the neighbor list) that is a member of the Remaining Set, the access terminal shall perform the following:
    - If the addition of this neighbor list pilot to the Neighbor Set would not cause the size of the Neighbor Set size to increase beyond the maximum Neighbor Set size supported by the access terminal, then the access terminal shall add this neighbor list pilot to the Neighbor Set with its AGE set to NeighborMaxAge.
    - If the addition of this neighbor list pilot would cause the size of the Neighbor Set to increase beyond the maximum Neighbor Set size supported by the access terminal and the Neighbor Set contains at least one pilot with AGE greater than NeighborMaxAge associated with the pilot's channel, then the access terminal shall delete the pilot in the Neighbor Set for which the difference between its AGE and the NeighborMaxAge associated with that pilot's channel (i.e., AGE - NeighborMaxAge) is the greatest and shall add this neighbor list pilot to the Neighbor Set with its AGE set to NeighborMaxAge associated with the pilot's channel.

[31] The order in which pilots of the same AGE are deleted does not matter in this case.

– If the addition of this neighbor list pilot would cause the size of the Neighbor Set to increase beyond the maximum Neighbor Set size supported by the access terminal and the Neighbor Set does not contain a pilot with AGE greater than NeighborMaxAge associated with the pilot's channel, the access terminal shall not add this neighbor list pilot to the Neighbor Set.

6.6.5.3.7 Remaining set management

The access terminal shall initialize the Remaining Set to contain all of the pilots whose PN offset index is an integer multiple of PilotIncrement and are not already members of any other set. In Connected State, the remaining set shall consist of pilots on at least all channels assigned in the ActiveSetAssignment message.

The access terminal shall add a pilot to the Remaining Set if it deletes the pilot from the Neighbor Set and if the pilot was not added to the Active Set or Candidate Set.

The access terminal shall delete the pilot from the Remaining Set if it adds it to another set.

6.6.5.4 Message sequence numbers

The access network shall validate all received PilotReport messages as specified in 6.6.5.4.1.

The access terminal shall validate all received ActiveSetAssignment messages as specified in 6.6.5.4.2.

The PilotReport message and the ActiveSetAssignment message carry a MessageSequence field that serves to flag duplicate or stale messages.

The MessageSequence field of the PilotReport message is independent of the MessageSequence field of the ActiveSetAssignment message.

6.6.5.4.1 PilotReport message validation

When the access terminal first sends a PilotReport message, it shall set the MessageSequence field of the message to zero. Subsequently, the access terminal shall increment this field each time it sends a PilotReport message.

The access network shall consider all PilotReport messages it receives in the Idle State as valid.

The access network shall initialize the receive pointer, *V(R)*, to the MessageSequence field of the first PilotReport message it received in the Idle State, and the access network shall subsequently set it to the MessageSequence field of each received PilotReport message.

When the access network receives a PilotReport message in the Connected State, it shall validate the message using the procedure defined in 10.7. The access network shall discard the message if it is invalid.

6.6.5.4.2 ActiveSetAssignment message validation

The access network shall set the MessageSequence field of the ActiveSetAssignment message it sends in the Idle State to zero. Subsequently, each time the access network sends a new ActiveSetAssignment message in the Connected State, it shall increment this field. If the access network is sending the same message multiple times, it shall not change the value of this field between transmissions.[32]

The access terminal shall initialize the receive pointer, *V(R)*, to the MessageSequence field of the first ActiveSetAssignment message that it receives in the Connected State.

When the access terminal receives an ActiveSetAssignment message in the Connected State, it shall validate the message using the procedure defined in 10.7. The access terminal shall discard the message if it is invalid.

6.6.5.5 Inactive state

Upon entering this state, the access terminal shall perform the following:

- The access terminal shall set the Active Set, the Candidate Set, and the Neighbor Set to NULL.
- The access terminal shall initialize the Remaining Set to contain all of the pilots whose PN offset index is an integer multiple of PilotIncrement .
- The access terminal shall set (xL,yL), the longitude and latitude of the sector in whose coverage area the access terminal last sent a PilotReport message, to (NULL, NULL).

6.6.5.6 Idle state

In this state, PilotReport messages from the access terminal are based on the distance between the sector where the access terminal last sent a PilotReport message and the sector currently in its active set, or on the time elapsed since the last PilotReport message was sent.

6.6.5.6.1 Active set maintenance

The access network shall not initially maintain an Active Set for the access terminal in this state.

The access terminal shall initially keep an Active Set of size one when it is in the Idle State. The Active Set pilot shall be the pilot associated with the Control Channel that the access terminal is currently monitoring. The access terminal shall return an *IdleHO* indication when the Active Set changes in the Idle State.

The access terminal shall not change its Active Set pilot at a time that causes it to miss a QuickPage packet. Other rules governing when to replace this Active Set pilot are beyond the scope of this specification.

If the access terminal receives an ActiveSetAssignment message, it shall set its Active Set to the list of pilots specified in the message.

---

[32] The access network may send a message multiple times to increase its delivery probability.

6.6.5.6.2 Pilot channel supervision in the idle state

The access terminal shall perform pilot channel supervision in the Idle State as follows:

- The access terminal shall monitor the pilot strength of the pilot in its active set, all of the pilots in the candidate set, and all of the pilots in the neighbor set that are on the same frequency.
- If the strength of all of the pilots that the access terminal is monitoring goes below the value specified by PilotDrop, the access terminal shall start a pilot supervision timer. The access terminal shall consider the timer to be expired after the time specified by PilotDropTimer.
- If the strength of at least one of the pilots goes above the value specified by PilotDrop while the pilot supervision timer is counting down, the access terminal shall reset and disable the timer.
- If the pilot supervision timer expires, the access terminal shall return a *NetworkLost* indication.

6.6.5.6.3 Processing access related indications in the Idle State

The following operation shall be performed at the access terminal if the *SharedSignalingMAC.AccessGrantReceived* Indication is received, and at the access network, if the *AccessChannelMAC.AccessProbeReceived* indication is received.

The Active Set Management Protocol at the access terminal shall construct the following ActiveSetAssignment block and place it in the public data. The fields in the block shall be decided based on the public data of the Overhead Messages Protocol, the received access grant, or the InitialSetupConfigurationAttribute.

The Active Set Management Protocol at the access network shall construct the following ActiveSetAssignment block and place it in the public data. The fields in the block shall be decided based on the public data of the Overhead Messages Protocol, the transmitted access grant, or the default values for the InitialSetupConfigurationAttribute.

Field	Length (bits)	Value
ChannelBandIncluded	1	1
ChannelBand	ChannelBandRecord Type Dependent	ChannelBand of the current Active Set Member
MaxPHYSubPacketSize	1	0
SingleServingSector	1	InitialSetupAttribute
RPICHEnabled	1	InitialSetupAttribute
NumRLControlSubbandsUser	4	Min(4,NumRLControlSubbands)[33]
RLControlSubbandUserOffset	4	0
EnhancedPilotReportEnabled	1	InitialSetupAttribute
EnhancedPilotReportRatio	4	InitialSetupAttribute
EnhancedPilotReportThreshold	4	InitialSetupAttribute
MinRequestInterval	2	InitialSetupAttribute
CQIReportInterval	2	InitialSetupAttribute
CQIReportPhase	3	InitialSetupAttribute
CQIPilotnterval	2	InitialSetupAttribute
CQIPilotPhase	3	InitialSetupAttribute
BFCHReportRate	4	InitialSetupAttribute
SFCHReportRate	4	InitialSetupAttribute
BFCHPowerOffset	3	InitialSetupAttribute
NumBFCHBits	2	InitialSetupAttribute
SFCHPowerOffset	3	InitialSetupAttribute
NumSFCHBits	2	InitialSetupAttribute
MandatoryCQICHCTRLReportingPeriod	3	InitialSetupAttribute
NumPilots	4	1
NumPilots occurrences of the following record:		
PilotPN	12	Pilot PN of sector that sent Access Grant
ActiveSetIndex	3	0

[33] NumRLControlSubbands is public data of the Overhead Messages Protocol for the sector and carrier the access grant was received from.

Field	Length (bits)	Value
SynchronousWithNextPilot	1	0
MAC ID	11	From AccessGrant in public data of SS MAC
AccessSequenceIDIncluded	1	0
AccessSequenceID	0	N/A
PowerControlStepUp	3	InitialSetupAttribute
PowerControlStepDown	3	InitialSetupAttribute
RDCHGainMin	6	InitialSetupAttribute
RDCHGainMax	2	InitialSetupAttribute

6.6.5.6.4 Pilot report rules

The access terminal shall send PilotReport messages to update its location with the access network.

The access terminal shall not send a PilotReport message if the connection timer is active.

The access terminal shall comply with the following rules regarding PilotReport messages:

- The access terminal shall send a PilotReport message upon receiving a *SendPilotReport* command.
- The access terminal shall send a PilotReport message whenever it receives a PilotReportRequest message.
- The access terminal shall include in the PilotReport message the pilot PN, pilot strength, and drop timer status for every pilot in the Active Set and Candidate Set.
- The access terminal shall send a PilotReport message if the IdleStateRegistrationTimeOut attribute is nonzero, and the last PilotReport message was sent more than IdleStateRegistrationTimeOut time ago.
- If the RegistrationRadiusFlag is set to ‘0’, the access terminal shall send a PilotReport message if the sector that is currently providing coverage to the access terminal has a latitude and longitude that is different from the latitude and longitude of the sector where the access terminal last sent a PilotReport message.
- If the RegistrationRadiusFlag is set to ‘1’, the access terminal shall send a PilotReport message if the computed value r is greater than the value provided in the RegistrationRadius field of the SectorParameters message transmitted by the sector in which the access terminal last sent a PilotReport message.

If (x_L, y_L) are the longitude and latitude of the sector in whose coverage area the access terminal last sent a PilotReport message, and (x_C, y_C) are the longitude and latitude of the sector currently providing coverage to the access terminal, then r is given by[34]

$$r = \left\lfloor \frac{\sqrt{\left[(x_C - x_L) \times \cos\left(\frac{\pi}{180} \times \frac{y_L}{14400}\right)\right]^2 + \left[y_C - y_L\right]^2}}{16} \right\rfloor$$

The access terminal shall compute r with an error of no more than ±5% of its true value when $|y_L/14400|$ is less than 60 and with an error of no more than ±7% of its true value when $|y_L/14400|$ is between 60 and 70.[35]

6.6.5.7 Connected state

In this state, PilotReport messages from the access terminal are based on changes in the radio link between the access terminal and the access network, obtained through pilot strength measurements at the access terminal.

The access network determines the contents of the Active Set through ActiveSetAssignment messages.

6.6.5.7.1 Active set maintenance

6.6.5.7.1.1 Access network

Whenever the access network sends an ActiveSetAssignment message to the access terminal, it shall add to the Active Set any pilots listed in the message that are not currently in the Active Set. The access network shall place the most recently transmitted ActiveSetAssignment message in the public data of the Active Set Management Protocol.

The access network shall delete a pilot from the Active Set if the pilot was not listed in a ActiveSetAssignment message and if the access network received the ActiveSetComplete message, acknowledging that ActiveSetAssignment message.

The access network should send an ActiveSetAssignment message to the access terminal in response to changing radio link conditions, as reported in the access terminal's PilotReport messages.

The access network should only specify a pilot in the ActiveSetAssignment message if it has allocated the required resources in the associated sector. This means that the sector specified by the pilot is ready to receive data from the access terminal and is ready to transmit queued data to the access terminal should the access terminal directs a handoff request to that sector.

---

[34] The *x's* denote longitude and the *y's* denote latitude.

[35] x_L and y_L are given in units of 1/4 seconds. $x_L/14400$ and $y_L/14400$ are in units of degrees.

If the access network adds or deletes a pilot in the Active Set, it shall send an *ActiveSetUpdated* indication.

6.6.5.7.1.2 Access terminal

If the access terminal receives a valid ActiveSetAssignment message (see 6.6.5.4.2), it shall replace the contents of its current Active Set with the pilots specified in the message. The access terminal shall process the message as defined in 6.6.5.7.5.

6.6.5.7.2 ResetReport message

The access network may send a ResetReport message to reset the conditions under which PilotReport messages are sent from the access terminal. Access terminal usage of the ResetReport message is specified in the following section.

6.6.5.7.3 Pilot strength report rules

The access terminal sends a PilotReport message to the access network in this state to request addition or deletion of pilots from its Active Set. If DynamicThresholds is equal to ‘0’, the access terminal shall include in the PilotReport message the pilot PN, pilot strength, and drop timer status for every pilot in the Active Set and Candidate Set. If DynamicThresholds is equal to ‘1’, then the access terminal shall include in the PilotReport message the pilot PN, pilot strength, and drop timer status for every pilot in the Active Set, for each pilot in the Candidate Set whose strength is above the values specified by PilotAdd, and for each pilot in the Candidate set whose strength, PS, satisfies the following inequality:

$$10 \times \log_{10} PS > \frac{SoftSlope}{8} \times 10 \times \log_{10} PS_{N_A} + \frac{AddIntercept}{2}$$

The access terminal shall send a PilotReport message if any one of the following occurs:

- The access terminal receives a PilotReportRequest message. The access terminal shall set the DetailedInfoIncluded field of the PilotReport message to 1 if it receives a PilotRequest message with the ReportFormat set to 0x01.
- The Default Active Set Management Protocol receives a *SendPilotReport* command.
- If DynamicThresholds is equal to ‘0’ and the strength of a Neighbor Set or Remaining Set pilot is greater than the value specified by PilotAdd.
- If DynamicThresholds is equal to ‘1’ and the strength of a Neighbor Set or Remaining Set pilot, PS, satisfies the following inequality:

$$10 \times \log_{10} PS > \max\left(\frac{SoftSlope}{8} \times 10 \times \log_{10} PS_{N_A} + \frac{AddIntercept}{2}, -\frac{PilotAdd}{2}\right)$$

- If DynamicThresholds is equal to ‘0’ and the strength of a Candidate Set pilot is greater than the value specified by PilotCompare above an Active Set pilot, and a PilotReport message carrying this information has not been sent since the last ResetReport message was received.
- If DynamicThresholds is equal to ‘0’ and the strength of a Candidate Set pilot is above PilotAdd, and a PilotReport message carrying this information has not been sent since the last ResetReport message was received.

- If DynamicThresholds is equal to '1' and
  - The strength of a Candidate Set pilot, PS, satisfies the following inequality:

$$10 \times \log_{10} PS > \frac{SoftSlope}{8} \times 10 \times \log_{10} PS_{N_A} + \frac{AddIntercept}{2}$$

  - A PilotReport message carrying this information has not been sent since the last ResetReport message was received.
- If DynamicThresholds is equal to '1' and
  - The strength of a Candidate Set pilot is greater than the value specified by PilotCompare above an Active Set pilot, and
  - The strength of a Candidate Set pilot, PS, satisfies the following inequality:

$$10 \times \log_{10} PS > \frac{SoftSlope}{8} \times 10 \times \log_{10} PS_{N_A} + \frac{AddIntercept}{2}$$

  - A PilotReport message carrying this information has not been sent since the last ResetReport message was received.
- The pilot drop timer of an Active Set pilot has expired, and a PilotReport message carrying this information has not been sent since the last ResetReport message was received.
- If the Active Set Size is greater than one, and
  - EnhancedPilotReportEnabled=1, and
  - The strength of a pilot in the active set has changed (increased or decreased) by more than EnhancedPilotReportThreshold since the last PilotReport was sent.
- If the Active Set Size is one, and
  - EnhancedPilotReportEnabled=1, and
  - The strongest current non-active pilot (for example, pilot j) has strength more than the EnhancedPilotReportRatio fraction of the total interference, and
  - The strength of pilot j differs by more than EnhancedPilotReportThreshold from the strength of the second strongest pilot at the time the last pilot report was sent (regardless of whether the second strongest pilot at that time was pilot j or some other pilot).

6.6.5.7.4 VCQI report rules

The access terminal shall send a VCQI report every VCQIReportInterval superframes, where VCQIReportInterval shall be part of the ActiveSetAssignment message. If the VCQIReportInterval is set to zero, the access terminal shall not send any VCQI reports. The VCQI report shall be a VCQIReportSISO if the CQIReportingMode of the ActiveSetAssignment message is set to CQISISO. The VCQI report shall be a VCQIReportMIMO if the CQIReportingMode of the ActiveSetAssignment message is set to CQISCW or CQIMCW.

6.6.5.7.5 Processing the ActiveSetAssignment message in the connected state

The access terminal shall process a valid ActiveSetAssignment message (see 6.6.5.4.2) as follows:

- The access terminal shall return an *ActiveSetUpdated* indication.
- The access terminal shall update its Active Set as defined in 6.6.5.7.1.2.
- The access terminal shall tune to the frequency defined by the ChannelBand record, if this record is included in the message. If more than one ChannelBand record is specified in the message, the access terminal may autonomously tune to one of the frequencies defined by the ChannelBand.
- The Active Set Management Protocol at the access terminal shall inform the physical layer about the synchronous status of sectors in the ActiveSetAssignment message according to the following rules. Two sectors shall be said to be synchronous both sectors are listed adjacent to each other in the ActiveSetAssignment message, and the sector listed first has the SynchronousWithNextSector bit set to '1'.

  In all other cases, the Active Set Management Protocol shall inform the physical layer that the two sectors are not synchronous.
- The access terminal shall place the ActiveSetAssignment message in the public data. If the ActiveSetAssignment message has a non-zero value for the AdditionalFieldsStatus field for some pilot, the access terminal shall fill the additional fields (as defined in the ActiveSetAssignment message) according to rules followed by the access network for setting the AdditionalFieldsStatus field (as defined in the ActiveSetAssignment message).
- The access terminal shall send the access network an ActiveSetComplete message specifying the MessageSequence value received in the ActiveSetAssignment message.

6.6.5.7.6 Processing the ActiveSetComplete message

The access network should set a transaction timer when it sends an ActiveSetAssignment message. If the access network sets a transaction timer, it shall reset the timer when it receives an ActiveSetComplete message containing a MessageSequence field equal to the one sent in the ActiveSetAssignment message.

If the timer expires, the access network should return an *ActiveSetManagement.ConnectionLost* indication.

6.6.6 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes.

6.6.6.1 PilotReport

The access terminal sends the PilotReport message to notify the access network of its current location and provide it with an estimate of its surrounding radio link conditions.

Field	Length (bits)
MessageID	8
MessageSequence	8
ReferencePilotPN	12
ReferencePilotStrength	8
ReferenceKeep	1
NumPilots	5
NumPilots occurrences of the following record:	
PilotPN	12
ChannelBandIncluded	1
ChannelBand	0 or ChannelBandRecordType Dependent
PilotStrength	8
Keep	1
ExtendedChannelInfoAvailable	1
DetailedInfoIncluded	1
FineTimingOffset	0 or 16
SuperframeOffset	0 or 16
SectorID	0 or 128
Reserved	Variable

MessageID — The access terminal shall set this field to 0x00.

MessageSequence — The access terminal shall set this field to the sequence number of this message. The sequence number of this message is 1 more than the sequence number of the last PilotReport message (modulo 256) sent by this access terminal. If this is the first PilotReport message sent by the access terminal, it shall set this field to 0x00.

ReferencePilotPN — The access terminal shall set this field to the access terminal's time reference (the reference pilot), relative to the zero offset pilot PN sequence in units of 64 PN chips.

ReferencePilotStrength — The access terminal shall set this field to $\lfloor -2 \times 10 \times \log_{10}PS \rfloor$, where PS is the strength of the reference pilot, measured as specified in 6.6.5.3.2. If this value is less than 0, the access terminal shall set this field to '00000000'. If this value is greater than '11111111', the access terminal shall set this field to '11111111'.

ReferenceKeep — If the pilot drop timer corresponding to the reference pilot has expired, the access terminal shall set this field to '0'; otherwise, the access terminal shall set this field to '1'.

NumPilots — The access terminal shall set this field to the number of pilots that follow this field in the message.

PilotPN — The PN offset in resolution of 1 chip of a pilot in the Active Set or Candidate Set of the access terminal that is not the reference pilot.

ChannelBandIncluded — The access terminal shall set this field to '1' if the channel for this pilot offset is not the same as the current channel. Otherwise, the access terminal shall set this field to '0'.

ChannelBand — The access terminal shall include this field if the ChannelBandIncluded field is set to '1'. The access terminal shall set this to the ChannelBand record corresponding to this pilot (see 10.1). Otherwise, the access terminal shall omit this field for this pilot offset.

PilotStrength — The access terminal shall set this field to $\lfloor -2 \times 10 \times \log_{10}PS \rfloor$, where PS is the strength of the pilot (PilotStrength) in the above field, measured as specified in 6.6.5.3.2. If this value is less than 0, the access terminal shall set this field to '00000000'. If this value is greater than '11111111', the access terminal shall set this field to '11111111'.

Keep — If the pilot drop timer corresponding to the pilot in the above field has expired, the access terminal shall set this field to '0'; otherwise, the access terminal shall set this field to '1'.

ExtendedChannelInfoAvailable — If the access terminal has a ExtendedChannelInfo block for the sector, and the ExtendedChannelInfo block is valid, as defined by the Overhead Messages Protocol, the access terminal shall set this field to '1'. Otherwise the access terminal shall set this field to '0'.

DetailedInfoIncluded — The setting for this field is described in the rules for transmission of a PilotReport message.

FineTimingOffset — This field shall be included if DetailedInfoIncluded is set to '1'. If the beginning of a superframe m from the pilot being reported is during superframe n of the reference pilot, this field shall be the time between the beginning of superframes n and m, in units of microseconds. The all ones value of this field is reserved, and indicates that timing information is not available.

SuperframeOffset — This field shall be included if DetailedInfoIncluded is set to ‘1’. This field shall be set to the value 2^{15} + m-n, where m and n are described in the discussion of the FineTimingOffset. The all ones value of this field is reserved, and indicates that timing information is not available.

SectorID — This field shall be set to the SectorID of the sector that transmits the pilot being reported.

Reserved — The number of bits in this field is equal to the number needed to make the message length an integer number of octets. This field shall be set to all zeros.

Channels	RTC	**SLP**	Reliable	Best Effort
Addressing	Unicast	**Security**	Required	

6.6.6.2 VCQIReportSISO

The access terminal shall send this message if the CQIReportingMode is set to CQISISO.

Field	Length (bits)
MessageID	8
NumPilots	3
NumPilots Occurrences of the following record{	
ActiveSetIndex	3
NumInterlaces	4
NumInterlaces instances of the following record{	
InterlaceID	4
NumSubbands	4
NumSubbands occurrences of the following record{	
SubbandID	4
VCQI	4
}}}	
Reserved	Variable

MessageID — The access terminal shall set this field to 0x01.

NumPilots — The access terminal shall set this field to the number of sectors for which the report is being sent.

ActiveSetIndex — The access terminal shall set this field to the ActiveSetIndex corresponding to the sector for which VCQI is being reported.

NumInterlaces — The access terminal shall set this field to the number of interlaces in the system.

InterlaceID	The access terminal shall set this field to the interlace number corresponding to the following record.
NumSubbands	The number of subbands in the system.
SubbandID	This field shall be set to the subband corresponding to the following VCQI.
VCQI	This field shall be set to the VCQIValueSISO for this subband and interlace. VCQIValueSISO is defined in the CQICH Physical Layer Channel Procedures for the Reverse Control Channel MAC Protocol.
Reserved	The number of bits in this field is equal to the number needed to make the message length an integer number of octets. This field shall be set to all zeros.

Channels	RTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

6.6.6.3 VCQIReportMIMO

The access terminal shall send this message if CQIReportingMode is set to CQISCW or CQIMCW.

Field	Length (bits)
MessageID	8
NumPilots	3
NumPilots Occurrences of the following record{	
ActiveSetIndex	3
NumInterlaces	4
NumInterlaces instances of the following record{	
InterlaceID	4
NumSubbands	4
NumSubbands occurrences of the following record{	
SubbandID	4
NumEffectiveAntennas	2
NumEffectiveAntennas occurrences of the following field{	
VCQI	4
}}}}	
Reserved	Variable

MessageID	The access terminal shall set this field to 0x02.
NumPilots	The access terminal shall set this field to the number of sectors for which the report is being sent.

ActiveSetIndex	The access terminal shall set this field to the ActiveSetIndex corresponding to the sector for which VCQI is being reported.
NumInterlaces	The access terminal shall set this field to the number of interlaces in the system.
InterlaceID	The access terminal shall set this field to the interlace number corresponding to the following record.
NumSubbands	The number of subbands in the system.
SubbandID	This field shall be set to the subband corresponding to the following VCQI.
NumEffectiveAntennas	This field shall be set as determined to the value in the QuickChannelInfo block for this sector, stored in the public data of the Overhead Parameters Protocol.
VCQI	If CQIReportingMode is set to CQISCW, this field shall be set to the VCQIValueSCW for this interlace, subband and rank. If CQIReportingMode is set to CQIMCW, this field shall be set to the VCQIValueMCW for this interlace, subband and layer. VCQIValueSCW and VCQIValueMCW are defined in the CQICH Physical Layer Channel Procedures for the Reverse Control Channel MAC Protocol.
Reserved	The number of bits in this field is equal to the number needed to make the message length an integer number of octets. This field shall be set to all zeros.

Channels	RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

6.6.6.4 ActiveSetAssignment

The access network sends the ActiveSetAssignment message to manage the access terminal's Active Set.

Field	Length (bits)
MessageID	8
MessageSequence	8
NumChannelBands	3
NumChannelBands instances of the following record{	
ChannelBandIncluded	1
ChannelBand	0 or ChannelBandRecordType Dependent

Field	Length (bits)
MaxPHYSubPacketSize	1
SingleServingSector	1
RPICHEnabled	1
NumRLControlSubbandsUser	4
RLControlSubbandUserOffset	4
VCQIMeasureInterval	2
EnhancedPilotReportEnabled	1
EnhancedPilotReportRatio	4
EnhancedPilotReportThreshold	4
MinRequestInterval	2
CQIReportInterval	2
CQIReportPhase	2
CQIPilotnterval	2
CQIPilotPhase	2
BFCHReportRate	4
SFCHReportRate	4
NumBFCHBits	2
NumSFCHBits	2
MandatoryCQICHCTRLReportingPeriod	3
NumPilots	3
NumPilots occurrences of the following record{	
PilotPN	12
ActiveSetIndex	3
SynchronousWithNextPilot	1
AdditionalFieldsStatus	2
The following eight fields shall be included if AdditionalFieldsStatus is equal to '00'{	
MAC ID	11
AccessSequenceIDIncluded	1
AccessSequenceID	0 or 4
PowerControlStepUp	3
PowerControlStepDown	3
RDCHGainMin	6
RDCHGainMax	4
}}}	
Reserved	Variable

1 MessageID The access network shall set this field to 0x03.

MessageSequence	The access network shall set this field to 1 higher than the MessageSequence field of the last ActiveSetAssignment message (modulo 2^S, S=8) sent to this access terminal.
NumChannelBands	The access network shall set this field to the number of ChannelBands included in the message. For each channel, the access network shall use the NumPilots field to determine the number of pilots included in the Active Set for that channel.
ChannelBandIncluded	The access network shall set this field to '1' if the ChannelBand record is included for these pilots. Otherwise, the access network shall set this field to '0'.
ChannelBand	If the ChannelBandIncluded field is set to '0' the access network shall omit this field. The access network shall include this field if the ChannelBandIncluded field is set to '1'. The access network shall set this to the ChannelBand record corresponding to the following pilots (see 10.1). If the optional parameters of this ChannelBand record are different from the parameters for the sector transmitting this message, the access network shall include the optional parameters in the ChannelBand record. ..
MaxPHYSubPacketSize	This field shall determine the maximum subpacket size used by the Physical Layer. This parameter shall take the value 4096 if this field is set to '0' and shall take the value 8192 if this field is set to '1'. If the access network changes the value of this parameter at the access terminal, the access network should use a new set of MACIDs in this message.
SingleServingSector	The access network shall set this field to '1' if the access terminal is required to maintain the same forward and reverse link serving sectors. The access network shall set this field to '0' otherwise.
RPICHEnabled	This field shall determine whether the R-PICH is transmitted.
NumRLControlSubbandsUser	This field shall determine the number of subbands over which the access terminal transmits the control segment waveform. This parameter shall take the value n+1. The access network shall set this parameter to be the same for sectors in a synchronous subset.
RLControlSubbandUserOffset	This field shall determine the offset used to generate the set of subbands over which the access terminal transmits the control segment waveform. This is an integer between 0 and 15. The access network shall set this parameter to be the same for sectors in a synchronous subset.
CQIReportingMode	This field shall specify the configuration of MIMO CQI reports sent by the access terminal.

Table 51 Interpretation of CQIReportingMode

CQIReportingMode	Interpretation
00	SISO CQI Report
01	CQISCW CQI Report
10	CQISCW CQI Report
11	Reserved

VCQIReportInterval The access network shall set this field to the interval at which the access terminal shall send a VCQIReport message, in units of 0.25 seconds.

VCQIMeasureInterval The access network shall set this field to control VCQI reports. This parameter shall take the value (2*n+1) superframes.

VCQIMeasureThreshold
The access network shall set this field to control VCQI reports.

MinVCQIReportInterval
The access network shall set this field to control VCQI reports.

MaxVCQIReportInterval
The access network shall set this field to control VCQI reports.

EnhancedPilotReportEnabled
The access network shall use this field to determine if enhanced pilot reports shall be triggered.

EnhancedPilotReportRatio
The access network shall set this field to control pilot reports for the case when EnhancedPilotReportEnabled is set to ‘1’. This parameter shall take the value n*0.5 dB.

EnhancedPilotReportThreshold
The access network shall set this field to control pilot reports for the case when EnhancedPilotReportEnabled is set to ‘1’, in units of 1 dB.

MinRequestInterval The access network shall set this field to specify the minimum number of PHY Frames between two request transmissions by the access terminal. For example, if the field is set to zero, the access terminal may send a request transmission in PHY Frames that are one control period segment apart.

CQIReportInterval This field shall determine the periodicity at which an access terminal shall report a CQI value. For example, a value of 4 shall mean that every fourth control segment period shall contain a CQI report.

Table 52 Interpretation of CQIReportInterval

CQIReportInterval	Periodicity of CQI report in number of control segment periods
00	1
01	2
10	3
11	4

CQIReportPhase — This field shall determine the phase of the CQI reports that are transmitted with periodicity CQIReportInterval. The access terminal shall transmit CQI report in control segment period number CQIReportPhase in superframe 0. For example, if CQIReportInterval is 2 and CQIReportPhase is 1, the access terminal shall transmit CQI in control segments number 1, 3, 5… starting at superframe 0. If CQIReportInterval is 2 and CQIReportPhase is 0, the access terminal shall transmit CQI in control segments number 0, 2, 4… starting at superframe 0.

CQIPilotInterval — This field shall determine the periodicity at which an access terminal shall report a default CQICHPilot. For example, a value of 3 shall mean that every third CQI report shall be set to a CQIPilot report.

Table 53 Interpretation of CQIPilotInterval

CQIPilotInterval	Periodicity of default CQI report in number of CQIReportIntervals
00	2
01	4
10	6
11	8

CQIPilotPhase — This field shall determine the phase of the CQICHPilot reports that are transmitted with periodicity CQIPilotInterval. The access terminal shall transmit CQICHPilot report in CQI report number CQICHPilotPhase in superframe 0.

BFCHReportRate — This field shall be set to the number of R-BFCH reports per super-frame.

SFCHReportRate — This field shall be set to the number of R-SFCH reports per super-frame.

NumBFCHBits — This field shall determine the number and type of bits sent on the R-BFCH. If this field is set to ‘11’ the access terminal shall send all bits. If this field is set to ‘10’, the access terminal sets the fields corresponding to the CQI value to 0. The values ‘00’ and ‘01’ are reserved.

NumSFCHBits — This field shall determine the number and type of bits sent on the R-SFCH. If this field is set to ‘11’ the access terminal shall send all bits. If this field is set to ‘10’, the access terminal sets the fields corresponding to the CQI value to 0. The values ‘00’ and ‘01’ are reserved.

MandatoryCQICHCTRLReportingPeriod — This field shall determine the reporting period in multiples of CQIReportInterval, where the AT is mandated to transmit R-CQICHCTRL report in the R-CQICH channel. The value ‘000’ is reserved.

NumPilots — The access network shall set this field to the number of pilots included in this message.

PilotPN — The access network shall set this field to the PilotPN of a pilot included in the message.

ActiveSetIndex — The access network shall set this field to enable CQI reporting to different sectors in the active set.

SynchronousWithNextPilot — The access network shall set this field to ‘1’ if this sector is synchronous with the next sector listed in the message. The access network shall set this field to ‘0’ if this sector is the last sector listed in the message, or if this sector is not synchronous with the next sector listed in the message. Rules for determining if two sectors are synchronous are given in the synchronization and timing section of the physical layer (9.3.3).

AdditionalFieldsStatus — This field shall describe the status of the additional fields described in the message definition. AdditionalFieldsStatus shall be set to ‘00’ if the additional fields are included in the message. AdditionalFieldsStatus shall be set to ‘01’ if the additional fields are not included in the message, and the values of the additional fields for this sector are identical to the values in the last transmitted ActiveSetAssignment message. AdditionalFieldsStatus shall be set to ‘01’ if the additional fields are not included in the message, and values of the additional fields for this sector are identical to the values of the additional fields for the first sector listed in the message. The value ‘11’ shall be reserved.

MAC ID — The access network shall set this field to the MAC ID assigned to the user in this sector.

AccessSequenceIDIncluded — The access network shall set this field to ‘1’ if an AccessSequenceID is included with the message.

AccessSequenceID — This field shall be set to the access sequence that the access terminal shall use to acquire reverse link timing for this sector. This field shall take values from zero to eight.

PowerControlStepUp — This field shall determine the power increase at the access terminal when it receives a power up command from the access network. This parameter shall take the value (n+1)*0.25 dB.

PowerControlStepDown — This field shall be determine the power decrease at the access terminal when it receives a power down command from the access network. This parameter shall take the value (n+1)*0.25 dB.

RDCHGainMin — This field shall determine the lower limit of the delta value at the access terminal. This parameter shall take the value (0.25*n – 4) dB.

RDCHGainMax — This field shall determine the upper limit of the delta value at the access terminal. This parameter shall take the value (RDCHGainMin + n) dB.

Reserved — The number of bits in this field is equal to the number needed to make the message length an integer number of octets. This field shall be set to all zeros. The access terminal shall ignore this field.

Channels	FTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

6.6.6.5 ActiveSetComplete

The access terminal sends the ActiveSetComplete message to provide an acknowledgment for the ActiveSetAssignment message.

Field	Length (bits)
MessageID	8
MessageSequence	8

MessageID — The access terminal shall set this field to 0x04.

MessageSequence — The access terminal shall set this field to the MessageSequence field of the ActiveSetAssignment message whose receipt this message is acknowledging.

Channels	RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

6.6.6.6 ResetReport

The access network sends the ResetReport message to reset the PilotReport transmission rules at the access terminal.

Field	Length (bits)
MessageID	8

MessageID The access network shall set this field to 0x05.

Channels	FTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

6.6.6.7 PilotReportRequest

The access network sends a PilotReportRequest message to request the access terminal to send a PilotReport message.

Field	Length (bits)
MessageID	8
ReportFormat	8

MessageID The access network shall set this field to 0x06.

ReportFormat The access network shall set this field to indicate the format of the PilotReport it is requesting from the access terminal. The valid values for this field are 0x00 and 0x01, where 0x01 indicates that a detailed pilot report is requested.

Channels	FTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

6.6.7 Interface to other protocols

6.6.7.1 Commands

This protocol does not send any commands.

6.6.7.2 Indications

This protocol registers to receive the following indications:

- *OverheadMessages.SectorParametersUpdated*
- *SharedSignalingMAC.AccessGrantReceived*

- *ConnectedState.RegistrationRadiusUpdated*
- *IdleState.RegistrationRadiusUpdated*

6.6.8 Configuration attributes

The following complex attributes and default values are defined (see 10.3 for attribute record definition). The access terminal should not initiate modification of the following attributes.

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Active Set Management Protocol.

6.6.8.1 SearchParameters attribute

Field	Length (bits)	Default Value
Length	8	N/A
AttributeID	8	N/A
PilotIncrement	4	4

Length — Length of the complex attribute in octets. The access network shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The access network shall set this field to 0x00.

PilotIncrement — The access network shall set this field to the pilot PN sequence increment, in units of PN chips, that access terminals are to use for searching the Remaining Set. The access network should set this field to the largest increment such that the pilot PN sequence offsets of all its neighbor access networks are integer multiples of that increment. The access terminal shall support all of the valid values for this field.

6.6.8.2 SetManagementSameChannelBandParameters attribute

The access terminal shall use these attributes if the pilot being compared is on the same channel as the active set pilots' channel.

Field	Length (bits)	Default Value
Length	8	N/A
AttributeID	8	N/A
PilotAdd	6	0x0e
PilotCompare	6	0x05
PilotDrop	6	0x12
PilotDropTimer	4	3
DynamicThresholds	1	0
SoftSlope	0 or 6	N/A
AddIntercept	0 or 6	N/A
DropIntercept	0 or 6	N/A
NeighborMaxAge	4	0
Reserved	variable	N/A

Length — Length of the complex attribute in octets. The access network shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The access network shall set this field to 0x01.

PilotAdd — This value is used by the access terminal to trigger a PilotReport in the Connected State. The access network shall set this field to the pilot detection threshold, expressed as an unsigned binary number equal to $\lfloor - 2 \times 10 \times \log 10\ Ec/I_0 \rfloor$. The value used by the access terminal is –0.5 dB times the value of this field. The access terminal shall support all of the valid values specified by this field.

PilotDrop — This value is used by the access terminal to start a pilot drop timer for a pilot in the Active Set or the Candidate Set. The access network shall set this field to the pilot drop threshold, expressed as an unsigned binary number equal to $\lfloor - 2 \times 10 \times \log 10\ Ec/I_0 \rfloor$. The value used by the access terminal is –0.5 dB times the value of this field. The access terminal shall support all of the valid values specified by this field.

PilotCompare — Active Set versus Candidate Set comparison threshold, expressed as a 2's complement number. The access terminal transmits a PilotReport message when the strength of a pilot in the Candidate Set exceeds that of a pilot in the Active Set by this margin. The access network shall set this field to the threshold Candidate Set pilot to Active Set pilot ratio, in units of 0.5 dB. The access terminal shall support all of the valid values specified by this field.

PilotDropTimer — Timer value after which an action is taken by the access terminal for a pilot that is a member of the Active Set or Candidate Set, and whose strength has not become greater than the value specified by PilotDrop. If the pilot is a member of the Active Set, a PilotReport message is sent in the Connected State. If the pilot is a member of the Candidate Set, it will be moved to the Neighbor Set. The access network shall set this field to the drop timer value shown in Table 54 corresponding to the pilot drop timer value to be used by access terminals. The access terminal shall support all of the valid values specified by this field.

Table 54 Pilot drop timer values

PilotDropTimer	Timer Expiration (seconds)	PilotDropTimer	Timer Expiration (seconds)
0	< 0.1	8	27
1	1	9	39
2	2	10	55
3	4	11	79
4	6	12	112
5	9	13	159
6	13	14	225
7	19	15	319

DynamicThresholds — This field shall be set to ‘1’ if the following three fields are included in this record. Otherwise, this field shall be set to ‘0’.

SoftSlope — This field shall be included only if DynamicThresholds is set to ‘1’. This field shall be set to an unsigned binary number, which is used by the access terminal in the inequality criterion for adding a pilot to the Active Set or dropping a pilot from the Active Set. The access terminal shall support all of the valid values specified by this field.

AddIntercept — This field shall be included only if DynamicThresholds is set to ‘1’. This field shall be set to a 2’s complement signed binary number in units of dB. The access terminal shall support all of the valid values specified by this field.

DropIntercept — This field shall be included only if DynamicThresholds is set to ‘1’. This field shall be set to a 2’s complement signed binary number in units of dB. The access terminal shall support all of the valid values specified by this field.

NeighborMaxAge — The access network shall set this field to the maximum AGE value beyond which the access terminal is to drop members from the Neighbor Set. The access terminal shall support all of the valid values specified by this field.

Reserved — The access network shall set this field to zero. The access terminal shall ignore this field. The length of this field shall be such that the entire record is octet-aligned.

6.6.8.3 SetManagementDifferentChannelBandParameters attribute

The access terminal shall use these attributes if the pilot being compared is on a channel that is different from the active set pilots' channel.

Field	Length (bits)	Default Value
Length	8	N/A
AttributeID	8	N/A
PilotAdd	6	0x0e
PilotCompare	6	0x05
PilotDrop	6	0x12
PilotDropTimer	4	3
DynamicThresholds	1	0
SoftSlope	0 or 6	N/A
AddIntercept	0 or 6	N/A
DropIntercept	0 or 6	N/A
NeighborMaxAge	4	0
Reserved	variable	N/A

Length — Length of the complex attribute in octets. The access network shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — The access network shall set this field to 0x02.

PilotAdd — This value is used by the access terminal to trigger a PilotReport in the Connected State. The access network shall set this field to the pilot detection threshold, expressed as an unsigned binary number equal to $\lfloor - 2 \times 10 \times \log10 Ec/I_0 \rfloor$. The value used by the access terminal is –0.5 dB times the value of this field. The access terminal shall support all of the valid values specified by this field.

PilotDrop — This value is used by the access terminal to start a pilot drop timer for a pilot in the Active Set or the Candidate Set. The access network shall set this field to the pilot drop threshold, expressed as an unsigned binary number equal to $\lfloor - 2 \times 10 \times \log10 Ec/I_0 \rfloor$. The value used by the access terminal is –0.5 dB times the value of this field. The access terminal shall support all of the valid values specified by this field.

PilotCompare	Active Set versus Candidate Set comparison threshold, expressed as a 2's complement number. The access terminal transmits a PilotReport message when the strength of a pilot in the Candidate Set exceeds that of a pilot in the Active Set by this margin. The access network shall set this field to the threshold Candidate Set pilot to Active Set pilot ratio, in units of 0.5 dB. The access terminal shall support all of the valid values specified by this field.
PilotDropTimer	Timer value after which an action is taken by the access terminal for a pilot that is a member of the Active Set or Candidate Set, and whose strength has not become greater than the value specified by PilotDrop. If the pilot is a member of the Active Set, a PilotReport message is sent in the Connected State. If the pilot is a member of the Candidate Set, it will be moved to the Neighbor Set. The access network shall set this field to the drop timer value shown in Table 54 corresponding to the pilot drop timer value to be used by access terminals. The access terminal shall support all of the valid values specified by this field.
DynamicThresholds	This field shall be set to '1' if the following three fields are included in this record. Otherwise, this field shall be set to '0'.
SoftSlope	This field shall be included only if DynamicThresholds is set to '1'. This field shall be set to an unsigned binary number, which is used by the access terminal in the inequality criterion for adding a pilot to the Active Set or dropping a pilot from the Active Set. The access terminal shall support all of the valid values specified by this field.
AddIntercept	This field shall be included only if DynamicThresholds is set to '1'. This field shall be set to a 2's complement signed binary number in units of dB. The access terminal shall support all of the valid values specified by this field.
DropIntercept	This field shall be included only if DynamicThresholds is set to '1'. This field shall be set to a 2's complement signed binary number in units of dB. The access terminal shall support all of the valid values specified by this field.
NeighborMaxAge	The access network shall set this field to the maximum AGE value beyond which the access terminal is to drop members from the Neighbor Set. The access terminal shall support all of the valid values specified by this field.
Reserved	The access network shall set this field to zero. The access terminal shall ignore this field. The length of this field shall be such that the entire record is octet-aligned.

6.6.8.4 InitialSetupAttribute

This attribute shall be used to construct an ActiveSetAssignment locally at the access terminal. This ActiveSetAssignment shall be used between the times the protocol receives an *AccessChannelMAC.AccessGrantReceived* indication and an ActiveSetAssignment message. The access terminal should use default values for this attribute and the use of non-default values is not recommended.

Field	Length (bits)	Default
AttributeID	8	N/A
SingleServingSector	1	0
RPICHEnabled	1	0
CQIReportingMode	2	00
VCQIReportInterval	4	0000
VCQIMeasureInterval	2	00
EnhancedPilotReportEnabled	1	0
EnhancedPilotReportRatio	4	0000
EnhancedPilotReportThreshold	4	1111
MinReportInterval	2	01
CQIReportInterval	2	00
CQIReportPhase	3	000
CQIPilotInterval	2	00
CQIPilotPhase	3	000
BFCHReportRate	4	0000
SFCHReportRate	4	0000
BFCHPowerOffset	3	000
NumBFCHBits	2	11
SFCHPowerOffset	3	000
NumSFCHBits	2	11
MandatoryCQICHCTRLReportingPeriod	3	111
PowerControlStepUp	3	111
PowerControlStepDown	3	111
RDCHGainMin	6	001000
RDCHGainMax	4	1000

AttributeID This field shall be set to 0x03

The remaining fields shall be interpreted as in the ActiveSetAssignment message.

6.6.8.5 IdleStateRegistrationTimeOut attribute

The default value of this attribute shall be 0x0000.

Attribute ID	Attribute	Values	Meaning
0x04	IdleStateRegistrationTimeOut	***0x0000***	Timer based registration disabled
		0x0001-0xffff	Timer based registration timeout in units of seconds

6.6.9 Protocol numeric constants

Constant	Meaning	Value
$N_{ASMPType}$	Type field for this protocol	Table 9
$N_{ASMPDefault}$	Subtype field for this protocol	0x0000
$N_{ASMPActive}$	Maximum size of the Active Set	8
$N_{ASMPCandidate}$	Maximum size of the Candidate Set	6
$N_{ASMPNeighbor}$	Minimum size of the Neighbor Set	20

6.6.10 Session state information

The Session State Information record (see 10.10) consists of parameter records.

This protocol defines the following parameter record in addition to the configuration attributes for this protocol.

6.6.10.1 ActiveSetManagement parameter

The following parameter shall be included in the Session State Information record only if the Session State Information is being transferred while the connection is open.

Table 55 The Format of the parameter record for the ActiveSetManagement parameter

Field	Length (bits)
ParameterType	8
Length	8
ASCMessageSequence	8
ASAMessageSequence	8
ASAMessageLength	16
ASAMessage	Variable
PRMessageSequence	8
PRMessageLength	8
PRMessage	Variable
NumPilots	8
NumPilots occurrences of the following fields	
Reserved	4
PilotPN	12
SectorID	128

ParameterType This field shall be set to 0x01 for this parameter record.

Length This field shall be set to the length of this parameter record in units of octets excluding the Length field.

ASCMessageSequence This field shall be set to the MessageSequence field of the last ActiveSetComplete message received by the access network.

ASAMessageSequence

This field shall be set to the MessageSequence field of the last ActiveSetAssignment message that was sent by the source access network.

ASAMessageLength This field shall be set to the length of the last ActiveSetAssignment message that was sent by the source access network.

ASAMessage Last ActiveSetAssignment message that was sent by the source access network.

PRMessageSequence This field shall be set to the MessageSequence field of the last PilotReport message that was received by the source access network.

PRMessageLength This field shall be set to the length of the last PilotReport message that was received by the source access network.

PRMessage	This field shall be set to the last PilotReport message that was received by the source access network.
NumPilots	This field shall be set to the NumPilots field in the last ActiveSetAssignment message that was sent by the source access network.
Reserved	This field shall be set to '0000'.
PilotPN	This field shall be set to the corresponding PilotPN field in the last ActiveSetAssignment message that was sent by the source access network.
SectorID	This field shall be set to the SectorID corresponding to the sector associated with the PilotPN specified above.

6.7 Default Initialization State Protocol

6.7.1 Overview

The Default Initialization State Protocol provides the procedures and messages required for an access terminal to acquire a serving network. This protocol imposes the following two requirements on an access terminal

- The access terminal follows channel band information provided in a Redirect message of the Air Link Management Protocol.
- Prevents the access terminal from attempting to connect to access networks with out of range Revision number (as defined through MaximumRevision and MinimumRevision).

At the access network, this protocol does not define any states.

At the access terminal, this protocol operates in one of the following four states:

- *Inactive State*: In this state the protocol waits for an *Activate* command.
- *Network Determination State*: In this state the access terminal chooses an access network on which to operate.
- *Pilot Acquisition State*: In this state the access terminal acquires a Forward Pilot Channel.
- *Read SystemInfo State*: In this state the access terminal reads the SystemInfo block and determines if the MaximumRevision and MinimumRevision fields in the SystemInfo block can be supported by the access terminal.

Protocol states and events causing transition between states are shown in Figure 56.

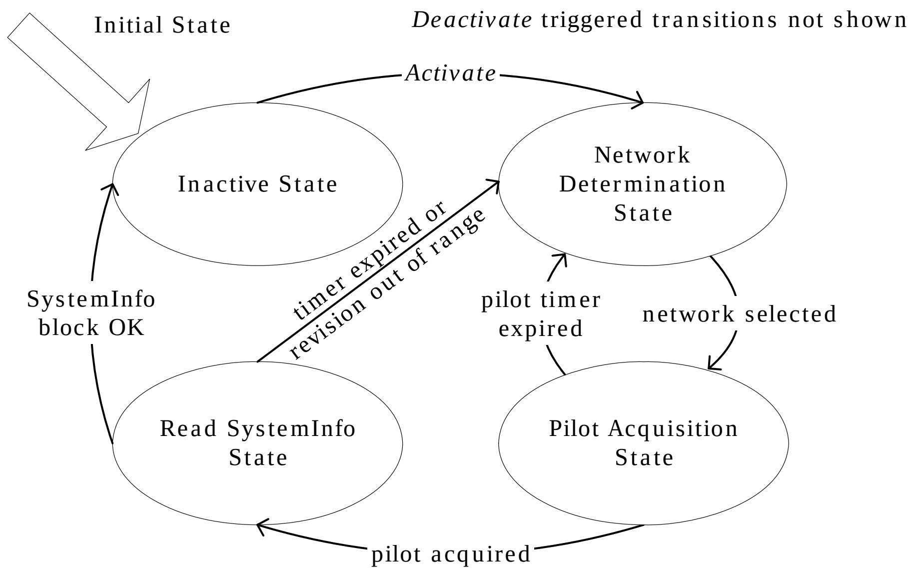

Figure 56 Default Initialization State Protocol state diagram (access terminal)

6.7.2 Primitives and public data

6.7.2.1 Commands

This protocol defines the following commands:

- *Activate* (an optional ChannelBand record can be specified with the command)
- *Deactivate*

6.7.2.2 Return indications

This protocol returns the following indications:

- *NetworkAcquired*

6.7.3 Public data

6.7.3.1 Static public data

- Selected ChannelBand
- System time
- PilotPN of the selected sector
- The following fields of the SystemInfo block:
  - MaximumRevision
  - MinimumRevision

6.7.3.2 Dynamic public data

- Subtype for this protocol

6.7.4 Protocol initialization and swap procedures

6.7.4.1 Protocol initialization

Upon initialization at the access terminal:

- The value for each attribute for this protocol instance shall be set to the default value for that attribute.
- The protocol shall enter the Inactive State.

Upon initialization at the access network:

- The value for each attribute for this protocol instance shall be set to the default value for that attribute.

6.7.4.2 Protocol swap

Upon swap at the access terminal, the protocol shall enter the Inactive State.

The access network shall define an empty swap procedure.

6.7.5 Procedures

The access network need not keep state for this protocol.

6.7.5.1 Command processing

The access network shall ignore all commands.

6.7.5.1.1 Activate

If the protocol receives an *Activate* command in the Inactive State, the access terminal shall transition to the Network Determination State.

If the protocol receives this command in any other state, the access terminal shall ignore it.

6.7.5.1.2 Deactivate

If the protocol receives a *Deactivate* command in the Inactive State, the access terminal shall ignore it.

If the protocol receives this command in any other state, the access terminal shall transition to the Inactive State.

6.7.5.2 Inactive state

In the Inactive State the access terminal waits for the protocol to receive an *Activate* command.

6.7.5.3 Network determination state

In the Network Determination State, the access terminal selects an OFDMA ChannelBand (see 10.1) on which to try and acquire the access network.

If a ChannelBand record was provided with the *Activate* command, the access terminal should select the system and channel specified by the record. Such a record may be provided by the access network when the Initialization State Protocol is activated due to a redirect message received by the Air Link Management Protocol.

The specific mechanisms to provision the access terminal with a list of preferred networks and with the actual algorithm used for network selection are beyond the scope of this specification.

Upon selecting a OFDMA Channel the access terminal shall enter the Pilot Acquisition State.

6.7.5.4 Pilot acquisition state

In the Pilot Acquisition State, the access terminal acquires the F-ACQCH of the selected OFDMA ChannelBand.

Upon entering the Pilot Acquisition State, the access terminal shall tune to the selected OFDMA ChannelBand and shall search for the pilot. If the access terminal acquires the pilot, it shall enter the Read SystemInfo State.[36] If the access terminal fails to acquire the pilot within $T_{ISPPilotAcq}$ seconds of entering the Pilot Acquisition State, it shall enter the Network Determination State.

6.7.5.5 Read SystemInfo state

Upon entering this state, the access terminal shall issue the *ControlChannelMAC.Activate* and *OverheadMessages.Activate* commands.

If the access terminal fails to receive the SystemInfo block within $T_{ISPSyncAcq}$ seconds of entering the Synchronization State, the access terminal shall issue *ControlChannelMAC.Deactivate* and *OverheadMessages.Deactivate* commands and shall enter the Network Determination State. While attempting to receive the SystemInfo block, the access terminal shall discard any other messages received on the Control Channel.

When the access terminal receives a SystemInfo block:

- If the access terminal's revision number is not in the range defined by the MinimumRevision and MaximumRevision fields (inclusive) specified in the message, the access terminal shall issue *ControlChannelMAC.Deactivate* and *OverheadMessages.Deactivate* commands and enter the Network Determination State.

---

[36] The Access Terminal Minimum Performance Requirements contains specifications regarding pilot acquisition performance.

- Otherwise, the access terminal shall:
  - Set the access terminal time to the time specified in the SystemInfo block. The time specified in the block is the time at the beginning of the superframe where the message transmission started. Note that this time is accurate only to the twelve least significant digits, and is enough for the access terminal to read the ExtendedChannelInfo message (on the Forward Traffic Channel MAC) that contains the complete time.
  - Return a *NetworkAcquired* indication.
  - Enter the Inactive State.

6.7.6 Message formats

No messages are defined for this protocol.

6.7.7 Interface to other protocols

6.7.7.1 Commands

This protocol issues the following commands:

- *ControlChannelMAC.Activate*
- *ControlChannelMAC.Deactivate*
- *OverheadMessages.Activate*
- *OverheadMessages.Deactivate*

6.7.7.2 Indications

This protocol does not register to receive any indications.

6.7.8 Configuration attributes

No configuration attributes are defined for this protocol.

6.7.9 Protocol numeric constants

Constant	Meaning	Value	Comments
$N_{ISPType}$	Type field for this protocol	Table 9	
$N_{ISPDefault}$	Subtype field for this protocol	0x0000	
$T_{ISPPilotAcq}$	Time to acquire pilot in access terminal	60 seconds	
$T_{ISPSyncAcq}$	Time to acquire SystemInfo block in access terminal	5 seconds	

6.7.10 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

7 Lower MAC Sublayer

7.1 Introduction

7.1.1 General overview

The Lower MAC Sublayer contains rules for the formulation of Lower MAC Sublayer packets for transmission over physical channels and interpretation of Lower MAC Sublayer packets provided from Physical Layer channels. In particular, the Lower MAC Sublayer contains the rules governing the operation of the Forward Traffic Channel and the Reverse Traffic Channel. In addition to these data channels, the Lower MAC Sublayer specifies rules for controlling and processing physical layer signaling channels on both the forward and reverse links, as described below. This section presents the protocols for the Lower MAC Sublayer. Each of these protocols can be independently negotiated at the beginning of the session. A description of the various physical channels controlled by each protocol may be found in the Overview chapter of this specification.

The Lower MAC Sublayer contains the following protocols:

- *Control Channel MAC Protocol*: This protocol contains the rules governing the operation of transmissions over the following Physical Layer channels: the other-sector interference channel (F-OSICH), and the two primary broadcast channels (F-pBCH0 and F-pBCH1).
- *Access Channel MAC Protocol*: This protocol contains the rules governing access terminal (AT) transmission timing and power characteristics for the R-ACH physical layer channel.
- *Shared Signaling MAC Protocol*: This protocol contains the rules governing the operation of the F-SSCH physical layer channel. The protocol handles H-ARQ, power control, channel assignment, and access grant delivery.
- *Forward Traffic Channel MAC Protocol*: This protocol contains the rules governing the operation of the F-DCH physical layer channel.
- *Reverse Control Channel MAC Protocol*: This protocol contains the rules governing the operation of the R-REQCH, R-CQICH, R-BFCH, R-SFCH, R-ACKCH, and R-PICH physical layer channels.
- *Reverse Traffic Channel MAC Protocol*: This protocol contains the rules governing the operation of the R-DCH physical layer channel.

7.1.2 Data encapsulation

In the transmit direction at the access network, the Lower MAC Sublayer receives Security Sublayer packets, uses the Forward Traffic Channel MAC to add layer-related headers and trailers, and forwards the resulting packet to the Physical Layer for transmission on the F-DCH.

In the transmit direction at the access terminal, the Lower MAC Sublayer receives Security Sublayer packets, uses the Reverse Traffic Channel MAC to add layer-related headers and trailers, and forwards the resulting packet to the Physical Layer for transmission on the R-DCH.

In the receive direction at the access terminal, the Lower MAC Sublayer receives Lower MAC packets from the Physical Layer F-DCH, uses the Forward Traffic Channel MAC to remove layer-related headers and trailers, and forward the contained Security Sublayer packets to the Security Sublayer.

In the receive direction at the access network, the Lower MAC Sublayer receives Lower MAC packets from the Physical Layer R-DCH, uses the Reverse Traffic Channel MAC to remove layer-related headers and trailers, and forwards the contained Security Sublayer packets to the Security Sublayer.

Figure 57 illustrates the relationship between Security Sublayer packets, Lower MAC packets, and Physical Layer packets. The content and formulation of Lower MAC headers and trailers is specified for each Lower MAC Sublayer protocol in this chapter.

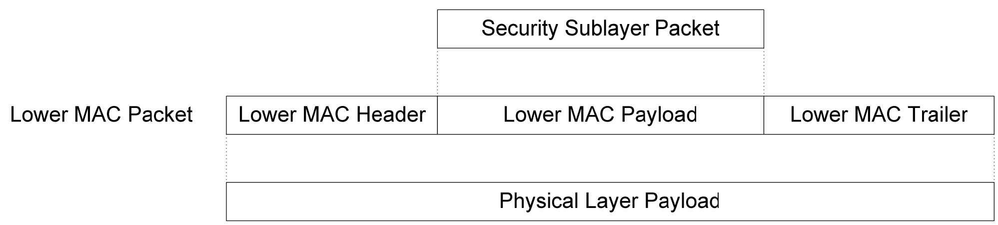

Figure 57 Lower MAC Sublayer packet encapsulation

7.1.3 Superframe timing

Forward and reverse link transmissions are divided into units of superframes. Superframes are further divided into units of PHY Frames. Frequency Division Duplex (FDD) and Time Division Duplex (TDD) superframe timing are described in the following sections.

Both forward and reverse link transmissions support H-ARQ. To provide time for the transmission and reception of acknowledgements, PHY Frames on the forward and reverse links are grouped into sets of PHY Frames called “interlaces,” and successive H-ARQ transmissions of MAC packets are confined to a single interlace. The Lower MAC Sublayer chapter of this specification uses a PHY Frame indexing scheme that is convenient for the descriptions herein, but may not be consistent with indexing schemes used in other layers and sublayers in the specification. Refer to the FTC and RTC MAC Protocols for the exact specification of H-ARQ operation.

7.1.3.1 FDD

7.1.3.1.1 Superframe timing

An FDD forward link superframe consists of a superframe preamble followed by $N_{FDD,FLPHYFrames}$ FL PHY Frames, and an FDD reverse link superframe consists of $N_{FDD,RLPHYFrames}$ RL PHY Frames, where $N_{FDD,FLPHYFrames}$ represents the number of FDD FL PHY Frames per FL superframe and $N_{FDD,RLPHYFrames}$ represents the number of FDD RL PHY Frames per RL superframe. These are constants in the system and $N_{FDD,FLPHYFrames} = N_{FDD,RLPHYFrames} = 24$.

The first RL PHY Frame of each RL superframe is lengthened by the duration of the FL superframe preamble to ensure superframe timing alignment between the forward link and reverse link.

Each superframe shall be uniquely identified by a superframe index that is incremented every superframe.

- Each FL PHY Frame shall be uniquely identified by a FL PHY Frame Index. The FL PHY Frame Index of the i^{th} FL PHY Frame, $i = 0, 1, \ldots, N_{FDD,FLPHYFrames}-1$, in the j^{th} superframe, $j = 0, 1, \ldots$, shall be given by $j{*}N_{FDD,FLPHYFrames} + i$.
- Each RL PHY Frame shall be uniquely identified by a RL PHY Frame Index. The RL PHY Frame Index of the i^{th} RL PHY Frame, $i = 0, 1, \ldots, N_{FDD,RLPHYFrames}-1$, in the j^{th} superframe, $j = 0, 1, \ldots$, shall be given by $j{*}N_{FDD,RLPHYFrames} + i$.

Henceforth, a PHY Frame (FL or RL) with PHY Frame index *k* is referred to as (FL or RL) PHY Frame *k*. The exact timing details are provided by the Physical Layer Specification. The structure of FDD forward link and reverse link superframes with $N_{FDD,FLPHYFrames} = 24$ and $N_{FDD,RLPHYFrames} = 24$ is shown in Figure 58.

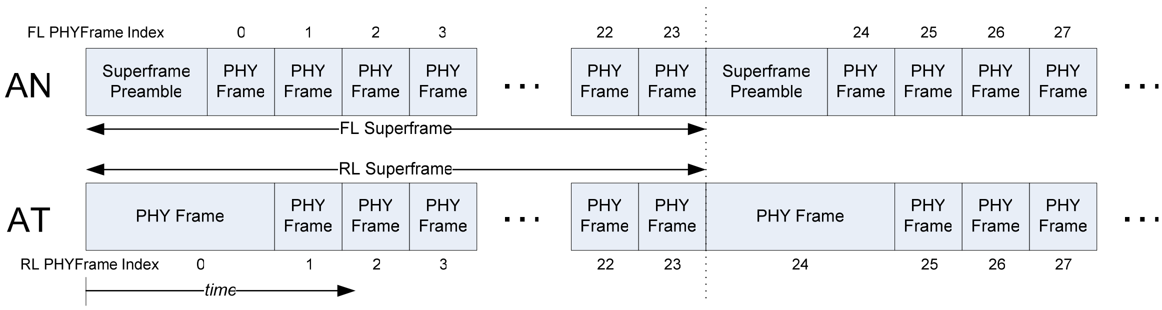

Figure 58 FDD superframe timing

7.1.3.1.2 H-ARQ interlace structure

There are six interlaces on both the forward and reverse links, and PHY Frame *k* is a member of interlace *k* mod *6*. Forward link assignments that arrive in FL PHY Frame *k* shall apply to the interlace containing FL PHY Frame *k* and starting with FL PHY Frame *k*. A FL transmission of a MAC packet on FL PHY Frame *k* is acknowledged on RL PHY Frame $k+3$. HARQ retransmissions associated with the MAC packet that starts in PHY Frame *k* occur in PHY Frames $k+6n$ where *n* is the retransmission index, n=0,1, … .

Figure 59 shows examples of the timing relationship between FL packet transmissions and the associated acknowledgement transmissions for the FDD mode with $N_{FDD,FLPHYFrames} = 24$ and $N_{FDD,RLPHYFrames} = 24$.

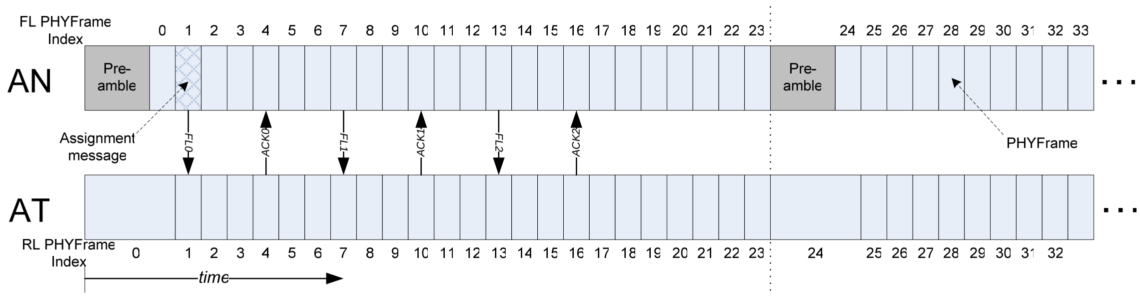

Figure 59 FDD FL H-ARQ interlace structure

Reverse link assignments that arrive in FL PHY Frame *k* shall apply to the interlace containing RL PHY Frame *k+2* and starting with RL PHY Frame *k+2*. A RL transmission of a MAC packet on RL PHY Frame *k* is acknowledged on FL PHY Frame *k+4*. HARQ retransmissions associated with the MAC packet that starts in PHY Frame *k* occur in PHY Frames *k+6n* where *n* is the retransmission index, *n*=0,1, … .

Figure 60 shows the timing relationship between the RL packet transmissions and the associated acknowledgment transmissions for the FDD mode with $N_{FDD,FLPHYFrames} = 24$ and $N_{FDD,RLPHYFrames} = 24$.

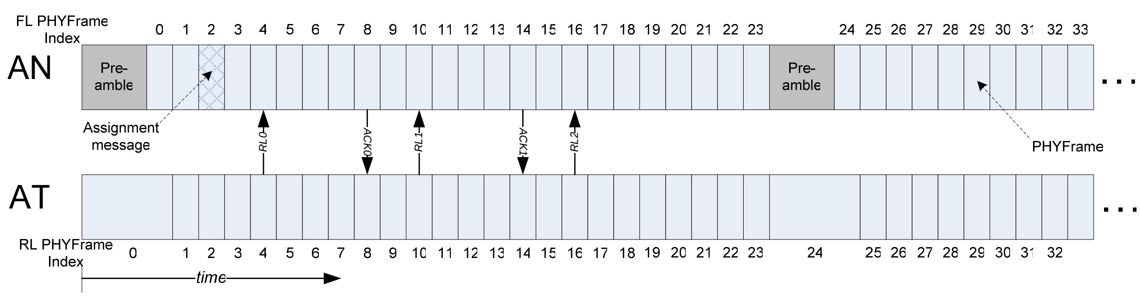

Figure 60 FDD RL H-ARQ interlace structure

7.1.3.1.3 H-ARQ interlace structure for Extended Transmission Duration Assignments

For FDD, in addition to the standard H-ARQ interlace structure, the system supports the use of "Extended Transmission Duration" assignments. Such assignments utilize extended PHY Frames (denoted E-PHY Frames, where E-PHY Frame *k* consists of PHY Frames *k* through *k+5*) and alter the timing of transmissions and corresponding ACK transmissions relative to the standard assignments. Extended Transmission Duration assignments create a potential for resource assignment collisions with standard assignments, and the AN should manage resource assignments to prevent such collisions.

Forward link assignments that arrive in FL PHY Frame *k* shall apply to the E-PHY Frame and the associated Extended Transmission Duration interlace starting with FL PHY Frame *k*. Note that with Extended Transmission Duration assignments, the associated interlace can begin in any FL PHY Frame. A FL transmission of a MAC packet using an Extended Transmission Duration assignment that starts on FL PHY Frame *k* continues for six consecutive PHY Frames (an E-PHY Frame). The E-PHY Frame is acknowledged using a combination of RL PHY Frames *k+8* and *k+9*. HARQ retransmissions associated with the MAC packet that starts in FL PHY Frame *k* occur in FL PHY Frames *k+12n* where *n* is the retransmission index.

Figure 61 shows examples of the timing relationship between FL packet transmissions and the associated acknowledgement transmissions for the FDD mode with $N_{FDD,FLPHYFrames}$ = 24 and $N_{FDD,RLPHYFrames}$ = 24.

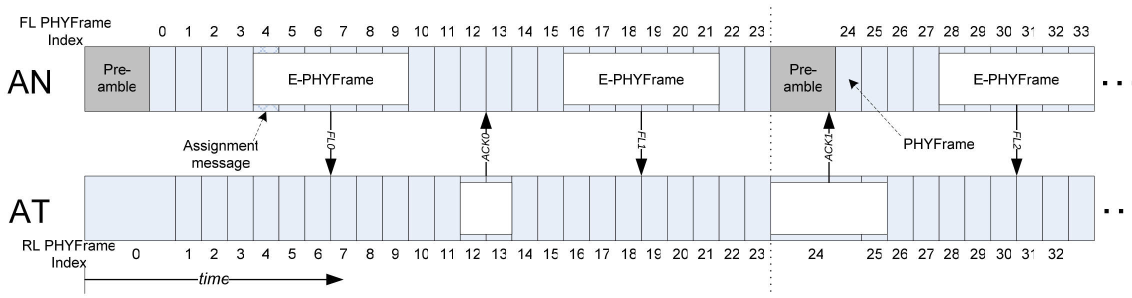

Figure 61 FDD FL H-ARQ interlace structure for Extended Transmission Duration Assignments

Reverse link assignments that arrive in FL PHY Frame *k* shall apply to the E-PHY Frame and the associated Extended Transmission Duration interlace starting with RL PHY Frame *k+3*. A RL transmission of a MAC packet using an Extended Transmission Duration assignment that starts on RL PHY Frame *k* continues for six consecutive PHY Frames (an E-PHY Frame). The E-PHY Frame is acknowledged on FL PHY Frame *k+9*. HARQ retransmissions associated with the MAC packet that starts in PHY Frame *k* occur in PHY Frames *k+12n* where *n* is the retransmission index.

Figure 62 shows examples of the timing relationship between RL packet transmissions and the associated acknowledgement transmissions for the FDD mode with $N_{FDD,FLPHYFrames}$ = 24 and $N_{FDD,RLPHYFrames}$ = 24.

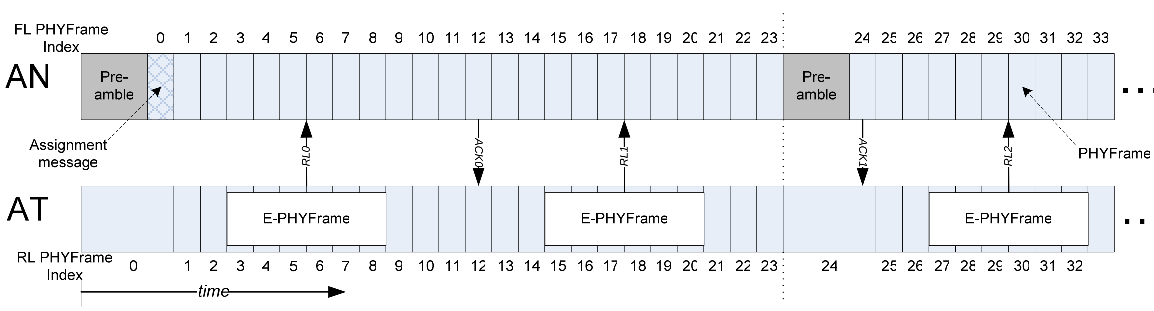

Figure 62 FDD RL H-ARQ interlace structure for Extended Transmission Duration Assignments

7.1.3.2 TDD

TDD mode has two associated variables called N_{FL_BURST} and N_{RL_BURST} (respectively given by the N_FLBurst and N_RLBurst parameters which are part of the public data of the Overhead Messages Protocol) that determine FL and RL partitioning. Specifically, the N_{FL_BURST}:N_{RL_BURST} partitioning refers to a partitioning wherein N_{FL_BURST} FL PHY Frames are transmitted on the FL followed by N_{RL_BURST} RL PHY Frames on the RL and so on in time. Both N_{FL_BURST} and N_{RL_BURST} are limited to the range 1-4 and are specified by 2 bit fields (see Overhead Messages Protocol).

A TDD forward link superframe consists of a superframe preamble and $N_{TDD,FLPHYFrames}$ FL PHY Frames, and a TDD reverse link superframe consists of $N_{TDD,RLPHYFrames}$ RL PHY Frames where $N_{TDD,FLPHYFrames}$ is the number of FL PHY Frames per FL superframe, and $N_{TDD,RLPHYFrames}$ is the number of RL PHY Frames per RL superframe. $N_{TDD,FLPHYFrames}$ and $N_{TDD,RLPHYFrames}$ are computed as follows:

$$N_{TDD,FLPHYFrames} = N_{FL\_BURST} \left\lceil \frac{24}{N_{FL\_BURST} + N_{RL\_BURST}} \right\rceil$$

$$N_{TDD,RLPHYFrames} = N_{RL\_BURST} \left\lceil \frac{24}{N_{FL\_BURST} + N_{RL\_BURST}} \right\rceil$$

The HARQ structure for the special case $N_{FL\_BURST} = N_{RL\_BURST} = 1$, or the 1:1 partitioning is described first. Then the 2:1 partitioning, followed by a general N_{FL_BURST}:N_{RL_BURST} partitioning is described. Note that while the general partition description does reduce to the 1:1 partitioning with $N_{FL\_BURST} = N_{RL\_BURST} = 1$, the 2:1 partitioning does not follow from the general partitioning because some optimizations have been used in the 2:1 partitioning to reduce H-ARQ retransmission latency on the FL.

7.1.3.2.1 Superframe timing for 1:1 partitioning

In the TDD mode, FL and RL transmissions alternate in time. Excluding the superframe preamble at the beginning of each FL superframe, the transmissions alternate between FL PHY Frames and RL PHY Frames throughout the superframe. An appropriate amount of guard interval is inserted between each FL-RL transition.

Each superframe shall be uniquely identified by a superframe index that is incremented every superframe.

- Each FL PHY Frame shall be uniquely identified by a FL PHY Frame Index. The FL PHY Frame Index of the i^{th} FL PHY Frame, i = 0, 1,…, $N_{TDD,FLPHYFrames}$-1, in the j^{th} superframe, j = 0, 1,…, shall be given by $j*N_{TDD,FLPHYFrames} + i$.
- Each RL PHY Frame shall be uniquely identified by a RL PHY Frame Index. The RL PHY Frame Index of the i^{th} RL PHY Frame, i = 0, 1,…, $N_{TDD,RLPHYFrames}$-1, in the j^{th} superframe, j = 0, 1,…, shall be given by $j*N_{TDD,RLPHYFrames} + i$.

Henceforth, a PHY Frame (FL or RL) with PHY Frame index *k* is referred to as (FL or RL) PHY Frame *k*. The exact timing details are provided by the Physical Layer specification. The structure of the TDD forward link and reverse link superframes with $N_{TDD,FLPHYFrames}$ = 12 and $N_{TDD,RLPHYFrames}$ = 12 is shown in Figure 63.

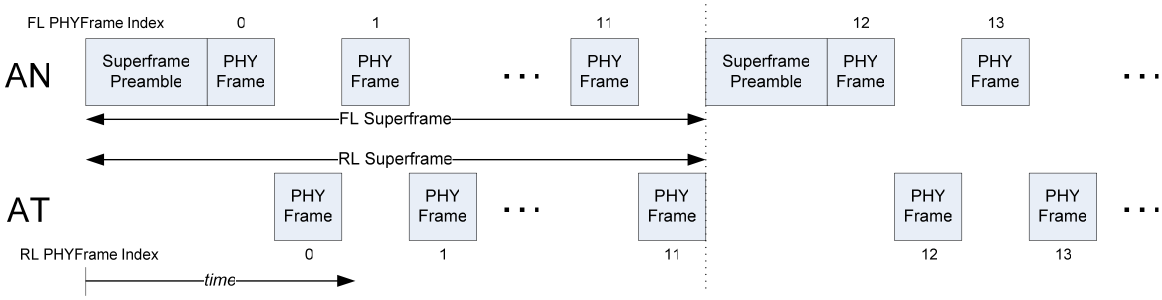

Figure 63 TDD superframe timing for 1:1 partitioning

7.1.3.2.2 H-ARQ interlace structure for 1:1 partitioning

There are three interlaces on both the forward and reverse links. Forward link assignments that arrive in FL PHY Frame *k* shall apply to the interlace containing FL PHY Frame *k* and starting with FL PHY Frame *k*. A FL transmission of a MAC packet on FL PHY Frame *k* is acknowledged on RL PHY Frame *k+1*. HARQ retransmissions associated with the MAC packet that starts in PHY Frame *k* occur in PHY Frames *k+3n* where *n* is the retransmission index, *n*=0,1, … .

Figure 64 shows the timing relationship between the FL packet transmissions and the associated acknowledgment transmissions for the TDD mode with $N_{TDD,FLPHYFrames}$ = 12 and $N_{TDD,RLPHYFrames}$ = 12.

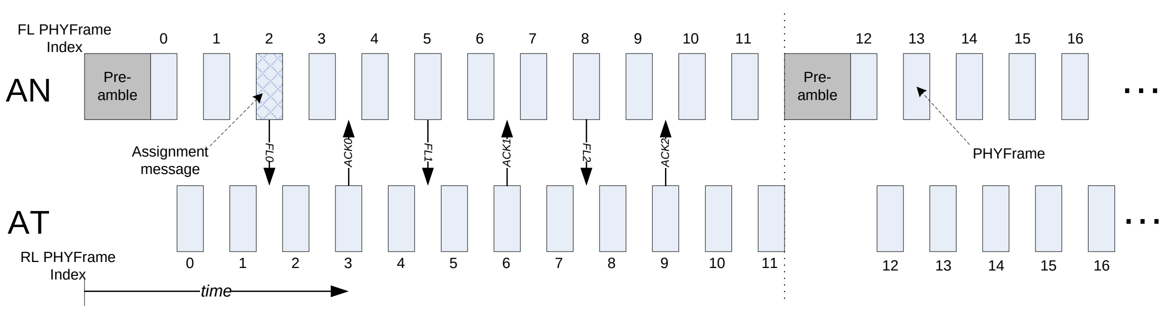

Figure 64 TDD FL H-ARQ interlace structure for 1:1 partitioning

Reverse link assignments that arrive in FL PHY Frame *k* shall apply to the interlace containing RL PHY Frame *k+1* and starting with RL PHY Frame *k+1*. A RL transmission of a MAC packet on RL PHY Frame *k* is acknowledged on FL PHY Frame *k+2*. HARQ retransmissions associated with the MAC packet that starts in PHY Frame *k* occur in PHY Frames *k+3n* where *n* is the retransmission index.

Figure 65 shows the timing relationship between the RL packet transmissions and the associated acknowledgment transmissions for the TDD mode with $N_{TDD,FLPHYFrames}$ = 12 and $N_{TDD,RLPHYFrames}$ = 12.

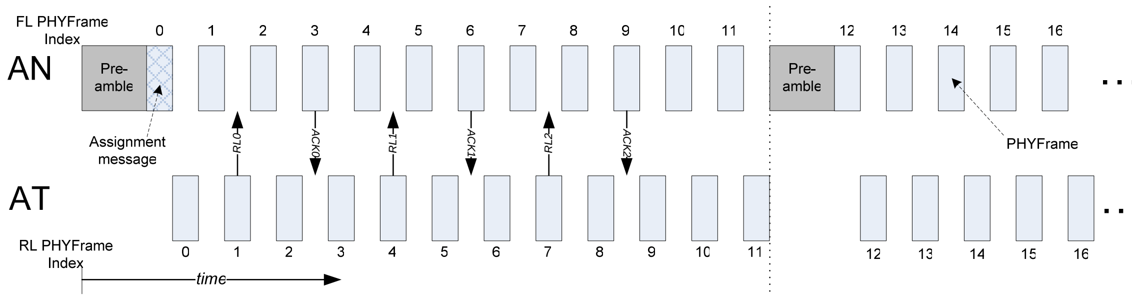

Figure 65 TDD RL H-ARQ interlace structure for 1:1 partitioning

7.1.3.2.3 Superframe timing for 2:1 partitioning

In the TDD 2:1 mode, FL and RL transmissions alternate in time, with 2 FL PHY Frames followed by 1 RL PHY Frame in time. Excluding the superframe preamble at the beginning of each FL superframe, the transmissions alternate between FL PHY Frames and RL PHY Frames throughout the superframe. An appropriate amount of guard interval is inserted between each FL-RL transition.

Each superframe shall be uniquely identified by a superframe index that is incremented every superframe.

- Each FL PHY Frame shall be uniquely identified by a FL PHY Frame Index. The FL PHY Frame Index of the i^{th} FL PHY Frame, $i = 0, 1,\ldots, N_{TDD,FLPHYFrames}-1$, in the j^{th} superframe, $j = 0, 1,\ldots$, shall be given by $j*N_{TDD,FLPHYFrames} + i$.
- Each RL PHY Frame shall be uniquely identified by a RL PHY Frame Index. The RL PHY Frame Index of the i^{th} RL PHY Frame, $i = 0, 1,\ldots, N_{TDD,RLPHYFrames}-1$, in the j^{th} superframe, $j = 0, 1,\ldots$, shall be given by $j*N_{TDD,RLPHYFrames} + i$.

Henceforth, a PHY Frame (FL or RL) with PHY Frame index k is referred to as (FL or RL) PHY Frame k. The exact timing details are provided by the Physical Layer specification. The structure of the TDD forward link and reverse link superframes with $N_{TDD,FLPHYFrames}$ = 16 and $N_{TDD,RLPHYFrames}$ = 8 is shown in Figure 66.

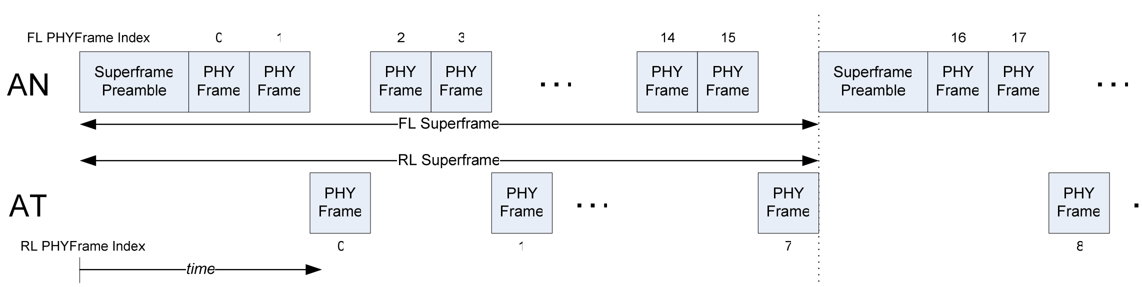

Figure 66 TDD superframe timing for 2:1 partitioning

7.1.3.2.4 H-ARQ interlace structure for 2:1 partitioning

There are five interlaces on the forward and two interlaces on the reverse link. Forward link assignments that arrive in FL PHY Frame *k* shall apply to the interlace containing FL PHY Frame *k* and starting with FL PHY Frame *k*. A FL transmission of a MAC packet FL PHY Frame *k* is acknowledged in RL PHY Frame $\lfloor k/2 \rfloor +1$. HARQ retransmissions associated with the MAC packet that starts in PHY Frame *k* occur in PHY Frames $k+5n$ where *n* is the retransmission index, n=0,1, … .

Figure 67 shows the timing relationship between the FL packet transmissions and the associated acknowledgment transmissions for the TDD mode with $N_{TDD,FLPHYFrames}$ = 16 and $N_{TDD,RLPHYFrames}$ = 8.

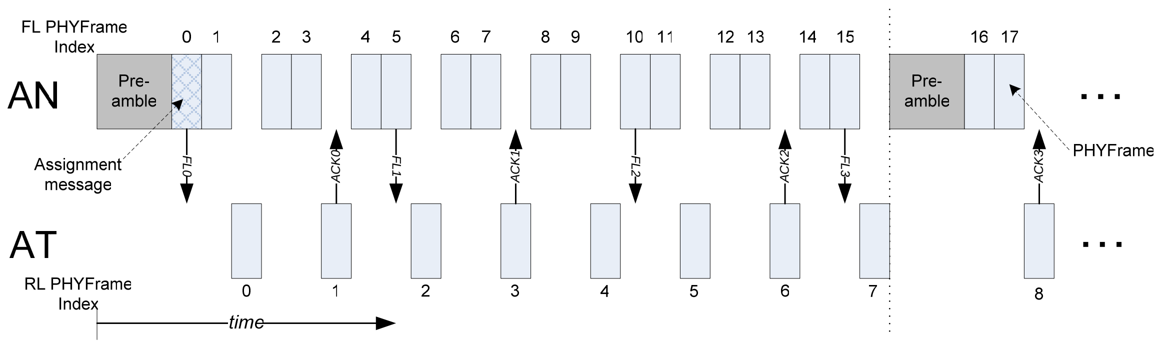

Figure 67 TDD FL H-ARQ interlace structure for 2:1 partitioning

Reverse link assignments that arrive in FL PHY Frame *k* shall apply to the interlace containing RL PHY Frame $\lfloor k/2 \rfloor$ and starting with the same RL PHY Frame. Note that due to assignment demodulation requirements at the AT, Assignment messages are only sent in FL PHY Frames with an even index. A RL transmission of a MAC packet on RL PHY Frame *j* is acknowledged on FL PHY Frame $2j+4$. HARQ retransmissions associated with the MAC packet that starts in PHY Frame *k* occur in PHY Frames $k+2n$ where *n* is the retransmission index.

Figure 68 shows the timing relationship between the RL packet transmissions and the associated acknowledgment transmissions for the TDD mode with $N_{TDD,FLPHYFrames}$ = 16 and $N_{TDD,RLPHYFrames}$ = 8.

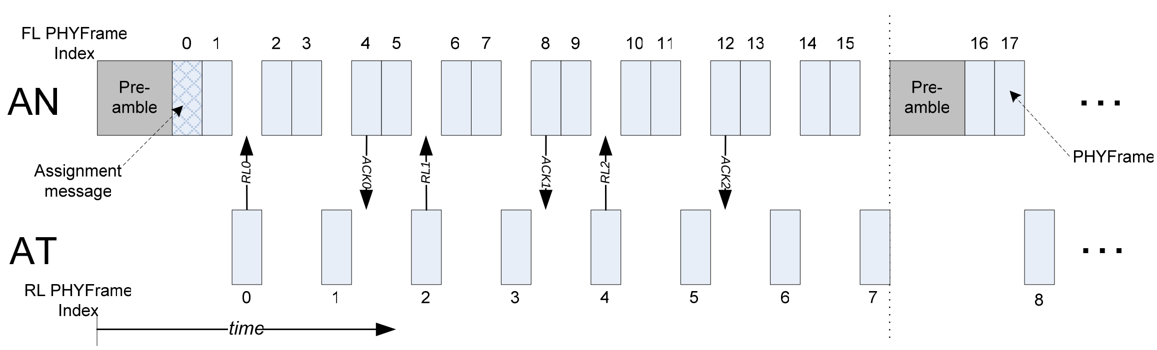

Figure 68 TDD RL H-ARQ interlace structure for 2:1 partitioning

7.1.3.2.5 Superframe timing and interlace structure for Generalized TDD partitioning

In addition to the 1:1 and 2:1 FL/RL TDD partitioning structures, the system supports a general TDD partitioning.[37] For convenience, let a FL burst be a set of N_{FL_BURST} PHY Frames that are transmitted continuously in time on the FL, and a RL burst be a set of N_{RL_BURST} PHY Frames that are transmitted continuously in time on the RL resulting in a N_{FL_BURST}:N_{RL_BURST} partitioning. FL and RL bursts then alternate in time, as shown in Figure 69. Let the "burst index" be an index of bursts, for example, FL Burst *i* consists of FL PHY Frames $i*N_{FL\_BURST}$ through $(i+1)N_{FL\_BURST} - 1$.

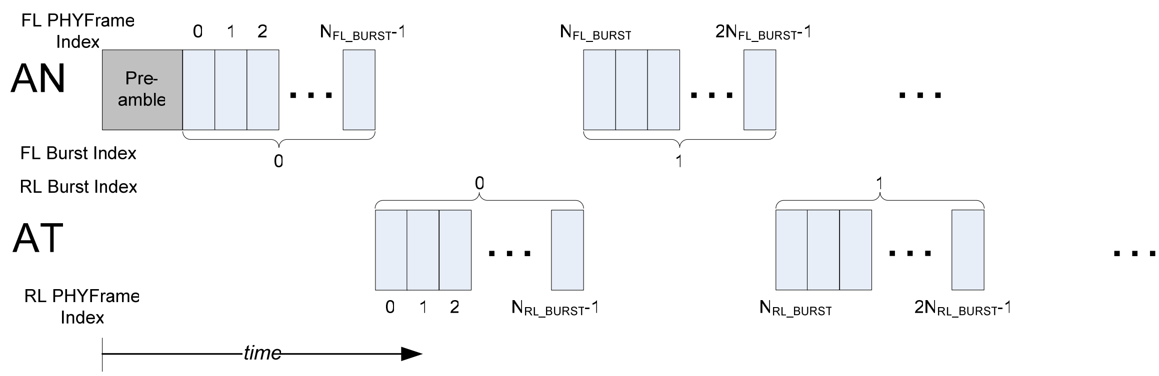

Figure 69 TDD Superframe Structure for other partitionings

Assignment, acknowledgement, and retransmission timing can then be specified using the following formulas, where *i* is the FL Burst Index, and *j* is the RL Burst index.

Assignments:

FL PHY Frames in *i* are assigned in FL burst *i*.

RL PHY Frames in *j* are assigned in FL burst $i=j$ if $N_{FL\_BURST}>1$ or $i=j-1$ if $N_{FL\_BURST} = 1$.

Note that for RL assignments, if $N_{FL\_BURST} \geq N_{RL\_BURST}$, then an assignment in the *k*th PHY Frame of the FL burst applies to the interlace containing *k*th PHY Frame of the relevant RL burst. However, if $N_{FL\_BURST} < N_{RL\_BURST}$, the assignment will contain additional bits enabling the AT to determine the interlace of the assignment.

Acknowledgement:

FL PHY Frames in *i* are acknowledged in RL burst $j=i+1$.

RL PHY Frames in *j* are acknowledged in FL burst $i=j+2$.

See 7.6.6.3.11.2 for the details of the ACK channelization within the relevant burst.

Retransmissions:

FL PHY Frames in *i* are retransmitted in FL burst $i+2$ if $N_{RL\_BURST}>1$ or $i+3$ if $N_{RL\_BURST}=1$.

RL PHY Frames in *j* are retransmitted in RL burst $j+2$ if $N_{FL\_BURST}>1$ or $j+3$ if $N_{FL\_BURST}=1$.

Note that retransmissions occur in the same PHY Frame within a burst. Thus, if a transmission occurs in the second PHY Frame of the burst, then all retransmissions shall also occur in the second PHY Frame of the relevant subsequent bursts.

---

[37] Note that this generalized partitioning reduces to the 1:1 partitioning described above in section 7.1.3.2.1, but does not reduce to the 2:1 partitioning of section 7.1.3.2.3. The 2:1 partitioning has been optimized on the FL to minimize retransmission latency.

Resulting number of interlaces:

$2*N_{FL\_BURST}$ FL interlaces if $N_{RL\_BURST}>1$ or $3*N_{FL\_BURST}$ FL interlaces if $N_{RL\_BURST}=1$.

$2*N_{RL\_BURST}$ RL interlaces if $N_{FL\_BURST}>1$ or $3*N_{RL\_BURST}$ RL interlaces if $N_{FL\_BURST}=1$.

7.1.4 Common definitions and terms

7.1.4.1 Channel trees

A channel tree is used to specify sets of hop-ports that are associated with each node identification number (NodeID). A set of hop-ports is said to be "mapped to a node" and a node "maps" a set of hop-ports. The following terms are used in association with channel trees in this specification:

- **Hop-port.** The fundamental unit of resource assignment. Each hop-port maps to one unique subcarrier. The mapping of hop-ports to subcarriers varies with time according to hopping rules specified in the Physical Layer specification.
- **Node.** A node corresponds to a single NodeID.
- **Children, Descendants.** Nodes that map a subset of the hop-ports mapped by a node.
- **Parents, Ancestors.** Nodes that map a superset of the hop-ports mapped by a node.
- **Base-nodes.** Nodes with no children. Base-nodes are assigned specific Physical Layer resources – in this case, hop-ports.
- **MinHopPortsPerNode.** The minimum number of hop-ports mapped to a node.

The indices of hop-ports follows a convention used throughout the Lower MAC Sublayer. Namely, hop-port indices are allocated sequentially starting from index 0. The base nodes are ordered in a left-to-right ordering as drawn, and hop-port indices are allocated sequentially starting from index 0 to the base nodes from left to right.

For example, the channel tree in Figure 70 has 14 nodes, numbered from 0 to 13. The base-nodes are NodeIDs 0 – 7. There are a total of 35 hop-ports available. Consider the node associated with NodeID 10. This node has parent NodeID 12 and children NodeIDs 4 and 5. The node maps 11 hop-ports, namely hop-ports 21-31.

Because nodes define orthogonal assignments, the use of a node in the tree can restrict use of other nodes. Thus, if a node is in use, then all descendants and ancestors of the node are unavailable for use and are called "restricted" nodes.

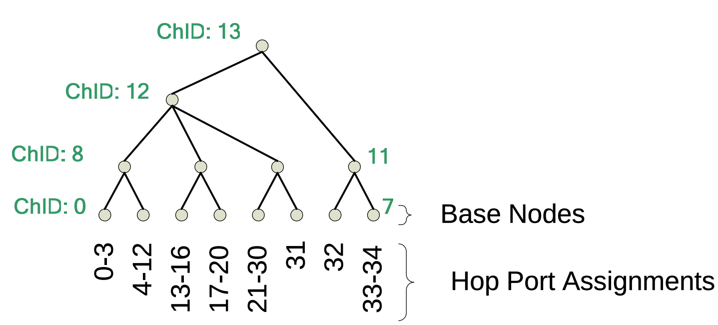

Figure 70 Example channel tree

7.1.4.2 Power density

The Lower MAC Layer computes the power settings of Physical Layer channels and passes these settings to the Physical Layer for use. In some cases, "power density" is used to communicate transmission power levels, as defined below:

Power density	The expected energy per modulation symbol[38]

where "modulation symbol" refers to frequency domain (pre-IFFT) symbols, as defined in the Physical Layer specification. The "expected" power is used to cover the case of transmissions using non-constant modulus signal constellations. In all cases, the density shall be computed according to the parameters of the modulation symbols in the particular channel being specified.

7.2 Default Control Channel MAC Protocol

7.2.1 Overview

The Default Control Channel MAC Protocol provides the procedures and messages required for an access network (AN) to transmit and for an access terminal to receive the Control Channel. The protocol controls transmissions over the following Physical Layer channels: the other- sector interference channel (OSICH), and the primary broadcast channels (pBCH0 and pBCH1).

This specification assumes that the access network has one instance of this protocol for each sector in the network.

This protocol can be in one of two states:

- *Inactive State*: In this state, the protocol waits for an *Activate* command. This state applies only to the access terminal and occurs when the access terminal has not acquired an access network or is not monitoring the Control Channel.
- *Active State*: In this state, the access network transmits and the access terminal receives the Control Channel.

[38] Note that this definition of power density is not true in the strictest sense because of cyclic prefix and windowing overheads. However, it provides a convenient method of communicating power levels to the Physical Layer Protocol. The construction of transmitted waveforms as a function of power densities is specified by the Physical Layer Protocol.

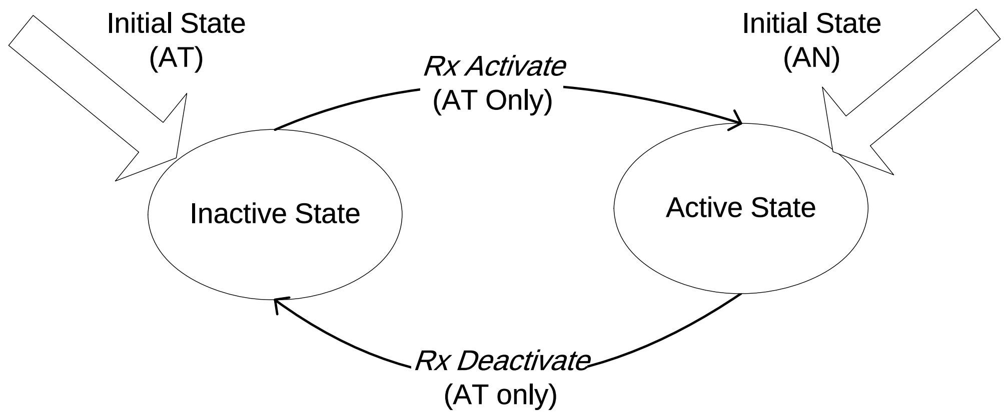

Figure 71 Default Control Channel MAC Protocol state diagram

7.2.2 Primitives

7.2.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

7.2.2.2 Return indications

This protocol returns the following indications:

- *SupervisionFailed*
- *PageReceived*
- *QuickPageReceived*

7.2.3 Public data

7.2.3.1 Static public data

This protocol does not define any static public data.

7.2.3.2 Dynamic public data

- Subtype for this protocol.

7.2.4 Protocol data unit

The transmission unit of this protocol is defined separately for the three physical channels used by the protocol.(pBCH0, pBCH1, and OSICH), as defined in 7.2.6.1, 7.2.6.2, and 7.2.6.3.

7.2.5 Protocol initialization and swap

7.2.5.1 Protocol initialization

Upon initialization at the access terminal,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Inactive State.

Upon initialization at the access network,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Active State.

7.2.5.2 Protocol swap

Upon swap at the access terminal, the protocol instance shall enter the Inactive State.

Upon swap at the access network, the protocol instance shall enter the Active State.

7.2.6 Procedures

The following sections specify procedures for transmission and reception of the F-pBCH0, F-pBCH1, and F-OSICH Physical Layer channels.

7.2.6.1 Procedures for transmission and reception of the F-pBCH1 Physical Layer channel

If multi-carrier mode is MultiCarrierOn, this protocol shall deliver a separate payload for transmission on each carrier. An F-pBCH1 channel transmission consists of the following payload:

Field	Length (bits)
LoadControl	2
DataCtrlOffset	4
ACKCtrlOffset	4
BlockType0	2
QuickPage block	0 or N_{QP_BLK}
QuickChannelInfo block	0 or as specified by Overhead Messages Protocol
BlockType1	0 or 2
QuickPage block	0 or N_{QP_BLK}
QuickChannelInfo block	0 or as specified by Overhead Messages Protocol
BlockType2	0 or 2

Field	Length (bits)
QuickPage block	0 or N_{QP_BLK}
QuickChannelInfo block	0 or as specified by Overhead Messages Protocol
BlockType3	0 or 2
QuickPage block	0 or N_{QP_BLK}
QuickChannelInfo block	0 or as specified by Overhead Messages Protocol
Reserved	Variable

LoadControl — This field shall determine the class of access terminals that are allowed to make an access attempt in the superframe.

DataCtrlOffset — This field shall determine the offset between the power spectral densities of the Reverse CDMA Control Segment and the Reverse Traffic Channel in units of 0.5 dB

ACKCtrlOffset — This field shall determine the offset between the power spectral densities of the Reverse CDMA Control Segment and the Reverse ACK Channel in units of 0.5 dB

BlockType*j* — This field shall indicate the type of payload carried by the transmission. The block types '00' and '11' are reserved.

QuickPage block — This field shall be included if BlockType preceding it is set to '01'. The transmitter shall construct the QuickPage block in accordance with the reception procedure 7.2.6.1.1.

QuickChannelInfo block — This field shall be included if BlockType preceding it is set to '10'. The transmitter shall obtain the QuickChannelInfo block from the Overhead Messages Protocol.

Reserved — The transmitter shall set the number of reserved bits to zero so that the packet given to the physical layer is $N_{CCMPpBCH1Bits}$ bits in length. The receiver shall ignore this field.

The fields above obey the following rules:

- BlockType*j* shall take the value '01' if it is followed by a QuickPage block.
- BlockType*j* shall take the value '10' if it is followed by a QuickChannelInfo block.
- The number of BlockType*j* fields shall be in accordance with $N_{CCMPpBCH1Bits}$.
- If a QuickChannelInfo block is included, it shall be the last block included.

The above rule makes it possible to carry, for example, two QuickPage and one QuickChannelInfo block in the same pBCH1 transmission. Note that the actual number of blocks that can be transmitted depends on the $N_{CCMPpBCH1Bits}$.

Also note that the above rules disallow more than one QuickChannelInfo blocks in the same pBCH1 transmission.

The transmitter shall set one of the BlockType fields to ‘10’ in superframes with an even superframe index. The transmitter shall set at least one of the BlockType to ‘01’ in superframes with an odd superframe index. At the transmit side, the protocol shall deliver the packet for transmission on F-pBCH1.

The receiver shall use 7.2.6.1.1 to parse the received QuickPage block. The receiver shall forward the received QuickChannelInfo block and reserved bits to the Overhead Messages Protocol.

7.2.6.1.1 Procedure for reception of the QuickPage block

The access terminal shall declare the QuickPage block to be in error if there is an error on the F-pBCH1 channel that contains a QuickPage block (this error information may be used by the Idle State Protocol). The access terminal shall process only those QuickPage blocks that occur in superframes with superframe number contained in the PageTimes array that is public data of theIdle State Protocol. The access terminal shall ignore QuickPage blocks in other superframes.

The QuickPage block may contain multiple QuickPages or a single Page (as decided by the access network). The access terminal shall process each received QuickPage block as described below.

The format of the QuickPage block shall depend on the first three bits (NumPages field) in the block. NumPages is the number of pages or quick pages in the QuickPage block. The access terminal performs the following actions upon receiving a QuickPage block.

- If NumPages=0 the QuickPage block shall be ignored.
- If NumPages=1 and the ATI in the QuickPage block matches the ReceiveATIList of the Address Management Protocol, the protocol shall issue a *PageReceived* indication.
- If 1<NumPages≤6, and the least significant bits of RQuickPage that is public data of the Idle State Protocol match one of the RQuickPage*i* fields in the QuickPage block, the protocol shall generate a *QuickPageReceived* indication.
- If NumPages=7, the protocol shall generate a *QuickPageReceived* indication if one of the following conditions hold:
  - The least significant bits of RQuickPage match one of the RQuickPage*i* fields in the QuickPage block.
  - RQuickPage7 is not equal to RQuickPage8, and the 4 least significant bits of RQuickPage are larger than RQuickPage8.

Table 56 QuickPage format with NumPages=0

Field	Length (bits)	Value
NumPages	3	0x0
Reserved	N_{QP_BLK} -3	

Table 57 QuickPage format with NumPages=1

Field	Length (bits)	Value
NumPages	3	0x1
ATI	32	ATI of paged terminal
Reserved	N_{QP_BLK} -35	

Table 58 QuickPage format with NumPages=2

Field	Length (bits)	Value
NumPages	3	0x2
RQuickPage1	16	RQuickPageLSBs
RQuickPage2	16	RQuickPageLSBs
Reserved	N_{QP_BLK} -35	

Table 59 QuickPage format with NumPages=3

Field	Length (bits)	Value
NumPages	3	0x3
RQuickPage1	10	RQuickPageLSBs
RQuickPage2	10	RQuickPageLSBs
RQuickPage3	10	RQuickPageLSBs
Reserved	N_{QP_BLK} -33	

Table 60 QuickPage format with NumPages=4

Field	Length (bits)	Value
NumPages	3	0x4
RQuickPage1	8	RQuickPageLSBs
RQuickPage2	8	RQuickPageLSBs
RQuickPage3	8	RQuickPageLSBs
RQuickPage4	8	RQuickPageLSBs
Reserved	N_{QP_BLK} -35	

Table 61 QuickPage format with NumPages=5

Field	Length (bits)	Value
NumPages	3	0x5
RQuickPage1	6	RQuickPageLSBs
RQuickPage2	6	RQuickPageLSBs
RQuickPage3	6	RQuickPageLSBs
RQuickPage4	6	RQuickPageLSBs
RQuickPage5	6	RQuickPageLSBs
Reserved	N_{QP_BLK} -33	

Table 62 QuickPage format with NumPages=6

Field	Length (bits)	Value
NumPages	3	0x6
RQuickPage1	5	RQuickPageLSBs
RQuickPage2	5	RQuickPageLSBs
RQuickPage3	5	RQuickPageLSBs
RQuickPage4	5	RQuickPageLSBs
RQuickPage5	5	RQuickPageLSBs
RQuickPage6	5	RQuickPageLSBs
Reserved	N_{QP_BLK} -33	

Table 63 QuickPage format with NumPages=7

Field	Length (bits)	Value
NumPages	3	0x7
RQuickPage1	4	RQuickPageLSBs
RQuickPage2	4	RQuickPageLSBs
RQuickPage3	4	RQuickPageLSBs
RQuickPage4	4	RQuickPageLSBs
RQuickPage5	4	RQuickPageLSBs
RQuickPage6	4	RQuickPageLSBs
RQuickPage7	4	RQuickPageLSBs
RQuickPage8	4	RQuickPageLSBs
Reserved	N_{QP_BLK} -35	

7.2.6.2 Procedures for transmission and reception of the F-pBCH0 Physical Layer channel

An F-pBCH0 channel transmission has the following payload.

Field	Length (bits)
SystemInfo block	Specified by Overhead Messages Protocol
Reserved	Variable

SystemInfo block — The transmitter shall obtain the SystemInfo block from the Overhead Messages Protocol.

Reserved — The transmitter shall set the number of reserved bits to make the transmitted packet (including the CRC added by the Physical Layer) octet aligned. The receiver shall ignore this field.

At the transmit side, every N_{pBCH0_Period} superframes, the protocol shall deliver the packet to the F-pBCH0 for transmission (where N_{pBCH0_Period} is defined in the Physical Layer). If multi-carrier mode is MultiCarrierOn, then this protocol shall deliver a separate packet for each carrier.

At the receive side, the protocol shall forward the received payload to the Overhead Messages Protocol.

7.2.6.3 Procedures for transmission and reception of the F-OSICH Physical Layer channel

The Access Network shall forward OSI value to the Physical Layer for transmission over the F-OSICH Physical Layer channel. The computation of the OSI value is beyond the scope of this specification. If multi-carrier mode is MultiCarrierOn, then this protocol shall deliver a separate OSI value for each carrier. Note that in addition to the F-OSICH, the OSI values may be transmitted on the F-SSCH.

An F-OSICH channel transmission has the following payload.

Field	Length (bits)
OSIValue	2
Reserved	6

OSIValue — The value shall indicate the level of interference at the sector. The value ‘11’ is reserved.

Reserved — The sender shall set this field to ‘000000’. The receiver shall ignore this field.

The Access Terminal shall monitor the OSICH channels of each sector in the OSIMonitorSet (see 7.7.6.4.1.6.1) and provide the received values to the Reverse Traffic Channel MAC Protocol.

7.2.6.4 Command processing

7.2.6.4.1 Activate

If this protocol receives an *Activate* command in the Inactive State, the access terminal shall transition to the Active State

If this protocol receives this command in the Active State, the command shall be ignored.

7.2.6.4.2 Deactivate

If this protocol receives a *Deactivate* command in the Inactive State, the command shall be ignored.

If this protocol receives this command in the Active State, the access terminal shall transition to the Inactive State

7.2.6.5 Inactive state

This state applies only to the access terminal.

When the protocol is in the Inactive State, the access terminal waits for an *Activate* command.

7.2.6.6 Active state

In this state, the access network transmits, and the access terminal monitors, the channels associated with the Control Channel MAC.

7.2.6.6.1 Access terminal requirements

7.2.6.6.1.1 F-pBCH0 supervision

Upon entering the active state, the access terminal shall set the F-pBCH0 supervision timer for $T_{CCMPSupervision0}$. If an F-pBCH0 MAC packet is received while the timer is active, the timer is restarted. If the timer expires, the protocol returns a *SupervisionFailed* indication and disables the timer.

7.2.6.6.1.2 F-pBCH1 supervision

Upon entering the active state, the access terminal shall set the F-pBCH1 supervision timer for $T_{CCMPSupervision1}$. If an F-pBCH1 MAC packet is received while the timer is active, the timer is restarted. If the timer expires, the protocol returns a *SupervisionFailed* indication and disables the timer.

7.2.7 Header and trailer formats

The headers and trailers for the F-pBCH0, F-pBCH1 and F-OSICH are described in the sections for procedures for transmission and reception for the respective channels (7.2.6.1, 7.2.6.2 and 7.2.6.3).

7.2.8 Interface to other protocols

7.2.8.1 Commands

This protocol does not issue any commands.

7.2.8.2 Indications

This protocol does not register to receive any indications.

7.2.9 Configuration attributes

No configuration attributes are defined for this protocol.

7.2.10 Protocol numeric constants

Constant	Meaning	Value
$N_{CCMPType}$	Type field for this protocol	Table 9
$N_{CCMPDefault}$	Subtype field for this protocol	0x0000
$T_{CCMPSupervision0}$	F-pBCH0 supervision timer value	12 x N_{pBCH0_Period} superframes
$T_{CCMPSupervision1}$	F-pBCH1 supervision timer value	12 superframes
N_{QP_BLK}	Number of bits in the QuickPagingBlock	35 bits
$N_{CCMPpBCH1Bits}$	Number of bits in the pBCH1 payload	52

7.2.11 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

7.3 Default Access Channel MAC Protocol

7.3.1 Overview

The Default Access Channel MAC Protocol provides the procedures and messages required for an access terminal to transmit, and an access network to receive, the Access Probe. An access probe may be used for initial access or handoff within an Active Set. The access network responds to an access probe with an Access Grant over the Shared Signaling MAC Protocol.

This specification assumes that the access network has one instance of this protocol for each sector in the network.

This protocol can be in one of two states at the access terminal:

- *Inactive State*: In this state, the protocol waits for an *Activate*command. This state occurs when the access terminal has not acquired an access network.

- *Active State*: In this state, the access terminal may transmit on the Access Channel.

This protocol can be in only one state at the access network.

- *Active State:* In this state, the access network monitors the Access Channel.

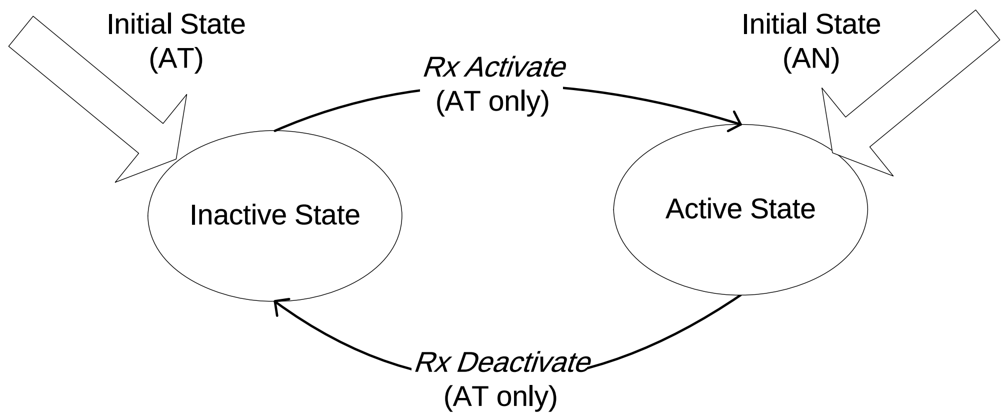

Figure 72 Default Access Channel MAC Protocol state diagram

This protocol uses the following parameters and attributes.

Parameter Name	Where Defined	Comments
AccessCycleDuration	OMP	ExtendedChannelInfo
AccessSequencePartition	OMP	ExtendedChannelInfo
MaxProbesPerSequence	OMP	ExtendedChannelInfo
ProbeRampUpStepSize	OMP	ExtendedChannelInfo
PilotThreshold1	OMP	ExtendedChannelInfo
PilotThreshold2	OMP	ExtendedChannelInfo
OpenLoopAdjust	OMP	ExtendedChannelInfo
AccessRetryPersistence ($N_{ACMPClass}$ values)	OMP	ExtendedChannelInfo
ReverseLinkSilencePeriod	OMP	ExtendedChannelInfo
ReverseLinkSilenceDuration	OMP	ExtendedChannelInfo
LoadControl	CC MAC	pBCH1
MaxProbeSequences	Configuration Attribute	
PageResponseBackoff	Configuration Attribute	
MaxProbeSequenceBackoff	Configuration Attribute	
RertyPersistenceOverride	Configuration Attribute	
TerminalAccessClass	Configuration Attribute	
RequestThreshold1	Configuration Attribute	
RequestThreshold2	Configuration Attribute	

7.3.2 Primitives

7.3.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*
- *AttemptAccess*

7.3.2.2 Return indications

This protocol returns the following indications:

- *AccessGrantReceived*
- *TransmissionAborted*
- *AccessProbeReceived*
- *AccessFailed*

7.3.3 Public data

7.3.3.1 Static public data

This protocol does not define any static public data.

7.3.3.2 Dynamic public data

- Subtype for this protocol
- AccessSequenceID
- PilotPN
- AccessCarrier
- ProbePower

7.3.4 Protocol data unit

This protocol does not carry any payload on behalf of other protocols.

7.3.5 Protocol initialization and swap

7.3.5.1 Protocol initialization

Upon initialization at the access terminal,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Inactive State.

Upon initialization at the access network,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Active State.

7.3.5.2 Protocol swap

Upon swap at the access terminal, the protocol instance shall enter the Inactive State.

Upon swap at the access network, the protocol instance shall enter the Active State.

7.3.6 Procedures

7.3.6.1 Command processing

The access network shall ignore all commands.

7.3.6.1.1 Activate

If this protocol receives an *Activate* command in the Inactive State,

- The access terminal shall transition to the Active State.
- The access network shall ignore this command.

If this protocol receives the command in the Active State, the command shall be ignored.

7.3.6.1.2 Deactivate

If this protocol receives a *Deactivate* command in the Inactive State, the command shall be ignored.

If this protocol at the access terminal receives the command in the Active State, the access terminal shall

- Immediately cease transmitting on the Access Channel if it is in the process of sending a probe.
- Return a *TransmissionAborted* indication if it was in the process of sending an access probe.
- Clear all public data.
- Transition to the Inactive State.

The access network shall ignore this command.

7.3.6.1.3 AttemptAccess

If the access terminal receives an *AttemptAccess* command, it shall invoke the procedure for processing an *AttemptAccess* command.

7.3.6.2 Access channel structure

Figure 73 illustrates the access probe structure and the access probe sequences. In the figure, Ns probe sequences are shown, where each probe sequence has Np probes. The Access Channel MAC Protocol transmits access probes by instructing the Physical Layer to transmit a probe. With the instruction, the Access Channel MAC Protocol provides the Physical Layer with a power level, AccessSequenceID, PilotPN of the sector to which the access probe is to be transmitted and an AccessCarrier field. The Physical Layer allows the transmission of access probes only once every ControlSegmentPeriod frames (see 7.6.6.3 for definition).

Each probe in a sequence is transmitted at increased power until any of the following conditions are met:

- The access terminal receives an access grant,
- Transmission is aborted because the protocol received a *Deactivate* command, or
- Maximum number of probes per sequence (MaxProbesPerSequence) has been transmitted. After a maximum number of probes has been transmitted, a new probe sequence may resume from a lower power level.

Prior to the transmission of the first probe of all probe sequences, the access terminal performs a persistence test (see section 7.3.6.4.1.3) that is used to control congestion on the Access Channel. This persistence test may return a zero value in some cases, depending on the cause for access.

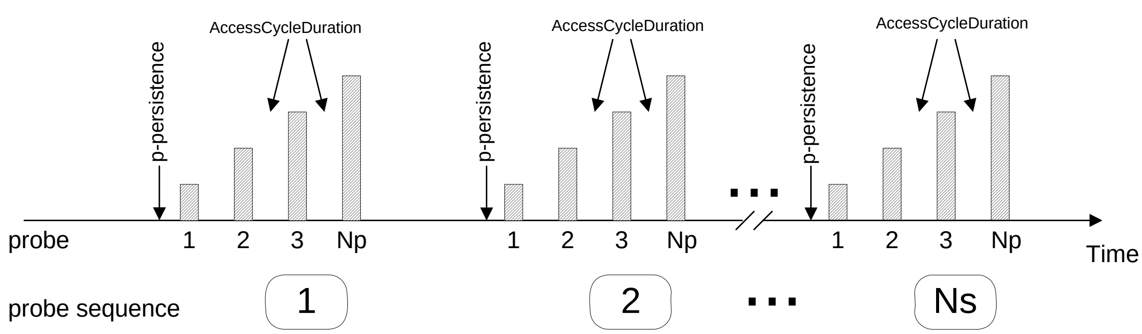

Figure 73 Access probe sequences. Ns sequences with Np probes per sequence

7.3.6.3 Inactive state

This state applies only to the access terminal.

In this state, the access terminal waits for an *Activate* command.

7.3.6.4 Active state

In this state, the access terminal is allowed to transmit on the Access Channel and the access network is monitoring the Access Channel.

7.3.6.4.1 Access terminal requirements

7.3.6.4.1.1 Procedure for processing the AttemptAccess command

The access terminal shall process only one *AttemptAccess* command at a time. *AttemptAccess* commands shall cause an existing probe transmission to end, and a new probe transmission to begin.

If any of the following events occurs while the *AttemptAccess* command is being processed, the access terminal shall invoke the procedure for ending probe transmission 7.3.6.4.1.2.

- The access terminal receives an access grant on the carrier on which the last access probe was transmitted (*SSMAC.AccessGrantReceived* indication).
- The access grant timer expires.
- The protocol receives an *IdleState.IdleHO* indication.
- The protocol receives a new *AttemptAccess* command (this command shall be processed after the current probe transmission ends).

When the procedure for processing the *AttemptAccess* command is invoked, the access terminal shall perform the following initial steps:

- Set ProbeSequenceNumber to 1.
- Set ProbeNumber to 1.
- Set TerminalAccessRetryPersistence to the TerminalAccessClass number value of the AccessRetryPersistence values. For example, if the TerminalAccessClass is 0, the field shall be set to the first AccessRetryPersistence value in the AccessParametersGroup.
- Set PilotPN to the Active Set public data field of the Idle State Protocol, or if a PilotPN was specified with the *AttemptAccess* command, set PilotPN to the specified value.
- Wait till the beginning of the next ControlSegmentPeriod (as defined in 7.6.6.3).

After performing the initial steps, the access terminal shall perform the following steps to transmit probes:

1. If ProbeSequenceNumber is greater than MaxProbeSequences, it shall set an access grant timer for $T_{ACMPANProbeTimeout}$ duration and not perform additional steps.
2. Determine the AccessSequenceID by invoking procedure 7.3.6.4.1.4**.**
3. Add the AccessSequenceID to the public data.
4. If ProbeNumber is greater than MaxProbesPerSequence it shall:
    a. Set ProbeNumber to 1.
    b. Increment ProbeSequenceNumber by 1.

   c. Determine the AccessCarrier by monitoring the LoadControl bits on the different carriers. For the remainder of the procedures, the access terminal shall use overhead parameters corresponding to the selected AccessCarrier.

   d. Add the AccessCarrier to the public data.

5. If ProbeNumber is 1, it shall determine DelayToNextProbe by invoking the procedure for determining probe sequence backoff time 7.3.6.4.1.3. Otherwise, it shall set DelayToNextProbe to the AccessCycleDuration.

6. Start a timer for DelayToNextProbe frames. After the timer is expired, the access terminal shall proceed to the next step. This timer shall be decremented only in frames that satisfy the following requirements:

   a. The ReverseLinkSilenceDuration and ReverseLinkSilencePeriod for the current sector is not active in the frame.

   b. The superframe containing that frame has the LoadControl bits transmitted on the Control Channel MAC set to a value less than or equal to the TerminalAccessClass configuration attribute.

   c. The QuickChannelInfoUpToDate and ExtendedChannelInfoUpToDate public data parameters of the Overhead Messages Protocol are both set to one.

7. Calculate the InitialProbePower using the procedures given in 7.3.6.4.1.5.

8. Transmit a probe using AccessSequenceID, PilotPN, AccessCarrier, and power calculated as:

   ProbePower = InitialAccessPower + ProbeRampUpStepSize * (ProbeNumber – 1)

9. Increment ProbeNumber and return to step 1.

7.3.6.4.1.2 Procedure for ending probe transmissions

If the access terminal receives an access grant (*SSMAC.AccessGrantReceived* indication), it shall:

- Terminate the probe transmission procedure.
- Return an *AccessGrantReceived*indication.
- Place the ProbePower (the power at which the last probe was transmitted) in the public data
- Clear the AccessSequenceID from the public data.
- Terminate this procedure.

If the access grant timer expires, it shall:

- Return an *AccessFailed* indication.
- Clear the public data.
- Terminate this procedure.

If the protocol receives an *ActiveSetManagement.IdleHO* indication, it shall:

- Return an *AccessFailed* indication.
- Terminate the probe transmission procedure.
- Clear the public data.
- Terminate this procedure.

7.3.6.4.1.3 Procedure for determining probe sequence backoff time

This procedure shall compute the persistence interval in units of ControlSegmentPeriods. If the following returns an integer value n, this procedure shall return a probe sequence backoff time of n ControlSegmentPeriods (see 7.6.6.3 for definition of ControlSegmentPeriod).

If the access attempt was made in response to a page and ProbeSequenceNumber is 1, the persistence interval shall be set as follows:

- If the access terminal detected only one QuickPage bit set in the superframe where the access terminal received the page, the persistence interval shall be set to zero.
- If the access terminal detected more than one QuickPage bit set in the superframe where the access terminal received the page, the persistence interval shall be set to a random integer drawn uniformly between 0 and PageResponseBackoff*3.

Otherwise, if the access attempt was made in response to an *OpenConnection* command from the Air Link Management Protocol at the access terminal and the ProbeSequenceNumber is 1, then the persistence interval shall be set to zero.

Otherwise, the persistence interval shall be set as follows:

- Generate a geometric random variable n with parameter p = TerminalAccessRetryPersistence, i.e., the random variable takes value n with probability p(1-p)^n.
- Set the persistence interval to the minimum of n and MaxProbeSequenceBackoff.

7.3.6.4.1.4 Procedure for determining AccessSequenceID

The sequences available for initial access shall be divided into $N_{ACMPNumPartitions}$ partitions. The access terminal shall determine the partition to be used for the access attempt based on the observed pilot power and buffer level (number of bytes in the transmit buffer). Once the partition is determined, the access terminal shall select the AccessSequenceID using a uniform probability distribution over the partition. Note that of the $N_{ACMPNumSequences}$ available for access, sequences 0, 1, 2 …, $N_{ACMPSpecialSequences}$-1 are reserved for active set operations, and sequences $N_{ACMPSpecialSequences}$ through $N_{ACMPNumSequences}$ - 1 are available for initial access.

7.3.6.4.1.4.1 Partitioning the access sequence space

The size of each partition shall be determined by the AccessSequencePartition field in the ExtendedChannelInfo block. Partition number N shall consist of AccessSequenceIDs ranging from PartitionNLower to PartitionNUpper. PartitionNLower and PartitionNUpper are determined using PartitionNSize (see Table 64 for a description of PartitionNSize) and the following algorithm.

- Partition1Lower = $N_{ACMPSpecialSequences}$
- For N between 1 and 8
  - Set PartitionNUpper = min($N_{ACMPNumSequences}$ -1 , PartitionNLower + PartitionNSize)
  - Set Partition(N+1)Lower = min($N_{ACMPNumSequences}$ -1 , PartitionNUpper + 1)
- Partition9Upper = $N_{ACMPNumSequences}$ - 1

The mapping of AccessSequencePartition to the values of PartitionNSize is contained in Table 64.

Table 64 Access sequence partitions

AccessSequence Partition	PartitionNSize (N from 1 to 9)									
	N	1	2	3	4	5	6	7	8	9
00000		0	0	0	0	0	0	0	0	remaining
00001		S2	S2	S2	S2	S2	S2	S2	S2	remaining
00010		S3	S3	S3	S1	S1	S1	S1	S1	remaining
00011		S1	S1	S1	S3	S3	S3	S1	S1	remaining
00100		S1	S1	S1	S1	S1	S1	S3	S3	remaining
00101		S3	S1	S1	S3	S1	S1	S3	S1	remaining
00110		S1	S3	S1	S1	S3	S1	S1	S3	remaining
00111		S1	S1	S3	S1	S1	S3	S1	S1	remaining
01000		S3	S3	S1	S3	S1	S1	S1	S1	remaining
01001		S1	S1	S1	S3	S3	S1	S3	S1	remaining

Where the constants S1 through S3 are interpreted as,

Table 65 Constants for interpreting the access sequence partition table

Constant	Value
S1	floor($N_{ACMPNumSequences}$/18)
S2	S1*2
S3	S1*4

7.3.6.4.1.4.2 Determining AccessSequenceID in Access State of Idle State Protocol

If the access probe transmission procedure is invoked when the Idle State Protocol is in the Access State, the Access Channel MAC Protocol shall select a partition space based on PilotLevel and RequestLevel as defined in Table 66. The access terminal shall then select an AccessSequenceID using a uniform probability distribution over sequences in the selected partition.

Table 66 Mapping the BufferLevel and PilotLevel to access sequence segment

PilotLevel	RequestLevel		
	1	**2**	**3**
1	1	2	3
2	4	5	6
3	7	8	9

The access terminal shall select its RequestLevel based on the number of bits it needs to transmit on the ReverseTrafficChannelMAC. The Request Level shall be set as follows:

- RequestLevel shall be set to 1 for RequestThreshold1 or fewer bytes.
- RequestLevel shall be set to 2 for RequestThreshold1+1 to RequestThreshold2 bytes.
- RequestLevel shall be set to 3 for more than RequestThreshold2 bytes.

To determine the PilotLevel, the access terminal shall use PilotThreshold1 and PilotThreshold2 as described below.

The access terminal shall select its PilotLevel based on the ratio (measured in dB) of the acquisition pilot power from the sector where the access attempt is being made to the total power received in the acquisition channel time slot (acquisition pilot power plus interference power).

- If this ratio is below PilotThreshold1, the PilotLevel is set to 1.
- If the ratio is not below PilotThreshold1 and is below PilotThreshold2, the PilotLevel is set to 2.
- Otherwise the PilotLevel is set to 3.
- In case the partition of the sequence space is not available (because, for example, the AccessSequencePartition field of the Access Parameters has not been read), the AccessSequenceID shall be selected uniformly from between ($N_{ACMPNumSequences}$-S1-1) and ($N_{ACMPNumSequences}$-1). From Table 64 it may be observed that the minimum size of Partition 9 is S1, and the procedure in this paragraph always selects a sequence in Partition 9.

The acquisition pilot power referred to above is the received power of the second OFDM symbol of the F-ACQCH, as described in the Physical Layer specification (OFDM symbol with index 6 from the superframe preamble). See 9.3.2.4.4. The acquisition channel time slot refers to the same symbol, i.e., the OFDM symbol with index 6 from the superframe preamble.

7.3.6.4.1.4.3 Determining AccessSequenceID outside Access State of Idle State Protocol

The following describes the access probe transmission procedure when the Idle State Protocol is not in Access State. These procedures are used for

- "hard" handoff between different sectors of a synchronous subset
- handoff between different synchronous subsets
- handoff between sectors having different frequencies
- timing and power correction for a sector

The AccessSequenceID shall depend on the identity of the handoff target sector, as follows.

The AccessSequenceID for the purpose of handoff shall be set as follows

- If the ActiveSetAssignment message has AccessSequenceIDIncluded set to 0 for the target sector, the AccessSequenceID shall be set using the PilotLevel and RequestLevel fields (described in 7.3.6.4.1.4.2) as follows:

$$\text{AccessSequenceID} = \lceil N_{\text{ACMPSpecialSequences}} / 2 \rceil + 3*(\text{PilotLevel}-1)+\text{RequestLevel}-1.$$

- If the ActiveSetAssignment message has AccessSequenceIDIncluded set to 1 for the target sector, the AccessSequenceID shall be set to the AccessSequenceID field for the target sector in the ActiveSetAssignment message.

The AccessSequenceID for the purpose of timing or power correction shall be set as follows

- The AccessSequenceID shall be set using the PilotLevel field described in 7.3.6.4.1.4.2.

$$\text{AccessSequenceID} = 3*(\text{PilotLevel}-1).$$

Note that the Physical Layer will scramble any transmission of a special access sequence (one with AccessSequenceID from 0..$N_{\text{ACMPSpecialSequences}}$-1) with the MACID of the transmitting AT in the target sector.

7.3.6.4.1.5 Procedure for determining InitialAccessPower

The parameter InitialAccessPower shall be determined based on the OpenLoopAdjust parameter and the received power of the pilot from the sector where the access attempt is being made.

$$\text{InitialAccessPower} = -\text{ MeanRxPower (dBm)} + \text{OpenLoopAdjust}$$

where the MeanRxPower shall be updated throughout the access procedure based on the received traffic channel acquisition pilot power, as measured at the receive antenna of the access terminal.

7.3.6.4.2 Access network requirements

If the access network receives an access probe, it shall generate an *AccessProbeReceived* indication, and place the AccessSequenceID in its public data.

If the protocol receives a *SharedSignalingMAC.AccessGrantSent* indication, it shall issue the following commands:

- *ReverseTrafficChannelMAC.Activate*
- *ReverseControlChannelMAC.Activate*

The access network should monitor and control the load on the R-ACH. The access network may control the load by adjusting the LoadControl bits.

7.3.7 Header formats

This protocol does not define any headers.

7.3.8 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes.

7.3.9 Interface to other protocols

7.3.9.1.1 Commands

This protocol issues the following commands (access network only):

- *ReverseTrafficChannelMAC.Activate*
- *ReverseControlChannelMAC.Activate*

7.3.9.1.2 Indications

This protocol registers to receive the following indications:

- *SharedSignalingMAC.AccessGrantSent*
- *SharedSignalingMAC.AccessGrantReceived*

7.3.10 Configuration attributes

The negotiable simple attributes for this protocol are listed in Table 67. The access terminal and the access network shall use as default the values in Table 67 listed in ***bold italics***.

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Access Channel MAC Protocol.

Table 67 Configurable values

Attribute ID	Attribute	Values	Meaning
0xff	RetryPersistenceOverride	***0xff***	The access terminal shall use the persistence probability value as specified by the appropriate AccessRetryPersistence field of the AccessParameters message.
		0x3f	The access terminal shall use zero as the persistence probability.
		0x00 to 0x3e	The access terminal shall use $2^{-n/4}$ as the persistence probability.
		All other values	Reserved
0xfe	TerminalAccessClass	***0x00***-0x03	The access class of the user. This parameter controls the AccessRetryPersistence used by the terminal.
0xfd	MaxProbeSequences	***0x03***	The maximum number of probe sequences that can be transmitted as part of one access attempt.
		0x01, 0x02, 0x04 to 0x0f	Other allowed values for this parameter.
		All other values	Reserved
0xfc	PageResponseBackoff	***0x01***	Expedited response to pages not enabled
		0x00,	Expedited response to pages enabled
		All other values	Reserved
0xfb	MaxProbeSequenceBackoff	***0x08***	Maximum backoff between probe sequences
		0x09 to 0x20	Other allowed values
		All other values	Reserved
0xfa	RequestThreshold1	***0x30***	Used to determine the RequestLevel when making an access attempt in idle state.
		Other values	Other allowed values
0xfa	RequestThreshold2	***0x80***	Used to determine the RequestLevel when making an access attempt in idle state.
		Other values	Other allowed values

7.3.11 Protocol numeric constants

Constant	Meaning	Value
$N_{ACMPType}$	Type field for this protocol	Table 9
N_{S1ACMP}	Subtype field for this protocol	0x0000
$N_{ACMPClass}$	Number of different persistence values	4
$N_{ACMPNumSequences}$	Number of available access sequences	1024
$N_{ACMPNumPartitions}$	Number of partitions of the access sequence space	9
$N_{ACMPSpecialSequences}$	Number of access sequences reserved for handoff or power and timing correction.	18
$T_{ACMPANProbeTimeout}$	Maximum time to send an acknowledgment for a probe at the access network	5 PHYFrames

7.3.12 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

7.4 Default Shared Signaling Channel MAC Protocol

7.4.1 Overview

The Default Shared Signaling MAC Protocol provides the procedures and messages required for Lower MAC Sublayer signaling. This includes access network transmissions on the F-SSCH Physical Layer channels.

This specification assumes that the access network has one instance of this protocol for each sector in the network. However, any implementation that behaves identically is compliant.

This protocol can be in one of two states:

- *Inactive State*: In this state, the protocol waits for an *Activate* command. This state applies only to the access terminal and occurs when the access terminal has not acquired an access network or is not monitoring the F-SSCH.
- *Active State*: In this state, the access network transmits and the access terminal receives the F-SSCH.

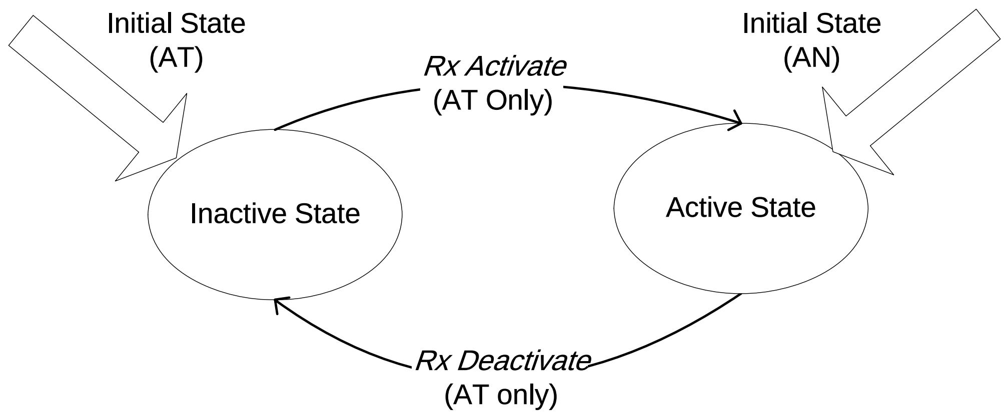

Figure 74 Default Shared Signaling Channel MAC Protocol state diagram

This protocol shall use the following parameters and attributes.

Parameter Name	Where Defined	Comments
SSCHNumHopports	OMP	QuickChannelInfo block
EffectiveNumAntennas	OMP	QuickChannelInfo block
FLFirstRestrictedSetSubband	OMP	QuickChannelInfo block
FLNumRestrictedSetSubbands	OMP	QuickChannelInfo block
FLChannelTreeIndex	OMP	QuickChannelInfo block
RLChannelTreeIndex	OMP	ExtendedChannelInfo
FLPCReportInterval	OMP	ExtendedChannelInfo
MACIDRange	OMP	ExtendedChannelInfo
RLCtrlPCMode	OMP	ExtendedChannelInfo
FastOSIEnabled	OMP	ExtendedChannelInfo
AccessCarrier	Access Channel MAC Protocol	Public data
AccessSequenceID	Access Channel MAC Protocol	Public data
FLSS	RCC MAC Protocol	Public data
DFLSS	RCC MAC Protocol	Public data
RLSS	RCC MAC Protocol	Public data
DRLSS	RCC MAC Protocol	Public data
TuneAwayStatus	Connected State Protocol	Public data
SelectedInterlaceMode	Connected State Protocol	Public data
SelectedInterlaceAssignment	Connected State Protocol	Public data
MultiCarrierOn	Physical Layer Protocol	Public data

7.4.2 Primitives

7.4.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

7.4.2.2 Return indications

This protocol returns the following indications:

- *AccessGrantSent*
- *AccessGrantReceived*
- *SupervisionFailed*

7.4.3 Public data

7.4.3.1 Static public data

This protocol does not define any static public data.

7.4.3.2 Dynamic public data

- Subtype for this protocol
- AccessGrant message most recently transmitted or received by the protocol
- ActiveCarriers: For each MACID, the public data shall store ActiveCarriers, four bits which specify the carriers in which the access terminal can operate. If the i^{th} bit for a particular access terminal is equal to '1', that indicates the access terminal can operate in the i^{th} carrier. (Note that ActiveCarriers is only relevant when multi-carrier mode is equal to MultiCarrierOn.)
- REQCarrier: For each MACID, the public data shall store REQCarrier, two bits which specify on which carrier the access terminal shall send requests. The two bits represent an integer from 0 to 3, which directly specifies the carrier index. (Note that this is only relevant when multi-carrier mode is equal to MultiCarrierOn.)

7.4.4 Protocol data unit

This protocol does not carry payload on the behalf of other protocols.

7.4.5 Protocol initialization and swap

7.4.5.1 Protocol initialization

Upon initialization at the access terminal,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Inactive State.

Upon initialization at the access network,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Active State.

7.4.5.2 Protocol swap

Upon swap at the access terminal, the protocol instance shall enter the Inactive State.

Upon swap at the access network, the protocol instance shall enter the Active State.

7.4.6 Procedures

The following sections specify procedures for transmission and reception of signals on the F-SSCH Physical Layer channel.

7.4.6.1 Command processing

The access network shall ignore all commands.

7.4.6.1.1 Activate

If this protocol receives an *Activate* command in the Inactive State, the access terminal shall transition to the Active State.

If this protocol receives this command in the Active State, the command shall be ignored.

7.4.6.1.2 Deactivate

If this protocol receives a *Deactivate* command in the Inactive State, the command shall be ignored.

If this protocol receives this command in the Active State, the access terminal shall transition to the Inactive State.

7.4.6.2 Inactive state

This state applies only to the access terminal.

When the protocol is in the Inactive State, the access terminal waits for an *Activate* command.

7.4.6.3 Active state

In this state, the access network transmits, and the access terminal monitors, the Physical Layer channels managed by this MAC protocol.

7.4.6.3.1 Access network requirements

7.4.6.3.1.1 Hop-port assignment for the F-SSCH

The F-SSCH is a physical layer channel, and is present in all FL PHY Frames.

In the following, restricted hop-ports are those belonging to restricted subbands, as specified by FLFirstRestrictedSetSubband and FLNumRestrictedSetSubbands, and those that are not in the usable hop-ports, as specified by the PHY Layer Protocol.

If multi-carrier mode is equal to MultiCarrierOff, then the F-SSCH operates over a set of hop-ports that map to a set of base nodes in the FL channel tree with index FLChannelTreeIndex. The exact set of hop-ports shall be determined based on SSCHNumHopports, as follows.

Consider the hop-ports indexed from 0 to $N_{CARRIER\_SIZE}$-1, where $N_{CARRIER\_SIZE}$ is defined in the Physical Layer Protocol. Let q be a hop-port index in this range. Let y be the least integer that is greater than or equal to $N_{CARRIER\_SIZE}$/(MinHopPortsPerNode*SSCHNumHopports), where MinHopPortsPerNode is a characteristic of the channel tree in use and is defined in 7.1.4.1, and let K be the number of hop-ports already allocated to F-SSCH by the following algorithm.

1. Set K=0, and q=0.

2. Let the base node mapped by hop-port q be specified by base node Q. The access network shall check if the following condition holds: Base node Q maps to no hop-ports that are restricted, and maps to no hop-ports that have been already allocated to the F-SSCH.

   If this condition is satisfied, the access network shall allocate all the hop-ports mapped by base node Q to F-SSCH, and shall increment K by the number of hop-ports mapped by base node Q. Proceed to step 4.

   If this condition is not satisfied, proceed to step 3:

3. Increment q by 1. If q > $N_{CARRIER\_SIZE}$-1, let q=0 and return to step 2.

4. If K>=SSCHNumHopports, all necessary hop-ports for F-SSCH have been allocated, and this algorithm shall end.

   If K < SSCHNumHopports, let q=q+y. If q > $N_{CARRIER\_SIZE}$-1, let q=0. Return to step 2.

If multi-carrier mode is equal to MultiCarrierOn, then the F-SSCH shall operate over a set of hop-ports in each carrier. The set of hop-ports in the carrier j allocated to the SSCH shall be determined using the procedure described above setting the values of FLChannelTreeIndex and SSCHNumHopports to those sent in the overhead parameters message of carrier j. Note that these values may be different for each carrier.

The Physical Layer Protocol shall be passed the set of hop-ports to be used for F-SSCH on each carrier.

7.4.6.3.1.2 Blocks for the F-SSCH

The F-SSCH can send multiple blocks. The fields of each block are defined in the descriptions of the blocks that follow the table. Exceptions are the following commonly used fields:

MACID	Sector-specific access terminal identifier.
NodeID	Field used to identify a set of hop-ports. Mapping of NodeID to hop-ports is defined in 7.5.6.6 for the forward link and in 7.7.6.7 for the reverse link.
FrameResID	Field used to resolve the interlace of applicability for AccessGrants and RLABs when duplexing mode is TDD and $N_{FL\_BURST} < N_{RL\_BURST}$.
PF	Field used to identify packet formats, and is defined in 7.5.6.7 and 7.7.6.8.

The length of the FrameResID is zero (0) bits if duplexing mode is FDD. If duplexing mode is TDD, the length of the FrameResID, N_{FRID} is

$$N_{FRID} = \begin{cases} \left\lceil \log_2\left(\frac{N_{RL\_BURST}}{N_{FL\_BURST}}\right) \right\rceil & , N_{RL\_BURST} > N_{FL\_BURST} \\ 0 & , N_{RL\_BURST} \le N_{FL\_BURST} \end{cases}$$

To resolve the RL interlace of applicability of an RLAB/AccessGrant, let $i \in [0, N_{FL\_BURST} - 1]$ indicate the FL PHY frame of arrival of the RLAB/AccessGrant within an FL burst, and let $j \in [0, N_{RL\_BURST} - 1]$ indicate the RL PHY frame within the following RL burst that is in the interlace of applicability. Then,

$$j = \left\lfloor i * \frac{N_{RL\_BURST}}{N_{FL\_BURST}} \right\rfloor + \text{FrameResID}$$

The 'Extended Transmission' field, although it may be present in the blocks as described below, is absent if the duplexing mode is TDD.

The parameter $N_{EFF\_TX\_ANT}$ is used in the tables, where

$$N_{EFF\_TX\_ANT} = \min(\text{EffectiveNumAntennas}, 4)$$

The length of the NodeID field used to identify forward link hop-ports is given by $N_{FL\_NODEID} = \lceil \log_2(NumFLNodeIDs) \rceil$, where $NumFLNodeIDs$ is the number of nodes in the forward link channel tree with index FLChannelTreeIndex in the carrier of interest. The length of the NodeID field used to identify reverse link hop-ports is given by $N_{RL\_NODEID} = \lceil \log_2(NumRLNodeIDs) \rceil$, where $NumRLNodeIDs$ is the number of nodes in the reverse link channel tree with index RLChannelTreeIndex in the carrier of interest.

The length of the MACID field is given by $N_{MACID=}$ ceil(log2($N_{CARRIER\_SIZE}$*$N_{CARRIERS}$)), where $N_{CARRIERS}$ is given in the Physical Layer Protocol.

1 **Table 68 F-SSCH blocks**

BLOCK Name	Header (binary)	Length (bits)	Fields [#bits]
AccessGrant	0000	6+ N_{MACID} + N_{RL_NODEID} + N_{FRID}	MACID [N_{MACID}] NodeID [N_{RL_NODEID}] FrameResID [N_{FRID}] TimingAdjust [6]
NS-FLAB	0001	9+ N_{MACID} + N_{FL_NODEID}	MACID [N_{MACID}] NodeID [N_{FL_NODEID}] PF [6] Duration [2] ExtendedTransmission [1]
NS-MCWFLAB1	0010	8+ N_{MACID} + N_{FL_NODEID}	MACID [N_{MACID}] NodeID [N_{FL_NODEID}] PFLayer1 [5] Duration [2] Extended Transmission [1]
NS-MCWFLAB2	0011	N_{MACID} + 4*($N_{EFF\_TX\_ANT}$-1)	MACID [N_{MACID}] PFLayer2 [4] … PFLayer$N_{EFF_TX_ANT}$ [4]
NS-SCWFLAB	0100	10+ N_{MACID} + N_{FL_NODEID}	MACID [N_{MACID}] NodeID [N_{FL_NODEID}] PF [5] NumLayers [2] Duration [2] Extended Transmission [1]
FLAB	0101	8+ N_{MACID} +N_{FL_NODEID}	MACID [N_{MACID}] NodeID [N_{FL_NODEID}] PF [6] Extended Transmission [1] Supplemental [1]
MCWFLAB1	0110	6+ N_{MACID} + N_{FL_NODEID}	MACID [N_{MACID}] NodeID [N_{FL_NODEID}] PFLayer1 [5] Extended Transmission [1] Supplemental [1]
MCWFLAB2	0111	N_{MACID} + 4*($N_{EFF\_TX\_ANT}$-1)	MACID [N_{MACID}] PFLayer2 [4] … PFLayer$N_{EFF_TX_ANT}$ [4]

BLOCK Name	Header (binary)	Length (bits)	Fields [#bits]
SCWFLAB	1000	9+ N_{MACID} + N_{FL_NODEID}	MACID [N_{MACID}] NodeID [N_{FL_NODEID}] PF [5] NumLayers [2] Extended Transmission [1] Supplemental [1]
RLAB	1001	10+ N_{MACID} + N_{RL_NODEID} + N_{FRID}	MACID [N_{MACID}] NodeID [N_{RL_NODEID}] FrameResID [N_{FRID}] PF [5] Extended Transmission [1] Supplemental [1] Delta [3]
NS-RLAB	1010	10+ N_{MACID} + N_{RL_NODEID} + N_{FRID}	MACID [N_{MACID}] NodeID [N_{RL_NODEID}] FrameResID [N_{FRID}] PF [5] Duration [1] Extended Transmission [1] Delta [3]
CCB	1011	6+ N_{MACID}	MACID [N_{MACID}] ActiveCarriersChange [4] REQCarrierChange [2]
FLAB-HO	1100	10+ N_{MACID} + N_{FL_NODEID}	MACID [N_{MACID}] NodeID [N_{FL_NODEID}] PF [6] Extended Transmission [1] DesiredSector [3]
RLAB-HO	1101	9+ N_{MACID} + N_{RL_NODEID} + N_{FRID}	MACID [N_{MACID}] NodeID [N_{RL_NODEID}] FrameResID [N_{FRID}] PF [5] Extended Transmission [1] DesiredSector [3]

7.4.6.3.1.2.1 AccessGrant

A block that is sent in response to a detected access sequence transmission that allocates a MACID to the access terminal and an initial NodeID for use by the access terminal. The Access Grant block is scrambled with a hash of the AccessSequenceID used by the access terminal during transmission of the associated access sequence. The hashing procedure is defined in the Physical Layer. A TimingAdjust field is provided to inform the access terminal of the timing offset to use for subsequent RL transmissions. The access terminal shall advance its transmission timing by the amount: offset = TimingAdjust*N_{FFT}/128 chips, where TimingAdjust is interpreted as an unsigned integer.

7.4.6.3.1.2.2 NS-FLAB

Non-sticky Forward Link Assignment Block. If the MACID in this block is not the broadcast MACID, this block informs the access terminal that holds the specific MACID that hop-ports specified by the NodeID field have been assigned to the access terminal, and informs that access terminal of the PF that should be used on its assigned hop-ports. The PF field is described in 7.5.6.7.1. The duration of the assignment is specified in the duration field, resulting in a duration of 3,6,9, or 12 PHY frames. If the MACID in this message is the broadcast MACID, then all access terminals are assigned the hop-ports specified by the NodeID field, with the given packet format.

If the extended transmission field is equal to '0', an NS-FLAB assigns hop-ports for a particular interlace consisting of standard PHY Frames, and the duration field specifies the number of standard PHY Frames to be used for transmission for this assignment. If the Extended Transmission field is equal to '1', an NS-FLAB assigns hop-ports for an interlace consisting of extended PHY Frames (6 contiguous standard PHY Frames), and the duration field specifies the number of extended PHY Frames to be used for transmission for this assignment.

7.4.6.3.1.2.3 NS-MCWFLAB

NS-MCWFLAB1	Non-sticky Multiple Code Word MIMO Forward Link Assignment Block, part one. This block informs the access terminal that holds a specific MACID that hop-ports specified by the NodeID field have been assigned to the access terminal. It also informs the access terminal, via the PFLayer1 field, which packet format should be used on the first MIMO layer. See the Physical Layer specification for the definition of MIMO layers. The packet format field (PFLayer1) is described in 7.5.6.7.2. The packet formats that should be used on the remaining layers are specified in MCWFLAB2. The duration of the assignment is specified in the duration field, resulting in a duration of 3,6,9, or 12 PHY frames (standard or extended). If the extended transmission field is equal to'0', an NS-MCWFLAB1 assigns hop-ports for a particular interlace consisting of standard PHY frames, and the duration field specifies the number of standard PHY frames to be used for transmission on this assignment. If the Extended Transmission field is equal to'1', an NS-MCWFLAB1 assigns an hop-ports for an interlace consisting of extended PHY frames, and the duration field specifies the number of extended PHY frames to be used for transmission on this assignment.
NS-MCWFLAB2	Non-sticky Multiple Code Word MIMO Forward Link Assignment Block, part two. This block informs the access terminal that holds a specific MACID of the packet formats to be used for MIMO layer 2, up through $N_{EFF\_TX\_ANT,}$ via the fields PFLayer*2* … PFLayer$N_{EFF_TX_ANT}$. The packet format field (PFLayer$_i$) is described in 7.5.6.7.2.

7.4.6.3.1.2.4 NS-SCWFLAB

Non-sticky Single Code Word MIMO Forward Link Assignment Block. This block informs the access terminal that holds a specific MACID that hop-ports specified by the NodeID field have been assigned to the access terminal. It also informs the access terminal, via the PF field, which packet format should be used. The PF field is described in 7.5.6.7.3. The NumLayers field indicates the number of MIMO layers that shall be transmitted on the assignment. See the Physical Layer specification for the definition of MIMO layers.

The duration of the assignment is specified in the duration field, resulting in a duration of 3, 6, 9, or 12 PHY frames (standard or extended).

If the extended transmission field is equal to‘0’, an NS-SCWFLAB assigns hop-ports for a particular interlace consisting of standard PHY frames, and the duration field specifies the number of standard PHY frames to be used for transmission for this assignment. If the Extended Transmission field is equal to‘1’, an NS-SCWFLAB assigns hop-ports for an interlace consisting of extended PHY frames, and the duration field specifies the number of extended PHY frames to be used for transmission for this assignment.

7.4.6.3.1.2.5 FLAB

Forward Link Assignment Block. This block informs the access terminal that holds a specific MACID that hop-ports specified by the NodeID field have been assigned to the access terminal, and informs that access terminal of the PF that should be used on its assigned hop-ports. The PF field is described in 7.5.6.7.1.

If the extended transmission field is equal to ‘0’, an FLAB assigns hop-ports for a particular interlace consisting of standard PHY Frames. If the Extended Transmission field is equal to ‘1’, an FLAB assigns hop-ports for an interlace consisting of extended PHY Frames.

The access network shall set the Supplemental field in the message to ‘1’, if the assigned hop-ports should be added to the existing access terminal assignment on the interlace to be occupied by the given assignment, and to ‘0’, if the assignment should replace any existing assignment on the interlace to be occupied by the given assignment.

Note that an existing extended transmission duration assignment should only be supplemented with another extended transmission duration assignment (i.e., an FLAB with the Extended Transmission field set to ‘1’), and an existing standard (i.e., non-extended) assignment should only be supplemented with another standard assignment (i.e., an FLAB with the Extended Transmission field set to ‘0’.)

7.4.6.3.1.2.6 MCWFLAB

MCWFLAB1: Multiple Code Word MIMO Forward Link Assignment Block , part one. This block informs the access terminal that holds a specific MACID that hop-ports specified by the NodeID field have been assigned to the access terminal. It also informs the access terminal, via the PFLayer1 field, which packet format should be used on the first MIMO layer. See the Physical Layer specification for the definition of MIMO layers. The packet format field (PFLayer1) is described in 7.5.6.7.2. The packet formats that should be used on the remaining layers are specified in MCWFLAB2.

If the extended transmission field is equal to ‘0’, a MCWFLAB1 assigns hop-ports for a particular interlace consisting of standard PHY Frames. If the Extended Transmission field is equal to ‘1’, a MCWFLAB assigns hop-ports for an interlace consisting of extended PHY Frames.

The access network shall set the Supplemental field in the message to ‘1’, if the assigned hop-ports should be added to the existing access terminal assignment on the interlace to be occupied by the given assignment, and to ‘0’, if the assignment should replace any existing assignment on the interlace to be occupied by the given assignment.

Note that an existing extended transmission duration assignment should only be supplemented with another extended transmission duration assignment (i.e., an MCWFLAB1 with the Extended Transmission field set to ‘1’), and an existing standard (i.e., non-extended) assignment should only be supplemented with another standard assignment (i.e., an MCWFLAB1 with the Extended Transmission field set to ‘0’.)

MCWFLAB2: Multiple CodeWord MIMO Forward Link Assignment Block, part two. This block informs the access terminal that holds a specific MACID of the packet formats to be used for MIMO layer 2, up through $N_{EFF\_TX\_ANT,}$ via the fields PFLayer2 … PFLayer$N_{EFF_TX_ANT}$. The packet format field (PFLayer$_i$) is described in 7.5.6.7.2.

7.4.6.3.1.2.7 SCWFLAB

Single Code Word MIMO Forward Link Assignment Block. This block informs the access terminal that holds a specific MACID that hop-ports specified by the NodeID field have been assigned to the access terminal. It also informs the access terminal, via the PF field, which packet format should be used. The PF field is described in 7.5.6.7.3. The NumLayers field indicates the number of MIMO layers that shall be transmitted on the assignment. See the Physical Layer specification for the definition of MIMO layers.

If the extended transmission field is equal to ‘0’, a SCWFLAB assigns hop-ports for a particular interlace consisting of standard PHY Frames. If the Extended Transmission field is equal to ‘1’, a SCWFLAB assigns hop-ports for an interlace consisting of extended PHY Frames.

The access network shall set the Supplemental field in the message to ‘1’, if the assignment should be added to the existing access terminal assignment on the interlace to be occupied by the given assignment, and to ‘0’, if the assignment should replace any existing assignment on the interlace to be occupied by the given assignment.

Note that an existing extended transmission duration assignment should only be supplemented with another extended transmission duration assignment (i.e., an SCWFLAB with the Extended Transmission field set to ‘1’), and an existing standard (i.e., non-extended) assignment should only be supplemented with another standard assignment (i.e., an SCWFLAB with the Extended Transmission field set to ‘0’.)

7.4.6.3.1.2.8 RLAB

Reverse Link Assignment Block. This block informs the access terminal that holds a specific MACID that hop-ports specified by the NodeID field have been assigned to the access terminal, and informs the access terminal, via the PF field, of the packet format to be used for transmission on its assignment. The PF field is described in 7.7.6.8.

If the extended transmission field is equal to‘0’ , an RLAB assigns hop-ports for a particular interlace consisting of standard PHY frames. If the Extended Transmission field is equal to‘1’, an RLAB assigns hop-ports for an interlace consisting of extended PHY frames.

The access network shall set the Supplemental field in the message to ‘1’, if the assignment should be added to the existing access terminal assignment on the interlace to be occupied by the given assignment, and to ‘0’, if the assignment should replace any existing assignment on the interlace to be occupied by the given assignment.

Note that an existing extended transmission duration assignment should only be supplemented with another extended transmission duration assignment (i.e., an RLAB with the Extended Transmission field set to ‘1’), and an existing standard (i.e., non-extended) assignment should only be supplemented with another standard assignment (i.e., an RLAB with the Extended Transmission field set to ‘0’.)

The Delta field is used by the access terminal to determine the power at which to send reverse link data, as specified by the RTC MAC.

7.4.6.3.1.2.9 NS-RLAB

Non-sticky Reverse Link Assignment Block.. This block informs the access terminal that holds a specific MACID that hop-ports specified by the NodeID field have been assigned to the access terminal, and informs the access terminal, via the PF field, of the packet format to be used for transmission on its assignment. The PF field is described in 7.7.6.8.

The duration of the assignment is specified in the duration field, resulting in a duration of 3 or 6 PHY frames (standard or extended).

If the extended transmission field is equal to‘0’, an NS-RLAB assigns hop-ports for a particular interlace consisting of standard PHY frames, and the duration field specifies the number of standard PHY frames to be used for transmission for this assignment. If the Extended Transmission field is equal to‘1’, an NS-RLAB assigns hop-ports for an interlace consisting of extended PHY frames, and the duration field specifies the number of extended PHY frames to be used for transmission for this assignment.

The Delta field is used by the access terminal to determine the power at which to send reverse link data, as specified by the RTC MAC.

7.4.6.3.1.2.10 CCB

ChangeCarrierBlock. This block only applies when multi-carrier mode is MultiCarrierOn. It notifies the access terminal that it should change the carriers on which it is operating. The ActiveCarriersChange field indicates the carriers on which the AT should now operate. This field is a four bit field. When the i^{th} bit is equal to '1', this indicates that the access terminal should operate on the i^{th} carrier. When this block is sent, ActiveCarriers, for the MACID specified in the MACID field, shall be updated to the value in the ActiveCarriersChange field.

The REQCarrierChange field notifies the access terminal that it should change the carrier on which it is sending requests to the carrier specified by the index value in this field. When this block is sent, REQCarrier, for the MACID specified in the MACID field, shall be updated to the value in the REQCarrierChange field. In addition, when this block is sent, other sectors in the access terminal's Active Set shall be notified of the updated values of the ActiveCarriers and REQCarrier for the access terminal.

7.4.6.3.1.2.11 FLAB-HO

Forward Link Assignment Block Handoff. This block informs the access terminal that holds a specific MACID that hop-ports specified by the NodeID field have been assigned to the access terminal, for the sector indicated in the DesiredSector field. (If the NodeID provided is equal to $NodeID_{DEASSIGN}$, then no resources are allocated). The DesiredSector field shall be interpreted as an active set index, and indicates that the FLSS will switch to the sector specified in this field. The PF field informs the access terminal of the packet format that should be used on its assigned hop-ports. The PF field is described in 7.5.6.7.1.

If the extended transmission field is equal to '0', an FLAB-HO assigns hop-ports for a particular interlace consisting of standard PHY Frames. If the Extended Transmission field is equal to '1', an FLAB-HO assigns hop-ports for an interlace consisting of extended PHY Frames.

7.4.6.3.1.2.12 RLAB-HO

Reverse Link Assignment Block Handoff. This block informs the access terminal that holds a specific MACID that hop-ports specified by the NodeID field have been assigned to the access terminal, for the sector indicated in the DesiredSector field. (If the NodeID provided is equal to $NodeID_{DEASSIGN}$, then no resources are allocated). The DesiredSector field shall be interpreted as an active set index, and indicates that the RLSS will switch to the sector specified in this field. The PF field informs the access terminal of the packet format to be used for transmission on its assignment. The PF field is described in 7.7.6.8.

If the extended transmission field is equal to'0', an RLAB-HO assigns hop-ports for a particular interlace consisting of standard PHY frames. If the Extended Transmission field is equal to'1', an RLAB-HO assigns hop-ports for an interlace consisting of extended PHY frames.

7.4.6.3.1.3 General rules for F-SSCH

In addition to the above blocks, the F-SSCH also transmits ACK bits for RL traffic, Power Control (PC) and CQI Erasure Indication (CEI) bits, and Fast OSI value.

Hop-ports allocated from different F-SSCH blocks constitute assignments of different types. Note however, that MCWFLAB1 and MCWFLAB2 allocate hop-ports for the same assignment type. Similarly, NS-MCWFLAB1 and NS-MCWFLAB2 allocate hop-ports for the same assignment type.

The access network should not send an assignment block to an access terminal with the supplemental bit set to '1' for an interlace where the access terminal has no assignment of the same type on that interlace, as such blocks will be ignored.

The access network should not send an assignment to an access terminal for an interlace, with the supplemental bit set to '1' which does not change the access terminal's assignment for that interlace; such blocks will be ignored. (See the FTC-MAC protocol and RTC-MAC protocol for assignment management rules.)

If the access network wants to assign more hop-ports (for an assignment of a single type) to an access terminal via F-SSCH on a single PHY Frame than can be specified by a single NodeID, the access network should send multiple Link Assignment Blocks of the same type to the access terminal's MACID. Except for the NodeID field, all of the fields in these Link Assignment Blocks should be identical. In the case of MCWFLABs, only 1 MCWFLAB2 need be sent, since this information is common to all the MCWFLAB1s that are sent to a particular user in the same PHY Frame. Similarly, for the case of NS-MCWFLABs, only 1 NS-MCWFLAB2 need be sent, since this information is common to all the NS-MCWFLAB1s that are sent to a particular user in the same PHY frame.

The access network should not transmit any F-SSCH blocks, ACK bits, or power control bits intended for an access terminal while TuneAwayStatus is equal to '1'. NS-FLABs with broadcast MACIDs may still be sent since these blocks are not directed to a particular access terminal.

If the access terminal has SelectedInterlaceMode set to '1', and SelectedInterlaceAssignment contains the PilotPN of the current sector, then the access network should only send blocks on F-SSCH to this access terminal over the interlaces specified in SelectedInterlaceAssignment. Additionally, the access network should only send assignment blocks to this access terminal if the Extended Transmission field is equal to '0'. NS-FLABs with broadcast MACIDs may still be sent on the restricted interlaces, and may have the Extended Transmission field set to '1', since these blocks are not directed to a particular access terminal.

If multi-carrier mode is MultiCarrierOn, the access network should not send any F-SSCH blocks, ACKs, or power control bits over F-SSCH intended for a particular MACID on carriers that are not marked as active carriers in the ActiveCarriers parameter of the public data for that MACID.

The AN should not send an RLAB-HO with the DesiredSector field set to a sector which is outside the synchronous subset to which the current RLSS belongs, unless the NodeID field of the RLAB-HO is $NodeID_{DEASSIGN}$. Similarly, the AN should not send an FLAB-HO with the DesiredSector field set to a sector which is outside the synchronous subset to which the current FLSS belongs, unless the NodeID field of the FLAB-HO is $NodeID_{DEASSIGN}$..

If the AN sends an access grant over SSCH, it should also send assignment block/blocks to deassign/decrement the assignments of any access terminal/terminals which would otherwise be using the hop-ports being assigned by the access grant.

If the AN sends an RLAB-HO over SSCH, it should ensure that assignment block/blocks are sent from the sector specified in the DesiredSector field of the RLAB-HO, which would deassign/decrement the assignments of any access terminal/terminals which would otherwise be using the hop-ports being assigned by the RLAB-HO. Similarly, If the AN sends an FLAB-HO over SSCH, it should ensure that assignment block/blocks are sent from the sector specified in the DesiredSector field of the FLAB-HO, which would deassign/decrement the assignments of any access terminal/terminals which would otherwise be using the hop-ports being assigned by the FLAB-HO.

7.4.6.3.1.4 Framing F-SSCH blocks

The framing of blocks in SSCH packets follows a specific format as defined in 7.4.6.3.1.2. An SSCH packet consists of a 4-bit header followed by the block, followed by reserved bits which are used to make all the SSCH packets the same size. The PHY shall add the CRC to each packet.

Table 69 F-SSCH Block Structure

Field	Length (bits)
Header	4
Block	variable
Reserved	variable

Header — A field that identifies the subsequent Block fields. See table in 7.4.6.3.1.2 for Header values and the respective blocks and block lengths.

Block — See table in 7.4.6.3.1.2.

Reserved — The access network shall set this field to zero. The access terminal shall ignore this field. The length of this field shall be such that the total number of bits in the SSCH packets is $10+N_{MACID}+N_{FL\_NODEID}+N_{FRID}+$ 4 header bits.

For each PHY Frame, the access network shall decide the specific blocks to be sent on SSCH and shall compute the power density at which each SSCH block is transmitted, for use by the Physical Layer. The framed blocks shall be passed to the Physical Layer in order of decreasing transmit power density (blocks with higher transmit power density passed first). Algorithms for choosing which blocks to be sent, and computing the transmit power density for each block are beyond the scope of this specification.

7.4.6.3.1.5 Procedures for sending an access grant

The access network should send an access grant message when it receives an *AccessChannelMAC.AccessProbeReceived* indication.

If the AccessSequenceID placed in the public data of the AccessChannelMAC is the $0,\ldots N_{ACMPSpecialSequences}-1$ AccessSequenceID, then the AccessGrant shall contain the MACID that was used to generate the Physical Layer sequence corresponding to the received access probe, and may contain $NodeID_{DEASSIGN}$. Refer to the Access Channel MAC Protocol for the procedures for determining AccessSequenceID.

If the AccessSequenceID placed in the public data of the AccessChannelMAC is higher than $N_{ACMPSpecialSequences}$-1, then the AccessGrant shall contain a MAC ID that is not in use in the system. Based on the space to which the received AccessSequenceID corresponds, the access network may decide the NodeID in the access grant. When the access grant is sent, the protocol shall generate a *SharedSignalingMAC.AccessGrantSent* indication.

7.4.6.3.1.6 Procedures for the power control bits

For each access terminal that contains the sector in the active set, and whose MACID is within MACIDRange, the access network shall transmit one CQI Erasure Indication (CEI) bit on F-SSCH every FLPCReportInterval frames. The access network shall determine the value of the CEI bits for all MACIDs within MACIDRange, and the power at which each is to be transmitted, for use by the Physical Layer. The MAC Layer Protocol shall pass the CEI bits in order of increasing MACID to the Physical Layer Protocol.

In addition to the CEI bits, if RLCtrlPCMode is set to ‘UpDown’, the access network shall transmit one Power Control (PC) bit on F-SSCH every FLPCReportInterval frames, for each access terminal whose RLSS is this sector, and whose MACID is within the MACIDRange. The access network shall determine the value of the PC bits for all MACIDs within MACIDRange, and the power at which each is to be transmitted, for use by the Physical Layer. The MAC Layer Protocol shall pass the PC bits in order of increasing MACID to the Physical Layer Protocol.

The algorithms for determining the values of the CEI and PC bits and the power at which they are to be transmitted are outside the scope of this specification.

7.4.6.3.1.7 Procedures for the Fast OSI value

If FastOSIEnabled is set to ‘1’, then, in addition to the F-OSICH, the access network shall transmit the OSI value on the F-SSCH, every PHY frame. The computation of the OSI value is beyond the scope of this specification. If multi-carrier mode is MultiCarrierOn, then this protocol shall transmit a separate value for each carrier.

7.4.6.3.1.8 Procedures for sending ACKs

Acknowledgements for RL traffic are sent on F-SSCH. The timing for sending ACKs is given in 7.1.3.

The access terminal’s ACKNode for a particular assignment on a particular interlace shall be the base node mapped by the hop-port with the lowest numerical index in the access terminal’s assignment on the carrier which has the most hop-ports for that assignment. If more than one carrier meets this criteria, then the carrier in this subset that has the smallest carrier index shall be the carrier on which the ACKNode is chosen.

Each PHY frame, the access network shall transmit ACKs intended for certain access terminals. In order to positively acknowledge a particular access terminal, the Access network shall set the bit corresponding to access terminal’s ACKNode to ‘1’, and shall set the bits corresponding to the rest of the base nodes in the access terminal’s assignment to ‘0’. After the bits corresponding to the base nodes for the users receiving acknowledgements have been set in this way, the bits corresponding to all other base nodes shall be set to '0'. The MAC Layer Protocol shall pass to the Physical layer protocol the sequence of bits corresponding to all base nodes: the base nodes for each carrier shall be ordered from left to right, and the carriers shall be ordered via increasing carrier index. The access network shall also determine the power used to transmit each of these bits, for use by the Physical Layer. Algorithms for determining this power are outside the scope of this specification. In addition, for every positive acknowledgement, the MAC shall pass to the Physical layer a valid MACID corresponding to the acknowledged AT which is needed for MACID scrambling.

7.4.6.3.1.9 Procedures for maintaining ActiveCarriers and REQCarriers at the AN

The access network shall initialize ActiveCarriers for a particular MACID to be the carrier specified by AccessCarrier.

When the access network sends a CCB block in the F-SSCH to a particular access terminal, ActiveCarriers (for the MACID specified in the MACID field of the CCB) shall be updated to the value in the ActiveCarriersChange field of the CCB. The value of ActiveCarriers for a particular access terminal may also change if another sector indicates the necessary change.

The access network shall initialize REQCarrier for a particular MACID to be the carrier specified by AccessCarrier.

When the access network sends a CCB block in the F-SSCH to a particular access terminal, REQCarrier (for the MACID specified in the MACID field of the CCB) shall be updated to the value in the REQCarrierChange field of the CCB. The value of REQCarrier for a particular access terminal may also change if another sector indicates the necessary change.

7.4.6.3.2 Access terminal requirements

When this protocol is in active state, the following apply:

If the access terminal has an outstanding access probe and the AccessSequenceID is $N_{ACMPSpecialSequences}/2$ through $N_{ACMPSpecialSequences}-1$, then the access terminal shall, at least, demodulate all F-SSCH packets, as well as the relevant ACK and power control bits, in the F-SSCH of the access terminal's DFLSS each PHY Frame. If the access terminal has no outstanding access probe with AccessSequenceID $N_{ACMPSpecialSequences}/2$ through $N_{ACMPSpecialSequences}-1$, then the access terminal shall demodulate all F-SSCH packets, as well as the relevant ACK and power control bits, in the F-SSCHs of its FLSS, DFLSS, RLSS, and DRLSS (see 7.6.6.3 for definitions of FLSS, DFLSS, RLSS, DFLSS) every PHY Frame. An access terminal that has a DFLSS in a different synchronous subset than its FLSS, and is not capable of demodulating the F-SSCH in both its DFLSS and its FLSS, should immediately issue an access probe with AccessSequenceID $N_{ACMPSpecialSequences}/2$ through $N_{ACMPSpecialSequences}-1$.

If multi-carrier mode equals MultiCarrierOn, the access terminal shall demodulate F-SSCH packets, ACKs and power control bits, on those carriers specified by its ActiveCarriers parameter in the public data.

If the access terminal has its parameter TuneAwayStatus in the public data of the Connected State Protocol set to '1', the access terminal need not monitor any F-SSCH.

The following are potential error events and specified actions:

- Each block that does not correctly pass CRC shall be discarded.

The access terminal shall perform the following actions on the remaining blocks/bits:

- If the access terminal's ACK bit has an ACK value of 1, then the access terminal has received a positive ACK signal for its assignment on the corresponding interlace. Otherwise, the access terminal has received an implicit negative ACK signal for the assignment on the corresponding interlace. Actions contingent upon the reception of a positive or negative acknowledgement shall be performed as specified in the RTC MAC Protocol.
- The access terminal shall determine its transmit power based on its power control bits received on F-SSCH as specified in the RCC MAC Protocol.
- For each sector in the active set, if FastOSIEnabled is set to '1', the access terminal shall monitor the Fast OSI value and provide the received values to the Reverse Traffic Channel MAC Protocol.
- For each sector in the active set, if an AccessGrant block is received, the access terminal shall:
  - If all the following are true
    - the access terminal has no assigned MACID on the sector,
    - and the AccessSequenceID used to scramble the AccessGrant matches the AccessSequenceID in the public data of the Access Channel MAC Protocol,
    - and the PilotPN of the sector that transmitted the AccessGrant matches the PilotPN in the public data of the Access Channel MAC Protocol,
    - and if multi-carrier mode is set to MultiCarrierOn, the carrier on which the AccessGrant is received matches the AccessCarrier in the public data of the Access Channel MAC Protocol,

    the access terminal shall:
    - Return an *AccessGrantReceived* indication specifying the applicable sector.
    - Place the received AccessGrant in the public data.
    - In the Physical Layer, adjust the timing for the sector that transmitted the access grant.
    - Instruct the RCC MAC to initialize the transmit power as described in 7.6.6.3.5.
  - If all of the following are true:
    - the MAC ID in the AccessGrant matches a MACID assigned to the access terminal on the sector,
    - the AccessSequenceID used to scramble AccessGrant matches the AccessSequenceID in the public data of the Access Channel MAC Protocol,
    - the PilotPN of the sector that transmitted the AccessGrant matches the PilotPN in the public data of the Access Channel MAC Protocol
    - and if multi-carrier mode is set to MultiCarrierOn, the carrier on which the AccessGrant is received matches the AccessCarrier in the public data of the Access Channel MAC Protocol,

the access terminal shall:

- Return an *AccessGrantReceived* indication specifying the applicable sector.
- Place the received AccessGrant in the public data.
- In the Physical Layer, adjust the timing for the sector that transmitted the access grant.
- Instruct the RCC MAC to initialize the transmit power as described in 7.6.6.3.5.

❐ The access terminal shall invoke the procedures for processing an access grant in the access terminal Assignment Management section of the RTC MAC Protocol, unless the NodeID field is equal to $NodeID_{DEASSIGN,}$ in which case the access grant will not be passed to the RTC MAC Protocol for further processing.

- If an FLAB is received, the access terminal shall invoke the procedures for processing an FLAB in the access terminal Assignment Management section of the FTC MAC Protocol.
- If a NS-FLAB is received, the access terminal shall invoke the procedures for processing a NS-FLAB in the access terminal Assignment Management section of the FTC MAC Protocol.
- If an RLAB is received, the access terminal shall invoke the procedures for processing an RLAB in the access terminal Assignment Management sections of the RTC MAC Protocol.
- If an NS-RLAB is received, the access terminal shall invoke the procedures for processing an NS-RLAB in the access terminal Assignment Management sections of the RTC MAC Protocol.
- If an MCWFLAB is received (part 1 or part 2), the access terminal shall invoke the procedures for processing a MCWFLAB in the access terminal Assignment Management section of the FTC MAC Protocol.
- If an NS-MCWFLAB is received (part 1 or part 2), the access terminal shall invoke the procedures for processing a NS-MCWFLAB in the access terminal Assignment Management section of the FTC MAC Protocol.
- If a SCWFLAB is received, the access terminal shall invoke the procedures for processing an SCWFLAB in the access terminal Assignment Management section of the FTC MAC Protocol.
- If a NS-SCWFLAB is received, the access terminal shall invoke the procedures for processing an NS-SCWFLAB in the access terminal Assignment Management section of the FTC MAC Protocol.
- If an FLAB-HO is received, the access terminal shall invoke the procedures for processing an FLAB-HO in the access terminal Assignment Management section of the FTC MAC Protocol.
- If an RLAB-HO is received, the access terminal shall invoke the procedures for processing an RLAB-HO in the access terminal Assignment Management section of the RTC MAC Protocol.

7.4.6.3.2.1 Procedures for maintaining ActiveCarriers and REQCarrier at the AT

The access terminal shall initialize ActiveCarriers to be the carrier specified by AccessCarrier.

The access terminal shall initialize REQCarrier to be the carrier specified by AccessCarrier l.

If multi-carrier mode is equal to MultiCarrierOff, then the access terminal shall ignore all CCB blocks sent on F-SSCH.

If multi-carrier mode is equal to MultiCarrierOn, then the following apply:

When the access terminal receives a CCB block in the F-SSCH, the access terminal shall update ActiveCarriers to the value in the ActiveCarriersChange field of the CCB.

When the access terminal receives a CCB block in the F-SSCH, REQCarrier shall be updated to the value in the REQCarrierChange field of the CCB.

7.4.6.3.3 Supervision Procedures

The access terminal shall return a *SupervisionFailed* indication if the F-SSCH channel is not received for a time period of length $T_{SSCMSupervision}$.

7.4.7 Header and trailer formats

This protocol does not carry higher-layer payload, and thus header and trailer formats are not defined.

7.4.8 Interface to other protocols

7.4.8.1.1 Commands

This protocol does not issue any commands.

7.4.8.1.2 Indications

This protocol registers to receive the following indication:

- *AccessChannelMAC.AccessProbeReceived*

7.4.9 Configuration attributes

No configuration attributes are defined for this protocol.

7.4.10 Protocol numeric constants

Constant	Meaning	Value
$N_{SSCMPType}$	Type field for this protocol	Table 9
$N_{SSCMPDefault}$	Subtype field for this protocol	0x0000
$MACID_{BROADCAST}$	Broadcast MACID	0x0
$T_{SSCMSupervision}$	Supervision timer	1 s

7.4.11 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

7.5 Default Forward Traffic Channel MAC Protocol

7.5.1 Overview

The Default Forward Traffic Channel MAC Protocol provides the procedures and messages required for an access network to transmit, and an access terminal to receive, the Forward Traffic Channel.

The access network maintains an instance of this protocol for every assigned MAC ID.

This protocol at the access terminal operates in one of two states:

- *Inactive State*: In this state, the access terminal cannot receive any packets on the Forward Traffic Channel. When the protocol is in this state, it waits for an *Activate* command.
- *Active State*: In this state, the access terminal receives the forward traffic channel based on link assignment blocks received from SS MAC.

The protocol states and allowed transitions between the states are shown in Figure 75. The rules governing these transitions are provided in 7.5.6.3 and 7.5.6.4.

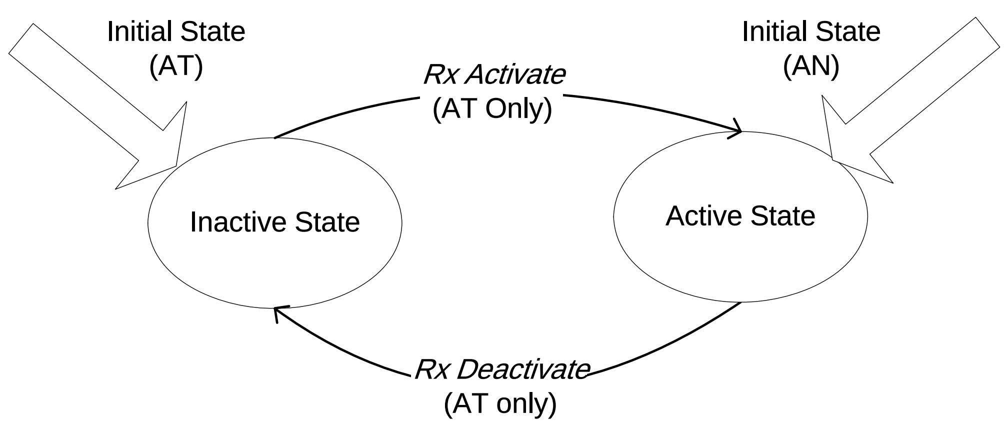

Figure 75 Default Forward Traffic Channel MAC Protocol state diagram

This protocol shall use the following parameters and attributes:

Parameter Name	Where Defined	Comments
FLChannelTreeIndex	OMP	QuickChannelInfo block
FLNumSDMADimensions	OMP	QuickChannelInfo block
FLImplicitDeassignEnabled	Connected State Protocol	Public data
SelectedInterlaceMode	Connected State Protocol	Public data
TuneAwayStatus	Connected State Protocol	Public data
FLSS	RCC MAC Protocol	Public data
DFLSS	RCC MAC Protocol	Public data
MultiCarrierOn	Physical Layer Protocol	Public data

7.5.2 Primitives

7.5.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

7.5.2.2 Return indications

This protocol returns the following indication:

- *ForwardTrafficPacketsMissed*
- *SupervisionFailed*
- *PageReceived*
- *UATIReceived*
- *SessionLost*

7.5.3 Public data

7.5.3.1 Static public data

This protocol does not define any static public data.

7.5.3.2 Dynamic public data

- Subtype for this protocol.

7.5.4 Protocol data unit

The transmission unit of this protocol is a Forward Traffic Channel Lower MAC Sublayer packet. Each packet consists of one Security Sublayer packet.

7.5.5 Protocol initialization and swap

7.5.5.1 Protocol initialization

Upon initialization at the access terminal,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Inactive State.

Upon initialization at the access network.

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Active State.

7.5.5.2 Protocol Swap

Upon swap, the protocol instance shall enter the Inactive State.

7.5.6 Procedures

This protocol constructs a Forward Traffic Channel Lower MAC Sublayer packet out of the Security Sublayer packet by adding the Lower MAC Sublayer header and trailer as defined in 7.5.7. The protocol then sends the packet for transmission to the Physical Layer.

7.5.6.1 Command processing

7.5.6.1.1 Activate

If this protocol receives an *Activate* command in the Inactive State, the access terminal and the access network shall transition to the Active State.

If this protocol receives the command in any other state, the command shall be ignored.

7.5.6.1.2 Deactivate

If the protocol receives a *Deactivate* command in the Active State:

- The access terminal shall cease monitoring the Forward Traffic Channel and shall transition to the Inactive State.
- The access network shall ignore the command.

If this command is received in the Inactive State, the command shall be ignored.

7.5.6.2 Forward traffic channel addressing

Transmission on the Forward Traffic Channel is multiplexed in time and frequency. An assignment on the Forward Traffic Channel shall be specified by a set of hop-ports and an interlace. Each hop-port is specified by a hop-port index as well as a carrier index. If the duplex mode is FDD, then the interlace may be composed of standard PHY Frames or extended PHY Frames, (as specified by the assignment blocks received from the SS MAC protocol). Extended PHY Frames are defined in 7.1.3.1.3.

The duration of an assignment of hop-ports may or may not be pre-specified. Assignments whose durations are pre-specified are known as non-sticky assignments, and assignments whose durations are not pre-specified are known as sticky assignments.

The set of hop-ports assigned for a particular interlace for a particular access terminal via sticky assignment blocks (received from SS MAC) is referred to as a "Forward Link Access Terminal Assignment" or FL-ATA. Sticky assignment blocks include the following three types: FLABs (and FLAB-HOs), SCWFLABs, and MCWFLABs. (See 7.4.6.3.1 for complete definitions of these block types.) A FL-ATA for a particular interlace may only be assigned via a single type of sticky assignment block. A FL-ATA that has been assigned via a particular type of assignment block is also associated with that type. More precisely, a FL-ATA may be of type FLAB, SCWFLAB, or MCWFLAB. The access terminal may have multiple FL-ATAs, one for each nonoverlapping interlace (note that overlapping interlaces can be created only by the use of extended PHY Frames).

A non-sticky assignment block with a unicast MACID field (referred to as a non-sticky unicast assignment block) assigns hop-ports for a particular access terminal for a particular interlace. The set of hop-ports assigned for a particular access terminal for a particular interlace via non-sticky unicast assignment blocks (received from SS MAC) is referred to as a "Forward Link Non-Sticky Access Terminal Unicast Assignment" or FL-NS-ATA_{UC}. Non-sticky unicast assignment blocks include the following three types: NS-FLABs, NS-SCWFLABs, and NS-MCWFLABs (all with unicast MACID fields). See 7.4.6.3.1 for complete definitions of these block types. A FL-NS-ATA_{UC} for a particular interlace may only be assigned via a single type of non-sticky unicast assignment block. A FL-NS-ATA_{UC} that has been assigned via a particular type of assignment block is also associated with that type. More precisely , a FL-NS-ATA_{UC} may be of type NS-FLAB, NS-SCWFLAB, or NS-MCWFLAB. An access terminal may have multiple FL-NS-ATA_{UC}s, one for each nonoverlapping interlace.

A non-sticky assignment block with a broadcast MACID field (referred to as a non-sticky broadcast assignment block) assigns hop-ports for multiple access terminals. The set of hop-ports used by a particular access terminal on a particular interlace, assigned via non-sticky broadcast assignment blocks is referred to as a "Forward Link Non Sticky Access Terminal Broadcast Assignment" or FL-NS-ATA_{BC}. Non-sticky broadcast assignment blocks include only NS-FLABs. An access terminal may have multiple FL-NS-ATA_{BC}s, one for each nonoverlapping interlace on each active carrier.

Sets of hop-ports assigned in assignment blocks received from SS MAC are specified using the channel tree that is currently in use for the FL. The channel tree that is in use is specified by FLChannelTreeIndex. Channel trees are specified in 7.5.6.6.

Packets transmitted over the Forward Traffic Channel are transmitted over the F-DCH physical layer channel. F-DCH is the primary channel for user data transmission, and access terminals are assigned F-DCH resources (FL-ATAs/-FL-NS-ATAs) via assignment blocks that are sent over the F-SSCH.

The following rules apply regarding the coexistence of FL-ATAs/FL-NS-ATA$_{BC}$s/FL-NS-ATA$_{UC}$s:

- An access terminal shall have no more than one FL-ATA per interlace. If duplex mode is FDD, then additionally, an access terminal shall not have any FL-ATAs that overlap in time.
- An access terminal shall have no more than one FL-NS-ATA$_{UC}$ per interlace. If duplex mode is FDD, then additionally, an access terminal shall not have any FL-NS-ATA$_{UC}$ s that overlap in time.
- An access terminal shall have no more than one FL-NS-ATA$_{BC}$ per interlace per carrier. If duplex mode is FDD, then additionally, an access terminal shall not have any FL-NS-ATA$_{BC}$ s that overlap in time on the same carrier.
- An access terminal shall not have an a non-empty FL-ATA and a non-empty FL-NS-ATA$_{UC}$ in the same interlace. If duplex mode is FDD, then additionally, an access terminal shall not have a non-empty FL-ATA and a non-empty FL-NS-ATA$_{UC}$ that overlap in time.

The FL-ATA for an interlace can be accumulated via multiple assignment blocks of the same type, sent over multiple PHY Frames, as specified in 7.5.6.4.1.1. The FL-NS-ATA$_{UC}$ for an interlace may be accumulated over multiple assignment blocks of the same type, but these blocks must be sent in the same PHY Frame, as specified in 7.5.6.4.1.1.1. Similarly, The FL-NS-ATA$_{BC}$ for an interlace may be accumulated over multiple assignment blocks of the same type, but these blocks must be sent in the same PHY frame, as specified in 7.5.6.4.1.1.1. All hop-ports in an FL-ATA/FL-NS-ATA$_{UC}$/ FL-NS-ATA$_{BC}$ for a particular interlace shall be combined for transmission over the Physical Layer channel (F-DCH). Different interlaces shall always carry separate MAC packets with independent H-ARQ termination. An example is illustrated in Figure 76.

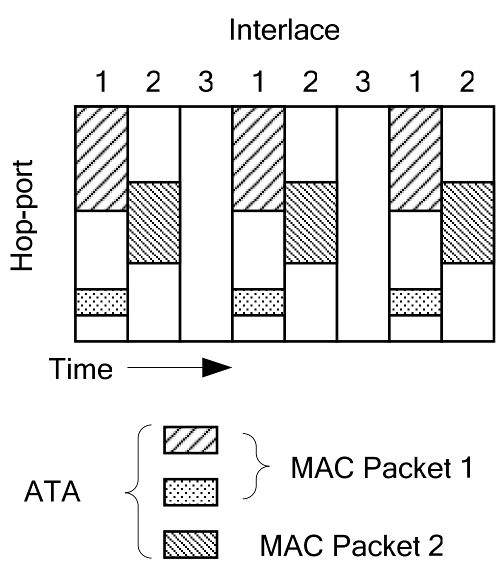

Figure 76 F-DCH addressing example

7.5.6.3 Inactive state

When the protocol is in the Inactive State, the access terminal waits for an *Activate* command.

The Inactive State is not defined for the Access Network.

7.5.6.4 Active state

In the Active State, the access network shall transmit over the F-DCH using the FL-ATAs or FL-NS-ATAs, and PFs selected by the access network and signaled to the access terminal over the F-SSCH Physical Layer channel. The access terminal processes assignment blocks from the SS MAC protocol to maintain its FL-ATAs/FL-NS-ATAs, configures the Physical Layer for reception of packets according to the ATAs, and controls transmission of ACK/NACK information via the RCC MAC Protocol based on pass or fail of the MAC packet, as indicated by the PHY. The MACID assigned to the access terminal for each sector in its active set shall be given in the ActiveSetAssignment message that is public data of the Active Set Management Protocol. Assignment and H-ARQ logic for the access terminal and access network are specified in 7.5.6.4.1 and 7.5.6.4.2, respectively.

Note that an FL-NS-ATA shall be used for transmission of a single Lower MAC packet (see 7.1.2).

7.5.6.4.1 Access terminal requirements

An access terminal shall keep a variable NumLayers for each interlace, which shall be initialized to zero when this protocol enters the Active state. On reception of a SCWFLAB/NS-SCWFLAB for a particular interlace from the SS MAC protocol that is not discarded due to assignment management rules given in 7.5.6.4.1.2, NumLayers for that interlace shall be set to the NumLayers field of the SCWFLAB/NS-SCWFLAB. On reception of a MCWFLAB/NS-MCWFLAB for a particular interlace from the SS MAC protocol that is not discarded due to assignment management rules given in 7.5.6.4.1.2, NumLayers for that interlace shall be set to the number of non-zero packet formats given in the MCWFLAB/NS-MCWFLAB. NumLayers for a particular interlace shall also be modified to reflect rank adjustments sent in the FL MAC header.

7.5.6.4.1.1 Access terminal assignment management for sticky assignments

In this section, the term FLAB will be used to indicate all types of sticky FL assignment blocks, including FLAB, MCWFLAB, and SCWFLAB, unless otherwise specified. (FLAB-HOs are discussed in 7.5.6.4.1.3) Similarly, the term NS-FLAB will be used to indicate all types of non-sticky FL assignment blocks, including NS-FLAB, NS-MCWFLAB, and NS-SCWFLAB, unless otherwise specified.

The access terminal shall maintain and manage its FL-ATAs by monitoring FLABs and NS-FLABs delivered from the SS MAC protocol.

The logic in this section assumes that all of the FLABs/NS-FLABs are being sent from the same serving sector, the FLSS (see 7.6.6.3 for the definition of FLSS). The logic for access terminal assignment management during handoff is found in 7.5.6.4.1.3.

If FLImplicitDeassignEnabled is equal to ‘1’, then upon receiving a *TunedAway* indication from the Connected State Protocol, the access terminal shall expire all its FL-ATAs.

If SelectedInterlaceMode is equal to '1', then the access terminal shall ignore all FLABs that have the Extended Transmission field set to '1'.

If the extended transmission field is equal to '0', an FLAB assigns an interlace consisting of standard PHY Frames, as shown in 7.1.3.1.2. If the Extended Transmission field is equal to '1', an FLAB assigns an interlace consisting of extended PHY Frames , as shown in 7.1.3.1.3.

7.5.6.4.1.1.1 Simultaneous assignments

If duplex mode is FDD, FLABs received in the same PHY frame could be for different interlaces because some of the FLABs could be for the interlace consisting of standard PHY frames, and some of the FLABs could be for the interlace consisting of extended PHY frames. If multiple FLABs for the access terminal's MACID are received in the same PHY frame from SS MAC, and they are not all for the same interlace, then the access terminal shall assume an error has occurred, and shall ignore all of these FLABs, unless there is an FLAB with NodeID set to $\text{NodeID}_{\text{DEASSIGN}}$, in which case this FLAB shall not be ignored.

If multiple FLABs for the access terminal's MACID are received from the SS MAC Protocol in the same PHY frame, for the same interlace, and one of the FLABs has a NodeIDs set to $\text{NodeID}_{\text{DEASSIGN}}$, then all FLABs except for the latter shall be discarded. This rule trumps all those which follow in this section.

If multiple FLABs of the same type, which have the access terminal's MACID, are received from the SS MAC protocol in the same PHY Frame, for the same interlace, and all the FLABs have the same values in all the fields except the NodeID field, then the access terminal shall treat these FLABs as a single FLAB assigning the union of the hop-ports mapped by the constituent NodeIDs.

If multiple FLABs of the same type, which have the access terminal's MACID, are received from the SS MAC protocol in the same PHY Frame, for the same interlace, and if the values in at least one of the fields (excluding the NodeID field) are not the same, then the access terminal shall assume an error has occurred, and shall ignore all of these FLABs.

If multiple FLABs, which have the access terminal's MACID, are received from the SS MAC protocol in the same PHY frame, for the same interlace, and all of the FLABs are not of the same type, the access terminal shall assume an error has occurred, and shall ignore all of these FLABs.

7.5.6.4.1.1.2 Supplemental and non-supplemental assignments

The following assumes that an FLAB/FLABs have been received from the SS MAC with the MACID matching that of the access terminal.

If the Supplemental field of an FLAB for a particular interlace is equal to '1', then the new FL-ATA on that interlace shall be the union of hop-ports included in the old FL-ATA on that interlace and the hop-ports specified by the new NodeID, provided the old FL-ATA and the new FLAB are of the same type, and the old FL-ATA is non-empty. The PF specified in the received FLAB shall be used in place of any PFs that may have been specified in any previous assignment of NodeIDs (hop-ports) on the interlace.

If the Supplemental field of an FLAB for a particular interlace is equal to '0', then the FL-ATA for the relevant interlace shall be cleared before adding the hop-ports specified by the NodeID in the FLAB to the FL-ATA for the interlace.

If the duplex mode is FDD, and the supplemental field of an FLAB for a particular interlace is equal to ‘0’, then the following apply. If the extended transmission field of the FLAB is equal to ‘1’, all FL-ATAs shall be expired except for the FL-ATA, should there be one, which is an extended transmission duration assignment and does not overlap in time with the new assignment. The hop-ports specified by the NodeID in the FLAB shall then be given to the FL-ATA in the corresponding interlace.

If the duplex mode is FDD, and the supplemental field of an FLAB, for a particular interlace, is equal to ‘0’, and if the extended transmission field of the FLAB is equal to ‘0’, all FL-ATAs that are extended transmission duration assignments shall be expired. The hop-ports specified by the NodeID in the FLAB shall be given to the FL-ATA in the corresponding interlace.

If an FLAB is received for a particular interlace with the supplemental field set to ‘1’ when the FL-ATA is empty on that interlace, then the access terminal shall ignore this FLAB.

If an FLAB is received for a particular interlace with the supplemental field set to ‘1’, and the current FL-ATA on that interlace has a different type then the FLAB, the access terminal shall ignore this FLAB.

If an FLAB is received for a particular interlace with the supplemental field set to ‘1’ that does not change the access terminal’s FL-ATA for that interlace, then the access terminal shall ignore this FLAB.

7.5.6.4.1.1.3 Decrementing Assignments

If duplex mode is TDD, then the access terminal shall decrement its FL-ATAs as follows. If an FLAB or NS-$FLAB_{UC}$ is received that contains a MACID other than the access terminal’s MACID with an assignment for a particular interlace, then all of the hop-ports in the FL-ATA for that interlace that intersect with hop-ports specified by the NodeID included in the FLAB/NS-$FLAB_{UC}$ shall be expired (removed from the FL-ATA) for that interlace.

If duplex mode is FDD, then the access terminal shall decrement its FL-ATAs as follows: If an FLAB or NS-$FLAB_{UC}$ is received that contains a MACID other than the access terminal’s MACID with an assignment for a particular interlace, any FL-ATA whose interlace overlaps in time with this interlace, and whose hop ports intersect with the hop ports assigned by the FLAB/NS-$FLAB_{UC}$, shall have its overlapping hop-ports removed from the FL-ATA.

If the access terminal receives, in the same PHY frame, an FLAB with its MACID, and an NS-$FLAB_{UC}$ with a MACID other than its MACID, for the same interlace, and the hop ports assigned by the FLAB intersect with the hop ports assigned by the NS-$FLAB_{UC}$, then the access terminal shall expire the intersecting hop ports from its FL-ATA.

If duplex mode is FDD, the following applies. If the access terminal receives, in the same PHY frame, an FLAB with its MACID, and an NS-$FLAB_{UC}$ with a MACID other than its MACID, for time overlapping interlaces, and the hop ports assigned by the FLAB intersect with the hop ports assigned by the NS-$FLAB_{UC}$, then the access terminal shall expire the intersecting hop ports from its FL-ATA.

If the access terminal receives, in the same PHY frame, an FLAB with its MACID, and an FLAB with a MACID other than its MACID, for the same interlace, and the hop ports assigned by the FLABs intersect, then the access terminal shall ignore the FLAB with its MACID.

If duplex mode is FDD, the following applies. If the access terminal receives, in the same PHY frame, an FLAB with its MACID, and an FLAB with a MACID other than its MACID, for time overlapping interlaces, and the hop ports assigned by the FLABs intersect, then the access terminal shall ignore the FLAB with its MACID.

7.5.6.4.1.1.4 Deassigning Assignments

If an FLAB is received for a particular interlace, for the access terminal's MACID, that assigns the reserved $NodeID_{DEASSIGN}$, then the FL-ATA on that interlace shall be expired.

7.5.6.4.1.1.5 Overlapping NS broadcast assignments and sticky assignments

If an access terminal receives (from SS MAC) a $NS\text{-}FLAB_{BC}$ for a particular interlace such that its $FL\text{-}NS\text{-}ATA_{BC}$ on that interlace overlaps with its (sticky) FL-ATA on that interlace, then the access terminal shall update its $FL\text{-}NS\text{-}ATA_{BC}$ according to the $NS\text{-}FLAB_{BC}$, and shall remove any overlapping hop-ports (between the $FL\text{-}NS\text{-}ATA_{BC}$ and FL-ATA) from the FL-ATA on that interlace for the duration of that particular $FL\text{-}NS\text{-}ATA_{BC}$.

If duplex mode is FDD, the following apply. If an access terminal receives (from SS MAC) a $NS\text{-}FLAB_{BC}$ for a particular interlace such that its $FL\text{-}NS\text{-}ATA_{BC}$ on that interlace overlaps in frequency with a (sticky) FL-ATA on a time overlapping interlace, then the access terminal shall update its $FL\text{-}NS\text{-}ATA_{BC}$ according to the $NS\text{-}FLAB_{BC}$, and shall remove any overlapping hop-ports (between the $FL\text{-}NS\text{-}ATA_{BC}$ and FL-ATA) from the FL-ATA for the duration of that particular $FL\text{-}NS\text{-}ATA_{BC}$.

If an access terminal receives (from SS MAC) a $NS\text{-}FLAB_{BC}$, and an FLAB in the same PHY Frame, such that its $FL\text{-}NS\text{-}ATA_{BC}$ overlaps with its FL-ATA on that interlace, then the access terminal shall update its $FL\text{-}NS\text{-}ATA_{BC}$ according to the $NS\text{-}FLAB_{BC}$, and shall remove any overlapping hop-ports (between the $FL\text{-}NS\text{-}ATA_{BC}$ and FL-ATA) from the FL-ATA on that interlace for the duration of that particular $FL\text{-}NS\text{-}ATA_{BC}$.

If duplex mode is FDD, the following apply. If an access terminal receives (from SS MAC) a $NS\text{-}FLAB_{BC}$ for a particular interlace, and an FLAB in the same PHY Frame, such that its $FL\text{-}NS\text{-}ATA_{BC}$ overlaps in frequency with the FL-ATA on a time overlapping interlace, then the access terminal shall update its $FL\text{-}NS\text{-}ATA_{BC}$ according to the $NS\text{-}FLAB_{BC}$, and shall remove any overlapping hop-ports (between the $FL\text{-}NS\text{-}ATA_{BC}$ and FL-ATA) from the FL-ATA for the duration of that particular $FL\text{-}NS\text{-}ATA_{BC}$.

7.5.6.4.1.1.6 Time overlapping NS unicast and sticky assignments

If the access terminal receives a unicast NS-FLAB of any type with its MACID for a particular interlace, while it already has a non-empty FL-ATA on that interlace, the access terminal shall keep the resulting $FL\text{-}NS\text{-}ATA_{UC}$ and clear the FL-ATA on that interlace.

Furthermore, if duplex mode is FDD, and if the access terminal receives a unicast NS-FLAB of any type with its MACID, that has the Extended Transmission field set to '1' , the access terminal shall expire all FL-ATAs that it has, unless the FL-ATA is also an extended transmission duration assignment, and occupies an interlace whose extended PHY Frame does not overlap with the extended PHY Frame of the new non-sticky unicast assignment. If the access terminal gets an $NS\text{-}FLAB_{UC}$ with the Extended Transmission field set to '0', then all FL-ATAs that are extended duration assignments shall be expired.

If the access terminal receives, in the same PHY Frame, a NS-FLAB with its MACID, giving it a non-empty FL-NS-ATA_{UC}, and an FLAB giving it a non-empty FL-ATA, then it shall ignore the FLAB.

7.5.6.4.1.1.7 Multi-code word assignments

If the access terminal receives a MCW-FLAB1 with its MACID from the SS MAC Protocol with the supplemental field set to '0', then there should be a MCW-FLAB2 with its MACID received in the same PHY Frame: otherwise, the access terminal shall ignore the MCW-FLAB1. If the access terminal receives a MCW-FLAB1 with its MACID from the SS MAC protocol with the supplemental bit set to '1', then if no MCW-FLAB2 with its MACID is received in the same PHY Frame, the access terminal shall use the current packet formats for layers 2 and above (provided there is an existing FL-ATA of type MCWFLAB). Otherwise, if a MCWFLAB2 with its MACID is received in the same PHY frame, the access terminal shall update the packet formats for layers 2 and above according to the MCWFLAB2.

7.5.6.4.1.1.8 Inband packet format switch

If the access terminal receives an InBandPacketFormatSwitch message, it shall update its packet format for the specified interlace, and use this updated packet format to try to decode all subsequent packets on that interlace, provided the InBandPacketFormatSwitch message is consistent with the assignment type on that interlace. If the InBandPacketFormatSwitch message is not consistent with the assignment type on that interlace, the access terminal shall ignore the InBandPacketFormatSwitch message.

7.5.6.4.1.2 Access terminal assignment management for non-sticky assignments

In this section the term NS-FLAB will be used to indicate all types of non-sticky FL assignment blocks, including NS-FLAB, NS-MCWFLAB, and NS-SCWFLAB, unless otherwise specified. The term NS-FLAB shall also be used for both broadcast and unicast NS-FLABs, unless otherwise specified. Note that unicast NS-FLABs may be any type, but broadcast NS-FLABs may not be NS-SCWFLABs or NS-MCWFLABs. Additionally, the term FLAB will be used to indicate all types of sticky FL assignment blocks, including FLAB, MCWFLAB, and SCWFLAB, unless otherwise specified.

The access terminal shall maintain and manage its FL-NS-ATAs by monitoring FLABs and NS-FLABs delivered from the SS MAC protocol. (Unless otherwise specified, FL-NS-ATA includes both FL-NS-ATA_{BC}s and FL-NS-ATA_{UC}s).

The logic in this section assumes that all of the FLABs/NS-FLABs are being sent from the FLSS. The logic for access terminal assignment management during handoff is found in 7.5.6.4.1.3.

If FLImplicitDeassignEnabled is equal to '1', then upon receiving a *TunedAway* indication from the Connected State Protocol, the access terminal shall expire all its FL-NS-ATAs.

If SelectedInterlaceMode is equal to '1', then the access terminal shall ignore all NS-FLABs that have the Extended Transmission field set to '1'.

When the access terminal receives a non-sticky forward link assignment block with broadcast MACID (NS-$FLAB_{BC}$) from the SS MAC Protocol for a particular interlace on a particular carrier, the access terminal shall then temporarily maintain a FL-NS-ATA_{BC} (non-sticky broadcast assignment) for that interlace and carrier. When the access terminal receives, from the SS MAC Protocol, a non-sticky forward link assignment block with unicast MACID (NS-$FLAB_{UC}$) for a particular interlace, where the MACID matches its MACID, the access terminal shall temporarily maintain a FL-NS-ATA_{UC} (non-sticky access terminal unicast assignment) for that interlace.

If the extended transmission field is equal to '0', an NS-FLAB assigns hop-ports for a particular interlace consisting of standard PHY Frames as shown in 7.1.3.1.2. The duration field specifies the number of standard PHY Frames to be used for transmission of this assignment. If the Extended Transmission field is equal to '1', an NS-FLAB assigns hop-ports for an interlace consisting of extended PHY Frames ,as shown in 7.1.3.1.3. The duration field specifies the number of extended PHY Frames to be used for transmission of this assignment.

If multi-carrier mode is equal to MultiCarrierOn, the access terminal may maintain a FL-NS-ATA_{BC} for each non-overlapping interlace for each active carrier. Active Carriers are specified by the ActiveCarriers parameter in the public data of the SS MAC Protocol.

The access terminal shall give up its FL-NS-ATA_{BC} for a particular interlace after it has kept it for the assignment duration specified in the corresponding NS-$FLAB_{BC}$. Similarly, the access terminal shall give up its FL-NS-ATA_{UC} for a particular interlace after it has kept it for the assignment duration specified in the corresponding NS-$FLAB_{UC}$.

7.5.6.4.1.2.1 Simultaneous assignments

If duplex mode is FDD, NS-FLABs received in the same PHY frame could be for different interlaces because some of the NS-FLABs could be for the interlace consisting of standard PHY frames, and some of the NS-FLABs could be for the interlace consisting of extended PHY frames. If multiple NS-$FLAB_{UC}$s for the access terminal's MACID are received in the same PHY frame from SS MAC, and they are not all for the same interlace, then the access terminal shall assume an error has occurred, and shall ignore all of these NS-$FLAB_{UC}$s, unless there is an NS-$FLAB_{UC}$ with NodeID set to $NodeID_{DEASSIGN}$, in which case this NS-$FLAB_{UC}$ shall not be ignored. If multiple NS-$FLAB_{BC}$s for the access terminal's MACID are received from SS MAC in the same PHY frame, for the same carrier, and the NS-$FLAB_{BS}$s are not all for the same interlace, then the access terminal shall assume an error has occurred, and shall ignore all of these NS-$FLAB_{BC}$s.

If multiple NS-$FLAB_{UC}$s for the access terminal's MACID are received from the SS MAC Protocol in the same PHY frame, and one of the NS-$FLAB_{UC}$s has NodeID set to $NodeID_{DEASSIGN}$, then all the NS-$FLAB_{UC}$s except for the latter shall be discarded. This rule trumps all those which follow in this section.

If multiple NS-$FLAB_{BC}$s are received from the SS MAC Protocol in the same PHY Frame and carrier, and the values of all the fields, except for the NodeID field, are the same, the access terminal shall treat these NS-$FLAB_{BC}$s as a single NS-$FLAB_{BC}$, assigning the union of the hop-ports mapped by the constituent NodeIDs. If multiple NS-$FLAB_{BC}$s are received from the SS MAC protocol in the same PHY Frame and carrier, and if all the fields (other than the NodeID field) are not the same, the access terminal shall consider an error has occurred and ignore these blocks.

If multiple NS-$FLAB_{UC}$s of the same type are received from the SS MAC Protocol in the same PHY-frame, for the access terminal's MACID, and the values of all the fields, except for the NodeID field, are the same, the access terminal shall treat these NS-$FLAB_{UC}$s as a single NS-$FLAB_{UC}$ assigning the union of the hop-ports mapped by the constituent NodeIDs. If multiple NS-$FLAB_{UC}$s of the same type are received form the SS MAC protocol in the same PHY Frame, for the access terminal's MACID, and if all the fields (other than the NodeID field) are not the same, the access terminal shall consider an error has occurred and shall ignore these blocks.

If multiple NS-$FLAB_{UC}$s are received from the SS MAC Protocol in the same PHY frame for the access terminal's MACID, and these NS-$FLAB_{UC}$s are not of the same type, then the access terminal shall consider an error has occurred and ignore these blocks.

7.5.6.4.1.2.2 Deassigning Assignments

If an NS-FLAB for a particular interlace is received with the access terminal's MACID, and assigns the reserved $NodeID_{DEASSIGN}$, then the FL-NS-ATA on that interlace shall be expired.

7.5.6.4.1.2.3 Time Overlapping broadcast assignments

If a NS-$FLAB_{BC}$ is received form the SS MAC Protocol for a particular interlace and carrier, and the access terminal already has a FL-NS-ATA_{BC} for that interlace and carrier, then the new assignment block takes precedence: the access terminal shall stop trying to decode on the old FL-NS-ATA_{BC} for that interlace (shall clear this FL-NS-ATA_{BC}), and shall update its FL-NS-ATA_{BC} for that interlace according to the new NS-$FLAB_{BC}$.

If duplex mode is FDD, the following apply. If an NS-$FLAB_{BC}$ is received from the SS MAC Protocol for a particular carrier that has the Extended Transmission field set to '1', then all FL-NS-ATA_{BC}s on that carrier shall be expired, unless there is an FL-NS-ATA_{BC} that is an extended transmission duration assignment whose extended PHY Frames do not overlap with those of the new assignment. If an NS-$FLAB_{BC}$ is received from the SS MAC protocol for a particular carrier that has the Extended Transmission field set to '0', then all FL-NS-ATA_{BC}s on that carrier that are extended transmission duration assignments shall be expired. The access terminal shall update its FL-NS-ATA_{BC} according to the new NS-$FLAB_{BC}$.

7.5.6.4.1.2.4 Time Overlapping unicast assignments

If a NS-$FLAB_{UC}$ is received from the SS MAC protocol for a particular interlace with the access terminal's MACID, and the access terminal already has a FL-NS-ATA_{UC} for that interlace, then the new assignment block takes precedence: the access terminal shall stop trying to decode on the old FL-NS-ATA_{UC} for that interlace (shall clear this FL-NS-ATA_{UC}), and shall update its FL-NS-ATA_{UC} for that interlace according to the new FL-NS-$FLAB_{UC}$.

If duplex mode is FDD, the following apply. If an NS-$FLAB_{UC}$ is received from the SS MAC Protocol, that has the access terminal's MACID and has the Extended Transmission field set to '1', then all FL-NS-ATA_{UC}s shall be expired, unless there is an FL-NS-ATA_{UC} that is an extended transmission duration assignment whose extended PHY Frames do not overlap with those of the new assignment. If an NS-$FLAB_{UC}$ is received that has the access terminal's MACID, and has the Extended Transmission field set to '0', then all FL-NS-ATA_{UC}s that are extended transmission duration assignments shall be expired. The access terminal shall update its FL-NS-ATA_{UC} according to the new NS-$FLAB_{UC}$.

7.5.6.4.1.2.5 Overlapping broadcast and unicast assignments

If the access terminal receives a NS-FLAB$_{BC}$ from the SS MAC protocol for a particular interlace such that its FL-NS-ATA$_{BC}$ on that interlace overlaps with its FL-NS-ATA$_{UC}$ on that interlace, then the most recent assignment block takes precedence; that is, the access terminal shall expire the old FL-NS-ATA$_{UC}$ assignment, and update its FL-NS-ATA$_{BC}$ according to the new NS-FLAB$_{BC}$.

Furthermore, if duplex mode is FDD, and if the access terminal receives an NS-FLAB$_{BC}$ from the SS MAC protocol with the Extended Transmission field set to ‘1’, the access terminal shall expire all FL-NS-ATA$_{UC}$s that overlap in frequency with the FL-NS-ATA$_{BC}$, unless the FL-NS-ATA$_{UC}$ is also an extended transmission duration assignment, and occupies an interlace whose extended PHY Frame does not overlap with the extended PHY Frame of the new non-sticky broadcast assignment. If the access terminal receives an NS-FLAB$_{BC}$ from the SS MAC protocol with the Extended Transmission field set to ‘0’, then all FL-NS-ATA$_{UC}$s that are extended duration assignments and overlap in frequency with the FL-NS-ATA$_{BC}$ shall be expired.

If the access terminal receives a NS-FLAB$_{UC}$ with its MACID, from the SS MAC protocol, for a particular interlace such that its FL-NS-ATA$_{UC}$ on that interlace overlaps with its FL-NS-ATA$_{BC}$ on that interlace, then the most recent assignment takes precedence, that is, the access terminal shall expire the old assignment, the FL-NS-ATA$_{BC}$, and update its FL-NS-ATA$_{UC}$ according to the new NS-FLAB$_{UC}$

Furthermore, if duplex mode is FDD, and if the access terminal receives an NS-FLAB$_{UC}$ with its MACID from the SS MAC protocol with the Extended Transmission field set to ‘1’, the access terminal shall expire all FL-NS-ATA$_{BC}$s that overlap in frequency with the FL-NS-ATA$_{UC}$, unless the FL-NS-ATA$_{BC}$ is also an extended transmission duration assignment, and occupies an interlace whose extended PHY Frame does not overlap with the extended PHY Frame of the new non-sticky unicast assignment. If the access terminal receives an NS-FLAB$_{UC}$ with its MACID from the SS MAC Protocol with the Extended Transmission field set to ‘0’, then all FL-NS-ATA$_{BC}$s that are extended transmission duration assignments and overlap in frequency with the FL-NS-ATA$_{UC}$ shall be expired.

If the access terminal receives a NS-FLAB$_{BC}$ and a NS-FLAB$_{UC}$ in the same PHY frame for the same interlace, and the corresponding FL-NS-ATA$_{UC}$ and FL-NS-ATA$_{BC}$ overlap in frequency, then the access terminal shall assume an error has occurred, and shall ignore the NS-FLAB$_{UC}$.

If duplex mode is FDD, then if the access terminal receives a NS-FLAB$_{BC}$ and a NS-FLAB$_{UC}$ in the same PHY frame for time overlapping interlaces, and the corresponding FL-NS-ATA$_{UC}$ and FL-NS-ATA$_{BC}$ overlap in frequency, then the access terminal shall assume an error has occurred, and shall ignore the NS-FLAB$_{UC}$.

7.5.6.4.1.2.6 Time Overlapping NS unicast and sticky assignments

If an access terminal receives an FLAB with its MACID from the SS MAC protocol for a particular interlace while it already has a FL-NS-ATA$_{UC}$ on that interlace, then it shall give up the FL-NS-ATA$_{UC}$ on that interlace and only try to decode on the FL-ATA for that interlace.

Furthermore, if duplex mode is FDD, and if the access terminal receives an FLAB with its MACID from the SS MAC protocol with the Extended Transmission field set to ‘1’ , the access terminal shall expire all FL-NS-ATA$_{UC}$s that it has, unless the FL-NS-ATA$_{UC}$ is also an extended transmission duration assignment, and occupies an interlace whose extended PHY Frame does not overlap with the extended PHY Frame of the new assignment. If the access terminal receives an FLAB with its

MACID from the SS MAC protocol with the Extended Transmission field set to '0', then all FL-NS-ATA_{UC}s that are extended duration assignments shall be expired.

7.5.6.4.1.2.7 Overlapping NS broadcast and sticky assignments

If duplex mode is TDD, the following apply. If the Access terminal gets an FLAB such that its updated FL-ATA on an interlace overlaps with its FL-NS-ATA_{BC} for that interlace, the access terminal shall expire the FL-NS-ATA_{BC}.

If duplex mode is FDD, the following apply. If the Access terminal gets an FLAB such that its updated FL-ATA on an interlace overlaps (in time and frequency) with a FL-NS-ATA_{BC} , the access terminal shall expire the FL-NS-ATA_{BC}.

7.5.6.4.1.2.8 Overlapping assignments from other ATs

If an FLAB or NS-$FLAB_{UC}$ is received from the SS MAC protocol that contains a MACID other than the access terminal's MACID with an assignment for a particular interlace, and if the hop-ports specified by the NodeID in the FLAB/NS-FLAB $_{UC}$ intersect with the access terminal's FL-NS-ATA $_{UC}$ on that interlace, then the access terminal shall expire its FL-NS-ATA $_{UC}$ for that interlace.

Furthermore, if duplex mode is FDD, and if an FLAB/NS-$FLAB_{UC}$ is received from the SS MAC protocol that contains a MACID other than the access terminal's MACID and has the Extended Transmission field set to '1', the access terminal shall expire all FL-NS-ATA_{UC}s whose hop-ports overlap with the new assignment, unless the FL-NS-ATA_{UC} is also an extended transmission duration assignment, and occupies an interlace whose extended PHY Frame does not overlap with the extended PHY Frame of the new assignment. If an FLAB/NS-$FLAB_{UC}$ is received from the SS MAC protocol that contains a MACID other than the access terminal's MACID and has the Extended Transmission field set to '0', then all FL-NS-ATA_{UC}s that are extended transmission duration assignments, and have hop-ports that overlap with the new assignment, shall be expired.

If an FLAB or NS-FLAB $_{UC}$ is received from the SS MAC protocol that contains a MACID other than the access terminal's MACID with an assignment for a particular interlace, and if the hop-ports specified by the NodeID in the FLAB/NS-FLAB $_{UC}$ intersect with the access terminal's FL-NS-ATA $_{BC}$ on that interlace for a particular carrier, then the access terminal shall expire its FL-NS-ATA $_{BC}$ for that interlace and carrier.

Furthermore, if duplex mode is FDD, and if an FLAB/NS-$FLAB_{UC}$ is received from the SS MAC protocol that contains a MACID other than the access terminal's MACID and has the Extended Transmission field set to '1', the access terminal shall expire all FL-NS-ATA_{BC}s whose hop-ports overlap with the new assignment, unless the FL-NS-ATA_{BC} is also an extended transmission duration assignment, and occupies an interlace whose extended PHY Frame does not overlap with the extended PHY Frame of the new assignment. If an FLAB/NS-$FLAB_{UC}$ is received from the SS MAC protocol that contains a MACID other than the access terminal's MACID and has the Extended Transmission field set to '0', then all FL-NS-ATA_{BC}s that are extended transmission duration assignments, and have hop-ports that overlap with the new assignment , shall be expired.

If the access terminal receives, in the same PHY frame, an NS-$FLAB_{BC}$ with its MACID and an NS-$FLAB_{UC}$ with a MACID other than its MACID, for the same interlace, and the hop ports assigned by the NS-$FLAB_{UC}$ overlap with the hop ports assigned by the NS-$FLAB_{BC,}$ the access terminal assumes as error has occurred, and shall ignore the NS-$FLAB_{UC}$.

If duplex mode is FDD, the following holds. If the access terminal receives, in the same PHY frame, an NS-$FLAB_{BC}$ with its MACID and an NS-$FLAB_{UC}$ with a MACID other than its MACID, for time overlapping interlaces, and the hop ports assigned by the NS-$FLAB_{UC}$ overlap with the hop ports assigned by the NS-$FLAB_{BC,}$ the access terminal assumes as error has occurred, and shall ignore the NS-$FLAB_{UC}$.

If the access terminal receives, in the same PHY frame, an NS-$FLAB_{UC}$ with its MACID and an NS-$FLAB_{UC}$ with a MACID other than its MACID, for the same interlace, and the hop ports assigned by the NS-$FLAB_{UC}$ s overlap, the access terminal assumes as error has occurred, and shall ignore the NS-$FLAB_{UC}$ with its MACID.

If duplex mode is FDD, the following holds. If the access terminal receives, in the same PHY frame, an NS-$FLAB_{UC}$ with its MACID and an NS-$FLAB_{UC}$ with a MACID other than its MACID, for time overlapping interlaces, and the hop ports assigned by the NS-$FLAB_{UC}$ s overlap, the access terminal assumes as error has occurred, and shall ignore the NS-$FLAB_{UC}$ with its MACID.

If the access terminal receives, in the same PHY frame, an NS-$FLAB_{BC}$, and an FLAB with a MACID other than the access terminal's MACID, for the same interlace, and the hop ports assigned by the NS-$FLAB_{BC}$ intersect with the hop ports assigned by the FLAB, the access terminal shall ignore the FLAB (in the sense that the NS-ATA_{BC} assigned by the NS-$FLAB_{BC}$ will not be modified or expired because of the FLAB.)

If duplex mode is FDD, the following applies. If the access terminal receives, in the same PHY frame, an NS-$FLAB_{BC}$, and an FLAB with a MACID other than the access terminal's MACID, for time overlapping interlaces, and the hop ports assigned by the NS-$FLAB_{BC}$ intersect with the hop ports assigned by the FLAB, the access terminal shall ignore the FLAB (in the sense that the NS-ATA_{BC} assigned by the NS-$FLAB_{BC}$ will not be modified or expired because of the FLAB).

If the access terminal receives, in the same PHY frame, an NS-$FLAB_{UC}$ with its MACID, and an FLAB with a MACID other than the access terminal's MACID, for the same interlace, and the hop ports assigned by the NS-$FLAB_{UC}$ intersect with the hop ports assigned by the FLAB, the access terminal shall ignore the FLAB (in the sense that the NS-ATA_{UC} assigned by the NS-$FLAB_{UC}$ will not be modified or expired because of the FLAB).

If duplex mode is FDD, the following applies. If the access terminal receives, in the same PHY frame, an NS-$FLAB_{UC}$ with its MACID, and an FLAB with a MACID other than the access terminal's MACID, for time overlapping interlaces, and the hop ports assigned by the NS-$FLAB_{UC}$ intersect with the hop ports assigned by the FLAB, the access terminal shall ignore the FLAB (in the sense that the NS-ATA_{UC} assigned by the NS-$FLAB_{UC}$ will not be modified or expired because of the FLAB).

7.5.6.4.1.3 Access terminal assignment management during handoff

If an *FLSSChanged* indication from the RCC MAC protocol is received, the access terminal shall clear all FL-ATAs and FL-NS-ATAs associated with the old FLSS.

If the access terminal receives an FLAB-HO with its MACID, and the sector in the DesiredSector field is different from the FLSS, but in the same synchronous subset as the FLSS, the access terminal shall issue a *ReverseControlChannelMAC.ChangeFLSS* to change the FLSS to the sector in the DesiredSector field of the FLAB-HO. If the access terminal then receives an *FLSSChanged* indication from the RCC MAC Protocol, the access terminal shall process the *FLSSChanged* indication as specified above, and in addition shall update its new FL-ATA to the hop ports specified by the

NodeID field in the FLAB-HO for the interlace specified by the ExtendedTransmission field in the FLAB-HO, for the sector specified in the DesiredSector field. The packet format to be used is specified in the PF field of the FLAB-HO.

If the access terminal receives an FLAB-HO with its MACID, and the sector in the DesiredSector field is not different from the FLSS, or is in a different synchronous subset as the FLSS, the access terminal shall ignore this FLAB-HO.

If the access terminal receives multiple FLAB-HOs with its MACID, and all the fields are the same except for the NodeID fields, then the access terminal shall treat these FLAB-HOs as a single FLAB-HO assigning the union of the hop ports specified by the NodeID fields.

If the access terminal receives multiple FLAB-HOs with its MACID, and all the fields other than the NodeID field are not the same, then the access terminal shall ignore these FLAB-HOs.

If the access terminal receives an FLAB/NS-FLAB with the access terminal's MACID, that has the supplemental field set to '0', from the DFLSS, while the DFLSS is different from the FLSS, the access terminal shall issue a *ReverseControlChannelMAC.ChangeFLSS* command to change from the FLSS to the DFLSS. If the access terminal then receives an *FLSSChanged* indication from the RCC MAC Protocol, the access terminal shall process the *FLSSChanged* indication as specified above, and in addition shall update the appropriate FL-ATA/FL-NS-ATA according to the new FLAB/NS-FLAB.

The access terminal shall ignore all FLABs or NS-FLABs that come from sectors other than its current FLSS or its DFLSS.

7.5.6.4.1.4 Access terminal assignment processing for sticky assignments

The access terminal shall attempt to decode a MAC packet on each interlace with a non-empty FL-ATA (contains one or more hop-ports). The access terminal may attempt to detect erasure sequences that are transmitted by the access network whenever a MAC packet is not available for transmission. Note that if duplex mode is FDD, erasures may be sent on standard PHY Frames or extended PHY Frames. Exact algorithms for detecting erasure sequences and start-of-packet for MAC packets that span multiple PHY Frames are beyond the scope of this specification.

When an access terminal's FL-ATA on an interlace changes, and the duration of the change is not pre-specified (as would be the case if the FL-ATA was punctured due to a FL-NS-ATA_{BC}), the access terminal shall terminate any existing packet decoding on the interlace and restart packet decoding attempts on the interlace (as long as the new ATA is non-empty).

When an access terminal's FL-ATA on an interlace changes, and the duration of the change is pre-specified, as is the case if the FL-ATA is punctured due to a FL-NS-ATA_{BC}, the access terminal shall continue to decode as usual, except that the FL-ATA is reduced for the specified duration.

If a MAC packet on the FL-ATA is successfully decoded, as indicated by the PHY, the access terminal shall reset the supervision timer and transmit a positive ACK via the RCC MAC to the access network. Refer to 7.1.3 for detailed interlace structure and acknowledgment timing for both FDD and TDD modes. The payload of the packet is then passed up to the Security Sublayer for further processing.

If a MAC packet on the FL-ATA fails to decode successfully, as indicated by the PHY, and the access terminal determines that the packet has been transmitted for the maximum number of PHY Frames for the relevant PF (see 7.5.6.7), then the access terminal shall expire the FL-ATA for the interlace in which the packet failed. For an access terminal that is in MIMO MCW mode, the maximum number of PHY Frames is the maximum over all of the specified PFs. Exact algorithms to estimate the number of H-ARQ packet transmissions that have been sent prior to successful decode are beyond the scope of this specification.

7.5.6.4.1.5 Access terminal assignment processing for non-sticky assignments

For each interlace with a non-empty FL-NS-ATA_{UC}, the access terminal may attempt to decode a MAC packet. The access terminal shall not attempt to detect erasure sequences for FL-NS-ATA_{UC}. If a MAC packet on the FL-NS-ATA_{UC} is successfully decoded, as indicated by the PHY, the access terminal shall reset the supervision timer, transmit a positive ACK via the RCC MAC to the access network. The access terminal may choose to expire its FL-NS-ATA_{UC} at this time if it is not performing ACK to NACK detection. After sending an ACK, the access terminal may choose to expire its FL-NS-ATA_{UC} earlier than the time the assignment is set to expire (this time is derived from the duration field in the assignment block). Refer to 7.1.3 for detailed interlace structure and acknowledgment timing for both FDD and TDD modes. The payload of the packet is then passed up to the Security Sublayer for further processing.

For each active carrier, for each interlace with a non-empty FL-NS-ATA_{BC}, the access terminal may attempt to decode a MAC packet. The access terminal shall not attempt to detect erasure sequences for FL-NS-ATA_{BC}. Even if a MAC packet on the FL-NS-ATA_{BC} is successfully decoded, as indicated by the PHY, the access terminal shall not transmit a positive ACK message via the RCC MAC to the access network.

7.5.6.4.1.6 Header Processing

If a MAC packet on the FL-NS-ATA_{UC} is successfully decoded, as indicated by the PHY, the packet shall be processed according to the following rules:

- The access terminal shall parse the packet according to the header and trailer specified for unicast transmissions.
- The access terminal shall pass the payload of the packet to the Security Sublayer for further processing
- If the UATIInfoIncluded header field of the packet is equal to ‘1’,
    - The access terminal shall generate a *UATIReceived* indication accompanied by the contents of the MAC header.
    - If the FailureCode header field of the packet is set to SessionLost, the access terminal shall generate a *SessionLost* indication.

If a MAC packet on the FL-NS-ATA_{BC} is successfully decoded, as indicated by the PHY, the packet shall be processed according to the following rules:

- The access terminal shall parse the packet according to the header and trailer specified for broadcast transmissions.

- The access terminal shall forward the Security Sublayer packet to the Security Sublayer if either of the following two conditions are met:
  - If the ATIType field and the ATI field of the first ATI Record in the Lower MAC Sublayer header is equal to the ATIType and ATI fields of any member of the Address Management Protocol's ReceiveATIList.
  - If the ATIType of any of the ATI Record in the MAC Layer header of a Security Sublayer packet is equal to '00' (i.e., BATI) and the ReceiveATIList includes a record with ATIType set to '00'.
- If the ATIType field and the ATI field of any ATI Record in the Lower MAC Sublayer header of a is equal to the ATIType and ATI fields of any member of the Address Management Protocol's ReceiveATIList, and the OpenConnectionRequired field associated with any of the matching ATI field is set to "1", then this protocol shall return an *PageReceived* indication. This indicates that the access terminal has been paged.
- Otherwise, the access terminal shall discard the received packet.

7.5.6.4.2 Access network requirements

The access network shall keep a variable NumLayers for each interlace, which shall be initialized to zero when this protocol enters the Active state. On transmission of a SCWFLAB/NS-SCWFLAB for a particular interlace via the SS MAC protocol, NumLayers for that interlace shall be set to the NumLayers field of the SCWFLAB/NS-SCWFLAB. On transmission of a MCWFLAB/NS-MCWFLAB for a particular interlace via the SS MAC protocol, NumLayers for that interlace shall be set to the number of non-zero packet formats given in the MCWFLAB/NS-MCWFLAB. NumLayers for a particular interlace shall also be modified to reflect rank adjustments sent in the FL MAC header.

7.5.6.4.2.1 Access network assignment management

If FLImplicitDeassignEnabled is equal to '1', then upon receiving a *TunedAway* indication from the Connected State Protocol, the access network shall expire all the FL-ATAs and FL-NS-ATAs for that access terminal.

If the FLNumSDMADimensions > 1, then the access network shall ensure that assignments to different ATs that contain hop-ports that map to the same subcarriers have the same F-DPICH format, as described in 7.5.6.7.

7.5.6.4.2.2 F-DCH transmissions associated with sticky assignments

The access network may formulate and transmit a MAC packet on an interlace according to the FL-ATA on the interlace and the PF selected for transmission. The MAC shall pass the Physical Layer Protocol the set of hop ports specified by the FL-ATA, and shall specify whether or not the assignment is an Extended Transmission Duration Assignment (see 7.1.3.1.3). The access network informs the access terminal of its assignment using signaling blocks as described in 7.4.6.3.1.2.

If TuneAwayStatus is equal to '1', then the access network shall not send any MAC packets on any ATAs on any interlaces to this access terminal.

If a positive ACK addressed to the access terminal's MACID is received that corresponds to the transmitted packet, transmission of the MAC packet shall terminate, and the interlace is immediately available for the next packet. (Note that for multi-code word assignments assigned by MCW-FLABs, the ACKs for all the layers need to be received before the transmission of the MAC packet terminates, and the interlace is available for the next packet.) If the access network transmits a packet for the maximum number of transmissions of the PF selected for this MAC packet without receiving an ACK, the access network shall expire the FL-ATA for the interlace, and shall return a *ForwardTrafficPacketsMissed* indication along with parameters that uniquely identify the lost packet. The method of uniquely identifying the packet is out of the scope of this specification.

The access terminal should send no more than one MAC packet over a FL-ATA per interlace. If two interlaces overlap in time, the access terminal should only send a MAC packet over a FL-ATA on one of the interlaces.

If no packet is available for transmission in a given PHY Frame (standard or extended), then an erasure sequence shall be transmitted on the FL-ATA, or a subset of the FL-ATA, for that PHY Frame (as specified in the Physical Layer specification), or the access network shall expire the FL-ATA for the interlace.

7.5.6.4.2.3 F-DCH transmissions associated with non-sticky assignments

The access network may formulate and transmit a MAC packet on an interlace according to the NS-ATA on the interlace and the PF selected for transmission. The MAC shall pass the Physical Layer Protocol the set of hop ports specified by the FL-NS-ATA, and shall specify whether or not the assignment is an Extended Transmission Duration Assignment (see 7.1.3.1.3). The access network shall not start an F-DCH transmission containing a page in the first frame of a superframe. (A page can be sent using an FTC MAC packet transmission to a broadcast MACID wherein the MAC header includes the UATI field. See 7.5.7.2).

The number of H-ARQ retransmissions over a FL-NS-ATA_{UC} is equal to the duration of the FL-NS-ATA_{UC}, as specified by the duration field in the non-sticky assignment block (see 7.4.6.3.1.2). Note that the maximum number of transmissions may be more than 6, which is the number specified in 7.5.6.7.1.

If TuneAwayStatus is equal to '1', then the access network shall not send any MAC packets on any FL-NS-ATA_{UC}s on any interlaces to this access terminal. MAC packets being sent on FL-NS-ATA_{BC}s may still be sent.

The access network should send no more than one MAC packet over a FL-NS-ATA_{UC} per interlace. If two interlaces overlap in time, the access network should only send a MAC packet over a FL-NS-ATA_{UC} on one of the interlaces. The access network should send no more than one MAC packet over a FL-NS-ATA_{BC} per carrier ,per interlace. For a given carrier, if two interlaces overlap in time, the access network should only send a MAC packet over a FL-NS-ATA_{BC} on one of the interlaces. The access network should not send MAC packets to the same MACID over both an FL-ATA and a FL-NS-ATA_{UC} on the same interlace, or on time overlapping interlaces.

If a positive ACK corresponding to the access terminal's MACID is received that corresponds to the transmitted packet on a FL-NS-ATA_{UC}, transmission of the MAC packet shall terminate, and the access network shall expire the FL-NS-ATA_{UC}. (Note that for multi-code word assignments assigned by NS-MCW-FLABs, the ACKs for all the layers need to be received before the transmission of the MAC packet terminates, and the interlace is available for the next packet.) If no ACK is received for this packet, the access network shall return a *ForwardTrafficPacketsMissed* indication along with parameters that uniquely identify the lost packet. The method of uniquely identifying the packet is out of the scope of this specification.

FL-NS-ATAs shall be expired after the specified duration (see 7.4.6.3.1.2), if not already expired.

7.5.6.5 Supervision procedures

The access terminal shall maintain a supervision timer of duration InactivityTimeout. This timer shall be reset as described in 7.5.6.4.1.4 and 7.5.6.4.1.5. The access terminal shall issue a *SupervisionFailed* indication when the supervision timer expires.

7.5.6.6 Channel trees

A channel tree defines the mapping of each NodeID to a set of hop-ports. A channel tree on the FL is indexed by the FLChannelTreeIndex and the number of subcarriers mapped by the channel tree, $N_{CARRIER\_SIZE}$, a parameter that is defined by the Physical Layer protocol. See 7.1.4.1 for common terms used for describing channel trees in this specification. Hop-ports shall be numbered numerically from 0.

Q_{SDMA} equals FLNumSDMADimensions.

The set of hop-ports specified by a NodeID shall be the union of usable hop-ports mapped by all base-nodes that are descendants of the node specified by NodeID, minus the hop-ports allocated to F-SSCH (as defined in 7.4.6.3.1.1). "Usable hop-ports" are defined by the Physical Layer Protocol.

The number of hop-ports indexed by the tree shall equal $Q_{SDMA}*N_{CARRIER\_SIZE}$, and the total number of nodes in the tree shall be a function of $N_{CARRIER\_SIZE}$ and Q_{SDMA}.

Note that when multi-carrier mode is equal to MultiCarrierOn, there is an independent channel tree per carrier, and the channel tree in use for the carrier is signaled on the overhead channels of that carrier. Further, when a specific NodeID or set of hop-ports is communicated with other protocols in this specification, the associated carrier must also be communicated.

7.5.6.6.1 FL channel tree index 0

Channel trees associated with FLChannelTreeIndex 0 are illustrated in Figure 81 for $N_{CARRIER\_SIZE}$ = 512, 1024, and 2048. For $N_{CARRIER\_SIZE}$ = 512, all nodes above the dashed line marked with $N_{CARRIER\_SIZE}$ = 512 are included in the channel tree; For $N_{CARRIER\_SIZE}$ = 1024, all nodes above the dashed line marked with $N_{CARRIER\_SIZE}$ = 1024 are included in the channel tree; For $N_{CARRIER\_SIZE}$ = 2048, all nodes above the dashed line marked with $N_{CARRIER\_SIZE}$ = 2048 are included in the channel tree.

MinHopPortsPerNode equals 16 for FLChannelTreeIndex 0.

The figure shows only $1/Q_{SDMA}$ of the total tree, and there are Q_{SDMA} identical versions of the illustrated tree each with unique NodeIDs and mapping unique hop-ports. For example, if the total tree is composed of Q_{SDMA} identical trees indexed by $i=0, 1, \ldots, Q_{SDMA}-1$, then NodeIDs of nodes on the i th tree can be obtained from the illustrated tree by adding $i *N_{CARRIER\_SIZE}/(MinHopPortsPerNode/2)$ to the NodeID of the illustrated tree. The number of hop-ports indexed by the tree shall equal $Q_{SDMA}N_{CARRIER\_SIZE}$, and the total number of nodes in the tree shall be a function of $N_{CARRIER\_SIZE}$. Namely, the base-nodes are defined by the intervals NodeID = i* $N_{CARRIER\_SIZE}$/(MinHopPortsPerNode/2)+{ $N_{CARRIER\_SIZE}$/MinHopPortsPerNode - 1 , $N_{CARRIER\_SIZE}$/(MinHopPortsPerNode/2) - 2 }, $i=0, 1, \ldots, Q_{SDMA}-1$. Thus, for $N_{CARRIER\_SIZE}$=512 and Q_{SDMA}=4, there are 128 base-node NodeIDs, 31 through 62, 95 through 126, 159 through 190, and 223 through 254. For nodes on the same level of a channel tree, the NodeID associated with a node increases from left to right with step of 1. One deassignment NodeID, $NodeID_{DEASSIGN}$, is set to $N_{CARRIER\_SIZE}$/(MinHopPortsPerNode/2) - 1.

The mapping of hop-ports to each base-node is described as follows. Each base-node maps to MinHopPortsPerNode hop-ports, the first MinHopPortsPerNode hop-ports (indices 0 to MinHopPortsPerNode-1) to the base-node with the lowest NodeID, the second MinHopPortsPerNode hop-ports to the next base-node, etc., until all hop-ports are mapped. See Table 70 for an example of this mapping for $N_{CARRIER\_SIZE}$=512, MinHopPortsPerNode=16, and Q_{SDMA}=4.

Table 70 Base node NodeID to Hop-port Mapping Example ($N_{CARRIER\_SIZE}$=512, MinHopPortsPerNode=16, and Q_{SDMA}=4)

Base node NodeID	Hop-ports mapped
31	0-15
32	16-31
…	…
62	496-511
95	512-527
96	528-543
…	…
126	1008-1023
…	…
254	2032-2047

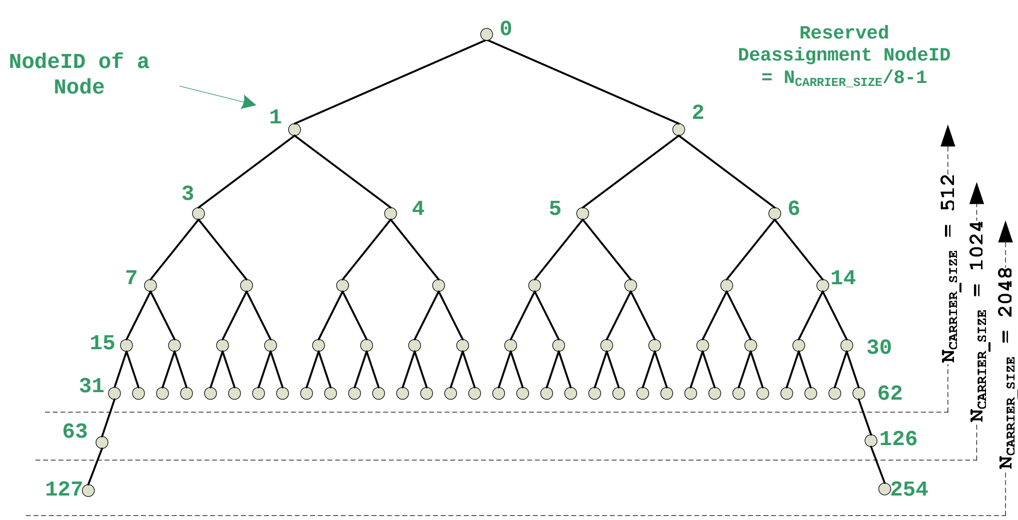

Figure 77 FL channel trees with index 0

7.5.6.7 Packet formats

A packet format (PF) specifies the spectral efficiency, maximum number of transmissions, and the modulation format to be used for each transmission of a data packet. Each packet format is indexed by a packet format index. The modulation format is specified by the number of bits in each modulation symbol, which is denoted by modulation order. Modulation orders of 2, 3, 4, and 6 corresponding to QPSK, 8PSK, 16QAM and 64QAM modulations respectively. These modulations are described in detail in the Physical Layer specification. The size of the MAC packet that is provided to the Physical Layer is a function of the packet format as well as the set of hop-ports that are assigned to the data packet (to be transmitted on the F-DCH Physical Layer channel.) The computation of the packet size as a function of the set of hop-ports and the packet format is described in the Physical Layer.

7.5.6.7.1 SISO mode packet formats (SSMAC)

The packet format consists of six bits. The first bit of the PF indicates the transmission mode. If the first bit is equal to '0' then the default mode is used for transmission. If the first bit is equal to '1' then the STTD mode is used for transmission. The second bit of the PF indicates the F-DPICH format as described in the F-DPICH section of the physical layer protocol. If the second bit is equal to '0', F-DPICH format 0 is used. If the second bit is equal to '1', F-DPICH format 1 is used. The remaining four bits index the spectral efficiency, maximum number of transmissions, and the modulation format to be used for each transmission of a data packet. The indexing is described in Table 71.

Table 71 FL packet formats – SISO mode

Packet Format Index	Spectral efficiency on 1st trans-mission	Max number of trans-missions	Modulation order for each transmission					
			1	2	3	4	5	6+
0	0.2	6	2	2	2	2	2	2
1	0.5	6	2	2	2	2	2	2
2	1.0	6	2	2	2	2	2	2
3	1.5	6	3	2	2	2	2	2
4	2.0	6	4	3	3	3	3	3
5	2.5	6	6	4	4	4	4	4
6	3.0	6	6	4	4	4	4	4
7	4.0	6	6	6	4	4	4	4
8	5.0	6	6	6	4	4	4	4
9	6.0	6	6	6	4	4	4	4
10	7.0	6	6	6	4	4	4	4
11	8.0	6	6	6	6	4	4	4
12	9.0	6	6	6	6	4	4	4
13	10.0	6	6	6	6	6	4	4
14	11.0	6	6	6	6	6	4	4
15	NULL							

PF index 15 is used to indicate NULL, or no PF, and is useful in MCW assignments that utilize fewer than the maximum number of layers for transmission (see 7.4.6.3.1).

7.5.6.7.2 MIMO MCW mode packet formats

The packet format consists of five bits. However, the first bit is sometimes omitted in the packet format field of SS MAC blocks when multiple blocks are used to communicate assignment information. In this case, the omitted first bit shall be assumed to be the same as the (non-omitted) first bit of the associated SS MAC block that contains all five packet format bits.

The first bit of the PF along with the parameter NumLayers (for the interlace being assigned) indicates the F-DPICH format as described in the F-DPICH section of the physical layer protocol. If NumLayers is equal to '1' or '2', then the F-DPICH format used is F-DPICH format 0 if the first bit is equal to '0' or F-DPICH format 1 if the first bit is equal to '1'. If NumLayers is equal to '3', the F-DPICH format used is F-DPICH format 0 (the first bit is ignored). If NumLayers is equal to '4', the F-DPICH format used is F-DPICH format 2 (the first bit is ignored). The remaining four bits index the spectral efficiency, maximum number of transmissions, and the modulation format to be used for each transmission of a data packet. The indexing is described in Table 71.

7.5.6.7.3 MIMO SCW mode packet formats

The packet format consists of five bits. The first bit of the PF along with the parameter NumLayers (for the interlace being assigned) indicates the F-DPICH format as described in the F-DPICH section of the physical layer protocol. If NumLayers is equal to '1' or '2', then the F-DPICH format used is F-DPICH format 0 if the first bit is equal to '0' or F-DPICH format 1 if the first bit is equal to '1'. If NumLayers is equal to '3', the F-DPICH format used is F-DPICH format 0 (the first bit is ignored). If NumLayers is equal to '4', the F-DPICH format used is F-DPICH format 2 (the first bit is ignored). The remaining four bits index the spectral efficiency, maximum number of transmissions, and the modulation format to be used for each transmission of a data packet. The indexing is described in Table 71.

7.5.7 Header and trailer formats

7.5.7.1 Header and trailer for unicast transmissions

The access network shall formulate a unicast packet for transmission over the Forward Traffic Channel using the following header and trailer:

7.5.7.1.1 Header

Field	Length (bits)
UATIInfoIncluded	1
IsSecure	1
KeyChange	1
InBandControlIncluded	1
Reserved	4
ATIdentifier	0 or 128
UATIStatus	0 or 4
FailureCode	0 or 4
0 or more of the following fields	
InBandControl	8

UATIInfoIncluded — The access network shall set this field to 1 if a UATI is included, and to zero otherwise. The access network shall set UATIInfoIncluded to ‘1’ for the first packet sent on the F-DCH after the receipt of a packet with UATIInfoIncluded equal to ‘1’ on the R-DCH.

IsSecure — The access network shall set this field to ‘1’ if the packet is secured by the Authentication and Encryption protocols. The access network shall set this field to ‘0’ otherwise.

KeyChange — This field shall be set by the Security Sublayer at the transmitter and communicated to the Security Sublayer along with the payload at the receiver.

InBandControlIncluded — Used to signal the existence of in-band control bits. The access network shall set this field to ‘1’ if the InBandControl field is present. Otherwise, the access network shall set this field to ‘0’. InBandControl messages are defined in 7.5.7.1.3.

Reserved — The sender shall set this field to ‘0000’. The receiver shall ignore this field.

ATIdentifier — If UATIInfoIncluded is 1, the access network shall set this field to the last ATIdentifier sent by the access terminal. Otherwise, the access network shall omit this field.

UATIStatus — The access network shall include this field if UATIInfoIncluded is set to 1.

Table 72 Encoding of the UATIStatus field

Field value	Description
0x0	Reset timer
0x1	Expire timer
0x2	Ignore
All other values are reserved	

FailureCode — The access network shall set this field to a non-zero value if UATIInfoIncluded is set to 1 and UATIStatus is set to 1.

Table 73 Encoding of the FailureCode field

Field value	Description
0x00	No failure
0x01	Token not supported
0x02	Network Busy
0x03	Authentication or billing failure
0x04	Desired QoS unavailable
0x05	No route to host
0x06	Network maintenance
0x07	Connection closed due to terminal request
0x08	SessionLost
0x09	General failure
All other values are reserved	

InBandControl — Included only if InBandControl set to ‘1’. InBandControl messages are defined in 7.5.7.1.3.

7.5.7.1.2 Trailer

This protocol does not add a trailer.

7.5.7.1.3 FLInBandControl

7.5.7.1.3.1 InBandPacketFormatSwitchSISO block

This block allows the access network to change the packet format being used for forward link transmissions. The block has the following format

Field	Length (bits)
ContinuationBit	1
BlockFormat	2
PacketFormat	6
Reserved	7

ContinuationBit — The access network shall set the continuation bit to 0 if this is the last FLInBandControl block in the MAC header. Otherwise, the access terminal shall set the continuation bit to 1.

BlockFormat — The access network shall indicate an InBandPacketFormat level block by setting the format indicator bits to 00.

PacketFormat — The access network shall set this field to indicate the packet format that is to be used on subsequent transmissions on the same interlace. See 7.5.6.7.1 for the definition of this field.

Reserved — The access network shall set this field to '0000000'. The access terminal shall ignore this field.

7.5.7.1.3.2 InBandPacketFormatSwitchMIMOSCW block

This block allows the access network to change the packet format being used for forward link transmissions. The block has the following format

Field	Length (bits)
ContinuationBit	1
BlockFormat	2
PacketFormat	5
NumLayers	2
Reserved	6

ContinuationBit — The access network shall set the continuation bit to 0 if this is the last FLInBandControl block in the MAC header. Otherwise, the access terminal shall set the continuation bit to 1.

BlockFormat — The access network shall indicate an InBandPacketFormat level block by setting the format indicator bits to 01.

PacketFormat — The access network shall set this field to indicate the packet format that is to be used on subsequent transmissions on the same interlace. See 7.5.6.7.3 for the definition of this field.

NumLayers — The access network shall set this field to indicate the number of layers that is to be used on subsequent transmission on the same interlace.

Reserved — The access network shall set this field to ‘000000’. The access terminal shall ignore this field.

7.5.7.1.3.3 InBandPacketFormatSwitchMIMOMCW block

This block allows the access network to change the packet format being used for forward link transmissions. The block has the following format

Field	Length (bits)
ContinuationBit	1
BlockFormat	2
PacketFormat*1*	4
PacketFormat*2*	4
PacketFormat*3*	4
PacketFormat*4*	4
Reserved	6

ContinuationBit — The access network shall set the continuation bit to 0 if this is the last FLInBandControl block in the MAC header. Otherwise, the access terminal shall set the continuation bit to 1.

BlockFormat — The access network shall indicate an InBandPacketFormat level block by setting the format indicator bits to 01.

PacketFormat*i* — The access network shall set this field to indicate the packet format that is to be used on layer *i* of subsequent transmissions on the same interlace. See 7.5.6.7.2 for the definition of this field.

Reserved — The access network shall set this field to ‘000000’. The access terminal shall ignore this field.

7.5.7.2 Header and trailer for broadcast transmissions

The access network shall formulate a broadcast packet for transmission using the following header and trailer:

7.5.7.2.1 Header

Field	Length (bits)
IsSecure	1
NumATIRecords	5
NumATIRecords number of the following 2 fields	
ATIRecord	2 or 34
OpenConnectionRequired	1
Reserved	0-7

IsSecure — The access network shall set this field to '1' if the packet is secured by the Authentication and Encryption protocols. The access network shall set this field to '0' otherwise.

NumATIRecords — This field shall be set to the number of ATIRecords included in the header.

ATI Record — Access Terminal Identifier Record. The access network shall set this field to the record specifying the access terminal's address. This record is defined in the Common Algorithms chapter (10) of this specification.

OpenConnectionRequired — If this field is set to '1', and the ATI record field matches as specified in 7.2.6.6.1.1, the terminal attempts to open a connection. This field may trigger a *PageReceived* indication at the access terminal.

Reserved — The length of this field shall be such that the entire message is octet-aligned. The sender shall set this field to zero. The receiver shall ignore this field.

7.5.7.2.2 Trailer

This protocol does not add a trailer.

7.5.8 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes.

7.5.9 Interface to other protocols

7.5.9.1 Commands sent

- *ReverseControlChannelMAC.ChangeFLSS*

7.5.9.2 Indications

This protocol registers to receive the following indications:

- *PhysicalLayer.ForwardDataCompleted*
- *ConnectedState.TunedAway*
- *RCCMAC.FLSSChanged*

7.5.10 Configuration attributes

The negotiable attributes for this protocol are listed in Table 74. The access terminal shall use as defaults the values in Table 74 that are listed in ***bold italics***.

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Forward Traffic Channel MAC Protocol.

Table 74 Configurable attributes

Attribute ID	Attribute	Values	Meaning
0x00	InactivityTimeout	0x0000, 0x0100-0xffff	Reserved
		0x0002	InactivityTimeout=2 seconds
		0x0001-0x00ff	IactivityTimeout in seconds

7.5.11 Protocol numeric constants

Constant	Meaning	Value
$N_{FTCMPType}$	Type field for this protocol	Table 9
$N_{FTCMPDefault}$	Subtype field for this protocol	0x0000
$MACID_{BROADCAST}$	Broadcast MACID	0x0

7.5.12 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

7.6 Default Reverse Control Channel MAC Protocol

7.6.1 Overview

The Reverse Control Channel MAC defines the procedures for transmissions on the following Physical Layer channels: R-CQICH, R-BFCH, R-SFCH, R-PICH, R-REQCH, and R-ACKCH.

The R-CQICH is used by the access terminal to transmit the FL quantized channel quality from different sectors to the access network. The R-SFCH is a feedback channel, that is used by the access terminal to transmit the quantized FL channel quality measured for a subband in the FL serving sector (FLSS). The R-BFCH is a feedback channel, that is used by the access terminal to transmit the beam index defined in the Appendix, as well as supplemental channel quality information to enable SDMA transmission. SDMA transmission is defined in the Physical Layer Protocol. The R-PICH is a broadband pilot channel, and the R-REQCH is used by the access terminal to request resources. The R-ACKCH is used by the access terminal to acknowledge the FL MAC packets.

The access network maintains an instance of this protocol for every access terminal.

This protocol operates in one of two states:

- *Inactive State*: In this state, the access terminal is not assigned a MACID and shall not transmit on the Reverse Traffic Channel. When the protocol is in this state, it waits for an *Activate* command.
- *Active State*: In this state, the access terminal is assigned a MACID and may transmit data on the Reverse Traffic Channel.

The protocol states and the indications and events causing the transition between the states are shown in Figure 78.

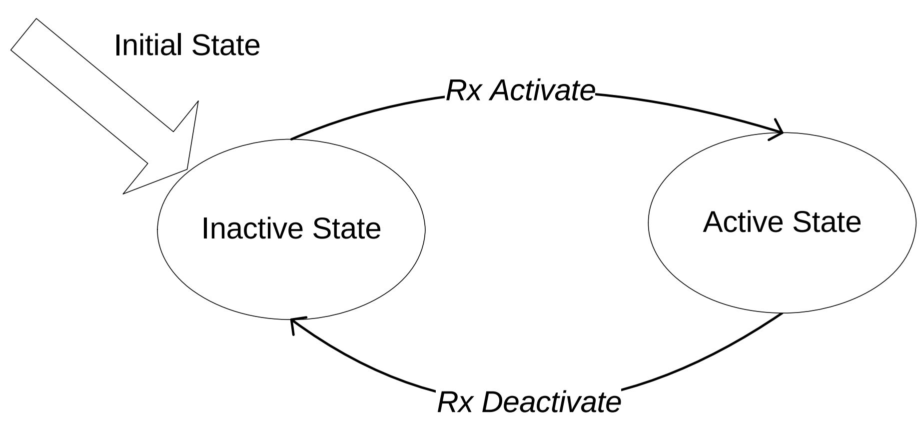

Figure 78 Default Reverse Control Channel MAC Protocol state diagram

This protocol shall use the following parameters and attributes.

Parameter Name	Where Defined	Comments
CQIReportingMode	ASMP	ActiveSetAssignmentMsg
CQIReportInterval	ASMP	ActiveSetAssignmentMsg
CQIReportPhase	ASMP	ActiveSetAssignmentMsg
CQIPilotReportInterval	ASMP	ActiveSetAssignmentMsg
CQIPilotReportPhase	ASMP	ActiveSetAssignmentMsg
BFCHReportRate	ASMP	ActiveSetAssignmentMsg
SFCHReportRate	ASMP	ActiveSetAssignmentMsg
BFCHPowerOffset	OMP	ExtendedChannelInfo
NumBFCHBits	ASMP	ActiveSetAssignmentMsg
SFCHPowerOffset	OMP	ExtendedChannelInfo
NumSFCHBits	ASMP	ActiveSetAssignmentMsg
BFCHBeamCodeBookIndex	OMP	ExtendedChannelInfo
MandatoryCQICHCTRLReportingPeriod	ASMP	ActiveSetAssignmentMsg
FLNumSDMADimensions	OMP	QuickChannelInfo block
RACKBandwidthFactor	OMP	ExtendedChannelInfo
ActiveSetIndex	ASMP	ActiveSetAssignmentMsg
MultiCarrierOn	Physical Layer Protocol	Public data
ActiveCarriers	SS MAC Protocol	Public data
REQCarrier	SS MAC Protocol	Public data
EffectiveNumAntennas	OMP	QuickChannelInfo block
MinRequestInterval	ASMP	ActiveSetAssignmentMsg
RPICHEnabled	ASMP	ActiveSet Assignment Msg
ProbePower	ACMP	public data
OpenLoopTransitionTime	Configuration Attribute	
RLCtrlPCMode	OMP	ExtendedChannelInfo
PowerControlStepUp	ASMP	ActiveSetAssignmentMsg
PowerControlStepDown	ASMP	ActiveSetAssignmentMsg
ACKCtrlOffset	pBCH1	Part of pBCH1 transmission
ACKChannelGain*j*, j = 0 to 9	Configuration Attribute	
ACKStepUpSize	Configuration Attribute	
ACKExtendedFrameGain	Configuration Attribute	
REQChannelGain*j*, j=0,1,2,3	OMP	ExtendedChannelInfo
CtrlAccessOffset	OMP	ExtendedChannelInfo
PICHPowerOffset	OMP	ExtendedChannelInfo
CQICHPowerBoostForHandoff	Configuration Attribute	
REQCHPowerBoostForHandoff	Configuration Attribute	
VCQIMeasureInterval	ASMP	ActiveSetAssignmentMsg
SingleServingSector	ASMP	ActiveSetAssignmentMsg

7.6.2 Primitives

7.6.2.1 Commands

This protocol defines the following commands:

- *ChangeFLSS*
- *ChangeRLSS*
- *Activate*
- *Deactivate*

7.6.2.2 Return indications

The protocol returns the following indications:

- *FLSSChanged*
- *RLSSChanged*
- *DFLSSChanged*
- *DRLSSChanged*

7.6.3 Public data

7.6.3.1 Static public data

This protocol does not define any static public data.

7.6.3.2 Dynamic public data

- Subtype for this protocol
- FLSS
- RLSS
- DFLSS
- DRLSS

7.6.4 Protocol data unit

This is a control protocol and does not carry any payload on the behalf of other protocols.

7.6.5 Protocol initialization and swap

7.6.5.1 Protocol initialization

Upon initialization at the access terminal,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Inactive State.

Upon initialization at the access network,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Inactive State.

7.6.5.2 Protocol swap

Upon swap, the protocol instance shall enter the Inactive State.

7.6.6 Procedures

7.6.6.1 Command processing

7.6.6.1.1 Activate

If the protocol receives an *Activate* command in the Inactive State, the access terminal and the access network shall perform the following:

- Transition to the Active State

If the protocol receives this command in any other state, the command shall be ignored.

7.6.6.1.2 Deactivate

If the protocol receives a *Deactivate* command in the Active State:

- The access terminal shall cease transmitting the Reverse Control Channel and shall transition to the Inactive State.
- The access network shall cease monitoring the Reverse Control Channel from this access terminal and shall transition to the Inactive State.

If the protocol receives a *Deactivate* command in the Inactive State, the command shall be ignored.

7.6.6.2 Inactive State

The access terminal shall not transmit on the physical channels governed by this protocol.

7.6.6.3 Active State

The access terminal shall transmit on the physical channels governed by this protocol according to the procedures in this section. This section makes frequent use of the following terms defined for each active access terminal:

Active Set	The set of sectors for which the access terminal has an assigned MACID.
DFLSS	(Desired Forward Link Serving Sector) The sector within the active set that the access terminal autonomously determines is the best sector for forward link data transmissions.

DRLSS	(Desired Reverse Link Serving Sector) The sector within the active set that the access terminal autonomously determines is the best sector for reverse link data transmissions.
FLSS	(Forward Link Serving Sector) The last DFLSS from which the access terminal successfully received a sticky FLAB with supplemental field ‘0’, or a non-sticky FLAB with a unicast MACID (see 7.5.6.4.1.2.1).
RLSS	(Reverse Link Serving Sector) The last DRLSS from which the access terminal successfully received an RLAB with supplemental field ‘0’ or a non-sticky RLAB with a unicast MACID (see 7.7.6.4.1.1.1).
Desired Subband	The subband that the access terminal autonomously determines is the best segment for forward link data transmission.
SSCH Subbands	Subbands where F-SSCH is transmitted.
Control Segment	Set of hop-ports in selected RL PHY Frames, used by the access terminal to transmit R-CQICH, R-REQCH, R-PICH, R-SFCH, and R-BFCH.
ControlSegmentPeriod	Time duration between consecutive Control Segment transmissions to the RLSS, in units of number of RL PHY Frames.
TDM2 Power Density	Power density of the second OFDM symbol in the F-ACQCH in the Physical Layer Protocol.

In this section, the parameters CQIReportingMode, CQIReportInterval, CQIReportPhase, CQIPilotReportInterval, CQIPilotReportPhase, BFCHReportRate, SFCHReportRate, NumSFCHBits, NumBFCHBits, MandatoryCQICHCTRLReportingPeriod, RPICHEnabled, and VCQIMeasureInterval correspond to the FLSS, and the parameter MinRequestInterval corresponds to the RLSS or DRLSS, depending on the sector to which the R-REQCH is being transmitted.

7.6.6.3.1 Synchronous Subsets

A synchronous subset contains a set of sectors that are determined to be synchronous according to the most recent Active Set Assignment message received by the access terminal. Each ChannelBand, as defined in Chapter 1, contains one or more synchronous subsets.

- The DRLSS, RLSS, and FLSS of the access terminal shall be members of the same synchronous subset.
- A synchronous subset that contain the FLSS is referred to as a serving synchronous subset. A synchronous subset that does not contain the FLSS is referred to as a non-serving synchronous subset.
- The access terminal shall not transmit R-CQICH, R-BFCH, R-SFCH, R-PICH, and R-REQCH on Control Segment of sectors that are not in the ChannelBand of the FLSS.
- The access terminal may transmit the R-CQICH on the Control Segment of sectors of non-serving synchronous subsets. The access terminal shall not transmit R-BFCH, R-SFCH, R-PICH, and R-REQCH on the Control Segment of sectors of non-serving synchronous subsets.

7.6.6.3.2 Serving Sector Maintenance

If the multi-carrier mode is set to MultiCarrierOn,

- the RLSS shall be the same for all carriers indicated by ActiveCarriers, and
- the FLSS shall be the same for all carriers indicated by ActiveCarriers, and
- the DFLSS shall be the same for all carriers indicated by ActiveCarriers, and
- the DRLSS shall be the same for all carriers indicated by ActiveCarriers.

When this protocol receives a *ChangeFLSS* command, it shall perform the following:

- The protocol shall update the FLSS in the Public data to be equal to the FLSS in the *ChangeFLSS* command,
- issue a *FLSSChanged* indication,
- the protocol shall update the DFLSS in the Public data of this protocol to be equal to the FLSS in the *ChangeFLSS* command,
- issue a *DFLSSChanged* indication.
- In addition, if the FLSS in the *ChangeFLSS* command is a member of a non-serving synchronous subset, or if the SingleServingSector bit is set to '1', the protocol shall additionally perform the following
    - the protocol shall update the RLSS in the Public data of this protocol, to be equal to the FLSS in the *ChangeFLSS* command,
    - issue a *RLSSChanged* indication,
    - the protocol shall update the DRLSS in the Public data of this protocol, to be equal to the FLSS in the *ChangeFLSS* command,
    - issue a *DRLSSChanged* indication.

When this protocol receives a *ChangeRLSS* command, it shall perform the following:

- If the RLSS in the *ChangeRLSS* command is a member of a non-serving synchronous subset, the protocol shall ignore the *ChangeRLSS* command.
- Otherwise,
    - it shall update the RLSS in the Public data of this protocol,
    - issue a *RLSSChanged* indication,
    - it shall update the DRLSS in the Public data of this protocol,
    - issue a *DRLSSChanged* indication.
    - If the SingleServingSector bit is set to '1' for the RLSS in the *ChangeRLSS* command, the protocol shall additionally perform the following
        - the protocol shall update the FLSS in the Public data of this protocol, to be equal to the RLSS in the *ChangeRLSS* command,
        - issue a *FLSSChanged* indication,

- the protocol shall update the DFLSS in the Public data of this protocol, to be equal to the RLSS in the *ChangeRLSS* command,
- issue a *DFLSSChanged* indication.

7.6.6.3.3 Reverse Link CQI Reporting Modes

The RCC MAC Protocol has multiple reporting modes. The access terminal's reporting mode shall be set based upon the CQIReportingMode. The access terminal's CQIReportingMode can be one of the following:

- the Single Code Word CQI Reporting Mode(CQISCW),
- the Multiple Code Word CQI Reporting Mode (CQIMCW),
- the SISO CQI Reporting Mode (CQISISO).

7.6.6.3.4 Access Terminal Procedures for Determining RL Channel Quality

For the purposes of determining the RL channel quality, the access terminal shall monitor the FL CQI Erasure Indication (CEI) bits from all sectors in each synchronous subset for which R-CQICH is transmitted on the corresponding Control Segment. If the multi-carrier mode is set to MultiCarrierOn, the access terminal shall independently monitor the FL CEI bit sent on carriers indicated by ActiveCarriers.

For each sector and carrier, the access terminal shall interpret the first FL CEI bit intended for the access terminal in the F-SSCH, following an R-CQICH message, according to the following interpretation rules:

- If the FL CEI bit value is equal to 1, then the access terminal shall interpret the FL CEI bit as indicating that the R-CQICH transmission was erased at the corresponding sector and carrier; otherwise, the access terminal shall interpret the FL CEI bit as indicating that the last R-CQICH transmission was not erased at the corresponding sector and carrier,
- If an R-CQICH report has not occurred since the previous FL CEI bit from a given sector and carrier, intended for the access terminal, then the current FL CEI bit from the corresponding sector and carrier does not indicate an R-CQICH erasure.
- If CQICHPowerBoostForHandoff is not zero, then the AT shall ignore the FL CEI bits from all the sectors within the synchronous subset that contains the DFLSS following an R-CQICH report that carries an FL handoff request (i.e., if the DFLSS is different from the FLSS, the CQI is a CQICHCTRL report, the ActiveSetIndex field in the CQI is the active set index of the DFLSS, and the DFLSS flag in the CQI is set to '1').

Following the above rules, the access terminal shall compute the estimated erasure rates for all sectors and carriers, for each synchronous subset for which R-CQICH is transmitted on the corresponding Control Segment.

7.6.6.3.5 Access Terminal Procedures for Power Control

If the multi-carrier mode is set to MultiCarrierOff, the access terminal shall initialize an independent parameter P_{CTRL} for each synchronous subset. If the multi-carrier mode is set to MultiCarrierOn, the access terminal shall initialize an independent parameter P_{CTRL} for each synchronous subset, for each carrier indicated by ActiveCarriers. The initial value of the P_{CTRL} is computed as

$$P_{CTRL} = ProbePower + CtrlAccessOffset$$

Independent ProbePower parameters are maintained for each synchronous subset. Each ProbePower parameter refers to the mean output power of the Access Channel preamble for the corresponding synchronous subset, at the end of the last successful Access Channel probe. If the multi-carrier mode is set to MultiCarrierOn, independent ProbePower parameters are maintained for each synchronous subset, for each carrier indicated by ActiveCarriers. In this case, each ProbePower parameter refers to the mean output power of the Access Channel preamble for the corresponding synchronous subset and carrier, at the end of the last successful Access Channel probe.

7.6.6.3.5.1 Open Loop Power Adjustment

If OpenLoopTransitionTime is set to zero, the access terminal shall not perform the open loop power adjustment procedures described in this section. Otherwise, the access terminal shall measure the mean received power during each superframe preamble of the RLSS, and compare it to the mean received power measured during the previous superframe preamble of the RLSS. Following a step change of ReceivedPowerChange dB in the mean received superframe preamble power since the previous superframe preamble of the RLSS, each P_{CTRL} value shall transition to its final value of P_{CTRL}-RceivedPowerChange according to the following rule:

$$P_{CTRL}(t) = P_{CTRL}(0) - ReceivedPowerChange \times \min\left\{\frac{t}{OpenLoopTransitionTime}, 1\right\},$$

where t denotes the time since the last superframe preamble of the RLSS and is measured in the same units as OpenLoopTransitionTime. The above expression defines the step response for this transition, i.e., if multiple changes are measured within a single transition time, their effects on each P_{CTRL} value shall be accumulated.

7.6.6.3.5.2 Closed Loop Power Adjustment

Define PCSector(non-serving synchronous subset) as the sector with the lowest estimated erasure rate in the non-serving synchronous subset. PCSector(non-serving synchronous subset) is defined for each non-serving synchronous subset for which R-CQICH is transmitted and erasure rate is measured.

The access terminal shall treat the FL CEI bit intended for the access terminal that is sent over the F-SSCH of the PCSector(non-serving synchronous subset) for each non-serving synchronous subset as the power control command for that non-serving synchronous subset. If RLCtrlPCMode for the RLSS is set to ‘UpDown’, then the access terminal shall treat the FL PC bit intended for the access terminal that is sent over the F-SSCH of the RLSS as the power control command for the synchronous subset that contains the RLSS. If RLCtrlPCMode for the RLSS is set to ‘ErasureBased’, then the access terminal shall treat the FL CEI bit intended for the access terminal that is sent in the F-SSCH of the RLSS as the power control command for the synchronous subset that contains the RLSS.

The access terminal shall adjust the appropriate P_{CTRL} in response to FL power control command (PC or CEI bits, as explained below) intended for the access terminal that is sent over the F-SSCH of the RLSS and PCSector (non-serving synchronous subset) for each non-serving synchronous subset. If the multi-carrier mode is set to MultiCarrierOn, the access terminal shall independently adjust appropriate P_{CTRL} for each carrier indicated by ActiveCarriers, in response to the power control commands (PC or CEI bits, as explained below) for the corresponding carrier.

Each P_{CTRL} is adjusted independently using the PowerControlStepUp or PowerControlStepDown parameters corresponding to the RLSS (for the serving synchronous subset) or PCSector (non-serving synchronous subset) (for each non-serving synchronous subset), according to the following rules.

- For each non-serving synchronous subset, and for the synchronous subset that contains the RLSS, if RLCtrlPCMode for the RLSS is set to 'ErasureBased',
  - When the CEI bit transmitted over the appropriate F-SSCH is '1', the access terminal shall increase the corresponding P_{CTRL} by PowerControlStepUp dB.
  - When the CEI bit transmitted over the appropriate F-SSCH is '0', the access terminal shall decrease the corresponding P_{CTRL} by PowerControlStepDown dB.
- For the synchronous subset that contains the RLSS, if RLCtrlPCMode for the RLSS is set to 'UpDown'
  - When the PC bit transmitted over the F-SSCH of the RLSS is '1', the access terminal shall increase the corresponding P_{CTRL} by PowerControlStepUp dB.
  - When the PC bit transmitted over the F-SSCH of the RLSS is '0', the access terminal shall decrease the corresponding P_{CTRL} by PowerControlStepUp dB.
- If CQICHPowerBoostForHandoff is not zero, and the DFLSS does not belong to the same synchronous subset as the RLSS, then the AT shall ignore the FL CEI bits from that synchronous subset, following an R-CQICH report that carries an FL handoff request (i.e., if the DFLSS is different from the FLSS, the CQI report is a CQICHCTRL, the ActiveSetIndex field in the CQI is the active set index of the DFLSS, and the DFLSS flag in the CQI is set to '1').
- If CQICHPowerBoostForHandoff is not zero, the DFLSS belongs to the same synchronous subset as the RLSS, and RLCtrlPCMode for the RLSS is set to 'ErasureBased', then the AT shall ignore the FL CEI bit from the RLSS, following an R-CQICH report that carries an FL handoff request (i.e., if the DFLSS is different from the FLSS, the CQI report is a CQICHCTRL, the ActiveSetIndex field in the CQI is the active set index of the DFLSS, and the DFLSS flag in the CQI is set to '1').

These changes in P_{CTRL} values shall be in addition to any changes dictated by the open-loop power control algorithm described in Section 7.6.6.3.5.1.

The mean output power of reverse control channels for the RLSS and PCSector(non-serving synchronous subset), and if the multi-carrier mode is set to MultiCarrierOn, for each carrier indicated by ActiveCarriers, shall be independently derived from the corresponding P_{CTRL}, as described below.

The R-CQICH transmit power shall be computed according to the following rules:

- If the DFLSS is different from the FLSS, the CQI report is a CQICHCTRL, the ActiveSetIndex field in the CQI is the active set index of the DFLSS, and the DFLSS flag in the CQI is set to '1', then

$$P_{CQICH} = P_{CTRL} + CQICHPowerBoostForHandoff$$

- Otherwise,

$$P_{CQICH} = P_{CTRL}$$

The R-PICH transmit power shall be computed for the RLSS as

$$P_{PICH} = P_{CTRL} + \text{PICHPowerOffset}$$

The R-BFCH transmit power shall be computed for the RLSS as

$$P_{BFCH} = P_{CTRL} + \text{BFCHPowerOffset}$$

The R-SFCH transmit power shall be computed only for the RLSS as

$$P_{SFCH} = P_{CTRL} + \text{SFCHPowerOffset}$$

The R-REQCH transmit power shall be computed for the RLSS and DRLSS according to the following rules:

- If the DRLSS is different from the RLSS, and the ActiveSetIndex field in the REQ is the active set index of the DRLSS, then

$$P_{REQCH} = P_{CTRL} + REQChannelGain_j + REQCHPowerBoostForHandoff$$

- Otherwise,

$$P_{REQ} = P_{CTRL} + REQChannelGain_j$$

where j is the QoSFlow field of the request message being transmitted (Table 90).

The R-ACKCH transmit power shall be computed only for the FLSS according to the following rules:

$$P_{ACK} = P_{CTRL} + ACKCtrlOffset + ACKChannelGain + ACKChannelGainAdjustment + ACKExtendedFrameGain$$

where ACKCtrlOffset is a parameter transmitted over pBCH1 of the FLSS. The access terminal shall use the estimated erasure rate seen by the sector for which the Acknowledgment is intended, and shall set the ACKChannelGain to ACKChannelGain*j* when the estimated erasure rate is between j*10% and (j+1)*10%, where j can take the integer values 0,1,2,..,8,9. The value of the ACKChannelGainAdjustment (dB) shall always be greater than or equal to 0 dB. The access terminal shall only use a non-zero ACKChannelGainAdjustment value if the access terminal is able to detect an ACK-to-NACK error event reliably with accuracy higher than 99 percent. ACKChannelGainAdjustment shall be set to zero at the beginning of every MAC packet transmission and shall increase by ACKStepUpSize dB every time an ACK-to-NACK error event is detected by the access terminal. The access terminal shall use a non-zero ACKExtendedFrameGain value if the access terminal is transmitting an acknowledgement for a FL MAC packet sent on an E-PHY Frame.

If the AT is transmitting packets on multiple interlaces, then it shall maintain independent values of ACKChannelGainAdjustment and ACKExtenedFrameGain for each interlace. Then, the AT shall use the interlace specific gain values when transmitting R-ACKCH associated with that interlace.

If the access terminal is unable to transmit the Physical Layer channels corresponding to the RCC MAC protocol and the RTC MAC protocol at the specified power due to access terminal's transmit power constraint, the access terminal shall transmit at the maximum transmit power. In this scenario, the access terminal may independently decide the relative transmission priority of these channels. In this scenario, the access terminal may independently decide to gate-off the transmission of one or more physical layer channels corresponding to RCC MAC protocol and RTC MAC protocol.

The access terminal shall not modify the power control variable P_{CTRL} after *ConnectedState.TunedAway* indication and until the *ConnectedState.TunedBack* indication. The access terminal shall not transmit the Physical Layer channels corresponding to the RCC MAC protocol and shall not monitor the FL PC, FL CEI, and Fast OSI value after *ConnectedState.TunedAway* indication and until the *ConnectedState.TunedBack* indication.

7.6.6.3.6 Procedures for the R-CQICH Physical Layer channel

The access terminal measures the FL channel quality on the SSCH subbands and transmits the quantized channel quality to the access network using the R-CQICH. The computation and signaling method are a function of the access terminal's CQIReportingMode, CQIReportInterval, CQIReportPhase, CQIPilotReportInterval, CQIPilotReportPhase and MandatoryCQICHCTRLReportingPeriod.

If the multi-carrier mode is set to MultiCarrierOn, the CQIReportingMode, CQIReportInterval, CQIPilotReportInterval, and MandatoryCQICHCTRLReportingPeriod are identical across carriers indicated by ActiveCarriers.

The RCC MAC Protocol shall provide the Physical Layer Protocol with the P_{CQICH}, RLCTRLCarrier, PilotPN, and MACID for each R-CQICH transmission. The PilotPNand MACID may correspond to different sectors depending on the CQI report and synchronous subset, as discussed in 7.6.6.3.6.1 The P_{CQICH} is determined according to the rules in 7.6.6.3.5. If the multi-carrier mode is set to MultiCarrierOn, the RLCTRLCarrier indicates the carrier, where the given R-CQICH is transmitted. If the multi-carrier mode is set to MultiCarrierOff, the RLCTRLCarrier is not defined.

7.6.6.3.6.1 Reporting Rules

A detailed description of the different CQI reports referred to in this section, can be found in 7.6.6.3.6.2.

The access terminal shall transmit R-CQICH on the Control Segment of the RLSS. In addition, the access terminal may transmit R-CQICH on the Control Segments of sectors in one or more non-serving synchronous subsets. If the DFLSS is in a non-serving synchronous subset, the access terminal shall transmit R-CQICH on the control segment corresponding to the DFLSS, or issue command ACMAC.AttemptAccess(DFLSS).

If the multi-carrier mode is set to MultiCarrierOn, the access terminal shall transmit independent R-CQICH on the carriers indicated by ActiveCarriers. For the synchronous subset containing the FLSS, the PilotPN, MACID, and RLCTRLCarrier shall correspond to the current FLSS, unless (RL PHY Frame Index – CQIPilotReportPhase) mod CQIPilotReportInterval =0. For non-serving synchronous subsets, the PilotPNand MACID shall correspond to the first sector of the synchronous subset (Indicated by the field PilotPN) listed in the last ActiveSetAssignment message received by the access terminal.

If (RL PHY Frame Index – CQIPilotReportPhase) mod CQIPilotReportInterval =0, the PilotPNand MACID, shall correspond to the first sector listed for each synchronous subset in the last ActiveSetAssignment message received by the access terminal.

The access terminal shall not transmit the R-CQICH after *ConnectedState.TunedAway* indication and until the *ConnectedState.TuneBack* indication.

If (RL PHY Frame Index – CQIPilotReportPhase) mod CQIPilotReportInterval =0 then the CQICHPilot shall be reported. If (RL PHY Frame Index – CQIPilotReportPhase) mod CQIPilotReportInterval ≠ 0 and (RL PHY Frame Index – CQIReportPhase) mod CQIReportInterval =0 then CQICHCTRL, CQICHSCW, or CQICHMCW shall be reported.

The CQIReportIndex shall be computed from the CQIReportInterval and CQIReportPhase as CQIReportIndex = (RL PHY Frame Index – CQIReportPhase) / CQIReportInterval.

If CQIReportIndex mod MandatoryCQICHCTRLReportingPeriod =0 and (RL PHY Frame Index – CQIPilotReportPhase) mod CQIPilotReportInterval ≠ 0, then CQICHCTRL shall be reported.

If (RL PHY Frame Index – CQIPilotReportPhase) mod CQIPilotReportInterval ≠ 0 and the CQI is reported for a sector other than the FLSS, then CQICHCTRL shall be reported.

If (RL PHY Frame Index – CQIPilotReportPhase) mod CQIPilotReportInterval ≠ 0 and DFLSS ≠ FLSS then CQICHCTRL shall be reported.

If 2 < EffectiveNumAntennas ≤ 4, the CQICHMCW report consists of 2 parts. The CQICHMCW report with the MCWIndex field set to 0, is the first part. However, if EffectiveNumAntennas ≤ 2, the CQICHMCW report consists of only the first part. The CQICHMCW report is not defined if EffectiveNumAntennas > 4.

Provided the above rules are not violated, then the AT may determine which CQI reports to use consistent with its CQIReportingMode.

7.6.6.3.6.2 CQI Report

The CQI reports for each CQIReportingMode and R-CQICH are shown in Table 75.

Table 75 CQI Reports for each CQIReportingMode

	CQIReportingMode		
	CQISISO	**CQISCW**	**CQIMCW**
CQI Report	CQICHPilot	CQICHPilot	CQICHPilot
	CQICHCTRL	CQICHCTRL	CQICHCTRL
		CQICHSCW	CQICHMCW

The format for the CQICHPilot report is shown in Table 76.

Table 76 Format for CQICHPilot report

Field	Length (bits)
ReservedValue	10

ReservedValue — The ReservedValue is set to 0.

The format for the CQICHCTRL is shown in Table 77.

Table 77 Format for CQICHCTRL report

Field	Length (bits)
FormatType	1
CQIValueSISO	4
DFLSSFlag	1
ActiveSetIndex	3
Reserved	1

FormatType — This bit is set to the value 0.

CQIValueSISO — Indicates FL SISO CQI value. See 7.6.6.3.6.3 for details.

DFLSSFlag — If the ActiveSetIndex is the current DFLSS, the DFLSSFlag bit shall be set to 1; otherwise, the DFLSSFlag bit shall be set to 0. If the multi-carrier mode is set to MultiCarrierOn, the DFLSSFlag shall be the same for the CQICHCTRL report transmitted on all carriers indicated by ActiveCarriers.

ActiveSetIndex — Indicates the sector to which the CQIValueSISO corresponds. If the multi-carrier mode is set to MultiCarrierOn, the ActiveSetIndex shall be the same for the CQICHCTRL report transmitted on all carriers indicated by ActiveCarriers.

Reserved — The sender shall set this field to ‘0’. The receiver shall ignore this field.

The format for the CQISCW is in Table 78.

Table 78 Format for CQICHSCW report

Field	Length (bits)
FormatType	1
CQIValueSCW	5
Rank	2
Reserved	2

FormatType: This bit is set to the value 1.

CQIValueSCW: Indicates FL MIMO SCW CQI value for the reported Rank. See 7.6.6.3.6.3 for details.

Rank: Indicates the desired number of MIMO layers in the FL MIMO SCW transmission.

Reserved: The sender shall set this field to '00'. The receiver shall ignore this field.

If $2 < \text{EffectiveNumAntennas} \le 4$, the CQICHMCW report consists of 2 parts. The format for the first part is shown in Table 79, and the format for the 2^{nd} part is shown in Table 80. However, if $\text{EffectiveNumAntennas} \le 2$, the CQICHMCW report consists of only the first part, shown in Table 79. The CQICHMCW report is not defined if $\text{EffectiveNumAntennas} > 4$.

Table 79 Format of first part of CQICHMCW report

Field	Length (bits)
FormatType	1
MCWIndex	1
CQIValueMCWLayer1	4
CQIValueMCWLayer2	4

FormatType: This bit is set to the value 1.

MCWIndex: This bit is set to the value 0 to indicate the CQI report is the first part of the CQICHMCW report.

CQIValueMCWLayer1: Indicates the FL MIMO MCW layer 1 CQI value. See 7.6.6.3.6.3 for details.

CQIValueMCWLayer2: Indicates the FL MIMO MCW layer 2 CQI value. See 7.6.6.3.6.3 for details.

Table 80 Format of second part of CQICHMCW report

Field	Length (bits)
FormatType	1
MCWIndex	1
CQIValueMCWLayer3	4
CQIValueMCWLayer4	4

FormatType This bit is set to the value 1.

MCWIndex This bit is set to the value 1 to indicate the CQI report is the second part of the CQICHMCW report.

CQIValueMCWLayer3
Indicates the FL MIMO MCW layer 3 CQI value. See 7.6.6.3.6.3 for details.

CQIValueMCWLayer4
Indicates the FL MIMO MCW layer 4 CQI value. See 7.6.6.3.6.3 for details.

7.6.6.3.6.3 CQIValue

Depending on the CQIReportingMode, the CQIValue can indicate the field CQIValueSISO in CQICHCTRL report, the field CQIValueSCW in CQICHSCW report, or the fields CQIValueMCWLayer1, CQIValueMCWLayer2, CQIValueMCWLayer3, and CQIValueMCWLayer4 in CQICHMCW report.

In the BlockHopping mode, the access terminal computes CQIValue assuming the FL PHY Frames are transmitted using beam index 0, in the code book specified by the BFCHBeamCodeBookIndex. The beam index 0 assumes no preferred transmit precoding matrix. See Appendix for details.

Based on an unrestricted observation interval, the access terminal shall report the highest tabulated CQIValue such that a packet transmitted by the access network to the access terminal over all of the hop-ports on the SSCH subbands (and on the corresponding RLCTRLCarrier if the multi-carrier mode is set to MultiCarrierOn) sent at TDM2 Power Density using the specified packet format and the specified number of FL-PHY Frames, terminating 1 PHY Frame before the start of the PHY Frame that the R-CQICH is transmitted, would result in a packet error probability ≤ 0.01. The CQI mappings are shown in Table 81.

Table 81 CQI Mapping to the FL Packet Format and Number of FL-PHY Frames

4-bit CQI Value	5-bit CQI Value	FL Packet Format	Number of FL-PHY Frames
0	0	N/A	0
	1	0	2
1	2	1	4
	3	2	3
2	4	2	1
	5	2	2
3	6	2	3
4	7	3	5
5	8	4	4
	9	4	3
6	10	10	2
	11	5	1
7	12	6	2
	13	8	2
8	14	11	1
	15	12	3
9	16	13	5
	17	14	4
10	18	9	4
	19	13	3
11	20	10	2
12	21	11	2
13	22	12	1
14	23	13	2
15	24	14	2
	25	N/A	N/A
	26	N/A	N/A
	27	N/A	N/A
	28	N/A	N/A
	29	N/A	N/A
	30	N/A	N/A
	31	N/A	N/A

7.6.6.3.6.4 VCQI Support

In this section, if the multi-carrier mode is set to MultiCarrierOn, the subband indices used for VCQI support, shall only include the indices corresponding to the carriers indicated by ActiveCarriers.

Define CQIValueSISOSubband[Interlace] to indicate the FL SISO CQI value for the given subband and FL interlace; CQIValueSCWSubband[Interlace][Rank] to indicate the FL SCW CQI value for the given subband, FL interlace and Rank; CQIValueMCWSubband[Interlace][Layer] to indicate the FL MCW CQI value for each MIMO layer.

Based on an unrestricted observation interval, the access terminal shall compute the CQIValueSISOSubband[Interlace], CQIValueSCWSubband[Interlace][Rank] or the CQIValueMCWSubband[Interlace][Layer] depending on the CQIReportingMode, such that a packet transmitted by the access network to the access terminal over all of the hop-ports on the subband, sent at TDM2 Power Density using the specified packet format and the specified number of FL-PHY Frames, terminating 1 PHY Frame before the start of the PHY Frame where CQIValueSISOSubband[Interlace], CQIValueSCWSubband[Interlace][Rank] or the CQIValueMCWSubband[Interlace][Layer] is computed, would result in a packet error probability ≤ 0.01. The CQI mappings are shown in Table 81.

If the CQIReportingMode is SISO, the access terminal shall compute the report VCQIValueSISO[Interlace][subband] by averaging the CQIValueSISOSubband computed for that particular interlace and subband, over an averaging interval specified by VCQIMeasureInterval, terminating 1 PHY Frame before the VCQI message is reported.

If the CQIReportingMode is SCW, the access terminal shall compute the report VCQIValueSCW[Interlace][subband][Rank] for each Rank, by averaging the CQIValueSCWSubband[Interlace][Rank] computed for that particular subband and Rank, over an averaging interval specified by VCQIMeasureInterval, terminating 1 PHY Frame before the VCQI message is reported.

If the CQIReportingMode is MCW, the access terminal shall compute the report VCQIValueMCW[Interlace] [subband][Layer] by averaging the CQIValueMCWSubband[Interlace] [Layer] computed for that particular subband and MIMO layer, over an averaging interval specified by VCQIMeasureInterval, terminating 1 PHY Frame before the VCQI message is reported.

7.6.6.3.7 Procedures for the R-SFCH Physical Layer channel

The access terminal uses the R-SFCH to transmit the subband index and the quantized FL channel quality corresponding to the reported subband index and the FLSS.

The RCC MAC Protocol shall provide the Physical Layer Protocol with the P_{SFCH}, RLCTRLCarrier, PilotPNand MACID for each R-SFCH transmission. The PilotPNand MACID correspond to the FLSS. The P_{SFCH} is determined according to the rules in 7.6.6.3.5. If the multi-carrier mode is set to MultiCarrierOn, the RLCTRLCarrier indicates the carrier where the R-SFCH is transmitted. If the multi-carrier mode is set to MultiCarrierOff, the RLCTRLCarrier is not defined.

7.6.6.3.7.1 Reporting Rules

The access terminal shall transmit the R-SFCH only on the Control Segment of the RLSS.

The access terminal shall send the R-SFCH reports so that the SFCHReportRate requirement is satisfied. The access terminal shall not transmit R-SFCH report, if SFCHReportRate is equal to the value 0.

If the multi-carrier mode is set to MultiCarrierOn, R-SFCH shall be transmitted on the carriers indicated by ActiveCarriers.

The access terminal shall not transmit the R-SFCH after *ConnectedState.TunedAway* indication and until the *ConnectedState.TunedBack* indication.

7.6.6.3.7.2 SFCH Report

The SFCH reports for each CQIReportingMode are shown in Table 82.

Table 82 SFCH Report for each CQIReportingMode

	CQIReportingMode		
	CQISISO	CQISCW	CQIMCW
SFCH Report	SFCHSISO	SFCHSCW	SFCHMCW

The format for the SFCHSISO report is shown in Table 83.

Table 83 Format for SFCHSISO report

Field	Length (bits)
SubBandIndex	4
SubBandCQIValueSISO	4
Reserved	2

SubBandIndex — Indicates the subband for which the SubBandCQIValueSISO is reported. If the multi-carrier mode is set to MultiCarrierOn, the two MSBs of this field is set to value 0.

SubBandCQIValueSISO — If NumSFCHBits is equal to 11, this field indicates the CQI Value for the reported SubBandIndex. See 7.6.6.3.7.3 for details. If NumSFCHBits is set to “10”, this field is set to the value 0.

Reserved — The sender shall set this field to ‘00’. The receiver shall ignore this field.

The format for the SFCHSCW report is shown in Table 84.

Table 84 Format for SFCHSCW report

Field	Length (bits)
SubBandIndex	4
SubBandCQIValueSCW	4
SubBandRank	2

SubBandIndex — Indicates the Subband for which the SubBandCQIValueSCW is reported. If the multi-carrier mode is set to MultiCarrierOn, the two MSBs of this field is set to value 0.

SubBandCQIValueSCW — If NumSFCHBits is equal to “11”, this field indicates the CQI Value for the FL MIMO SCW transmission for the reported Rank and the reported SubBandIndex. See 7.6.6.3.7.3 for details. If NumSFCHBits is equal to “10”, then this field is set to the value 0.

SubBandRank — Indicates the desired number of MIMO layers in the FL MIMO SCW transmission for the reported SubBandIndex.

If EffectiveNumAntennas = 4, the SFCHMCW report consists of 3 parts. The format for the first part is shown in Table 85, the format for the second part is shown in Table 86, and the format for the third part is shown Table 87. If EffectiveNumAntennas = 2 or EffectiveNumAntennas = 3, the SFCHMCW report consists of only the first two parts, shown in Table 85 and Table 86. If EffectiveNumAntennas = 1, the SFCHMCW report consists of only the first part, shown in Table 85. The SFCHMCW report is not defined if EffectiveNumAntennas > 4.

Table 85 Format of first part of SFCHMCW report

Field	Length (bits)
MCWIndex	2
SubBandIndex	4
SubBandCQIValueMCWLayer1	4

MCWIndex — This field is set to “00” to indicate the SFCH report is the first part of the SFCHMCW report.

SubBandIndex — Indicates the Subband for which the SubBandCQIValueMCWLayer1 is reported. If the multi-carrier mode is set to MultiCarrierOn, the two MSBs of this field is set to value 0.

SubBandCQIValueMCWLayer1 — If NumSFCHBits is equal to “11”, this field indicates the CQI Value for the FL MIMO MCW layer 1 transmission and the reported SubBandIndex. See 7.6.6.3.7.3 for details. If NumSFCHBits is equal to “10”, then this field is set to the value 0.

Table 86 Format of second part of SFCHMCW report

Field	Length (bits)
MCWIndex	2
SubBandCQIValueMCWLayer2	4
SubBandCQIValueMCWLayer3	4

MCWIndex — This field is set to "01" to indicate the SFCH report is the second part of the SFCHMCW report.

SubBandCQIValueMCWLayer2 — If NumSFCHBits is equal to "11", this field indicates the CQI Value for the FL MIMO MCW layer 2 transmission and the reported SubBandIndex. See 7.6.6.3.7.3 for details. If NumSFCHBits is equal to "10", then this field is set to the value 0.

SubBandCQIValueMCWLayer3 — If NumSFCHBits is equal to "11", this field indicates the CQI Value for the FL MIMO MCW layer 3 transmission and the reported SubBandIndex. See 7.6.6.3.7.3 for details. If NumSFCHBits is equal to "10", then this field is set to the value 0.

Table 87 Format of third part of SFCHMCW report

Field	Length (bits)
MCWIndex	2
SubBandCQIValueMCWLayer4	4
Reserved	4

MCWIndex — This field is set to "10" to indicate the SFCH report is the third part of the SFCHMCW report.

SubBandCQIValueMCWLayer4 — If NumSFCHBits is equal to "11", this field indicates the CQI Value for the FL MIMO MCW layer 4 transmission and the reported SubBandIndex. See 7.6.6.3.7.3 for details. If NumSFCHBits is equal to "10", then this field is set to the value 0.

Reserved — The sender shall set this field to '0000'. The receiver shall ignore this field.

7.6.6.3.7.3 SubBandCQIValue

Depending on the CQIReportingMode, the SubBandCQIValue can indicate the field SubBandCQIValueSISO in SFCHSISO report, the field SubBandCQIValueSCW in SFCHSCW report, or the fields SubBandCQIValueMCWLayer1, SubBandCQIValueMCWLayer2, SubBandCQIValueMCWLayer3 and SubBandCQIValueMCWLayer4 in SFCHSCW report.

The access terminal may autonomously choose to include single user transmit processing gains, such as STTD or transmission on a preferred precoding matrix, in the SubBandCQIValue. STTD and precoding are defined in the Physical Layer Protocol.

The access terminal shall not incorporate any form of multi-user transmit processing losses, such as SDMA losses, in the SubBandCQIValue. SDMA is defined in the Physical Layer Protocol.

Based on an unrestricted observation interval, the access terminal shall report the highest tabulated SubBandCQIValue such that a packet transmitted by the access network to the access terminal over all of the hop-ports on the subband corresponding to the SubBandIndex, sent at TDM2 Power Density using the specified packet format and the specified number of FL-PHY Frames, terminating 1 PHY Frame before the start of the PHY Frame that the R-SFCH is transmitted, would result in a packet error probability ≤ 0.01. The CQI mappings are shown in Table 81.

7.6.6.3.8 Procedures for the R-BFCH Physical Layer channel

The access terminal uses the R-BFCH to transmit the beam index, the subband index, and the CQI value offset necessary for SDMA transmission for the current FLSS. SDMA is defined in the Physical Layer Protocol. A description of the beam code books, indexed by the BFCHBeamCodeBookIndex is provided in 12.

The BFCH reports are sent using the BFCHBeamIndexType1 report or BFCHBeamIndexType2 report, the details of which are in section 7.6.6.3.8.1.

The RCC MAC Protocol shall provide the Physical Layer Protocol with the P_{BFCH}, RLCTRLCarrier, PilotPNand MACID for each R-BFCH transmission. The BFCHReportType is equal to the value 1 if BFCHBeamIndexType1 report is transmitted and equal to the value 2 if BFCHBeamIndexType2 report is transmitted. The PilotPN and MACID correspond to the FLSS. The P_{BFCH} is determined according to the rules in 7.6.6.3.5. If the multi-carrier mode is set to MultiCarrierOn, the RLCTRLCarrier indicates the carrier, where the R-BFCH is transmitted. If the multi-carrier mode is set to MultiCarrierOff, the RLCTRLCarrier is not defined.

7.6.6.3.8.1 Reporting Rules

The access terminal shall transmit the R-BFCH only on the Control Segment of the RLSS.

If the multi-carrier mode is set to MultiCarrierOn, R-BFCH shall be transmitted on the carriers, indicated by ActiveCarriers.

The access terminal shall not transmit the R-BFCH after *ConnectedState.TunedAway* indication and until the *ConnectedState.TunedBack* indication.

The access terminal shall send R-BFCH reports so that the BFCHReportRate requirement is satisfied. The access terminal shall not transmit R-BFCH report if BFCHReportRate is equal to the value ‘0’.

The access terminal shall transmit at most one BFCHBeamIndexType1 report per Control Segment and Carrier. In addition, the access terminal shall transmit the necessary number of BFCHBeamIndexType2 reports per Control Segment and Carrier, to satisfy the BFCHReportRate requirement.

7.6.6.3.8.2 BFCH Report

The following definition will be used in this section. The definition assumes that BFCHBeamIndexType1 report is transmitted on RL PHY Frame Index k and carrier j.

MasterSFCHReportSet — The set of most recently transmitted SFCH reports on carrier j and RL PHY Frame Index k and RL PHY Frame Index k - ControlSegmentPeriod.

The format for BFCHBeamIndexType1 report is shown in Table 88, and the format for BFCHBeamIndexType2 report is shown in Table 89.

Table 88 Format for BFCHBeamIndexType1

Field	Length (bits)
BeamIndex	6
SDMADeltaCQI	3
Reserved	1

BeamIndex — If the number of SFCH reports in the MasterSFCHReportSet is equal to ‘1’,, the BeamIndex indicates the desired beam index from the code book specified by the BFCHBeamCodeBookIndex, for the SubBandIndex specified in the SFCH report.

If the number of SFCH reports in the MasterSFCHReportSet is not equal to ‘1’, the access terminal shall set the BeamIndex field to the value ‘0’. The access point shall ignore this field.

If the SFCHReportRate is 0, the BeamIndex field is set to the value ‘0’.

SDMADeltaCQI — This field is defined only if NumBFCHBits is equal to “11”, FLNumSDMADimensions > 1 and the BeamIndex field corresponds to an SDMA beam. Otherwise, the access terminal shall set this field to the value ‘0’. The access point shall ignore this field.

If the SFCHReportRate > 0, the SDMADeltaCQI indicates the integer offset that is to be subtracted from the SubBandCQIValue field of the SFCH reports, to obtain the SDMACQIValue. See 7.6.6.3.8.3 for details on SDMACQIValue.

If the SFCHReportRate = 0, the SDMADeltaCQI indicates the integer offset that is to be subtracted from the appropriate CQIValue field of the CQI reports, to obtain the SDMACQIValue. See 7.6.6.3.8.3 for details on SDMACQIValue.

Reserved — The sender shall set this field to ‘0’. The receiver shall ignore this field.

Table 89 Format for BFCHBeamIndexType2

Field	Length (bits)
BeamIndex	6
SubBandIndex	4

BeamIndex — The BeamIndex indicates the desired beam index from the code book specified by the BFCHBeamCodeBookIndex, for the reported SubBandIndex.

SubBandIndex — Indicates the subband for which the BeamIndex is reported. If the multi-carrier mode is set to MultiCarrierOn, the two MSBs of this field is set to value 0.

7.6.6.3.8.3 SDMACQIValue

The SDMACQIValue is to be used by the Access Network for SDMA transmission.

In this section, the parameters used to compute SDMACQIValue are to be interpreted as integers. If SDMACQIValue is computed to be a number smaller than 0, the SDMACQIValue shall be made equal to 0. SDMACQIValue shall be interpreted as a 4-bit CQI value if the CQIReportingMode is CQISISO or CQIMCW, and as a 5-bit CQI value if the CQIReportingMode is CQISCW. The CQI mappings to be used for SDMACQIValue are shown in Table 81.

If the SFCHReportRate is 0, the SDMACQIValue is computed as follows.

- If the CQIReportingMode is CQISISO, SDMACQIValue = CQIValueSISO - SDMADeltaCQI, where CQIValueSISO is a field in the CQICHCTRL report of the R-CQICH, described in 7.6.6.3.6.
- If the CQIReportingMode is CQISCW, SDMACQIValue = CQIValueSCW - SDMADeltaCQI, where CQIValueSCW is a field in the CQICHSCW report of the R-CQICH, described in 7.6.6.3.6.
- If the CQIReportingMode is CQIMCW, the SDMACQIValue for each MIMO layer 'k' defined as SDMACQIValue = CQIValueMCWLayer'k' - SDMADeltaCQI, where CQIValueMCWLayer'k' is a field in the CQICHMCW report of the R-CQICH, described in 7.6.6.3.6.

If the SFCHReportRate is not 0, the SDMACQIValue is computed as follows.

- If the CQIReportingMode is CQISISO, SDMACQIValue = SubBandCQIValueSISO - SDMADeltaCQI, where SubBandCQIValueSISO is a field in the SFCHSISO report of the R-SFCH, described in 7.6.6.3.7.
- If the CQIReportingMode is CQISCW, SDMACQIValue = SubBandCQIValueSISO - SDMADeltaCQI, where SubBandCQIValueSCW is a field in the SFCHSCW report of the R-SFCH, described in 7.6.6.3.7.
- If the CQIReportingMode is CQIMCW, the SDMACQIValue for each MIMO layer 'k' defined as SDMACQIValue = SubBandCQIValueMCWLayer'k' - SDMADeltaCQI, where SubBandCQIValueMCWLayer'k' is a field in the SFCHMCW report of the R-SFCH, described in 7.6.6.3.6.

If the SFCHReportRate > 0, the access terminal shall report the lowest tabulated SDMADeltaCQI based on an unrestricted observation interval, such that the packet transmitted by the access network to the access terminal over all of the hop-ports referred by the SubBandIndex fields of the SFCH reports, sent at TDM2 Power Density using the specified packet format and the specified number of FL-PHY Frames corresponding to SDMACQIValue, terminating 1 PHY Frame before the start of the PHY Frame that the R-BFCH is transmitted, would result in a packet error probability ≤ 0.01.

If the SFCHReportRate = 0, the access terminal shall report the lowest tabulated SDMADeltaCQI based on an unrestricted observation interval, such that the packet transmitted by the access network to the access terminal over all of the hop-ports on the SSCH subbands used in the computation of R-CQICH, sent at TDM2 Power Density using the specified packet format and the specified number of FL-PHY Frames corresponding to SDMACQIValue, terminating 1 PHY Frame before the start of the PHY Frame that the R-BFCH is transmitted, would result in a packet error probability ≤ 0.01.

7.6.6.3.9 Procedures for the R-PICH Physical Layer channel

The access terminal uses the R-PICH to transmit the broadband pilot.

The access terminal shall transmit the R-PICH only on the Control Segment of the RLSS.

The access terminal shall transmit the R-PICH only if RPICHEnabled is equal to the value 1.

The RCC MAC Protocol shall provide the Physical Layer Protocol with P_{PICH} , RLCTRLCarrier, PilotPNand MACID for each R-PICH transmission. The PilotPNand MACID correspond to the FLSS. The P_{PICH} is determined according to the rules in 7.6.6.3.5. If the multi-carrier mode is set to MultiCarrierOn, the RLCTRLCarrier indicates the carrier, where the R-PICH is transmitted. If the multi-carrier mode is set to MultiCarrierOff, the RLCTRLCarrier is not defined. If the multi-carrier mode is set to MultiCarrierOn, independent R-PICH shall be transmitted on independent carriers indicated by ActiveCarriers.

The access terminal shall not transmit the R-PICH after *ConnectedState.TunedAway* indication and until the *ConnectedState.TunedBack* indication.

7.6.6.3.10 Procedures for the R-REQCH Physical Layer channel

The access terminal shall transmit the R-REQCH only on the Control Segment of the RLSS.

The RCC MAC Protocol shall provide the Physical Layer Protocol with the P_{REQCH} , RLCTRLCarrier, PilotPNand MACID. The PilotPNand MACID correspond to the FLSS. The P_{REQCH} is determined according to the rules in 7.6.6.3.5. If the multi-carrier mode is set to MultiCarrierOn, the RLCTRLCarrier is equal to the REQCarrier, that indicates the carrier on which the R-REQCH is transmitted. If the multi-carrier mode is set to MultiCarrierOff, the RLCTRLCarrier is not defined.

The access terminal shall not transmit the R-REQCH after *ConnectedState.TunedAway* indication and until the *ConnectedState.TunedBack* indication.

The access terminal shall wait a minimum duration of MinRequestInterval * ControlSegmentPeriod between consecutive R-REQCH transmissions to DRLSS. For instance, if the MinRequestInterval is two, and the access terminal last transmits R-REQCH to the DRLSS on RL PHY Frame with RL PHY Frame Index k, then the access terminal shall only be permitted to transmit R-REQCH to that same sector again on an RL PHY Frame with RL PHY Frame Indices ≥ ControlSegmentPeriod * 2 + k.

The R-REQCH message format is shown in Table 90.

Table 90 R-REQCH message format

Field	Length (bits)
QoSFlow	2
MaxNumSubCarriers	2
DRLSS	3
Reserved	3

QoSFlow — These bits specify the RLP QoS flow corresponding to the request. The access terminal shall indicate the QoS of the highest QoS flow that contains data available for transmission. The QoS priority order shall be 00 highest, 01 second, 10 third, and 11 lowest.

MaxNumSubCarriers — These bits specify the maximum number of subcarriers the access terminal can currently support. The access terminal shall specify the highest value from Table 91 such that both the buffer level of the QoS flow given by the QoSFlow bits and the number of subcarriers that the access terminal can support at RDCHGain using the available transmit power are satisfied.

Table 91 MaxNumSubCarriers lookup table

Num Subcarriers supportable at RDCHGain (Subcarriers)	Buffer size of QoSFlow (Bytes)	MaxNumSubCarriers Field
$X < 16$	$Y < 50$	00
$16 \le X < 32$	$50 \le Y < 100$	01
$32 \le X < 64$	$100 \le Y < 200$	10
Otherwise	Otherwise	11

DRLSS — This field shall be set to the 3-bit ActiveSetIndex corresponding to the access terminal's DRLSS.

Reserved — The sender shall set this field to '000'. The receiver shall ignore this field.

7.6.6.3.11 Procedures for the R-ACKCH Physical Layer channel

7.6.6.3.11.1 Definitions

The following definitions will be used in this section, and apply to a specified RL PHY Frame.

RACKCTRLCarrier — Indicates the carrier where the acknowledgement is transmitted.

NumACK — Number of FL PHY Frames acknowledged.

NumACKIndex	Index for each of the NumACK acknowledgements. The acknowledgements are assumed to be ordered so that so that the acknowledgment corresponding to the FL PHY Frame index k has a smaller NumACKIndex than the ACK corresponding to FL PHY Frame Index k+m, where m > 1. NumACKIndex satisfies 0 ≤NumACKIndex ≤ NumACK-1.
NumRACKBaseNodes	For the RACKCTRLCarrier, NumRACKBaseNodes = [$N_{CARRIER\_SIZE}$ / MinHopPortsPerNode] * [FLNumSDMADimensions / RACKBandwidthFactor] * NumACK.
BaseNodeIndex	Index for a base-node in the carrier. The base-nodes are assumed to be ordered in increasing order so that base-node mapped by the lowest index hop-port in the carrier has BaseNodeIndex =0 , and the base-node mapped by the highest index hop-port in the carrier has BaseNodeIndex = ([$N_{CARRIER\_SIZE}$ /MinHopPortsPerNode]* FLNumSDMADimensions)-1.
SpatialOrder	Number of MIMO-MCW layers.

7.6.6.3.11.2 ACK Reporting Rules

The access terminal shall not transmit the R-ACKCH after *ConnectedState.TunedAway* indication and until the *ConnectedState.TunedBack* indication.

In the FDD mode, for the FL transmission of a MAC packet on FL PHY Frame Index *k*, the access terminal shall send an ACK on the RL PHY Frame with RL PHY Frame Index *k+3,* resulting in NumACK = 1.

In the FDD mode, for the FL transmission of a MAC packet using an E-PHY Frame that starts on FL PHY Frame Index *k,* the access terminal shall send an ACK on the RL PHY Frames with RL PHY Frame Indices *k+8* and *k+9,* resulting in NumACK = 1.

Define:

$$A = \lceil N_{FL\_BURST} / N_{RL\_BURST} \rceil \text{ and } B = \lfloor N_{FL\_BURST} / N_{RL\_BURST} \rfloor \text{ in TDD mode.}$$

The parameters N_{FL_BURST} and N_{RL_BURST} are respectively given by the N_FLBurst and N_RLBurst parameters which are part of the public data of the Overhead Messages Protocol. The access terminal shall send ACKs according to the following rules:

- If $N_{FL\_BURST} \le N_{RL\_BURST}$, for the FL transmission of MAC packet sent on FL PHY Frame Index k, the access terminal shall send an ACK on RL PHY Frame with RL PHY Frame Index $(q+1)N_{RL\_BURST} + f$, where $q = \lfloor k / N_{FL\_BURST} \rfloor$ and $f = k \bmod N_{FL\_BURST}$. This results in NumACK = 1 for the first N_{FL_BURST} RL PHY Frames, and NumACK = 0 for the remaining RL PHY Frames, in the appropriate RL burst.
- If $N_{FL\_BURST} > N_{RL\_BURST}$, for the FL transmission of MAC packet sent on FL PHY Frame Index k, the access terminal shall send an ACK on RL PHY Frame with RL PHY Frame Index $(q+1)N_{RL\_BURST} + r$, where $q = \lfloor k / N_{FL\_BURST} \rfloor$ and $r = \lfloor (k \bmod N_{FL\_BURST}) / A \rfloor$. This results in NumACK = A, in the first $\lfloor N_{FL\_BURST} / A \rfloor$ RL PHY Frames, and NumACK = B in the subsequent RL PHY Frames, in the appropriate RL burst.

The RACKVal, RACKBaseNodeIndex, and RACKCTRLCarrier computation are described in 7.6.6.3.11.2.1. The P_{ACKCH} is determined according to the rules in 7.6.6.3.4.

If the multi-carrier mode is set to MultiCarrierOff, the MAC shall provide the PHY with NumRACKBaseNodes, P_{ACKCH} and RACKBaseNodeIndex and RACKVal, for each acknowledgement sent on the RL PHY Frame.

If the multi-carrier mode is set to MultiCarrierOn, the parameters NumRACKBaseNodes, RACKBaseNodeIndex, P_{ACKCH} and RACKVal are defined for the corresponding RACKCTRLCarrier.

If the multi-carrier mode is set to MultiCarrierOn, and the acknowledgement is for FL MIMO-MCW transmission restricted to one carrier or for FL MIMO-SCW or SISO transmission, the MAC shall provide the PHY with NumRACKBaseNodes, P_{ACKCH} , RACKBaseNodeIndex and RACKVal, for one RACKCTRLCarrier, sent on the RL PHY Frame.

If the multi-carrier mode is set to MultiCarrierOn, and the acknowledgement is for FL MIMO-MCW transmission across multiple carriers, the MAC may provide the PHY with NumRACKBaseNodes, P_{ACKCH} and RACKBaseNodeIndex and RACKVal, for multiple RACKCTRLCarriers.

The access point shall ensure that the base-node mapped by the lowest-indexed hop-port in the ATA for each RACKCTRLCarrier, has the corresponding BaseNodeIndex that is a multiple of RACKBandwidthFactor × MinHopPortsPerNode.

7.6.6.3.11.2.1 RACKVal, RACKBaseNodeIndex and RACKCTRLCarrier Computation

For each acknowledgement transmitted in the RL PHY Frame, the RACKVal, RACKBaseNodeIndex and RACKCTRLCarrier are determined as follows.

1. For the SISO and MIMO-SCW acknowledgement , the RACKVal, RACKBaseNodeIndex and RACKCTRLCarrier are determined as follows.

   a. If the FL transmission of MAC packet passes CRC, the RACKVal is set to the value 1; otherwise RACKVal is equal to the value 0.

   b. The RACKCTRLCarrier shall be the carrier with the most hop-ports in the ATA. If more than one carrier has the most hop-ports in the ATA, the RACKCTRLCarrier shall be the carrier corresponding to the lowest carrier index.

   c. The RACKBaseNodeIndex = [($N_{CARRIER\_SIZE}$ /MinHopPortsPerNode) * (FLNumSDMADimensions) / RACKBandwidthFactor * NumACKIndex] + [BaseNodeIndex / RACKBandwidthFactor], where the BaseNodeIndex corresponds to the base-node mapped by the lowest-indexed hop-port in the ATA in the RACKCTRLCarrier.

2. For MIMO-MCW acknowledgement, RACKBaseNodeIndex and RACKCTRLCarrier for MIMO layer 0 is chosen according to rules 1b and 1c. If the FL MIMO layer 0 transmission of a MAC packet passes CRC, then the corresponding RACKVal shall be equal to the value 1, otherwise the corresponding RACKVal shall be equal to the value 0.

3. In the MIMO-MCW mode, for the MIMO layer k (k >0), the RACKVal , RACKBaseNodeIndex and RACKCTRLCarrier are computed as follows. Initialize k = 1. Initialize RACKBaseNodeIndex and RACKCTRLCarrier according to rules 1b and 1c.

    a. If RACKBaseNodeIndex < NumRACKBaseNodes -1, increment RACKBaseNodeIndex by 1 and go to step 3c; otherwise go to step 3b.

    b. If RACKBaseNodeIndex ≥ NumRACKBaseNodes -1, determine RACKCTRLCarrier by incrementing the carrier index by 1, determine RACKBaseNodeIndex for MIMO layer k according to rule 1c for the updated RACKCTRLCarrier, and go to step 3c.

    c. If the FL MIMO layer 'k' transmission of a MAC packet passes CRC, then the corresponding RACKVal shall be equal to the value 1, otherwise the corresponding RACKVal shall be equal to the value 0. Go to step 3d.

    d. Increment k by 1. Go to step 3a if k < SpatialOrder; otherwise declare RACKCTRLCarrier and RACKBaseNodeIndex computation to be complete.

The MIMO-MCW acknowledgement shall have SpatialOrder total number of RACKBaseNodeIndex and RACKVal, counted across one or more RACKCTRLCarriers.

7.6.7 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes.

7.6.8 Interface to other protocols

7.6.8.1.1 Commands

This protocol issues the following command:

- ACMAC.AttemptAccess(DFLSS)

7.6.8.1.2 Indications

This protocol registers to receive the following indications:

- *ConnectedState.TunedAway*
- *ConnectedState.TunedBack*

7.6.9 Configuration attributes

The following complex attributes and default values are defined (see 10.3 for attribute record definition).

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Reverse Traffic Channel MAC Protocol.

7.6.9.1 PowerControl attribute

Field	Length (bits)	Default Value
Length	8	N/A
AttributeID	8	N/A
OpenLoopTransitionTime	8	0
ACKChannelGain0	8	152
ACKChannelGain1	8	160
ACKChannelGain2	8	168
ACKChannelGain3	8	176
ACKChannelGain4	8	182
ACKChannelGain5	8	188
ACKChannelGain6	8	196
ACKChannelGain7	8	204
ACKChannelGain8	8	212
ACKChannelGain9	8	224
ACKStepUpSize	8	136
ACKExtendedFrameGain	8	128
CQICHPowerBoostForHandoff	8	0
REQCHPowerBoostForHandoff	8	0

Length — Length of the complex attribute in octets. The access network shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — This field shall be set to 0x00.

ValueID — This field identifies this particular set of values for the attribute. The access network shall increment this field for each complex attribute-value record for a particular attribute.

OpenLoopTransitionTime — This field determines the transition time for open loop power adjustment, in msec. A zero value indicates that the open loop power adjustment procedures shall not be performed.

ACKChannelGain*j* This field is used to determine the gain of the R-ACKCH over the R-CQICH. The value of this parameter is (n-128)*0.125 dB and shall be used when the erasure rate is between j*10% and (j+1)*10%.

ACKStepUpSize This field is used to determine the gain of the R-ACKCH over the R-CQICH. The value of this parameter is (n-128)*0.125 dB.

ACKExtendedFrameGain This field determines the gain of the R-ACKCH over the R-CQICH. The value of this parameter is (n-128)*0.125 dB.

CQICHPowerBoostForHandoff This field determines the amount of power boost on the R-CQICH when signaling an FL handoff. The value of this parameter is n*0.125 dB.

REQCHPowerBoostForHandoff This field determines the amount of power boost on the R-REQCH when signaling an RL handoff. The value of this parameter is n*0.125 dB.

7.6.10 Protocol numeric constants

Constant	Meaning	Value
$N_{RCCMPType}$	Type field for this protocol	Table 9
$N_{RCCMPDefault}$	Subtype field for this protocol	0x0000

7.6.11 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

7.7 Default Reverse Traffic Channel MAC Protocol

7.7.1 Overview

The Default Reverse Traffic Channel MAC Protocol provides the procedures and messages required for an access terminal to transmit, and for an access network to receive, the Reverse Traffic Channel. The access network maintains an instance of this protocol for every access terminal.

This protocol operates in one of two states:

- *Inactive State*: In this state, the access terminal is not assigned a MACID and cannot transmit on the Reverse Traffic Channel. When the protocol is in this state, it waits for an *Activate* command.
- *Active State*: In this state, the access terminal is assigned a MACID and may transmit data on the Reverse Traffic Channel.

The protocol states and the indications and events causing the transition between the states are shown in Figure 79.

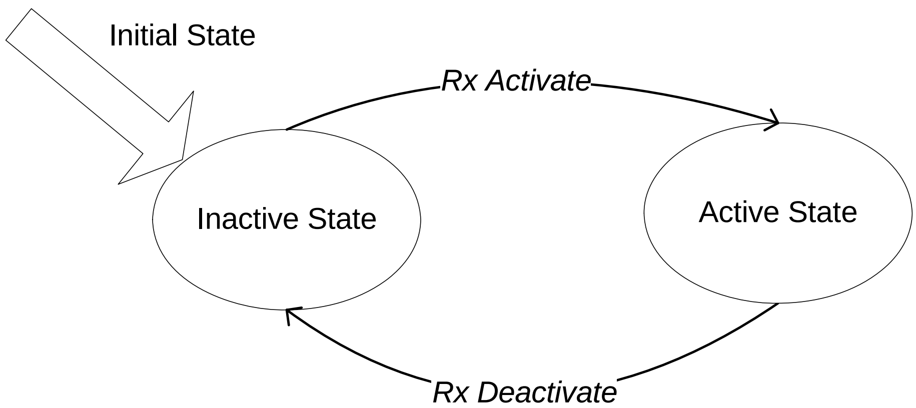

Figure 79 Default Reverse Traffic Channel MAC Protocol state diagram

This protocol shall use the following parameters and attributes.

Parameter Name	Where Defined	Comments
RLChannelTreeIndex	OMP	ExtendedChannelInfo
RLControlSegmentDuration	OMP	ExtendedChannelInfo
DataCtrlOffset	pBCH1	Part of pBCH1 transmission
RDCHGainMin	ASMP	ActiveSetAssignmentMsg
RDCHGainMax	ASMP	ActiveSetAssignmentMsg
RDCHInitialPacketFormat	OMP	ExtendedChannelInfo
ErasureGain*j*, j=0, 1, 2, 3	OMP	ExtendedChannelInfo
RLNumSDMADimensions	OMP	ExtendedChannelInfo
ReverseLinkSilenceDuration	OMP	Public data
ReverseLinkSilencePeriod	OMP	Public data
RLImplicitDeassignEnabled	Connected State Protocol	Public data
SelectedInterlaceMode	Connected State Protocol	Public data
TuneAwayStatus	Connected State Protocol	Public data
MultiCarrierOn	Physical Layer Protocol	Public data
UpDecisionThresholdMin	Configuration Attribute	
DownDecisionThresholdMin	Configuration Attribute	
ChanDiffMax	Configuration Attribute	
ChanDiffMin	Configuration Attribute	
UpDecisionValue	Configuration Attribute	
DownDecisionValue	Configuration Attribute	
DataGainStepUp	Configuration Attribute	

Parameter Name	Where Defined	Comments
DataGainStepDown	Configuration Attribute	
RDCHGainAdjustmentThreshold	Configuration Attribute	
OSIMonitorSetSize	Configuration Attribute	
OSIMonitorThreshold	Configuration Attribute	
OSI2SequenceMax	Configuration Attribute	
FastOSIChanDiffThreshold	Configuration Attribute	

7.7.2 Primitives

7.7.2.1 Commands

This protocol defines the following commands:

- *Activate*
- *Deactivate*

7.7.2.2 Return indications

This protocol returns the following indications:

- *ReverseTrafficPacketsMissed*
- *SupervisionFailed*
- *UATIReceived*

7.7.3 Public data

7.7.3.1 Static public data

This protocol does not define any static public data.

7.7.3.2 Dynamic public data

- Subtype for this protocol

7.7.4 Protocol data unit

The transmission unit of this protocol is a Reverse Traffic Channel Lower MAC Sublayer packet. Each packet contains one Security Sublayer packet.

7.7.5 Protocol initialization and swap

7.7.5.1 Protocol initialization

Upon initialization at the access terminal,

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Active State.

Upon initialization at the access network

- The values of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- The protocol shall enter the Active State.

7.7.5.2 Protocol Swap

Upon swap, the protocol instance shall enter the Inactive State.

7.7.6 Procedures

The protocol constructs a Reverse Traffic Channel Lower MAC Sublayer packet out of the Security Sublayer packet by adding the Lower MAC Sublayer header and trailer as defined in 7.7.7.

The protocol then sends the packet for transmission to the Physical Layer.

7.7.6.1 Command processing

7.7.6.1.1 Activate

If the protocol receives an *Activate* command in the Inactive State, the access terminal and the access network shall perform the following:

- Transition to the Active State
- Internal variables of the protocol shall be initialized as mentioned in the definition of these variables.

If the protocol receives this command in any other state, the command shall be ignored.

7.7.6.1.2 Deactivate

If the protocol receives a *Deactivate* command in the Active State:

- The access terminal shall cease transmitting the Reverse Traffic Channel and shall transition to the Inactive State.
- The access network shall cease monitoring the Reverse Traffic Channel from this access terminal and shall transition to the Inactive State.

If the protocol receives a *Deactivate* command in the Inactive State, the command shall be ignored.

7.7.6.2 Reverse traffic channel addressing

Transmission on the Reverse Traffic Channel is multiplexed in time and frequency. An assignment on the Reverse Traffic Channel shall be specified by a set of hop-ports and an interlace. Each hop-port is specified by a hop-port index as well as a carrier index. If the duplex mode is FDD, then the interlace may be composed of standard PHY Frames, or extended PHY Frames (as specified by the assignment blocks received from the SS MAC protocol). Extended PHY Frames are defined in 7.1.3.1.3.

The duration of an assignment of hop-ports may or may not be prespecified. Assignments whose durations are pre-specified are known as non-sticky assignments, and assignments whose durations are not pre-specified are knows as sticky assignments.

The set of hop-ports assigned for a particular interlace for a particular access terminal via sticky assignment blocks (RLABs, RLAB_HOs, or access grants) is referred to as the “Reverse Link Access Terminal Assignment” or RL-ATA for an interlace. An access terminal can have multiple RL-ATAs, one for each nonoverlapping interlace (note that overlapping interlaces can be created only by the use of extended PHY Frames).

The set of hop-ports assigned for a particular interlace for a particular access terminal via non-sticky assignment blocks (NS-RLABs) is referred to as the “Reverse Link Non-Sticky Access Terminal Assignment” or RL-NS-ATA for an interlace. An access terminal can have multiple RL-NS-ATAs, one for each nonoverlapping interlace

Sets of hop-ports in assignment blocks (from SS MAC) are specified using the channel tree specified by RLChannelTreeIndex. Channel tree tables are specified in 7.7.6.7.

Packets transmitted over the Reverse Traffic Channel are transmitted over the R-DCH Physical Layer channel. Access terminals are assigned R-DCH resources (RL-ATAs, RL-NS-ATAs) via assignment blocks (RLABs,RLAB-HOs, NS-RLABs) and AccessGrants that are sent over the F-SSCH.

The following rules apply regarding the coexistence of RL-ATAs and RL-NS-ATAs:

- An access terminal shall have no more than one RL-ATA per interlace. If duplex mode is FDD, then additionally, an access terminal shall not have any RL-ATA’s that overlap in time.
- An access terminal shall have no more that one RL-NS-ATA per interlace. If duplex mode is FDD, then additionally, an access terminal shall not have any RL-NS-ATA’s that overlap in time.
- An access terminal shall not have a non-empty RL-ATA and RL-NS-ATA on the same interlace. If duplex mode is FDD, then additionally, an access terminal shall not have a non-empty RL-ATA that overlaps in time with a non-empty RL-NS-ATA.

The RL-ATA for an interlace can be accumulated via multiple (sticky) assignment blocks as specified in 7.7.6.4.1.1. The RL-NS-ATA for an interlace can be accumulated via multiple (non-sticky) assignment blocks as specified in 7.7.6.4.1.4. All hop-ports in the RL-ATA/RL-NS-ATA for an access terminal in a single interlace shall be combined for transmission over the Physical Layer channel (R-DCH). Different interlaces shall always carry separate MAC packets with independent H-ARQ termination. An example is illustrated in Figure 80.

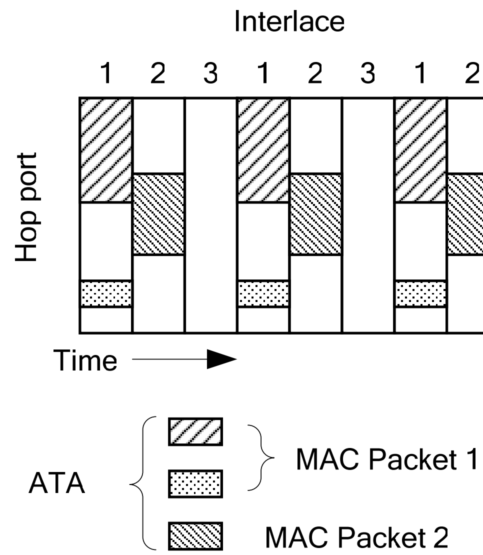

Figure 80 R-DCH addressing example

7.7.6.3 Inactive state

When the protocol is in the Inactive State, the access terminal and the access network wait for an *Activate* command.

7.7.6.4 Active state

In this section, the parameters RDCHGainMin and RDCHGainMax correspond to the RLSS.

In the Active State, the access terminal shall transmit over the R-DCH using the RL-ATAs or RL-NS-ATAs and PFs selected by the access network and signaled to the access terminal over the F-SSCH Physical Layer channel. The access terminal processes blocks from the SS MAC Protocol to maintain its RL-ATA and RL-NS-ATA, and configures the Physical Layer for transmission of packets according to the RL-ATA/RL-NS-ATA. The access network controls transmission of ACK/NACK information via the SS MAC Protocol (over the F-SSCH Physical Layer channel) based on pass or fail of the MAC packet, as determined by the PHY.

Note that an RL-NS-ATA shall be used for transmission of a single Lower MAC packet (see 7.1.2).

The MACID assigned to the access terminal for each sector in its active set shall be given in the ActiveSetAssignment message that is public data of the Active Set Management Protocol.

OSI2SequenceNum and PilotPNStrongest are an access terminal's locally maintained parameters. The initial value of OSI2SequenceNum shall be equal to 1, and the initial value of PilotPNStrongest shall be equal to -1.

RDCHGain is an access terminal's locally maintained parameter, whose initial value shall be equal to RDCHGainMin.

Assignment and H-ARQ logic for the access network and access terminal are specified in 7.7.6.4.1 and 7.7.6.4.2.

7.7.6.4.1 Access terminal requirements

7.7.6.4.1.1 Access terminal assignment management for sticky assignments

The access terminal shall maintain and manage its RL-ATA by monitoring RLABs, (and RLAB-HOs)and NS-RLABs, as well as AccessGrants received from the SS MAC protocol. For transmission on this ATA after an access grant, the access terminal shall use the packet format defined by RDCHInitialPacketFormat. After receiving an RLAB/NS-RLAB, the access terminal shall switch to the packet format specified in the RLAB/NS-RLAB.

If RLImplicitDeassignEnabled is equal to '1', then upon receiving a *TunedAway* indication from the Connected State Protocol, the access terminal shall expire all its RL-ATA's.

If SelectedInterlaceMode is equal to '1', then the access terminal shall ignore all RLABs that have the Extended Transmission field set to '1'.

In this section, it is assumed that all the RLABs/NS-RLABs are being sent from the same serving sector, the RLSS (see 7.6.6.3 for the definition of RLSS). The logic for access terminal assignment management during handoff is found in 7.7.6.4.1.3.

If the extended transmission field is set to '0', an RLAB assigns hop-ports for a particular interlace consisting of standard PHY frames as shown in 7.1.3.1.2. If the Extended Transmission field is equal to'1', an RLAB assigns hop-ports for an interlace consisting of extended PHY frames, as shown in 7.1.3.1.3.

7.7.6.4.1.1.1 Simultaneous assignments

If duplex mode is FDD, RLABs received in the same PHY frame could be for different interlaces because some of the RLABs could be for the interlace consisting of standard PHY frames, and some of the RLABs could be for the interlace consisting of extended PHY frames. If multiple RLABs for the access terminal's MACID are received in the same PHY frame from SS MAC, and they are not all for the same interlace, then the access terminal shall assume an error has occurred, and shall ignore all of these RLABs, unless there is an RLAB with the NodeID set to $\text{NodeID}_{\text{DEASSIGN}}$, in which case this RLAB shall not be ignored.

If multiple RLABs for the access terminal's MACID are received from the SS MAC protocol in the same PHY frame, for the same interlace, and one of the RLABs has a NodeID set to $\text{NodeID}_{\text{DEASSIGN}}$, then all RLABs except for the latter shall be discarded. This rule trumps all those which follow in this section.

If multiple RLABs for the access terminal's MACID are received from the SS MAC protocol in the same PHY frame, for the same interlace, and if all the values in all the fields except the NodeID field are the same, then the access terminal shall treat these RLABs as a single RLAB assigning the union of the hop-ports mapped by the constituent NodeIDs.

If multiple RLABs for the access terminal's MACID are received from the SS MAC protocol in the same PHY frame, for the same interlace, and if the values in at least one of the fields (excluding the NodeID field) are not the same, then the access terminal shall treat these RLABs as errors and shall ignore them.

7.7.6.4.1.1.2 Supplemental and non-supplemental assignments

If an RLAB is received from the SS MAC protocol for the access terminal's MACID, then the hop-ports associated with the NodeID and interlace assigned by the RLAB shall be added to the RL-ATA (refer to 7.4.6.3.1.2 for the RLAB format and interpretation), according to the following rules.

If the Supplemental field of the RLAB for a particular interlace (see the SS MAC specification) is equal to'1', then the new RL-ATA on that interlace is the union of hop-ports included in the old RL-ATA on that interlace and hop-ports specified by the new NodeID, provided the old RL-ATA is non-empty. The PF specified in the received RLAB shall be used in place of any PFs that may have been specified in any previous assignment of NodeIDs (hop-ports) on the interlace.

If the Supplemental field of the RLAB is equal to'0', then the RL-ATA for the relevant interlace shall be cleared before adding the hop-ports specified by the NodeID in the RLAB to the RL-ATA for the interlace. If duplex mode is FDD, then all RL-ATAs that overlap in time with the new assignment shall be cleared before adding the hop-ports specified by the NodeID in the RLAB to the RL-ATA for the interlace.

The access terminal shall ignore this RLAB if any of the following conditions are satisfied:

- If the Supplemental field of the RLAB for a particular interlace is equal to'1', and the access terminal has an empty RL-ATA on that interlace, or
- If an RLAB is received from the SS MAC protocol for the access terminal's MACID and the resulting combined RL-ATA has hop-ports that belong to two sub-channel trees (see 7.7.6.7), or
- If an RLAB is received from the SS MAC for the access terminal's MACID for a particular interlace that does not change the access terminal's RL-ATA for that interlace.

7.7.6.4.1.1.3 Decrementing assignments

If an RLAB or NS-RLAB is received from the SS MAC Protocol for a particular interlace that contains a MACID other than the access terminal's MACID, then all of the hop-ports in the RL-ATA on that interlace that intersect with hop-ports specified by the NodeID included in the RLAB/NS-RLAB shall be expired (removed from the RL-ATA) for that interlace. If duplex mode is FDD, and an RLAB/NS-RLAB is received from the SS MAC protocol for a particular interlace that contains a MACID other than the access terminal's MACID, then for each RL-ATA that overlaps in time and frequency with the hop-ports specified by this new assignment, the intersecting hop-ports shall be removed from the RL-ATA.

If the access terminal receives, in the same PHY frame, an RLAB with its MACID, and an NS-RLAB with a MACID other than its MACID, for the same interlace, and the hop ports assigned by the RLAB intersect with the hop ports assigned by the NS-RLAB, then the access terminal shall expire the intersecting hop ports from its RL-ATA.

If duplex mode is FDD, the following applies. If the access terminal receives, in the same PHY frame, an RLAB with its MACID, and an NS-RLAB with a MACID other than its MACID, for time overlapping interlaces, and the hop ports assigned by the RLAB intersect with the hop ports assigned by the NS-RLAB, then the access terminal shall expire the intersecting hop ports from its RL-ATA.

If the access terminal receives, in the same PHY frame, an RLAB with its MACID, and an RLAB with a MACID other than its MACID, for the same interlace, and the hop ports assigned by the RLABs intersect, then the access terminal shall ignore the RLAB with its MACID.

If duplex mode is FDD, the following applies. If the access terminal receives, in the same PHY frame, an RLAB with its MACID, and an RLAB with a MACID other than its MACID, for time overlapping interlaces, and the hop ports assigned by the RLABs intersect, then the access terminal shall ignore the RLAB with its MACID.

7.7.6.4.1.1.4 Deassigning assignments

If an RLAB is received from the SS MAC protocol for the access terminal's MACID that assigns the reserved $\text{NodeID}_{\text{DEASSIGN}}$, then the RL-ATA on that interlace shall be expired.

7.7.6.4.1.1.5 Time overlapping sticky and non-sticky assignments

If the access terminal receives, from SS MAC, a NS-RLAB with its MACID for a particular interlace, while it already has a non-empty RL-ATA on that interlace, the access terminal shall keep the resulting RL-NS-ATA, and clear the RL-ATA on that interlace.

Furthermore, if duplex mode is FDD, and the access terminal receives, from the SS MAC, a NS-RLAB with its MACID, then the access terminal shall clear any RL-ATA that overlaps in time with the new RL-NS-ATA.

If the access terminal receives, in the same PHY frame, a NS-RLAB giving it a non-empty RL-NS-ATA, and an RLAB giving it a non-empty RL-ATA, then the access terminal shall ignore the RLAB.

7.7.6.4.1.2 Access terminal assignment management for non-sticky assignments

The access terminal shall maintain and manage its RL-NS-ATAs by monitoring RLABs and NS-RLABs delivered from the SS MAC Protocol.

If RLImplicitDeassignEnabled is equal to '1', then upon receiving a *TunedAway* indication from the Connected State Protocol, the access terminal shall expire all its RL-NS-ATA's.

If SelectedInterlaceMode is equal to '1', then the access terminal shall ignore all NS-RLABs that have the Extended Transmission field set to '1'.

In this section, it is assumed that all the RLABs/NS-RLABs are being sent from the same serving sector, the RLSS (see 7.6.6.3 for the definition of RLSS). The logic for access terminal assignment management during handoff is found in 7.7.6.4.1.3.

If the extended transmission field is equal to'0', an NS-RLAB assigns hop-ports for a particular interlace consisting of standard PHY frames as shown in 7.1.3.1.2. The duration field specifies the number of standard PHY frames to be used for transmission of this assignment. If the Extended Transmission field is equal to'1', an NS-RLAB assigns hop-ports for an interlace consisting of extended PHY frames ,as shown in 7.1.3.1.3. The duration field specifies the number of extended PHY frames to be used for transmission of this assignment.

7.7.6.4.1.2.1 Simultaneous assignments

If duplex mode is FDD, NS-RLABs received in the same PHY frame could be for different interlaces because some of the NS-RLABs could be for the interlace consisting of standard PHY frames, and some of the NS-RLABs could be for the interlace consisting of extended PHY frames. If multiple NS-RLABs for the access terminal's MACID are received in the same PHY frame from SS MAC, and they are not all for the same interlace, then the access terminal shall assume an error has occurred, and shall ignore all of these NS-RLABs, unless there is an RLAB with the NodeID set to $NodeID_{DEASSIGN}$, in which case this RLAB shall not be ignored.

If multiple NS-RLABs for the access terminal's MACID are received from the SS MAC Protocol in the same PHY frame, and one of the NS-RLABs has a NodeIDs set to $NodeID_{DEASSIGN}$, then all NS-RLABs except for the latter shall be discarded. This rule trumps all those which follow in this section.

If multiple NS-RLABs for the access terminal's MACID are received from the SS MAC protocol in the same PHY frame, if all the values all the fields except the NodeID field are the same, then the access terminal shall treat these NS-RLABs as a single NS-RLAB assigning the union of the hop-ports mapped by the constituent NodeIDs.

If multiple NS-RLABs for the access terminal's MACID are received from the SS MAC Protocol in the same PHY frame, and if the values in at least one of the fields (excluding the NodeID field) are not the same, then the access terminal shall treat these NS-RLABs as errors and shall ignore them.

7.7.6.4.1.2.2 Deassigning assignments

If an NS-RLAB is received from the SS MAC protocol for the access terminal's MACID that assigns the reserved $NodeID_{DEASSIGN}$, then the RL-NS-ATA on that interlace shall be expired.

7.7.6.4.1.2.3 Time overlapping non-sticky assignments

If a NS-RLAB is received from the SS MAC Protocol for a particular interlace with the access terminal's MACID, and the access terminal already has a RL-NS-ATA for that interlace, then the new assignment block takes precedence: the access terminal shall stop trying to decode on the old RL-NS-ATA for that interlace (shall clear this RL-NS-ATA), and shall update its RL-NS-ATA for that interlace according to the new NS-RLAB.

If duplex mode is FDD, and a NS-RLAB is received form the SS MAC Protocol with the access terminal's MACID that assigns a non-empty RL-NS-ATA, then all RL-NS-ATAs that overlap in time with the new RL-NS-ATA shall be expired.

7.7.6.4.1.2.4 Time overlapping sticky and non-sticky assignments

If the access terminal receives, from SS MAC, a RLAB with its MACID for a particular interlace, while it already has a non-empty RL-NS-ATA on that interlace, the access terminal shall keep the resulting RL-ATA, and clear the RL-NS-ATA on that interlace.

Furthermore, if duplex mode is FDD, and the access terminal receives, from the SS MAC, an RLAB with its MACID, then the access terminal shall clear any RL-NS-ATA that overlaps in time with the new RL-ATA.

7.7.6.4.1.2.5 Overlapping assignments from other ATs

If an RLAB or NS-RLAB is received from the SS MAC Protocol for a particular interlace that contains a MACID other than the access terminal's MACID, then if the hop-ports specified by the NodeID included in the RLAB/NS-RLAB overlap with the RL-NS-ATA on that interlace, the RL-NS-ATA shall be expired. If duplex mode is FDD, then all RL-NS-ATAs that intersect in time and frequency with the new assignment shall be expired.

If the access terminal receives, in the same PHY frame, an NS-RLAB with its MACID, and an RLAB with a MACID other than the access terminal's MACID, for the same interlace, and the hop ports assigned by the NS-RLAB intersect with the hop ports assigned by the RLAB, the access terminal shall ignore the RLAB (in the sense that the NS-ATA resulting from the NS-RLAB will not be modified/expired because of the RLAB).

If duplex mode is FDD, the following applies. If the access terminal receives, in the same PHY frame, an NS-RLAB with its MACID, and an RLAB with a MACID other than the access terminal's MACID, for time overlapping interlaces, and the hop ports assigned by the NS-RLAB intersect with the hop ports assigned by the RLAB, the access terminal shall ignore the RLAB (in the sense that the NS-ATA resulting from the NS-RLAB will not be modified/expired because of the RLAB).

If the access terminal receives, in the same PHY frame, an NS-RLAB with its MACID and an NS-RLAB with a MACID other than its MACID, for the same interlace, and the hop ports assigned by the NS-RLABs overlap, the access terminal assumes as error has occurred, and shall ignore the NS-RLAB with its MACID.

If duplex mode is FDD, the following holds. If the access terminal receives, in the same PHY frame, an NS-RLAB with its MACID and an NS-RLAB with a MACID other than its MACID, for time overlapping interlaces, and the hop ports assigned by the NS-RLAB s overlap, the access terminal assumes as error has occurred, and shall ignore the NS-RLAB with its MACID.

7.7.6.4.1.3 Access terminal assignment management during handoff

If an *RLSSChanged* indication from the RCC MAC Protocol is received, the access terminal shall clear all RL-ATAs and RL-NS-ATAs associated with the old RLSS.

If the access terminal receives an RLAB-HO with its MACID, and the sector in the DesiredSector field is different from the RLSS, the access terminal shall issue a *ReverseControlChannelMAC.ChangeRLSS* to change the RLSS to the sector in the DesiredSector field of the RLAB-HO. If the access terminal then receives an *RLSSChanged* indication from the RCC MAC Protocol, the access terminal shall process the *RLSSChanged* indication as specified above, and in addition shall update its new RL-ATA to the hop ports specified by the NodeID field in the RLAB-HO for the interlace specified by the ExtendedTransmission field in the RLAB-HO, for the sector specified in the DesiredSector field. The packet format to be used is specified in the PF field of the RLAB-HO.

If the access terminal receives an RLAB-HO with its MACID, and the sector in the DesiredSector field is not different from the RLSS, the access terminal shall ignore this RLAB-HO.

If the access terminal receives multiple RLAB-HOs with its MACID, and all the fields are the same except for the NodeID fields, then the access terminal shall treat these RLAB-HOs as a single RLAB-HO assigning the union of the hop ports specified by the NodeID fields.

If the access terminal receives multiple RLAB-HOs with its MACID, and all the fields other than the NodeID field are not the same, then the access terminal shall ignore all these RLAB-HOs.

If the access terminal receives an RLAB/NS-RLAB with the access terminal's MACID, that has the supplemental field set to '0', from the DRLSS, while the DRLSS is different from the RLSS, the access terminal shall issue a *ReverseControlChannelMAC.ChangeRLSS* command to change from the RLSS to the DRLSS. If the access terminal then receives an *RLSSChanged* indication from the RCC MAC Protocol, the access terminal shall process the *RLSSChanged* indication as specified above, and in addition shall update the appropriate RL-ATA/RL-NS-ATA according to the new RLAB/NS-RLAB.

If the access terminal receives an access grant for handoff[39] with its MACID, the access terminal shall issue a *ReverseControlChannelMAC.ChangeFLSS* command to change the FLSS to the sector from which the access grant was sent, and shall also issue a *ReverseControlChannelMAC.ChangeRLSS* command to change the RLSS to the sector from which the access grant was sent. If the access terminal then receives an *RLSSChanged* indication from the RCC Protocol, the access terminal shall clear all RL-ATAs/RL-NS-ATAs it may have, and in addition shall update an RL-ATA to be equal to the hop ports specified by the NodeID field in the AccessGrant, for the sector from which the access grant was sent.

The access terminal shall ignore all RLABs or NS-RLABs that come from sectors other than its current RLSS or its DRLSS.

7.7.6.4.1.4 Access terminal transmission logic for sticky assignments

The access terminal may formulate and transmit a MAC packet on a PHY Frame on an interlace according to the RL-ATA on the interlace and the PF selected for transmission. The MAC shall pass the Physical Layer Protocol the set of hop ports specified by the RL-ATA, and shall specify whether or not the assignment is an Extended Transmission Duration Assignment (see 7.1.3.1.3). The power for transmission of a particular MAC packet is computed as specified in 7.7.6.4.1.6. The access network informs the access terminal of its assignment using signaling messages as described in the SS MAC Protocol specification.

If TuneAwayStatus is equal to '1', then the access terminal shall not send any MAC packets on any interlaces.

If a positive ACK signal corresponding to the RL-ATA is received (via the SS MAC Protocol) that corresponds to the transmitted packet, transmission of the MAC packet shall terminate, and the interlace is immediately available for the next packet. If the access terminal transmits a packet for the maximum number of transmissions of the PF selected for this MAC packet the access terminal shall automatically terminate transmission of the packet. If no ACK is received for this packet, the access terminal shall expire its RL-ATA, and return a *ReverseTrafficPacketsMissed*indication along with parameters that uniquely identify the lost packet. The method of uniquely identifying the packet is out of the scope of this specification.

[39] Some AccessGrants are sent for the purpose of obtaining timing and/or power offset information from a sector and should not be used for handoff logic. Such AccessGrants are scrambled using an AccessSequenceID that belongs to the set of IDs reserved for timing/power correction, as described in 7.3.6.4.1.4.3.

If an RLAB/NS-RLAB for a particular interlace is received from the SS MAC protocol that leaves the RL-ATA on that interlace non-empty, and if the access terminal is currently transmitting a packet on that interlace, then the access terminal shall return a *ReverseTrafficPacketsMissed* indication along with parameters that uniquely identify the lost packet and shall cease transmitting this packet. The method of uniquely identifying the packet is out of the scope of this specification. The access terminal shall then update its RL-ATA in accordance with the new RLAB/NS-RLAB.

If no packet is available for transmission in a given PHY Frame, then an erasure sequence shall be transmitted in the hop-ports, or subset of hop-ports, assigned to the access terminal in that PHY Frame as specified in section Physical Layer specification. The power for transmission of an erasure sequence is computed as specified in 7.7.6.4.1.6. If duplex mode is FDD, the PHY Frame may be either a standard PHY Frame or an extended PHY Frame.

7.7.6.4.1.5 Access terminal transmission logic for non-sticky assignments

The access terminal may formulate and transmit a MAC packet on a PHY Frame on an interlace according to the RL-NS-ATA on the interlace and the PF selected for transmission. The MAC shall pass the Physical Layer Protocol the set of hop ports specified by the RL-NS-ATA, and shall specify whether or not the assignment is an Extended Transmission Duration Assignment (see 7.1.3.1.3). The power for transmission of a particular MAC packet is computed as specified in 7.7.6.4.1.6. The access network informs the access terminal of its assignment using signaling messages as described in the SS MAC Protocol specification.

The number of H-ARQ retransmissions over a RL-NS-ATA is equal to the duration of the RL-NS-ATA, as specified by the duration field in the non-sticky assignment block (see 7.4.6.3.1.2).

If TuneAwayStatus is equal to ‘1’, then the access terminal shall not send any MAC packets on any interlaces.

If a positive ACK signal corresponding to the RL-NS-ATA is received (via the SS MAC Protocol) that corresponds to the transmitted packet, transmission of the MAC packet shall terminate, and the access terminal shall expire the RL-NS-ATA. If no ACK is received for this packet, the access terminal shall return an indication along with parameters that uniquely identify the lost packet. The method of uniquely identifying the packet is out of the scope of this specification.

7.7.6.4.1.6 R-DCH power control

During the transmission of the reverse link data, the power of the R-DCH, P_{DCH}, shall be:

$$P_{DCH} = P_{CTRL} - 10\log_{10}\left(N_{CTRL-SUBCARRIERS}\right) + 10\log_{10}\left(N_c\right) + RDCHGain + DataCtrlOffset$$

where P_{CTRL} is the reference value used by the access terminal in adjusting the mean output power of the reverse control channels and is given in 7.6.6.3.5, N_c is the number of hop-ports in the RL-ATA/RL-NS-ATA for this transmission, $N_{CTRL\text{-}SUBCARRIERS}$ is the number of subcarriers allocated for reverse control channels, DataCtrlOffset is a parameter transmitted over pBCH1 of the RLSS, and RDCHGain is as specified in 7.7.6.4.1.7. The R-DCH power shall further be subject to the access terminal’s transmit power limitation and shall remain constant for the entire transmission of each PHY Frame.

If no packet is available for transmission in a given PHY Frame, then an erasure sequence shall be transmitted with the following power:

$$P_{ERASURE} = P_{CTRL} - 10\log_{10}(N_{CTRL-SUBCARRIERS}) + 10\log_{10}(N_{c,erasure}) + \text{ErasureGain}_i + DataCtrlOffset$$

where $N_{c,erasure}$ is the number of subcarriers over which the erasure sequence is transmitted, and is always set to 16. ErasureGain_i i=0,1,2,3 is given in the OverheadParameterList of the Overhead Messages Protocol. ErasureGain_0 is used when $N_c>64$ and the erasure is being sent over a single PHY frame. ErasureGain_1 is used when $N_c>64$ and the erasure is being sent over an extended PHY frame. ErasureGain_2 is used when $N_c\leq 64$ and the erasure is being sent over a single PHY frame. ErasureGain_3 is used when $N_c\leq 64$ and the erasure is being sent over an extended PHY frame.

The reverse link erasure power shall further be subject to the access terminal's transmit power limitation and shall remain constant for the entire transmission of each PHY Frame. If the access terminal does not have enough power to send at $P_{ERASURE}$ due to power limitations, then it shall send at its maximum power.

7.7.6.4.1.6.1 OSIMonitorSet update

The access terminal shall update the OSIMonitorSet according to the following rules:

- If the access terminal is monitoring the Fast OSI value on the F-SSCH of any active set member other than the RLSS, then every FL PHY frame, the access terminal shall update the OSIMonitorSet with a list of PilotPNs of the sectors in the active set whose Fast OSI value are being monitored by the access terminal, and whose ChanDiff values, as defined in 7.7.6.4.1.7, are smaller than or equal to FastOSIChanDiffThreshold, where FastOSIChanDiffThreshold is a configuration attribute of the protocol.
- At the beginning of every superframe of the RLSS, the access terminal shall update the OSIMonitorSet with a list of PilotPN's of the sectors whose PilotStrength is larger than or equal to the OSIMonitorThreshold, where PilotPN and PilotStrength are parameters in the OverheadParameterList of the Overhead Messages Protocol and OSIMonitorThreshold is a configuration attribute of the protocol.

The OSIMonitorSet shall exclude the PilotPN of the RLSS. In addition, if the size of the list is larger than or equal to OSIMonitorSetSize (where OSIMonitorSetSize is a configuration attribute of the protocol), only OSIMonitorSetSize PilotPN's with strongest PilotStrength shall be kept.

7.7.6.4.1.7 RDCHGain determination

If an RLAB or NS-RLAB is received from SS MAC which results in a non empty RL-ATA or RL-NS-ATA, and the RLAB/NS-RLAB was received later than the last time RDCHGain was calculated, and additionally the Delta field in the RLAB/NS-RLAB is not the reserved Delta field, then RDCHGain will be set equal to the Delta specified by the Delta field. The encoding of the Delta field shall be as follows

- The value 7 shall be reserved
- Values between 0 and 6 shall be interpreted as a linear interpolation between RDCHGainMin and RDCHGainMax with RDCHGainIndex = floor[(6.5 / (RDCHGainMax – RDCHGainMin)) * (RDCHGain – RDCHGainMin)].

Otherwise, RDCHGain will be calculated as follows.

After each OSIMonitorSet update, the access terminal shall create an OSI vector whose i[th] element, i.e., OSI_i, corresponds to the most recent OSIValue from the sector whose PilotPN is indicated by the i[th] entry of the OSIMonitorSet. The most recent OSIValue can be a value received over the F-OSICH of the sector or over the Fast OSI Segment of the F-SSCH of the sector. In addition, the access terminal shall create a ChanDiff vector whose i[th] element, i.e., $ChanDiff_i$, corresponds to:

$$ChanDiff_i = \frac{RxPower_{RL,SS}}{TransmitPower_{RL,SS}} \times \frac{TransmitPower_i}{RxPower_i}$$

where $RxPower_{RLSS}$ and $RxPower_i$, contained in the public data of the Active Set Management Protocol, correspond to the average received power (across antenna) of the F-ACQCH of the RLSS, and the average received power (across antenna) of the F-ACQCH of the sector whose PilotPN is indicated by the i[th] entry of the OSIMonitorSet, respectively. $TransmitPower_{RLSS}$ and $TransmitPower_i$, specified in the OverheadParameterList of the Overhead Messages Protocol, correspond to the average transmit power of the F-ACQCH of the RLSS, and the average transmit power of the F-ACQCH of the sector whose PilotPN is indicated by the i[th] entry of the OSIMonitorSet, respectively. The above calculation shall be done in a linear unit. The access terminal shall determine RDCHGain as follows:

If the OSIMonitorSet is empty, the access terminal shall set RDCHGain to RDCHGainMax, OSI2SequenceNum to 1 and PilotPNStrongest to -1. RDCHGainMax is a parameter in the OverheadParameterList of the Overhead Messages Protocol.

If the OSIMonitorSet is not empty, the access terminal shall compute the RDCHGain as follows:

- The access terminal shall first compute a Decision Threshold vector, whose i[th] element, i.e., $DecisionThreshold_i$, $1 \le i \le OSIMonitorSetSize$, is given by:

$$DecisionThreshold_i = \begin{cases} \max\{UpDecisionThresholdMin, (1-a)b_i\} & if \quad OSI_i = 0 \\ \max\{DownDecisionThresholdMin, a(1-b_i)\} & if \quad OSI_i = 1 \\ 1 & if \quad OSI_i = 2 \end{cases}$$

  where UpDecisionThresholdMin and DownDecisionThresholdMin are configuration attributes of the protocol. Variables a and b_i are determined as follows:

$$a = \frac{\min\{RDCHGain, RDCHGainMax\} - RDCHGainMin}{RDCHGainMax - RDCHGainMin}, \text{ and}$$

$$b_i = \frac{\min\{ChanDiff_i, ChanDiffMax\} - ChanDiffMin}{ChanDiffMax - ChanDiffMin},$$

  where ChanDiffMax and ChanDiffMin are configuration attributes of the protocol, and all values in the above computations are in logarithmic scale (in units of dB).

- The access terminal shall produce a Decision vector whose i[th] element, i.e., $Decision_i$, $1 \le i \le OSIMonitorSetSize$, is given by:

$$Decision_i = \begin{cases} UpDecisionValue & \text{if} \quad x_i \le DecisionThreshold_i \quad \text{and} \quad OSI_i = 0 \\ -DownDecisionValue & \text{if} \quad x_i \le DecisionThreshold_i \quad \text{and} \quad OSI_i = 1 \text{ or } 2 \\ 0 & \text{otherwise} \end{cases}$$

where $0 \le x_i \le 1$ is a uniform random variable (generated using the procedure specified in the Common Algorithm Section) and UpDecisionValue and DownDecisionValue are configuration attributes of the protocol.

- The access terminal shall then compute a weighted decision, D_w, according to:

$$D_w = \frac{\sum_{i=1}^{OSIMonitorSetSize} \frac{1}{ChanDiff_i} Decision_i}{\sum_{i=1}^{OSIMonitorSetSize} \frac{1}{ChanDiff_i}}$$

The access terminal shall find the sector with the lowest ChanDiff in the OSIMonitorSet and call that sector as sector k. Then the access terminal shall set the variable OSIStrongest to the OSI value of sector k and PilotPNCurrent to the PilotPN of sector k. Then, the access terminal shall update OSI2SequenceNum and PilotPNStrongest as follows:

$$OSI2SequenceNum = \begin{cases} OSI2SequenceNum + 1 & \begin{array}{l} \text{if } PilotPNCurrent = PilotPNStrongest \text{ and} \\ OSI2SequenceNum \le OSI2SequenceNumMax - 1 \text{ and} \\ OSIStrongest = 2 \end{array} \\ 2 & \begin{array}{l} \text{if } PilotPNCurrent \ne PilotPNStrongest \text{ and} \\ OSIStrongest = 2 \end{array} \\ 1 & \text{otherwise} \end{cases}$$

$$PilotPNStrongest = \begin{cases} PilotPNCurrent, & \text{if } OSIStrongest = 2 \\ -1 & \text{, otherwise} \end{cases}$$

where OSI2SequenceNumMax is a configuration attribute of the protocol.

- The access terminal shall increase RDCHGain by DataGainStepUp dB if D_w is greater than RDCHGainAdjustmentThreshold and shall decrease RDCHGain by DataGainStepDown*OSI2SequenceNum dB if D_w is less than or equal to -RDCHGainAdjustmentThreshold, where DataGainStepUp, DataGainStepDown, and RDCHGainAdjustmentThreshold are configuration attributes of the protocol. Furthermore, the RDCHGain shall always lie between RDCHGainMin and RDCHGainMax. That is, the access terminal shall set RDCHGain to RDCHGainMin if the resulting RDCHGain is smaller than RDCHGainMin and to RDCHGainMax if the resulting RDCHGain is larger than RDCHGainMax.

7.7.6.4.1.8 OSIReport message procedures

If the access terminal receives a OSIReportRequest message, the access terminal shall respond with a OSIReport message within 2 superframes from the time of receipt of the OSIReportRequest message,. The access terminal shall include a history of duration RequestedHistory superframes where RequestedHistory is a field of the OSIReportRequest message.

7.7.6.4.2 Access network requirements for sticky assignments

For each interlace with a non-empty RL-ATA (contains one or more hop-ports), the access network may attempt to decode a MAC packet transmitted on the interlace. The access network may attempt to detect erasure sequences that are transmitted by the access terminal whenever a MAC packet is not available for transmission. Exact algorithms for detecting erasure sequences and the start-of-packet for MAC packets that span multiple PHY Frames are beyond the scope of this specification.

If TuneAwayStatus is equal to ‘1’, then the access network should not attempt to decode any packets from that access terminal.

If RLImplicitDeassignEnabled is equal to ‘1’, then upon receiving a *TunedAway* indication from the Connected State Protocol, the access network shall expire any RL-ATAs for that access terminal.

If a MAC packet is successfully decoded, as indicated by the PHY, the access network shall transmit a positive ACK on the F-SSCH channel via the SS MAC Protocol. If a MAC packet fails to decode and the access network determines that the packet has been transmitted for the maximum number of PHY Frames for the relevant PF, then the access network shall expire the RL-ATA for that interlace. Refer to 7.1.3 for detailed interlaced structure and acknowledgment timing for both FDD and TDD modes.

Exact algorithms to determine the number of H-ARQ attempts prior to successful decode are beyond the scope of this specification.

If a MAC packet is successfully decoded, the payload of the packet is then passed up to the Security Sublayer for further processing. If the MAC packet has a UATIInfoIncluded header field set to ‘1’, this protocol shall generate a *UATIReceived* indication, accompanied by the MAC header of the received packet.

If the RLNumSDMADimensions > 1, then the access network shall ensure that assignments to different ATs that contain hop-ports that map to the same subcarriers have the same F-DPICH format, as described in 7.7.6.8.

7.7.6.4.3 Access network requirements for non- sticky assignments

For each interlace with a non-empty RL-NS-ATA (contains one or more hop-ports), the access network may attempt to decode a MAC packet transmitted on the interlace.

If TuneAwayStatus is equal to '1', then the access network should not attempt to decode any packets from that access terminal.

If RLImplicitDeassignEnabled is equal to '1', then upon receiving a *TunedAway* indication from the Connected State Protocol, the access network shall expire any RL-NS-ATAs for that access terminal.

If a MAC packet is successfully decoded, as indicated by the PHY, the access network shall transmit a positive ACK on the F-SSCH channel via the SS MAC Protocol, and shall expire the RL-NS-ATA.

Exact algorithms to determine the number of H-ARQ attempts prior to successful decode are beyond the scope of this specification.

If a MAC packet is successfully decoded, the payload of the packet is then passed up to the Security Sublayer for further processing. If the MAC packet has a UATIInfoIncluded header field set to '1', this protocol shall generate a *UATIReceived* indication, accompanied by the MAC header of the received packet.

If the RLNumSDMADimensions > 1, then the access network shall ensure that (non-sticky) assignments to different ATs that contain hop-ports that map to the same subcarriers have the same F-DPICH format, as described in section 7.7.6.8.

The RL-NS-ATA shall expire after the specified duration (see 7.4.6.3.1.2), if it is not already expired. (This duration is specified by the assignment block received from SS MAC.)

7.7.6.5 Reverse link silence interval

The access terminal shall not transmit on any RL channel if the transmission of that channel would overlap with the Reverse Link Silence Interval[40]. This rule shall override any requirement for transmission stated elsewhere in this section or in the Reverse Control Channel MAC.

The Reverse Link Silence Interval is defined by ReverseLinkSilenceDuration and ReverseLinkSilencePeriod in the Overhead Messages Protocol.

7.7.6.6 Supervision procedures

The access terminal shall generate a *SupervisionFailed* indication when it does not receive a reverse traffic channel assignment during a time period of length $T_{RTCSupervision}$ while it has sent non-empty request bits during all R-REQCH transmissions during the time period.

[40] This implies that the access terminal must not even start transmission on the Reverse Traffic Data Channel if the transmission of the Reverse Traffic Channel packet would overlap with the Reverse Link Silence Interval.

7.7.6.7 Channel trees

A channel tree defines the mapping of each NodeID to a set of hop-ports and the grouping of hop-ports into port-sets. A channel tree on the RL is indexed by RLChannelTreeIndex and the number of subcarriers mapped by the channel tree, $N_{CARRIER\_SIZE}$, a parameter that is defined by the Physical Layer protocol. See 7.1.4.1 for common terms used for describing channel trees in this specification. Hop-ports shall be numbered numerically from 0.

Q_{SDMA} equals RLNumSDMADimensions.

The set of hop-ports specified by a NodeID shall be the union of all hop-ports mapped by all base-nodes that are descendants of the node specified by NodeID, minus unusable hop-ports, which are defined by the Physical Layer Protocol.

Hop-ports may be grouped into disjoint port-sets for frequency reuse purposes.

The number of hop-ports indexed by the tree shall equal Q_{SDMA} * $N_{CARRIER\_SIZE}$, and the total number of nodes in the tree shall be a function of $N_{CARRIER\_SIZE}$ and Q_{SDMA}, where Q_{SDMA} is the RL multiplexing factor.

Note that when multi-carrier mode is equal to MultiCarrierOn, there is an independent channel tree per carrier, and the channel tree in use for the carrier is signaled on the overhead channels of that carrier. Further, when a specific NodeID or set of hop-ports is communicated with other protocols in this specification, the associated carrier must also be communicated.

7.7.6.7.1 RL channel tree index 0

Channel trees associated with RLChannelTreeIndex 0 are illustrated in Figure 81 for $N_{CARRIER\_SIZE}$ = 512, 1024, and 2048. For $N_{CARRIER\_SIZE}$ = 512, all nodes above the dashed line marked with $N_{CARRIER\_SIZE}$ = 512 are included in the channel tree; For $N_{CARRIER\_SIZE}$ = 1024, all nodes above the dashed line marked with $N_{CARRIER\_SIZE}$ = 1024 are included in the channel tree; For $N_{CARRIER\_SIZE}$ = 2048, all nodes above the dashed line marked with $N_{CARRIER\_SIZE}$ = 2048 are included in the channel tree.

MinHopPortsPerNode equals 16 for RLChannelTreeIndex 0.

The figure shows only $1/Q_{SDMA}$ of the total tree, and there are Q_{SDMA} identical versions of the illustrated tree each with unique NodeIDs and mapping unique hop-ports. For example, if the total tree is composed of Q_{SDMA} identical trees indexed by $i=0, 1, ..., Q_{SDMA}-1$ then NodeIDs of nodes on the i th tree can be obtained from the illustrated tree by adding i *$N_{CARRIER\_SIZE}$/(MinHopPortsPerHode/2) to the NodeID of the illustrated tree. The number of hop-ports indexed by the tree shall equal $Q_{SDMA}N_{CARRIER\_SIZE}$, and the total number of nodes in the tree shall be a function of $N_{CARRIER\_SIZE}$. Namely, the base-nodes are defined by the intervals NodeID = i* $N_{CARRIER\_SIZE}$/(MinHopPortsPerNode/2)+{ $N_{CARRIER\_SIZE}$/MinHopPortsPerNode - 1 , $N_{CARRIER\_SIZE}$/(MinHopPortsPerNode/2) - 2 }, $i=0, 1, ..., Q_{SDMA}-1$. Thus, for $N_{CARRIER\_SIZE}$=512 and Q_{SDMA}=4, there are 128 base-node NodeIDs, 31 through 62, 95 through 126, 159 through 190, and 223 through 254. For nodes on the same level of a channel tree, the NodeID associated with a node increases from left to right with step of 1. One deassignment NodeID, $NodeID_{DEASSIGN}$, is set to $N_{CARRIER\_SIZE}$/(MinHopPortsPerNode/2) - 1.

The mapping of hop-ports to each base-node is described as follows. Each base-node maps to MinHopPortsPerNode hop-ports, the first MinHopPortsPerNode hop-ports (indices 0 to MinHopPortsPerNode-1) to the base-node with the lowest NodeID, the second MinHopPortsPerNode hop-ports to the next base-node, etc., until all hop-ports are mapped. See Table 92 for an example of this mapping for $N_{CARRIER\_SIZE}$=512, MinHopPortsPerNode=16, and Q_{SDMA}=4.

Table 92 Base node NodeID to Hop-port Mapping Example ($N_{CARRIER\_SIZE}$=512, MinHopPortsPerNode=16, and Q_{SDMA}=4)

Base node NodeID	Hop-ports mapped
31	0-15
32	16-31
…	…
62	496-511
95	512-527
96	528-543
…	…
126	1008-1023
…	…
254	2032-2047

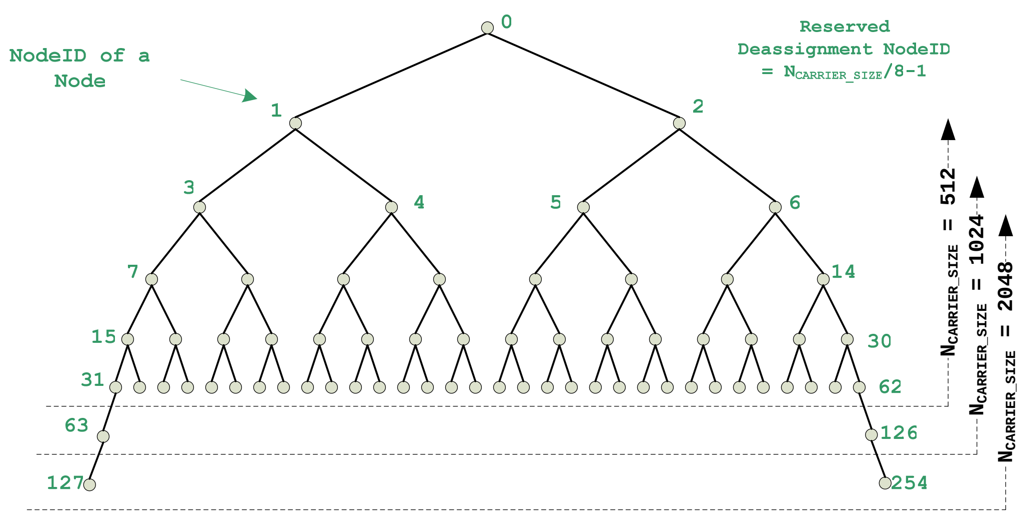

Figure 81 RL channel trees with index 0

7.7.6.8 Packet formats

A packet format (PF) specifies the pilot pattern, spectral efficiency, maximum number of transmissions, and the modulation format to be used for each transmission of a data packet. The packet format consists of five bits. The first bit of the PF indicates the F-DPICH format as described in the F-DPICH section of the physical layer protocol. If the first bit is equal to '0', F-DPICH format 0 is used. If the first bit is equal to '1', F-DPICH format 1 is used. The remaining four bits index the spectral efficiency, maximum number of transmissions, and the modulation format to be used for each transmission of a data packet. The indexing is described in Table 93.

The modulation format is specified by the number of bits in each modulation symbol, which is denoted by modulation order. Modulation orders of 2, 3, and 4 correspond to QPSK, 8PSK, and 16QAM modulations, respectively. The size of the MAC packet that is provided to the Physical Layer is a function of the packet format as well as the set of hop-ports that are assigned to the data packet (to be transmitted on the R-DCH Physical Layer channel).

Table 93 RL packet formats

Packet format index	Spectral efficiency on 1st transmission	Max number of transmissions	Modulation order for each transmission					
			1	2	3	4	5	6+
0	0.25	6	2	2	2	2	2	2
1	0.50	6	2	2	2	2	2	2
2	1.0	6	2	2	2	2	2	2
3	1.5	6	3	2	2	2	2	2
4	2.0	6	3	3	2	2	2	2
5	2.67	6	4	4	3	3	3	3
6	4.0	6	4	4	3	3	3	3
7	6.0	6	4	4	4	3	3	3
8	8.0	6	4	4	4	4	4	3
9	4.0	6	6	6	4	4	4	4
10	5.0	6	6	6	4	4	4	4
11	6.0	6	6	6	4	4	4	4
12	7.0	6	6	6	4	4	4	4
13	8.0	6	6	6	6	4	4	4
14	9.0	6	6	6	6	4	4	4

7.7.7 Header and trailer and formats

The access terminal shall formulate a packet for transmission over the Reverse Traffic Channel using the following header and trailer:

7.7.7.1 Header

Field	Length (bits)
UATIInfoIncluded	1
IsSecure	1
KeyChange	1
InBandControlIncluded	1
Reserved	4
The following six fields shall be included if UATIInfoIncluded is ‘1’	
SessionConfigurationToken	16
ATIdentifierType	1
ATIdentifier	128
ConnectCount	12
AccessReason	2
Reserved	1
The following field shall be included if InBandControlIncluded is ‘1’	
RLInBandControl	0 or N x 8

UATIInfoIncluded Used to signal the existence of access terminal fields in the header. These include SessionConfigurationToken, ATIdentiferType, and ATIdentifier. The access terminal shall set this field to ‘1’ if these fields are present. Otherwise, the access terminal shall set this field to ‘0’. The access terminal shall set UATIInfoIncluded to ‘1’ for the first packet sent on the R-DCH after the receipt of an AccessGrant.

IsSecure The access terminal shall set this field to ‘1’ if the packet is secured by the Authentication and Encryption protocols. The access terminal shall set this field to ‘0’ otherwise.

KeyChange This field shall be set by the Security Sublayer at the transmitter and communicated to the Security Sublayer along with the payload at the receiver.

InBandControlIncluded
Used to signal the existence of in-band control bits. The access network shall set this field to ‘1’ if the InBandControl field is present. Otherwise, the access network shall set this field to ‘0’.

Reserved This field shall be set to zero. The receiver shall ignore this field.

SessionConfigurationToken
This field shall be set to the SessionConfiguration public data of the Session Configuration Protocol.

ATIdentifierType — Used to signal the type of identifier to be signaled in the ATIdentifier field. The access terminal shall set this field to '1' if the ATIdentifier field contains a 128-bit UATI. Otherwise, the access terminal shall set this field to '0', indicating that ATIdentifier field contains the SessionSeed.

ATIdentifier — Used to signal the access terminal identifier record. If the ATIdentifierType field is 1 then this field shall be set to TransmitUATI public data of the Address Management Protocol. Otherwise, the lower bits of this field shall be set to the SessionSeed public data of the SessionManagementProtocol and the upper bits shall be set to zero.

ConnectCount — This field shall be set to the ConnectCount field that is public data of the Idle State Protocol.

AccessReason — This field shall be set to '00' if the access attempt was made in response to a page received by the access terminal. This field shall be set to '01' otherwise.

Reserved — The sender shall set this field to '0000'. The receiver shall ignore this field.

RLInBandControl — The RL in-band bits are used to transmit power control, buffer level, and packet latency information to the access network RL scheduler. The RLInBandControl blocks are given in 7.7.7.2.1.

7.7.7.2 Trailer

This protocol does not specify a trailer.

7.7.7.2.1 RLInBandControl

There are three possible RLInBandControl blocks. Multiple RLInBandControl blocks may be sent in one MAC Header.

7.7.7.2.1.1 InBandPowerControl block

The first InBandPowerControl block, shown below, transmits RL power control information. The access terminal may include more than one InBandPowerControl block in a MAC packet. The j^{th} InBandPowerControl block shall refer to the j^{th} carrier in the ActiveCarrier public data of the SSCH MAC Protocol (The ActiveCarrier public data shall be assumed to be sorted according to increasing carrier values).

Field	Length (bits)
ContinuationBit	1
BlockFormat	1
MaxSubCarriers	3
RDCHGainIndex	3

ContinuationBit — The access terminal shall set the continuation bit to 0 if this is the last RLInBandControl block in the MAC header. Otherwise, the access terminal shall set the continuation bit to 0.

BlockFormat — The access terminal shall indicate an RL power control block by setting the format indicator bit to 0.

MaxSubCarriers — The access terminal shall specify MaxSubCarriers, the maximum number of subcarriers that the access terminal can transmit at RDCHGain based on transmit power constraints with the following values:

Maximum Number of Supportable Subcarriers at RDCHGain	MaxSubCarriers
16	000
32	001
64	010
128	011
256	100
512	101
1024	110
2048	111

RDCHGainIndex — The access terminal shall set RDCHGainIndex to a 3-bit value between 0 and 7 as a linear interpolation between RDCHGainMin and RDCHGainMax as RDCHGainIndex = floor[(7.5 / (RDCHGainMax – RDCHGainMin)) * (RDCHGain – RDCHGainMin)].

7.7.7.2.1.2 InBandBufferLevel block

The second RLInBandControl block, shown below, transmits RL buffer level information. This block is used to provide the scheduler with a more accurate buffer level then the R-REQCH, as well as providing an in-band request channel. Multiple InBandBufferLevel blocks may be sent in one MAC packet for different QoS flows:

Field	Length (bits)
ContinuationBit	1
BlockFormat	2
BufferLevel	3
QoS	2

ContinuationBit — The access terminal shall set the continuation bit to 0 if this is the last RLInBandControl block in the MAC header. Otherwise, the access terminal shall set the continuation bit to 1.

BlockFormat — The access terminal shall indicate an RL buffer level block by setting the format indicator bits to 10.

BufferLevel — The access terminal shall specify the RLP QoS flow to which the buffer level corresponds. The buffer level is specified as follows:

RLP Buffer Level (Bytes)	BufferLevel (3bits)
X = 0	000
0 < X < 50	001
50 ≤ X < 400	010
400 ≤ X < 1000	011
1000 ≤ X < 2000	100
2000 ≤ X < 3000	101
3000 ≤ X < 9000	110
9000 ≤ X	111

QoS — The access terminal shall set the QoS to indicate one of the four negotiated QoS streams.

7.7.7.2.1.3 InBandLatencyInfo block

The third RLInBandControl block, shown in the following table, is used to transmit packet latency. Multiple InBandLatencyInfo blocks may be sent in one MAC packet for different QoS flows.

Field	Length (bits)
ContinuationBit	1
BlockFormat	2
LatencyLevel	3
QoS	2

ContinuationBit — The access terminal shall set the continuation bit to 0 if this is the last RLInBandControl block in the MAC header. Otherwise, the access terminal shall set the continuation bit to 1.

BlockFormat — The access terminal shall indicate an RL latency level block by setting the format indicator bits to 11.

LatencyLevel — The latency level is the largest latency of any packet in the specified QoS flow not including the bits sent in the same MAC packet as the RLInBandBits. The latency is specified in the number of RL-PHY Frames for which the packet has been waiting at the time the RLInBandBit packet starts transmission. The LatencyLevel value is given by the following lookup table.

Latency of RLP head of line packet (PHY Frames)	LatencyLevel (3 bits)
$X < 4$	000
$4 \le X < 10$	001
$10 \le X < 20$	010
$20 \le X < 40$	011
$40 \le X < 80$	100
$80 \le X < 150$	101
$150 \le X < 300$	110
$300 \le X$	111

QoS The access terminal shall specify the QoS flow to which the latency corresponds.

7.7.8 Message formats

The protocol uses the AttributeUpdateRequest, AttributeUpdateAccept, and AttributeUpdateReject messages of the Generic Attribute Update Protocol in 10.9 to update configurable attributes. In addition, this protocol defines the following messages.

7.7.8.1 OSIReportRequest

The access network sends a OSIReportRequest message to request the access terminal to send a OSIReport message.

Field	Length (bits)
MessageID	8
RequestedHistory	8

MessageID The access network shall set this field to 0x00.

RequestedHistory The access network shall set this field to indicate the number of measurements the access terminal should include in the OSIReport message.

Channels	FTC		**SLP**	Best Effort
Addressing	Unicast		**Security**	Required

7.7.8.2 OSIReport

The access terminal sends the PilotReport message to notify the access network of OSI values received from other sectors in the past.

Field	Length (bits)
MessageID	8
SuperframeReferenceNumber	34
NumPilots	5
NumPilots occurrences of the following record{	
PilotPN	12
CarrierID	2
NumOSIValues	8
NumOSIValues instances of the following two fields(	
OSIValue	2
ChanDiff	8
NumFastOSIValues	8
NumFastOSIValues instances of the following two fields(	
FastOSIValue	2
ChanDiff	8
}}	
Reserved	Variable

MessageID — The access terminal shall set this field to 0x01.

SuperframeReferenceNumber — The access terminal shall set this field to the superframe number when the latest OSIValue contained in the message was received.

NumPilots — The access terminal shall set this field to the number of pilots that follow this field in the message. The access terminal should include all pilots that have been in the Active Set or the OSIMonitorSet over the reported period.

PilotPN — The access terminal shall set this field to the PilotPN to which the following OSIValues correspond.

CarrierID — The access terminal shall set this field to the CarrierID of the carrier to which the following OSIValues correspond.

NumOSIValues — The access terminal sets this field to the number of OSIValues included for this pilot.

OSIValue — The i^{th} occurrence of this field for a given PilotPN refers to the OSI value received from this PilotPN in superframe number SuperframeReferenceNumber-i+1. The OSIValue ‘11’ shall correspond to an erased or unavailable OSIValue.

ChanDiff	The access terminal shall set this field to the ChanDiff for the given PilotPN in superframe number SuperframeReferenceNumber-i+1. This field shall be in a two's complement format in units of 1 dB (takes values between -128 and +127 dB). The ChanDiff field is defined in 7.7.6.4.1.7.
NumFastOSIValues	The access terminal sets this field to the number of FastOSIValues included for this pilot.
FastOSIValue	The i^{th} occurrence of this field for a given PilotPN refers to the Fast OSI value received from this PilotPN in FL PHY Frame frame number FirstFrameNumber-i+1, where FirstFrameNumber is the frame number of the first FL PHY frame of superframe number SuperframeReferenceNumber. The OSIValue '11' shall correspond to an erased or unavailable OSIValue.
ChanDiff	The access terminal shall set this field to the ChanDiff for the given PilotPN in FL PHY Frame frame number FirstFrameNumber-i+1, where FirstFrameNumber is the frame number of the first FL PHY frame of superframe number SuperframeReferenceNumber. This field shall be in a two's complement format in units of 1 dB (takes values between -128 and +127 dB). The ChanDiff field is defined in 7.7.6.4.1.7.
Reserved	The number of bits in this field is equal to the number needed to make the message length an integer number of octets. The sender shall set this field to zero. The receiver shall ignore this field.

Channels	RTC	**SLP**	Best Effort
Addressing	Unicast	**Security**	Required

7.7.9 Interface to other protocols

7.7.9.1.1 Commands

This protocol issues the following commands:

- *ReverseControlChannelMAC.ChangeFLSS*
- *ReverseControlChannelMAC.ChangeRLSS*

7.7.9.1.2 Indications

This protocol registers to receive the following indications:

- *ConnectedState.TunedAway*
- *ReverseControlChannelMAC.RLSSChanged*

7.7.10 Configuration attributes

The following complex attributes and default values are defined (see 10.3 for attribute record definition).

Unless specified otherwise, the access terminal and the access network shall use the Generic Attribute Update Protocol in 10.9 to update configurable attributes belonging to the Default Reverse Control Channel MAC Protocol.

7.7.10.1 PowerParameters attribute

Field	Length (bits)	Default Value
Length	8	N/A
AttributeID	8	N/A
UpDecisionThresholdMin	8	3
DownDecisionThresholdMin	8	7
ChanDiffMax	8	50
ChanDiffMin	8	0
UpDecisionValue	8	160
DownDecisionValue	8	80
DataGainStepUp	8	8
DataGainStepDown	8	8
RDCHGainAdjustmentThreshold	8	136
OSIMonitorSetSize	8	2
OSIMonitorThreshold	8	17
OSI2SequenceMax	3	4
FastOSIChanDiffThreshold	8	40

Length — Length of the complex attribute in octets. The access network shall set this field to the length of the complex attribute excluding the Length field.

AttributeID — This field shall be set to 0x00.

UpDecisionThresholdMin — This field is used to control parameters internal to the RTC MAC power control algorithm. The value of this field is $\min(1, 2^{-7} * n)$.

DownDecisionThresholdMin — This field is used to control parameters internal to the RTC MAC power control algorithm. The value of this parameter is $\min(1, 2^{-7} * n)$.

ChanDiffMax — This field determines maximum value of ChanDiff used in determining RDCHGain. The value of this parameter is 2*n dB.

ChanDiffMin — This field is determines the minimum value of ChanDiff used in determining RDCHGain. The value of this parameter is n dB.

UpDecisionValue — This field is used to control parameters internal to the RTC MAC power control algorithm. The value of this parameter is (n-128) * 2^{-5}.

DownDecisionValue — This field is used to control parameters internal to the RTC MAC power control algorithm. The value of this parameter is (n-128) * 2^{-5}.

DataGainStepUp — This field is used to control parameters internal to the RTC MAC power control algorithm. The value of this parameter is n * 2^{-5} dB.

DataGainStepDown — This field is used to control parameters internal to the RTC MAC power control algorithm. The value of this parameter is n * 2^{-5} dB.

RDCHGainAdjustmentThreshold — This field is used to control parameters internal to the RTC MAC power control algorithm. The value of this parameter is (n-128) * 2^{-5} dB.

OSIMonitorSetSize — This field is set to the size of the OSI Monitor Set.

OSIMonitorThreshold — This field determines a threshold such that only sectors with pilot signal to interference ratios above the threshold shall be added to the OSI monitor set. The value of this parameter is n dB.

OSI2SequenceMax — This field is set to control parameters internal to the RTC MAC power control algorithm.

FastOSIChanDiffThreshold — This field determines a threshold such that only active set members with a ChanDiff value smaller than this threshold shall be added to the OSI monitor set when the RDCHGain is being adjusted based on the Fast OSI values. The value of this parameter is n*0.125 dB.

7.7.11 Protocol numeric constants

Constant	Meaning	Value
$N_{RTCMPType}$	Type field for this protocol	
$N_{RTCMPDefault}$	Subtype field for this protocol	0x0000
$T_{RTCSupervision}$	Supervision timer	2 s

7.7.12 Session state information

The Session State Information record (see 10.10) consists of parameter records.

The parameter records for this protocol consist of the configuration attributes of this protocol.

8 Physical Layer

8.1 Default Physical Layer Protocol

8.1.1 Overview

This chapter contains the specification for the Default (Subtype 0) Physical Layer Protocol.

8.1.2 Primitives

8.1.2.1 Commands

This protocol does not define any commands.

8.1.2.2 Return indications

This protocol does not return any indications.

8.1.3 Public data

8.1.3.1 Static public data

This protocol does not define any static public data.

8.1.3.2 Dynamic public data

- Subtype for this protocol
- MultiCarrierOn mode

8.1.4 Protocol data unit

The transmission unit of this protocol is the Physical Layer packet. Each Physical Layer packet contains one MAC Layer packet.

8.1.5 Protocol initialization and swap procedures

8.1.5.1 Protocol initialization

Upon creation, the instance of this protocol in the access terminal and access network shall perform the following:

- The value of the attributes for this protocol instance shall be set to the default values specified for each attribute.
- This protocol shall determine the values of the following parameters
  - Duplexing mode (FDD or TDD)
  - Synchronization mode (Semi-synchronous or Asynchronous)
  - Multi-carrier mode (MultiCarrierOn or MultiCarrierOff)

8.1.6 Protocol swap

- This protocol defines an empty swap procedure.

8.1.7 Procedures

Procedures for the protocol are described in chapter 9.

8.1.8 Message formats

8.1.8.1 TimingCorrection

The access network shall send the timing correction message to correct the reverse link timing of the access terminal.

Field	Length (bits)
MessageID	8
NumSectors	2
NumSectors instances of the following fields	
PilotPN	12
TimingCorrection	16

MessageID — The access network shall set this field to 0x02.

NumSectors — The access network shall set this field to the number of sector records in the message.

PilotPN — The access network shall set this field to the PilotPN of the sector.

TimingCorrection — The access network shall set this field to the timing correction on the sector in twos complement format in units of 1/8 chips. A positive value shall advance the timing, and a negative value shall retard the timing.

8.1.9 Interface to other protocols

8.1.9.1 Commands

This protocol does not issue any commands.

8.1.9.2 Indications

This protocol does not register to receive any indications.

8.1.10 Configuration attributes

This protocol does not define any configuration attributes.

8.1.11 Protocol numeric constants and parameters

Constant	Meaning	Value
$N_{PHYPType}$	Type field for this protocol	Table 9
$N_{PHYPDefault}$	Subtype field for this protocol	0x0000
N_{FFT}	Number of subcarriers in an OFDM symbol	512, 1024, or 2048
T_{CHIP}	Basic unit of time for generating the OFDM waveform	Defined in Table 94 as a function of N_{FFT}
$N_{CARRIER\_SIZE}$	Number of subcarriers in one carrier	512 in MultiCarrierOn mode N_{FFT} in MultiCarrierOff mode
$N_{CARRIERS}$	Number of carriers	$N_{FFT}/N_{CARRIER\_SIZE}$ (= 1 in MultiCarrierOff mode)
$T_{CP,PR}$	Cyclic prefix duration for the superframe preamble	$N_{FFT}T_{CHIP}/4$
$N_{GUARD,PR}$	Number of guard subcarriers in the superframe preamble	Any multiple of $N_{CARRIER\_SIZE}/8$, ranging from $N_{CARRIER\_SIZE}/8$ through $7N_{CARRIER\_SIZE}/8$. This field shall be set by the access network, and is determined by the access terminal.
T_{WGI}	Duration of windowing guard interval	$N_{FFT}T_{CHIP}/32$
$N_{PREAMBLE}$	Number of OFDM symbols in the superframe preamble	8
$N_{FRAME,F}$	Number of OFDM symbols in a forward link PHY Frame	8
$N_{FRAME,R}$	Number of OFDM symbols in a reverse link PHY Frame	8
N_{BLOCK}	Number of subcarriers in a tile.	16 in BlockHopping mode (FL) 1 in SymbolRateHopping mode (FL) 16 for the RL
$T_{G,TDD,F}$	Guard time between a forward link PHY Frame and the subsequent reverse link PHY frames	$3N_{FFT}T_{CHIP}/4$
$T_{G,TDD,R}$	Guard time between a reverse link PHY Frame and the subsequent forward link PHY frames	$5N_{FFT}T_{CHIP}/32$
N_{pBCH0_Period}	Number of superframes over which F-pBCH0 is encoded	16
$N_{MaxErasureHopPorts,F}$	Maximum number of hop-ports to be used for transmitting a single erasure sequence on the forward link	16
$N_{MaxErasureHopPorts,R}$	Maximum number of hop-ports to be used for transmitting a single erasure sequence on the reverse link	16
$N_{CRC,pBCH}$	Number of CRC bits to be used for pBCH0 and pBCH1 packets.	12

Constant	Meaning	Value
$N_{CRC,SSCH}$	Number of CRC bits to be used for SSCH packets.	16
$N_{CRC,Data}$	Number of CRC bits to be used for F-DCH and R-DCH packets.	24
$N_{FastOSI}$	Number of modulation symbols in the Fast OSI segment.	8
$N_{BLOCK, R-ACKCH}$	Number of subcarriers in a R-ACKCH tile.	8
$N_{R-ACKCH-SUBTILE-DURATION}$	Number of OFDM symbols in a R-ACKCH subtile	2

Table 94 Chip duration as a function of N_{FFT}

N_{FFT}	T_{CHIP} in μs
512	1/4.9152
1024	1/9.8304
2048	1/19.6608

8.1.12 Session state information

This protocol does not define any parameter record to be included in a Session State Information record.

9 Default Physical Layer

9.1 Physical layer modes

The physical layer specification consists of two different duplexing modes, two different forward link hopping modes, two different synchronization modes and two different multi-carrier modes. The possible duplexing modes are Time Division Duplexing (TDD) and Frequency Division Duplexing (FDD). The different forward link hopping modes are SymbolRateHopping and BlockHopping. The forward link hopping mode to be used is given by the BlockHoppingEnabled field, which is part of the public data of the Overhead Messages Protocol. The possible synchronization modes are SemiSynchronous and Asynchronous. The possible multi-carrier modes are MultiCarrierOn and MultiCarrierOff.

Parts of the physical layer specification are described separately for different duplexing modes, different forward-link hopping modes, synchronization modes and/or multi-carrier modes. Except in these cases, the specification shall apply to all values of the corresponding mode.

The TDD mode has two associated variables called N_{FL_BURST} and N_{RL_BURST} which determine the time partitioning between forward and reverse links. The values of these variables are given by the N_FLBurst and N_RLBurst parameters respectively, which are part of the public data of the Overhead Messages Protocol.

In the MultiCarrierOn mode, the total transmission bandwidth is divided into a multiplicity of carriers, as specified in 9.3.2.2. The number of carriers is given by $N_{CARRIERS}$. For convenience of exposition, the same terminology is sometimes used in the MultiCarrierOff mode as well. In this case, the total transmission bandwidth is divided into $N_{CARRIERS} = 1$ carriers.

9.2 Encoding and modulation

This section describes the core encoding and modulation procedures, shown in Figure 82, that shall be used for constructing several of the physical layer channels. The procedures described in this section, namely packet-splitting, CRC insertion, encoding, channel interleaving, sequence repetition, scrambling, and modulation, together constitute a method for converting a k-bit packet (generated by an appropriate MAC protocol) into sequences of modulation symbols (one sequence per sub-packet), for any value of k that satisfies at least one of the following two conditions: (1) k is less than MaxPHYSubPacketSize (2) k is a multiple of 8. Here, MaxPHYSubPacketSize is public data of the Active Set Management Protocol.

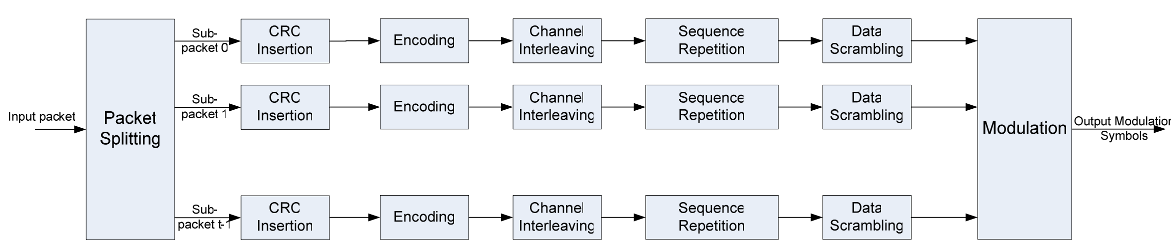

Figure 82 Encoding and modulation structure

9.2.1 Packet-splitting and CRC insertion

If the packet size k is larger than MaxPHYSubPacketSize, the packet shall be split into t sub-packets, indexed from 0 to t-1, where $t = \lceil k / MaxPHYSubPacketSize \rceil$. Here, MaxPHYSubPacketSize is public data of the Active Set Management Protocol. Let k_i denote the size in bits of sub-packet i. Define integers t_0 = (k/8) mod t and t_1 = t-t_0. Define two other integers $b_0 = 8\lceil k/(8t) \rceil$ and $b_1 = 8\lfloor k/(8t) \rfloor$. The first t_0 sub-packets shall consist of b_0 bits, i.e., $k_0 = k_1 = ... = k_{t_0-1} = b_0$, while the last t_1 sub-packets shall consist of b_1 bits, i.e., $k_{t_0} = k_{t_0+1} = ... = k_{t-1} = b_1$. The bits are distributed to the different sub-packets in order, i.e., bits 0 through k_0-1 form sub-packet 0, bits k_0 through k_0+k_1-1 form sub-packet 1, etc.

Each of the sub-packets so generated shall then be appended by a CRC as described in chapter 10. The number of CRC bits, denoted by N_{CRC}, to be appended is variable and is specified separately in the description of each physical layer channel using this procedure. The sizes of the resulting sub-packets at the end of this procedure are therefore given by $k_i' = k_i + N_{CRC}$, for i ranging from 0 through t-1.

At the receiver, a packet shall be declared to be in error if any of the constituent sub-packets are in error.

The operations described in Sections 9.2.2, 9.2.3, 9.2.4, and 9.2.5, namely encoding, channel interleaving, sequence repetition and scrambling operate independently on each of the sub-packets and are described only for the case t=1. The operation described in 9.2.6, namely modulation, operates jointly on all sub-packets and is described for all values of t.

9.2.2 Core encoders

The air-link shall support two basic encoding structures, namely a rate 1/5 parallel turbo code and a rate 1/3 convolutional code. The rate 1/5 turbo code shall be used for values of k larger than 128, while the rate 1/3 convolutional code shall be used for values of k less than or equal to 128.

9.2.2.1 Rate 1/3 convolutional encoding

The core rate-1/3 code is a non-systematic non-recursive convolutional code. The outputs of the convolutional code are punctured or repeated to achieve the desired number of convolutional encoder output bits.

The transfer function for the convolutional code shall be:

$$G(D) = [g_0(D)\ g_1(D)\ g_2(D)\ ]$$

where $g_0(D)= 1 + D^2 + D^3 + D^5 + D^6 + D^7 + D^8$, $g_1(D)= 1 + D + D^3 + D^4 + D^7 + D^8$, and $g_2(D)= 1 + D + D^2 + D^5 + D^8$, where D represents the delay operator.

The sequence of information bits shall be appended with a tail of eight 0s and input to the convolutional encoder. This code generates three code bits for each bit input to the encoder. Thus a total of 3(k+8) = 3k+24 coded bits are generated for a k-bit input packet. These code bits shall be output so that the code bit (c_0) encoded with generator function g_0 shall be output first, the code bit (c_1) encoded with generator function g_1 shall be output second, and the code bit (c_2) encoded with generator function g_2 shall be output last. The state of the convolutional encoder, upon initialization, shall be the all-zero state. The first code bit output after initialization shall be a code bit encoded with generator function g_0. The encoder for this code is illustrated in Figure 83.

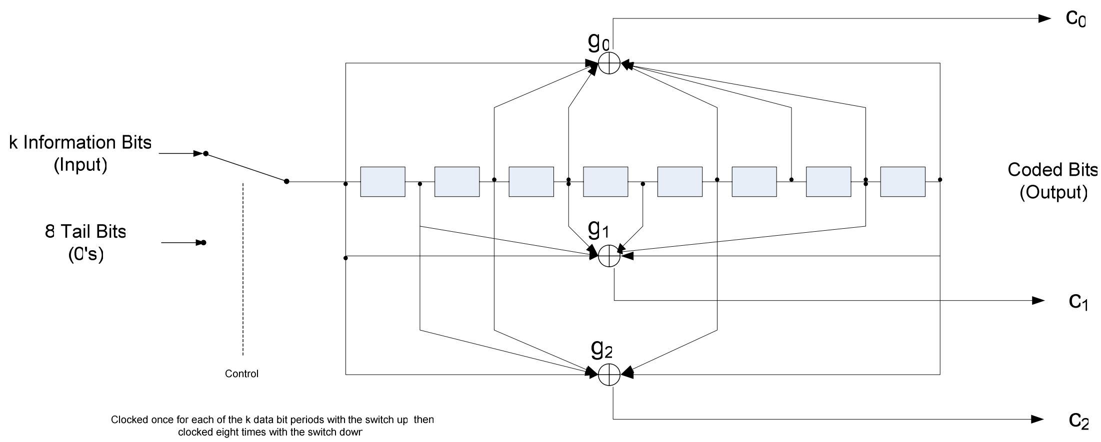

Figure 83 Rate 1/3 convolutional encoder

9.2.2.2 Rate 1/5 turbo encoding

The core turbo encoder is a rate 1/5 code that employs two systematic, recursive, convolutional encoders connected in parallel, with an interleaver—the turbo interleaver—preceding the second recursive convolutional encoder. The two recursive convolutional codes are called the constituent codes of the turbo code. The outputs of the constituent encoders are punctured or repeated to achieve the desired number of turbo encoder output bits.

The transfer function for the constituent code shall be:

$$G(D) = \begin{bmatrix} 1 & \frac{n_0(D)}{d(D)} & \frac{n_1(D)}{d(D)} \end{bmatrix}$$

where $d(D) = 1 + D^2 + D^3$, $n_0(D) = 1 + D + D^3$, and $n_1(D) = 1 + D + D^2 + D^3$, where D represents the delay operator.

The turbo encoder shall generate an output bit sequence that is identical to the one generated by the encoder shown in Figure 84. Initially, the states of the constituent encoder registers in this figure are set to zero. Then, the constituent encoders are clocked with the switches in the positions noted.

The encoded data output bits are generated by clocking the constituent encoders k times with the switches in the up positions, where k is the number of input bits into the turbo encoder. The constituent encoder outputs for each bit period shall be output in the sequence X, Y_0, Y_1, Y'_0, Y'_1 with the X output first. (The bit X' shall not be part of the output sequence.)

The turbo encoder shall generate 18 tail output bits following the encoded data output bits. This tail output bit sequence shall be identical to the one generated by the encoder shown in Figure 84. The tail output bits are generated after the constituent encoders have been clocked k times with the switches in the up position. The first 9 tail output bits are generated by clocking Constituent Encoder 1 three times with its switch in the down position while Constituent Encoder 2 is not clocked. The constituent encoder outputs for each bit period shall be output in the sequence X, Y_0, Y_1, with the X output first. The last 9 tail output bits are generated by clocking Constituent Encoder 2 three times with its switch in the down position while Constituent Encoder 1 is not clocked. The constituent encoder outputs for each bit period shall be output in the sequence X′, Y'_0, Y'_1, with the X' output first. The tail bit sequence ensures that both constituent encoders achieve the all-zeros state at the end of the encoding process.

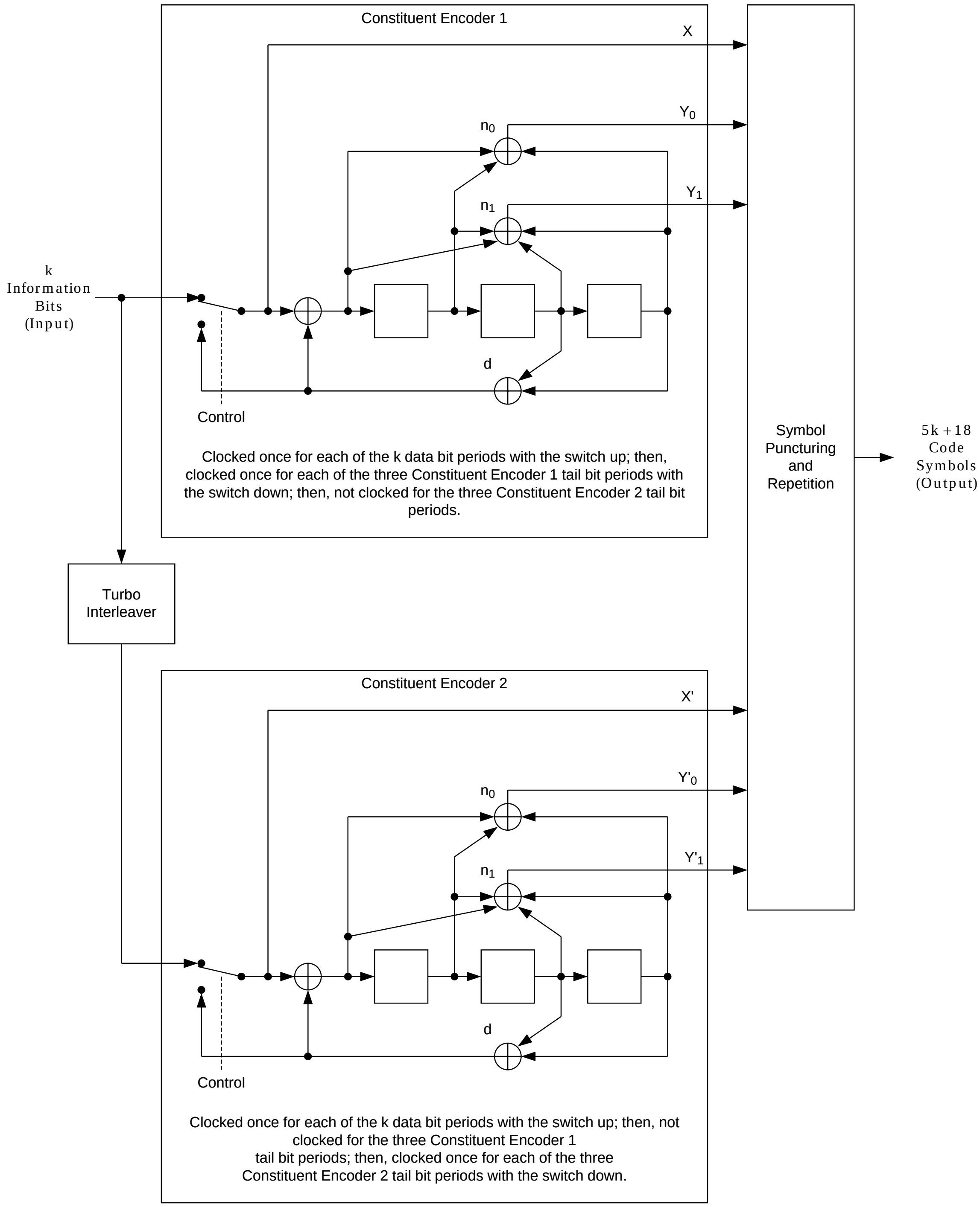

Figure 84 Turbo encoder

9.2.2.2.1 Turbo Interleaving

The turbo interleaver, which is part of the turbo encoder, shall block interleave the turbo encoder input data that is fed to Constituent Encoder 2.

The turbo interleaver shall be functionally equivalent to an approach where the entire sequence of turbo interleaver input bits are written sequentially into an array at a sequence of addresses, and then the entire sequence is read out from a sequence of addresses that is defined by the procedure described in the following.

Let the sequence of input addresses be from 0 to k – 1. Then, the sequence of interleaver output addresses shall be equivalent to those generated by the procedure illustrated in Figure 85 and described in the following.[41]

1. Determine the turbo interleaver parameter, n, where n is the smallest integer such that $k \le 2^{n+5}$.
2. Initialize an (n + 5)-bit counter x to 0.
3. Let $x' = (\lfloor x/32 \rfloor + 1) \bmod 2^n$. x' is generated by extracting the n most significant bits (MSBs) from the counter, adding one to form a new value, and then discarding all except the n least significant bits (LSBs) of this value.
4. Obtain the n-bit output of the table lookup defined in Table 95 with a read address equal to the five LSBs of the counter x, and call this output x". Note that this table depends on the value of n.
5. Multiply the values x' and x" obtained in Steps 3 and 4, and discard all except the n LSBs to get a number y. $y = x'x'' \bmod 2^n$.
6. Bit-reverse the five LSBs of the counter x to get a five-bit number y'.
7. Form a tentative output address z that has its MSBs equal to the value y' obtained in Step 6 and its LSBs equal to the value y obtained in Step 5.
8. Accept the tentative output address z as an output address if it is less than k; otherwise, discard it.
9. Increment the counter and repeat Steps 3 through 8 until all k interleaver output addresses are obtained.

---

[41] This procedure is equivalent to one where the counter values are written into a 2^5-row by 2^n-column array by rows, the rows are shuffled according to a bit-reversal rule, the elements within each row are permuted according to a row-specific linear congruential sequence, and tentative output addresses are read out by column. The linear congruential sequence rule is $x(i + 1) = (x(i) + c) \bmod 2^n$, where $x(0) = c$ and c is a row-specific value from a table lookup.

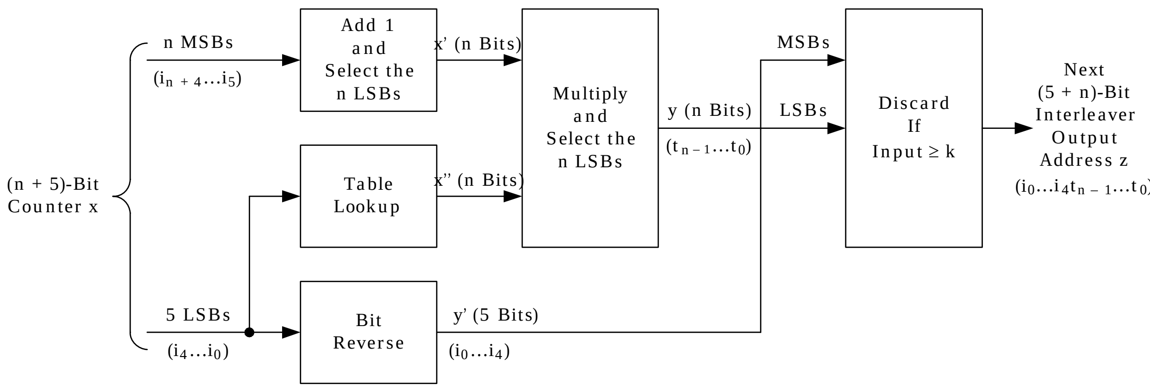

Figure 85 Turbo interleaver output address calculation procedure

Table 95 Turbo interleaver lookup table definition

Table Index	n = 2 Entries	n = 3 Entries	n = 4 Entries	n = 5 Entries	n = 6 Entries	n = 7 Entries	n = 8 Entries	n = 9 Entries
0	3	1	5	27	3	15	3	13
1	3	1	15	3	27	127	1	335
2	3	3	5	1	15	89	5	87
3	1	5	15	15	13	1	83	15
4	3	1	1	13	29	31	19	15
5	1	5	9	17	5	15	179	1
6	3	1	9	23	1	61	19	333
7	1	5	15	13	31	47	99	11
8	1	3	13	9	3	127	23	13
9	1	5	15	3	9	17	1	1
10	3	3	7	15	15	119	3	121
11	1	5	11	3	31	15	13	155
12	1	3	15	13	17	57	13	1
13	1	5	3	1	5	123	3	175
14	1	5	15	13	39	95	17	421
15	3	1	5	29	1	5	1	5
16	3	3	13	21	19	85	63	509
17	1	5	15	19	27	17	131	215
18	3	3	9	1	15	55	17	47
19	3	5	3	3	13	57	131	425
20	3	3	1	29	45	15	211	295
21	1	5	3	17	5	41	173	229
22	3	5	15	25	33	93	231	427
23	1	5	1	29	15	87	171	83
24	3	1	13	9	13	63	23	409

Table Index	n = 2 Entries	n = 3 Entries	n = 4 Entries	n = 5 Entries	n = 6 Entries	n = 7 Entries	n = 8 Entries	n = 9 Entries
25	1	5	1	13	9	15	147	387
26	3	1	9	23	15	13	243	193
27	1	5	15	13	31	15	213	57
28	3	3	11	13	17	81	189	501
29	1	5	3	1	5	57	51	313
30	1	5	15	13	15	31	15	489
31	3	3	5	13	33	69	67	391

9.2.3 Channel interleaving

The turbo or convolutional encoding shall be followed by channel interleaving, which consists of bit demultiplexing followed by bit permuting.

9.2.3.1 Bit demultiplexing

The output bits generated by the rate-1/3 convolutional encoder shall be reordered according to the following procedure:

1. All of the convolutional encoder output bits shall be demultiplexed into three sequences denoted V_0, V_1, V_2. The encoder output bits shall be sequentially distributed from the V_0 sequence to the V_2 sequence with the first bit going to the V_0 sequence, the second bit going to the V_1 sequence, the third to the V_2 sequence, the fourth to the V_0 sequence, etc.

2. The V_0, V_1, and V_2 sequences shall be ordered according to V_0V_1 V_2. That is, the V_0 sequence shall be first, the V_1 sequence shall be second, and the V_2 sequence shall be last.

The output bits generated by the rate-1/5 turbo encoder shall be reordered according to the following procedure:

1. All of the turbo encoder output data bits (i.e., the 5k bits output in the first k clock periods) shall be demultiplexed into five sequences denoted U, V_0, V_1, V'_0, and V'_1. The encoder output bits shall be sequentially distributed from the U sequence to the V'_1 sequence with the first encoder output bit going to the U sequence, the second to the V_0 sequence, the third to the V_1 sequence, the fourth to the V'_0 sequence, the fifth to the V'_1 sequence, the sixth to the U sequence, etc.

2. The 18 tail bits numbered 0 through 17 (i.e., the 18 bits generated during the last six clock periods) shall be distributed as follows: Tail bits numbered 0, 3, 6, 9, 12, and 15 shall go to the U sequence, the tail bits numbered 1, 4, and 7 shall go to the V_0 sequence, the tail bits numbered 2, 5, and 8 shall go to the V_1 sequence, the tail bits numbered 10, 13, and 16 shall go to the V'_0 sequence, and the tail bits numbered 11, 14, and 17 shall go to the V'_1 sequence. In other words, the tail bits of each non-systematic stream are allocated to the corresponding sequence.

3. The U, V_0, V_1, V'_0, and V'_1 sequences shall be ordered according to $UV_0V'_0V_1V'_1$. That is, the U sequence shall be first and the V'_1 sequence shall be last.

9.2.3.2 Bit permuting

The demultiplexed bits shall be permuted in three separate interleaver blocks with rate-1/5 coding and in one block with rate-1/3 coding. For the rate 1/5 turbo code, the permuter input blocks shall consist of the U sequence, the V_0 sequence followed by the V'_0 sequence (denoted as V_0/V'_0), and the V_1 sequence followed by the V'_1 sequence (denoted as V_1/V'_1). For the rate-1/3 convolutional code, the permuter input block shall consist of the V_0 sequence followed by the V_1 sequence followed by the V_2 sequence (denoted as $V_0/V_1/V_2$). A Pruned Bit-Reversal Interleaver (PBRI) shall be used for permuting each of the blocks.

The PBRI shall be functionally equivalent to an approach where the entire sequence of input bits in the block are written sequentially into an array at a sequence of addresses, and then the entire sequence is read out from a sequence of addresses that is defined by the procedure described in the following.

Let the number of bits in the input block be k_b, and let the sequence of input addresses be from 0 to k_b – 1. Then, the sequence of interleaver output addresses shall be equivalent to those generated by the procedure described in the following.

1. Determine the PBRI parameter, n, where n is the smallest integer such that $k_b \le 2^n$.
2. Initialize a counter j to 0.
3. Form a tentative output address that is equal to the bit-reversed value of j, using an n-bit binary representation. For example, if n = 4 and j = 3, then the bit reversed value of j is 12.
4. Accept the tentative output address as an output address if it is less than k_b; otherwise, discard it.
5. Increment the counter j and repeat Steps 3 through 5 until all k_b interleaver output addresses are obtained.

With rate-1/5 turbo coding, the interleaver output sequence shall be the interleaved U sequence of bits followed by the interleaved V_0/V'_0 sequence of bits followed by the interleaved V_1/V'_1 sequence of bits. With rate-1/3 convolutional coding, the interleaver output sequence shall be the interleaved $V_0/V_1/V_2$ sequence of bits.

9.2.4 Sequence repetition

Let $x_0, x_1, \ldots, x_{n-1}$ be the sequence of bits at the output of the channel interleaver. This sequence of bits shall be repeated to create a sequence of output bits $y_0, y_1, \ldots$ The output buffer $y_0, y_1, \ldots$ is read sequentially by the modulator, described in 9.2.6, until the required number of modulation symbols has been generated. The number of repetitions of the interleaver output sequence x_i shall be such that the modulator does not reach the end of the output buffer while generating modulation symbols.

The output sequence y_o, y_1, … shall be equivalent to an infinite sequence described by the formula $y_i = x_{i \bmod n}$, where n is the number of bits at the output of the channel interleaver.[42]

9.2.5 Data scrambling

The sequence y_0, y_1, … at the output of the sequence repetition stage shall be data-scrambled to randomize the data prior to modulation. The scrambling sequence shall be equivalent to one generated with a 17-tap linear feedback shift register with a generator sequence of $h(D) = 1 + D^{14} + D^{17}$, as shown in Figure 86. The n'th output s(n) of this shift register shall satisfy $s(n) = s(n\text{-}14) \oplus s(n\text{-}17)$. At the start of the physical layer packet, the shift register shall be initialized to the state $[b_{16}b_{15}b_{14}b_{13}b_{12}b_{11}b_{10}b_9b_8b_7b_6b_5b_4b_3b_2b_1b_0]$, which is a bitwise XOR of the vectors $[r_{10}r_9r_8r_7r_6r_5r_4r_3r_2r_1r_0\ d_5d_4d_3d_2d_1d_0]$ and $[t_{10}t_9t_8t_7t_6t_5t_4t_3t_2t_1t_0\ f_5f_4f_3f_2f_1f_0]$. Here, the $r_{10}r_9r_8r_7r_6r_5r_4r_3r_2r_1r_0$ bits shall be the bits of a 11-bit MACID (with r_0 being the LSB and r_{10} the MSB) that will be specified in the description of each physical layer channel that uses this procedure. If the length of the MACID is less than 11 bits, then the MACID shall be padded with 0's on the left (i.e., in the MSBs) to generate a 11-bit MACID.

The $d_5d_4d_3d_2d_1d_0$ bits shall be the 6 LSBs of a packet format index (with d_0 being the LSB and d_5 being the MSB), and will also be specified in the description of each physical layer channel that uses this procedure.[43] If the specified packet format is less than 6 bits, then it shall be padded with 0's in the beginning (i.e., the MSBs shall be set to 0) in order to achieve the desired length. The $t_{10}t_9t_8t_7t_6t_5t_4t_3t_2t_1t_0$ bits shall be the 11 LSBs of the superframe index of the superframe in which the first modulation symbol of this packet is transmitted. The $f_5f_4f_3f_2f_1f_0$ bits shall be the PHY Frame index (within the superframe) of the PHY Frame in which the first modulation symbol of this packet is transmitted. An all-zeros PHY Frame index is used if packet transmission begins during the superframe preamble.

The first bit in the scrambling sequence shall be generated by the initial state of the shift register. Each subsequent bit shall be generated by clocking the shift register once. Every bit at the output of the sequence repetition stage shall be XORed with the corresponding bit of the scrambling sequence to yield a scrambled bit.

The data-scrambling operation can be omitted for some physical layer channels. This shall be specified in the description of the relevant channel.

---

[42] This procedure is equivalent to repeating the original bits as often as required. The modulator, described in Section 9.2.6, will read this sequence sequentially, and the number of repetitions should be such that the modulator does not reach the end of the repeated sequence before the required number of modulation symbols is generated.

[43] The MACID bits will normally correspond to the MACID of the target user where this makes sense, and will be set to all 0's otherwise. The packet format bits will correspond to the packet format for a data packet, and will be set to all 0's otherwise.

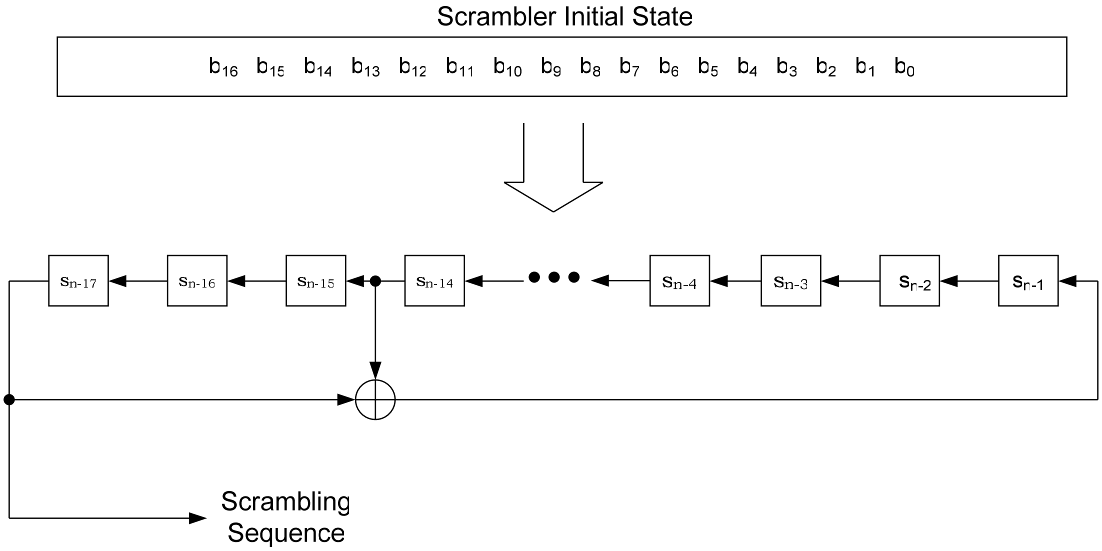

Figure 86 Data scrambler

9.2.6 Modulation

The outputs of the data-scrambler for all the different sub-packets, where a sub-packet is as defined in 9.2.1, shall be applied to a modulator that outputs complex numbers known as modulation symbols, which are modulated on to OFDM subcarriers. The modulator cycles between the different sub-packets while generating the modulation symbols, i.e., modulation symbol 0 is generated from sub-packet 0, modulation symbol 1 is generated from sub-packet 1, etc. The airlink supports QPSK, 8PSK, 16QAM and 64QAM modulation formats. The number of bits in one modulation symbol is called the modulation order. The modulation order is 2 for QPSK, 3 for 8PSK, 4 for 16QAM and 6 for 64QAM. The airlink also supports multiple modulation formats for a single encoded bit sequence. This section describes the procedure for generating a sequence of modulation symbols of varying modulation formats from an encoded bit sequence (after channel interleaving and scrambling). The number of modulation symbols and the modulation format of each symbol will be specified separately in the specification of each physical layer channel using this procedure.

The sequence of modulation symbols output from the modulator shall be equivalent to those generated by the following approach:

1. Let y(0,0), y(0,1), … be the infinite-length sequence of bits at the output of the scrambler corresponding to sub-packet 0, y(1,0), y(1,1), … the infinite-length sequence of bits at the output of the scrambler for sub-packet 1 and so on. Let t be the total number of sub-packets. Initialize t counters, denoted by $i_0, i_1, \ldots, i_{t-1}$, to 0. Initialize another set of t counters $j_0, j_1, \ldots, j_{t-1}$, to 0. Counter i_m counts the number of modulation symbols that have already been generated for the m^{th} sub-packet, while counter j_m is a pointer to the bits that were last modulated for the m^{th} sub-packet. Initialize another counter k = 0, which counts the total number of modulation symbols generated.

2. Let q be the desired modulation order of the next modulation symbol. Let m = k mod t. Collect the sequence of q bits $y(m,j_m), y(m,j_m+1), \ldots, y(m,j_m+q-1)$. A sequence of bits $z(0), \ldots, z(q-1)$ is obtained by rotating this sequence by the value i_m mod q, i.e., $z(p) = y(m,j_m+((i_m+p) \bmod q))$.

3. The rotated sequence z(0), z(1), …, z(q-1) is then mapped to a modulation symbol using the mapping described in 9.2.6.1 for QPSK, in 9.2.6.2 for 8PSK, in 9.2.6.3 for 16QAM, and in 9.2.6.4 for 64QAM.

4. Increment counter i_m by 1. Increment counter j_m by q. Increment counter k by 1.

5. Repeat steps 2 through 4 until the desired number of modulation symbols have been generated.

9.2.6.1 QPSK modulation

In the case of QPSK modulation, a group of 2 input bits (b_0, b_1) is mapped into a complex modulation symbol ($m_I(k)$, $m_Q(k)$), as specified in Table 96. Figure 87 shows the signal constellation of the QPSK modulator.

Table 96 QPSK modulation table

Input Bits		Modulation Symbols	
b_0	b_1	$m_I(k)$	$m_Q(k)$
0	0	A	A
0	1	–A	A
1	0	A	–A
1	1	–A	–A

Note: $A = 1/\sqrt{2}$

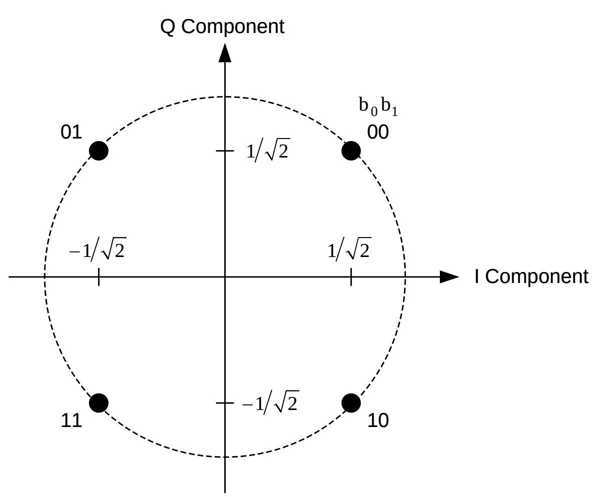

Figure 87 Signal constellation for QPSK modulation

9.2.6.2 8-PSK modulation

In the case of 8-PSK modulation, a group of 3 input bits (b_0, b_1, b_2) is mapped into a complex modulation symbol ($m_I(k)$, $m_Q(k)$) as specified in Table 97. Figure 88 shows the signal constellation of the 8-PSK modulator.

Table 97 8-PSK modulation table

Input Bits			Modulation Symbols	
b_0	b_1	b_2	$m_I(k)$	$m_Q(k)$
0	0	0	C	S
0	0	1	S	C
0	1	1	–S	C
0	1	0	–C	S
1	1	0	–C	–S
1	1	1	–S	–C
1	0	1	S	–C
1	0	0	C	–S

Note: $C = \cos(\pi/8) \approx 0.9239$ and $S = \sin(\pi/8) \approx 0.3827$

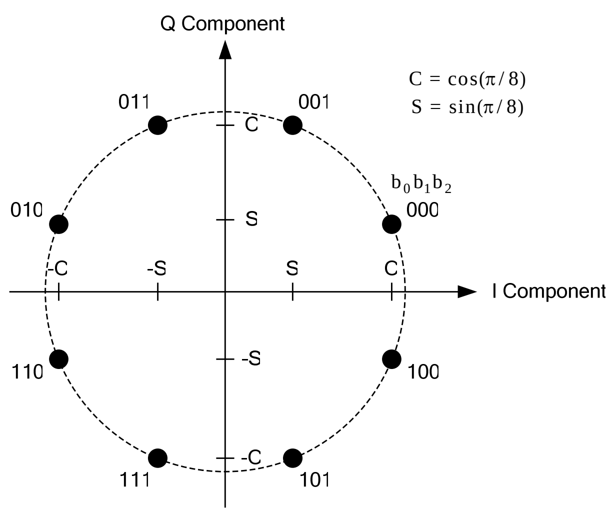

Figure 88 Signal constellation for 8-PSK modulation

9.2.6.3 16-QAM modulation

In the case of 16-QAM modulation, a group of 4 input bits (b_0, b_1, b_2, b_3) is mapped into a complex modulation symbol ($m_I(k)$, $m_Q(k)$), as specified in Table 98. Figure 89 shows the signal constellation of the 16-QAM modulator.

Table 98 16-QAM modulation table

Input Bits				Modulation Symbols	
b_0	b_1	b_2	b_3	$m_Q(k)$	$m_I(k)$
0	0	0	0	3A	3A
0	0	0	1	3A	A
0	0	1	1	3A	–A
0	0	1	0	3A	–3A
0	1	0	0	A	3A
0	1	0	1	A	A
0	1	1	1	A	–A
0	1	1	0	A	–3A
1	1	0	0	–A	3A
1	1	0	1	–A	A
1	1	1	1	–A	–A
1	1	1	0	–A	–3A
1	0	0	0	–3A	3A
1	0	0	1	–3A	A
1	0	1	1	–3A	–A
1	0	1	0	–3A	–3A

Note: $A = 1/\sqrt{10} \approx 0.3162$

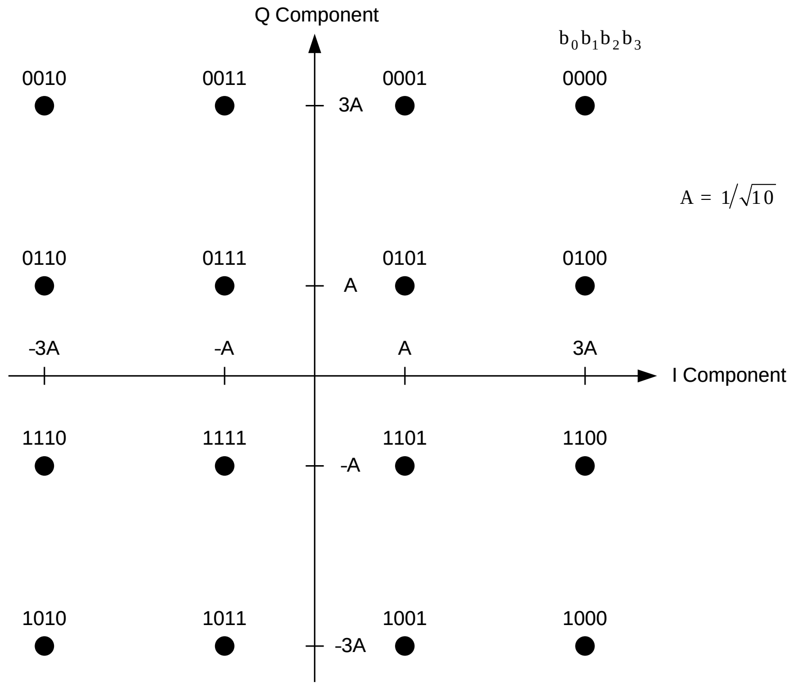

Figure 89 Signal constellation for 16-QAM modulation

9.2.6.4 64-QAM modulation

In the case of 64-QAM modulation, a group of 6 input bits (b_0, b_1, b_2, b_3, b_4, b_5) is mapped into a complex modulation symbol ($m_I(k)$, $m_Q(k)$) as specified in Table 99. Figure 90 shows the signal constellation of the 64-QAM modulator.

Table 99 64-QAM modulation table

Input Bits						Modulation Symbols	
b_0	b_1	b_2	b_3	b_4	b_5	$m_Q(k)$	$m_I(k)$
0	0	0	0	0	0	7A	7A
0	0	0	0	0	1	7A	5A
0	0	0	0	1	0	7A	A
0	0	0	0	1	1	7A	3A
0	0	0	1	0	0	7A	-7A
0	0	0	1	0	1	7A	-5A
0	0	0	1	1	0	7A	-A
0	0	0	1	1	1	7A	-3A
0	0	1	0	0	0	5A	7A
0	0	1	0	0	1	5A	5A
0	0	1	0	1	0	5A	A
0	0	1	0	1	1	5A	3A

Input Bits						Modulation Symbols	
b_0	b_1	b_2	b_3	b_4	b_5	$m_Q(k)$	$m_I(k)$
0	0	1	1	0	0	5A	-7A
0	0	1	1	0	1	5A	-5A
0	0	1	1	1	0	5A	-A
0	0	1	1	1	1	5A	-3A
0	1	0	0	0	0	A	7A
0	1	0	0	0	1	A	5A
0	1	0	0	1	0	A	A
0	1	0	0	1	1	A	3A
0	1	0	1	0	0	A	-7A
0	1	0	1	0	1	A	-5A
0	1	0	1	1	0	A	-A
0	1	0	1	1	1	A	-3A
0	1	1	0	0	0	3A	7A
0	1	1	0	0	1	3A	5A
0	1	1	0	1	0	3A	A
0	1	1	0	1	1	3A	3A
0	1	1	1	0	0	3A	-7A
0	1	1	1	0	1	3A	-5A
0	1	1	1	1	0	3A	-A
0	1	1	1	1	1	3A	-3A
1	0	0	0	0	0	-7A	7A
1	0	0	0	0	1	-7A	5A
1	0	0	0	1	0	-7A	A
1	0	0	0	1	1	-7A	3A
1	0	0	1	0	0	-7A	-7A
1	0	0	1	0	1	-7A	-5A
1	0	0	1	1	0	-7A	-A
1	0	0	1	1	1	-7A	-3A
1	0	1	0	0	0	-5A	7A
1	0	1	0	0	1	-5A	5A
1	0	1	0	1	0	-5A	A
1	0	1	0	1	1	-5A	3A
1	0	1	1	0	0	-5A	-7A
1	0	1	1	0	1	-5A	-5A
1	0	1	1	1	0	-5A	-A
1	0	1	1	1	1	-5A	-3A
1	1	0	0	0	0	-A	7A

Input Bits						**Modulation Symbols**	
b_0	**b_1**	**b_2**	**b_3**	**b_4**	**b_5**	**$m_Q(k)$**	**$m_I(k)$**
1	1	0	0	0	1	-A	5A
1	1	0	0	1	0	-A	A
1	1	0	0	1	1	-A	3A
1	1	0	1	0	0	-A	-7A
1	1	0	1	0	1	-A	-5A
1	1	0	1	1	0	-A	-A
1	1	0	1	1	1	-A	-3A
1	1	1	0	0	0	-3A	7A
1	1	1	0	0	1	-3A	5A
1	1	1	0	1	0	-3A	A
1	1	1	0	1	1	-3A	3A
1	1	1	1	0	0	-3A	-7A
1	1	1	1	0	1	-3A	-5A
1	1	1	1	1	0	-3A	-A
1	1	1	1	1	1	-3A	-3A

Note: $A = \sqrt{1/42} \approx 0.1543$

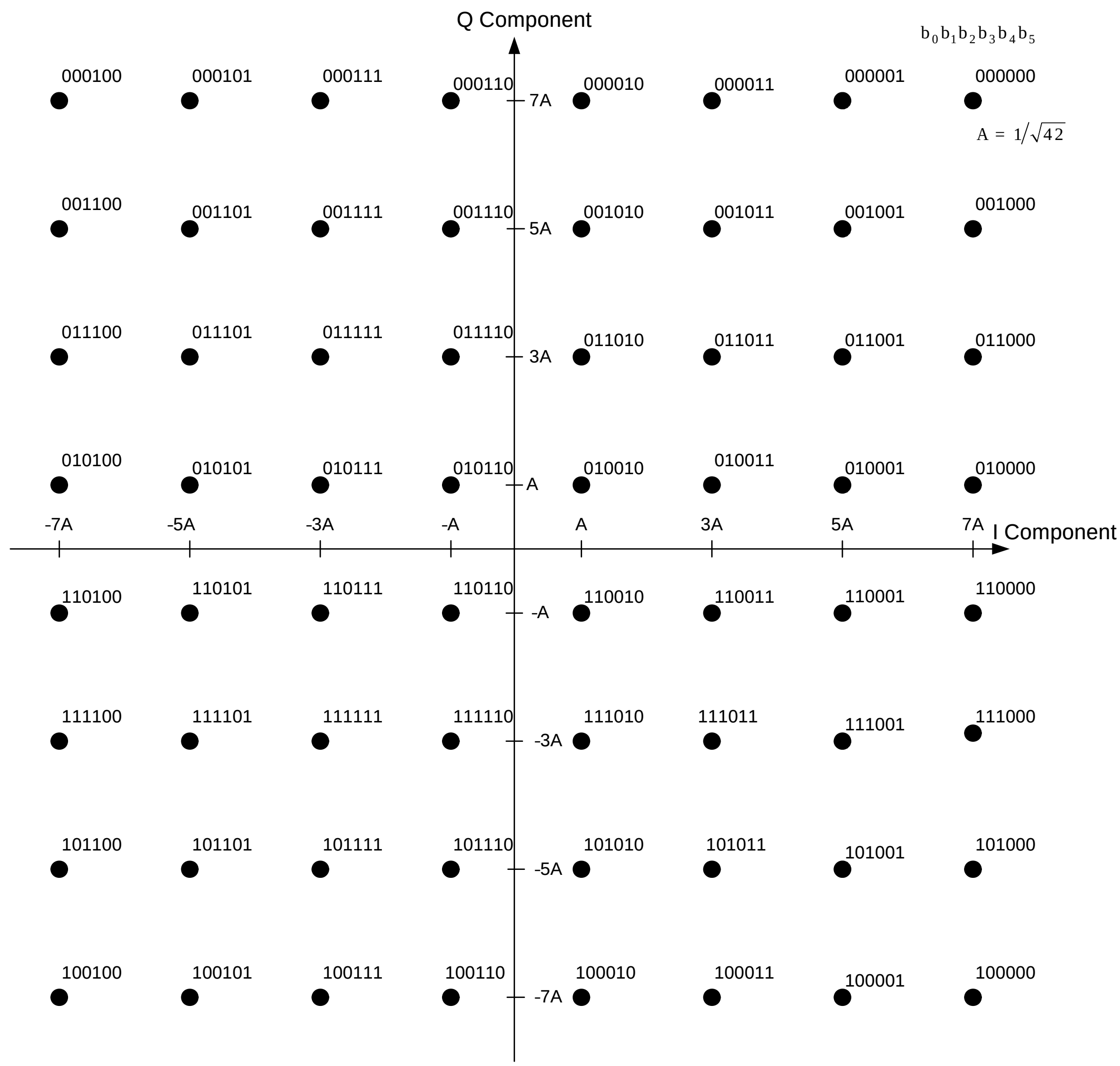

Figure 90 Signal constellation for 64-QAM modulation

9.3 Access network requirements

This section defines requirements specific to access network (AN) equipment and operation.

9.3.1 Transmitter

The transmitter shall reside in each sector of the access network. These requirements apply to the transmitter in each sector.

Each sector is assigned an integer identifier in the range 0-4095 (including 0 and 4095) called the PilotPN. This identifier may also be referred to as a 12-bit binary number with the leftmost bit being the MSB and the rightmost bit being the LSB.

9.3.2 Modulation characteristics

9.3.2.1 Superframe timing

The forward link transmission is divided into units of superframes. The structure of a forward link superframe shall be as shown in Figure 91 for FDD and as shown in Figure 92, Figure 93, Figure 94 for TDD for different values of the parameters N_{FL_BURST} and N_{RL_BURST}. A superframe shall consist of a superframe preamble followed by a series of $N_{FDD,FLPHYFrames}$ FL PHY Frames in FDD, and by $N_{TDD,FLPHYFrames}$ FL PHY Frames in TDD. In TDD mode with parameter N_{FL_BURST} and N_{RL_BURST}, N_{FL_BURST} FL PHY Frames alternate with the mute time reserved for RL PHY Frames. Here, $N_{FDD,FLPHYFrames}$ and $N_{TDD,FLPHYFrames}$[44] are as defined by the Lower MAC sublayer. The superframe preamble carries the F-CPICH, the F-pBCH, the F-ACQCH, and the F-OSICH. The FL PHY Frames carry the F-CPICH, F-AuxPICH, the F-SSCH and the F-DCH physical channels for SymbolRateHopping mode and the F-DPICH, the F-CPICH, the F-SSCH, and the F-DCH physical channels in BlockHopping mode. The structure of the superframe preamble and each FL PHYFrame shall be as shown in Figure 95 for SymbolRateHopping mode and in Figure 96 for BlockHopping mode.

Each superframe shall be identified by a superframe index that is incremented every superframe. The superframe index is related to the System Time as described in Chapter 1. Each superframe also has an associated quantity called the PilotPhase, defined as (PilotPN + Superframe Index) mod 4096.

The PHY layer chapter of this specification uses a FL PHY Frame indexing scheme that is convenient for the descriptions herein, but is not necessarily consistent with indexing schemes used in other layers and sublayers in the specification. In this indexing scheme, the FL PHY Frames in a given superframe shall be indexed sequentially from 0 through $N_{FDD,FLPHYFrames}$ -1 in FDD mode and from 0 through $N_{TDD,FLPHYFrames}$ -1 in TDD mode. The FL PHY Frame index is sometimes also referred to using its 6-bit binary representation.

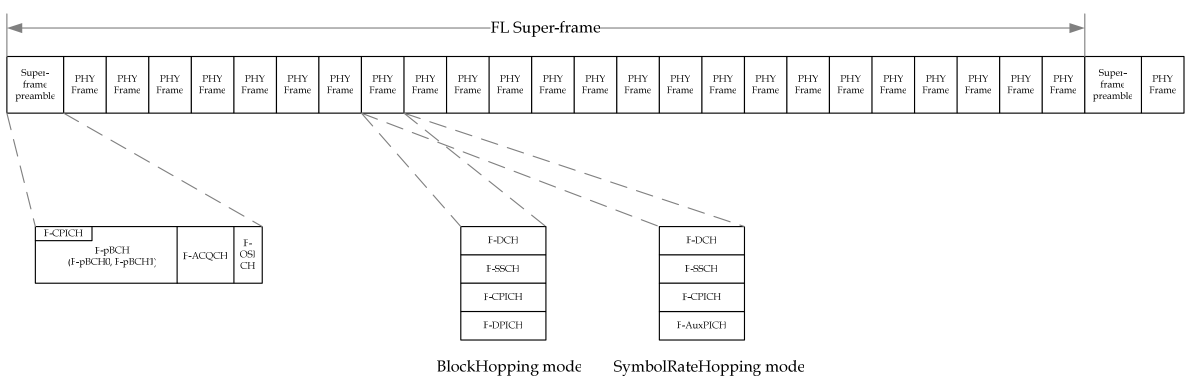

Figure 91 Forward link superframe structure: FDD

[44] Note that $N_{TDD,FLPHYFrames}$ is a function of N_{FL_BURST} and N_{RL_BURST}.

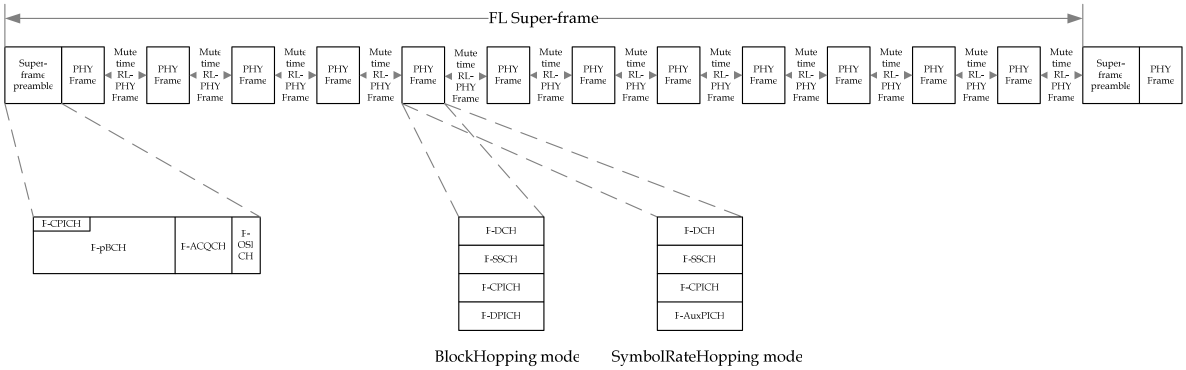

Figure 92 Forward link superframe structure: TDD (N_{FL_BURST} =1, N_{RL_BURST} =1)

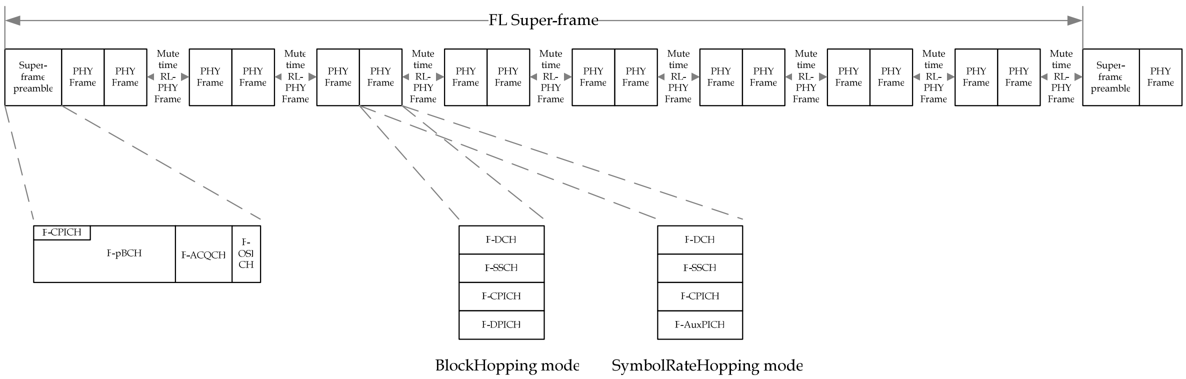

Figure 93 Forward link superframe structure: TDD (N_{FL_BURST} =2, N_{RL_BURST} =1)

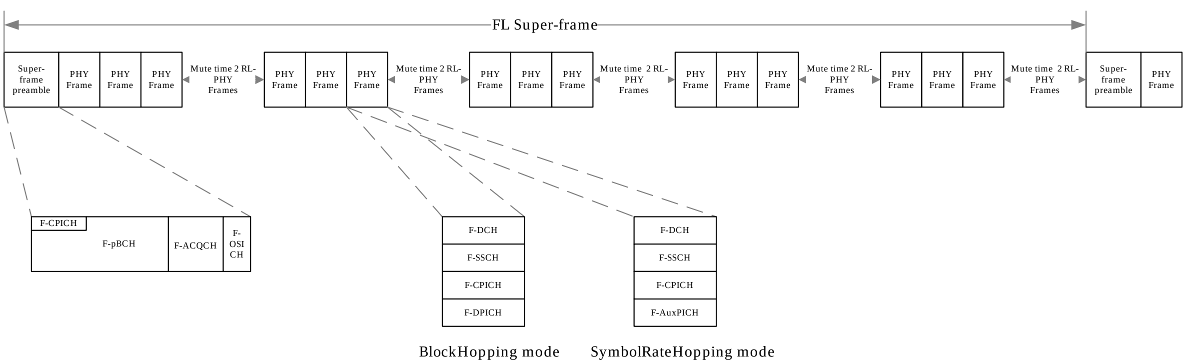

Figure 94 Forward link superframe structure: TDD (N_{FL_BURST} =3, N_{RL_BURST} =2)

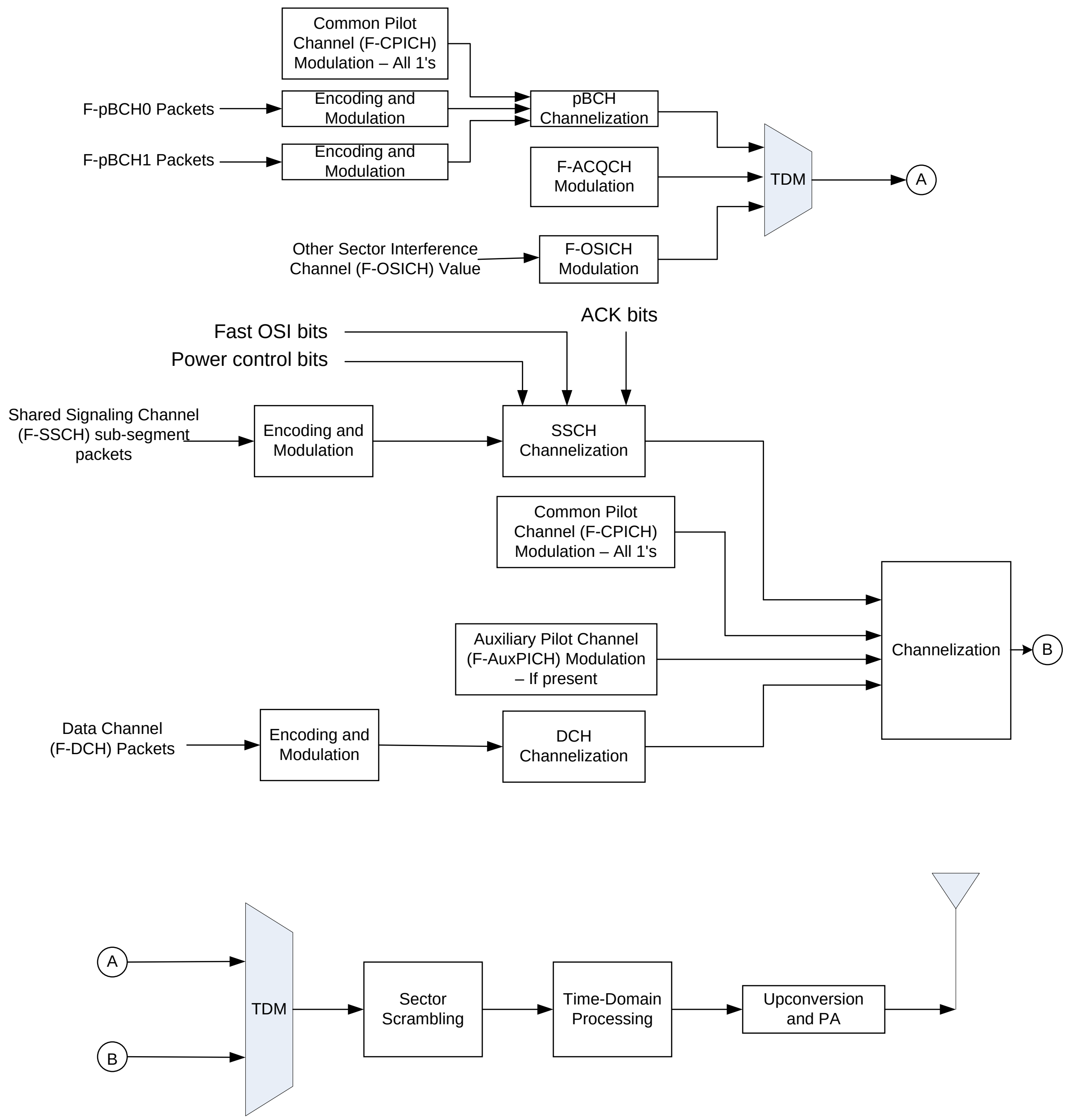

Figure 95 Forward channel structure: SymbolRateHopping mode

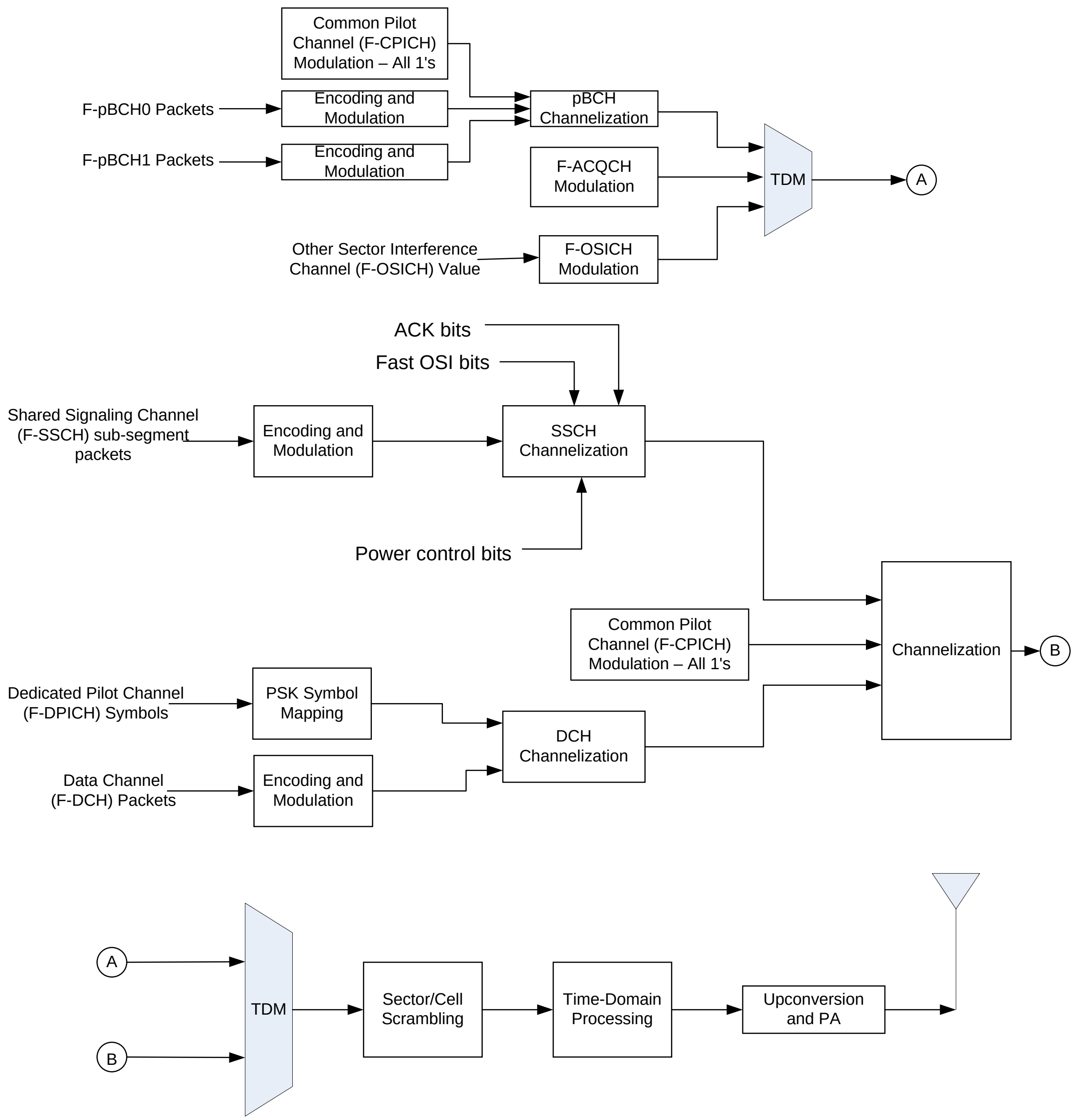

Figure 96 Forward channel structure: BlockHopping mode

9.3.2.2 OFDM symbol characteristics

The modulation used on the forward link is Orthogonal Frequency Division Multiplexing (OFDM). Both the superframe preamble as well as each FL PHY Frame shall be further subdivided into units of OFDM symbols. An OFDM symbol consists of N_{FFT} individually modulated subcarriers that carry complex-valued data. Complex-valued data is represented in the form $d = (d_{re}, d_{im})$, where d_{re} and d_{im} represent the real and imaginary components respectively. The subcarriers in each OFDM symbol shall be numbered 0 through N_{FFT}-1.

An additional indexing scheme may be used in MultiCarrierOn mode. The N_{FFT} subcarriers are split into $N_{CARRIERS}$ contiguous groups, each of which is referred to as a carrier. Each carrier consists of

$N_{CARRIER\_SIZE}$ subcarriers, where $N_{CARRIER\_SIZE} = N_{FFT} / N_{CARRIERS}$. Each carrier has an associated index, sometimes referred to as CarrierIndex, that ranges from 0 through $N_{CARRIERS}$ -1. The carrier with index c consists of subcarriers indexed $cN_{CARRIER\_SIZE}$ through $(c+1)N_{CARRIER\_SIZE}$ -1. In MultiCarrierOff mode, all N_{FFT} subcarriers belong to a single carrier having CarrierIndex 0. Furthermore, the subcarriers within each carrier may be indexed from 0 to $N_{CARRIER\_SIZE}$ -1 and the phrases “subcarrier f in carrier c” and “subcarrier with index f within carrier with index c” shall be equivalent to “subcarrier $cN_{CARRIER\_SIZE} + f$.” These two subcarrier indexing schemes are used interchangeably in the Physical Layer chapter of this specification.

9.3.2.2.1 Guard subcarriers

Some of the available subcarriers in an OFDM symbol are designated as guard subcarriers and shall not be modulated, i.e., no energy shall be transmitted on these subcarriers. The number of guard subcarriers in the superframe preamble and in each FL PHY Frame shall be $N_{GUARD,PR}$ and N_{GUARD} respectively. The quantity N_{GUARD} is given by the NumGuardSubcarriers parameter which is part of the public data of the Overhead Messages Protocol. The set of guard subcarriers in the superframe preamble shall be the subcarriers numbered 0 through $N_{GUARD,PR}/2 – 1$ and the subcarriers numbered $N_{FFT}-N_{GUARD,PR}/2$ through $N_{FFT}-1$. The set of guard subcarriers in each FL PHY Frame shall be the subcarriers numbered 0 through $N_{GUARD}/2 – 1$ and the subcarriers numbered $N_{FFT}-N_{GUARD}/2$ through $N_{FFT}-1$.

9.3.2.2.2 Quasi-guard subcarriers

In MultiCarrierOn mode, additional sub-carriers within each OFDM symbol are designated as quasi-guard subcarriers and shall not be modulated, i.e., no energy shall be transmitted on these subcarriers. The set of quasi-guard subcarriers in the superframe preamble shall be the subcarriers numbered $N_{CARRIER\_SIZE}*m- N_{GUARD,PR}/2$ through $N_{CARRIER\_SIZE}*m + N_{GUARD,PR}/2 – 1$ where $m = 1, …,N_{CARRIERS}-1$. The set of quasi-guard subcarriers in each FL PHY Frame shall be the subcarriers numbered $N_{CARRIER\_SIZE}*m- N_{GUARD}/2$ to $N_{CARRIER\_SIZE}*m + N_{GUARD}/2 – 1$ where $m = 1, …,N_{CARRIERS}-1$. Subcarriers that are not guard or quasi-guard subcarriers are also referred to as usable subcarriers.

9.3.2.2.3 OFDM symbol duration

The total OFDM symbol duration, denoted by $T_{s,PR}'$ during the superframe preamble and by T_s' during each FL PHY Frame, consists of four parts:

- A data part with duration T_{FFT}, where $T_{FFT} = N_{FFT} T_{CHIP}$.
- A flat guard interval, also known as a cyclic prefix, with duration $T_{CP,PR}$ in the superframe preamble and duration T_{CP} during each FL PHY Frame. Here, T_{CP} is given by the CPLength parameter which is part of the public data of the Overhead Messages Protocol.
- Two windowed guard intervals of duration T_{WGI} on the two sides of the OFDM symbol. There is an overlap of T_{WGI} between consecutive OFDM symbols (see Figure 106).

The effective OFDM symbol duration is $T_{s,PR} = T_{FFT} + T_{CP,PR} + T_{WGI}$ in the superframe preamble, and $T_s = T_{FFT} + T_{CP} + T_{WGI}$ in each FL PHY Frame. This effective symbol duration shall henceforth be referred to as the OFDM symbol duration.

9.3.2.2.4 Superframe duration

The superframe preamble consists of $N_{PREAMBLE}$ OFDM symbols. In FDD, each FL PHY Frame consists of $N_{FRAME,F}$ OFDM symbols and the total superframe duration is given by

$$T_{SUPERFRAME} = N_{PREAMBLE}T_{s,PR} + N_{FDD,FLPHYFrames}N_{FRAME,F}T_s.$$

In TDD, each FL PHY Frame consists of $N_{FRAME,F}$ OFDM symbols. The mute time between each set of N_{FL_BURST} contiguous FL PHY Frames in TDD is equal to the duration of N_{RL_BURST} contiguous RL PHY Frames plus guard times $T_{G,TDD,R}$ and $T_{G,TDD,F}$. The total superframe duration is given by

$$T_{SUPERFRAME} = N_{PREAMBLE}T_{s,PR} + N_{TDD,FLPHYFrames}N_{FRAME,F}T_s + N_{TDD,RLPHYFrames}N_{FRAME,R}T_s$$

$$+ (T_{G,TDD,F} + T_{G,TDD,R}) * (N_{TDD,RLPHYFrames}/N_{RL\_BURST})$$

9.3.2.2.5 Hop-port indexing

During the FL PHY Frame portion of the transmission, the subcarriers in each carrier of each OFDM symbol will also use a second indexing scheme known as hop-port indexing. In this scheme, each carrier in each OFDM symbol consists of $Q_{SDMA}N_{CARRIER\_SIZE}$ individually-modulated hop-ports. Here Q_{SDMA} is equal to FLNumSDMADimensions, which is part of the public data of the Overhead Messages Protocol on that carrier. The hop-ports in each carrier are indexed from 0 through $Q_{SDMA}N_{CARRIER\_SIZE}$ -1. The hop-port with index p in the carrier with CarrierIndex k is sometimes represented by the pair (k,p). An order is defined on the set of such pairs by saying that $(k_0,p_0) < (k_1,p_1)$ if either of the following two conditions is satisfied:

i. $k_0 < k_1$, or

ii. $k_0 = k_1$ and $p_0 < p_1$.

There is a mapping between the $Q_{SDMA}N_{CARRIER\_SIZE}$ hop-ports and the $N_{CARRIER\_SIZE}$ subcarriers in each carrier, called a hop-permutation. The hop-permutation may change as often as every OFDM symbol and is different for different sectors. The sequence of hop-permutations, also called the hopping sequence, is described in 9.3.2.5.1.

9.3.2.3 Multiple transmit antennas

Multiple transmit antennas may be present at the sector transmitter. The different transmit antennas shall have the same superframe timing (including the superframe index), the same OFDM symbol characteristics, and the same hop-permutation.

Modulation of some of the physical layer channels (for example the F-AuxPICH and the portion of the F-CPICH that lies in the FL PHY Frames) is specified separately for each transmit antenna. Here the term “transmit antenna” denotes an “effective transmit antenna” which is not necessarily the same as a physical antenna present at the sector.[45] The mapping between effective and physical transmit antennas is beyond the scope of this specification. Note that transmission on a single effective antenna may involve transmission on any or all of the physical antennas. The number of effective transmit antennas in the system is given by the EffectiveNumAntennas parameter, which is part of the

[45] Here, a physical antenna refers to an individual radiating element.

public data of the Overhead Messages Protocol. The different effective antennas in the system are indexed 0 through EffectiveNumAntennas – 1. Any reference to the term "transmit antenna" shall henceforth be interpreted as meaning an effective transmit antenna.

The modulation of some of the physical layer channels (for example all the channels in the superframe preamble, including the portion of the F-CPICH that lies in the superframe preamble) is specified only for a single effective antenna. If multiple effective antennas are present, the modulation symbols corresponding to these channels shall be transmitted only from the effective antenna with index 0.

Finally, the modulation of some of the physical layer channels (for example the F-DPICH) is described in terms of a concept called tile-antennas. A tile-antenna is a linear combination of the effective antennas present in the system. Multiple tile-antennas can be constructed using different linear combinations of the effective antennas. The process of constructing tile-antennas, i.e., the process of transmitting on a linear combination of effective antennas, is known as precoding. The mapping between effective antennas and tile-antennas can be described in terms of a matrix called the precoding matrix.

The physical layer also supports superposition of two waveforms on the same set of subcarriers, potentially using different precoding matrices. This happens when the hop-permutation maps two hop-ports to the same subcarrier, and is known as Space Division Multiple Access (SDMA).

The sector-specific scrambling, cell-specific scrambling and time-domain processing operations, described in 9.3.2.6, 9.3.2.7, and 9.3.2.8, respectively, shall be identical for each of the effective transmit antennas.

9.3.2.4 Superframe preamble modulation

The superframe preamble shall consist of $N_{PREAMBLE}$ = 8 OFDM symbols, which shall be indexed from 0 through $N_{PREAMBLE}$-1. The Common Pilot Channel (F-CPICH) shall be carried on the OFDM symbols with indices 0 and 1. The Primary Broadcast Channel (F-pBCH) shall be carried on the OFDM symbols with indices 0 through 4. The Acquisition Channel (F-ACQCH) shall be carried on the OFDM symbols with indices 5 and 6. Finally, the Other Sector Interference Channel (F-OSICH) shall be carried on the OFDM symbol with index 7.

9.3.2.4.1 $Offset_p$ definition

A variable $Offset_p$ is defined for each carrier and each OFDM symbol in the superframe. This variable shall be determined using a 13-bit PN-register with generator polynomial $h(D) = 1 + D^8 + D^{11} + D^{12} + D^{13}$, which is shown in Figure 97. For the carrier with CarrierIndex k_c, the shift-register shall be initialized to the state [1 p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0] before the beginning of the superframe. In SemiSynchronous mode, the bits $p_{11,}$ p_{10}, p_9, …, p_0 are the 12 bits of the quantity (PilotPhase + k_c) mod 4096, with p_{11} being the MSB and p_0 being the LSB. In Asynchronous mode, the bits $p_{11,}$ p_{10}, p_9, …, p_0 are the 12 bits of the quantity (PilotPN + k_c) mod 4096, with p_{11} being the MSB and p_0 being the LSB. $Offset_p$ shall be generated by clocking the shift-register 13 times for every even OFDM symbol in the superframe. More precisely, for the carrier with CarrierIndex k_c and the OFDM symbol with index j in the superframe, $Offset_p$ shall be chosen to be the value of this register after it has been clocked $\lfloor j/2 \rfloor * 13$ times. The value of the register shall be read from left to right; i.e., the left-most state shown in Figure 97 shall be treated as the MSB and the right-most state shall be treated as the LSB.

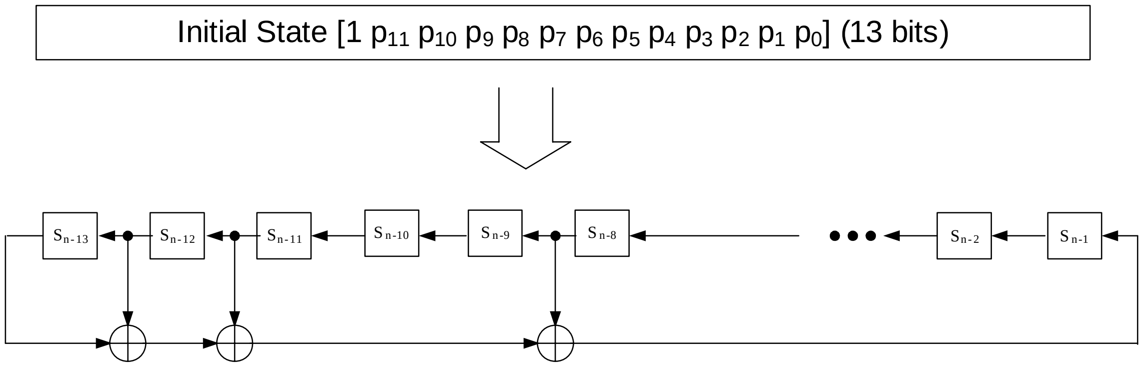

Figure 97 Offset$_p$ Determination

9.3.2.4.2 F-CPICH

The F-CPICH shall occupy an evenly spaced set of subcarriers in each carrier in the first two OFDM symbols of the superframe preamble. The spacing between neighboring pilot subcarriers for the superframe preamble shall be given by $D_{p,PR} = 4$. Let $Offset_{PR,k}$ be the value of $Offset_p$, as defined in 9.3.2.4.1, for the first two OFDM symbols in the superframe preamble and for the carrier with CarrierIndex k. For the OFDM symbol with index 0, the subcarrier with index i_{sc} in this OFDM symbol, where i_{sc} is such that this subcarrier lies in the carrier with CarrierIndex k, shall be occupied by the F-CPICH if it is a usable subcarrier and $i_{sc} \bmod D_{p,PR} = Offset_{PR,k} \bmod D_{p,PR}$. For the OFDM symbol with index 1, the subcarrier with index i_{sc} in this OFDM symbol, where i_{sc} is such that this subcarrier lies in the carrier with CarrierIndex k, shall be occupied by the F-CPICH if it is a usable subcarrier and $i_{sc} \bmod D_{p,PR} = (Offset_{PR,k} + D_{p,PR}/2) \bmod D_{p,PR}$.

Each subcarrier occupied by the F-CPICH shall be modulated with the complex value $(\sqrt{P}, 0)$, where P is the ratio of the power spectral density of the F-CPICH to the power spectral density of the second OFDM symbol in the F-ACQCH. This ratio is given by the PreamblePilotPower parameter, which is part of the public data of the Overhead Messages Protocol.

Any subcarrier in the superframe preamble occupied by the F-CPICH will be referred to as a pilot subcarrier.

9.3.2.4.3 F-pBCH

The F-pBCH shall be carried on the first five OFDM symbols in the superframe preamble. This channel consists of two sub-channels, namely F-pBCH0 and F-pBCH1.

9.3.2.4.3.1 F-pBCH0

One F-pBCH0 packet is transmitted over each carrier. An F-pBCH0 packet in the carrier with CarrierIndex k_c shall start in every superframe that satisfies (PilotPhase + k_c) mod $N_{pBCH0\_Period} = 0$, and shall be transmitted over a period of 16 superframes. Each F-pBCH0 packet is generated by the CC MAC Protocol and shall be appended with CRC, encoded, channel interleaved, repeated, and modulated using the procedures described in 9.2. A CRC length of $N_{CRC,pBCH}$ shall be used for this packet. No data-scrambling operation shall be performed on this packet. QPSK modulation shall be used for all of the modulation symbols. The modulation symbols for the F-pBCH0 packet to be transmitted in the carrier with CarrierIndex k_c shall be modulated on to OFDM subcarriers using the following procedure:

1. Initialize a superframe counter i to the superframe index corresponding to the first superframe carrying the F-pBCH0 packet. Initialize a subcarrier counter j to $N_{GUARD,PR}/2$.
2. If the subcarrier with index $N_{CARRIER\_SIZE}k_c + j$ in OFDM symbol index 0 of superframe index i is not a pilot subcarrier, then a QPSK modulation symbol s shall be generated using the procedure described in 9.2.6. This modulation symbol shall be modulated with energy P on the subcarrier with index $N_{CARRIER\_SIZE}k_c + j$, i.e., the value of the subcarrier shall be $\sqrt{P}\,s$. Here, P is the power density assigned to this packet by the CC MAC Protocol.
3. Increment j.
4. If $j = (N_{CARRIER\_SIZE}+N_{GUARD,PR})/4$, increment i and set $j = N_{GUARD,PR}/2$.
5. If i is larger than the superframe index of the last superframe carrying the F-pBCH0 packet, then stop. Otherwise repeat steps 2 through 5.

9.3.2.4.3.2 F-pBCH1

One F-pBCH1 packet is transmitted in each carrier in each superframe. Each F-pBCH1 packet is generated by the CC MAC Protocol and shall be appended with CRC, encoded, channel interleaved, repeated, and modulated using the procedures described in 9.2. A CRC length of $N_{CRC,pBCH}$ shall be used for this packet. No data-scrambling operation shall be performed on this packet. QPSK modulation shall be used for all of the modulation symbols. The modulation symbols for the F-pBCH1 packet to be transmitted in the carrier with CarrierIndex k_c in a given superframe shall be modulated on to OFDM subcarriers using the following procedure:

1. At the beginning of the superframe, initialize an OFDM symbol counter i to 0 and a subcarrier counter j to $(N_{CARRIER\_SIZE}+N_{GUARD,PR})/4$.
2. If the subcarrier with index $N_{CARRIER\_SIZE}k_c + j$ in the OFDM symbol with index i is not a pilot subcarrier, then a QPSK modulation symbol s shall be generated using the procedure described in 9.2.6. This modulation symbol shall be modulated with energy P on the subcarrier with index $N_{CARRIER\_SIZE}k_c + j$, i.e., the value of the subcarrier is $\sqrt{P}\,s$. Here, P is the power density assigned to this packet by the CC MAC Protocol.
3. Increment j.

4. If $j = N_{CARRIER\_SIZE} - N_{GUARD,PR}/2$ increment i and set $j = N_{GUARD,PR}/2$.

5. If i = 5, then stop. Else repeat steps 2 through 5.

9.3.2.4.4 F-ACQCH

The Acquisition Channel shall consist of the OFDM symbols with index 5 and 6 in each superframe preamble. In the OFDM symbol with index 5, all usable subcarriers with even-numbered indices shall be modulated with the value $(\sqrt{2}, 0)$. All subcarriers with odd-numbered indices shall be unmodulated; i.e., they shall have zero energy. In the OFDM symbol with index 6, all usable subcarriers shall be modulated with the value (1,0).[46]

9.3.2.4.5 F-OSICH

The F-OSICH shall be carried on the last OFDM symbol in each superframe preamble. The F-OSICH carries a three-state quantity x, taking the values 0, 1, and 2, in each carrier. The value of this quantity for each carrier is determined by the CC MAC Protocol. If x = 0, all usable subcarriers in the relevant carrier shall be modulated with the value (1,0). If x = 1, all usable subcarriers in the relevant carrier shall be modulated with the value (0,1). If x = 2, all usable subcarriers in the relevant carrier shall be modulated with the value (-1,0).

9.3.2.5 FL PHY Frame modulation

The superframe preamble shall be followed by a sequence of FL PHY Frames, each of which has an associated FL PHY Frame index, as described in 9.3.2.1. Each FL PHY Frame shall consist of $N_{FRAME,F}$ OFDM symbols, and its structure shall be as specified in Figure 95 (SymbolRateHopping mode) and Figure 96 (BlockHopping mode). Each carrier in each OFDM symbol in a FL PHY Frame consists of $Q_{SDMA}N_{CARRIER\_SIZE}$ hop-ports, numbered from 0 through $Q_{SDMA}N_{CARRIER\_SIZE}$-1, as described in 9.3.2.2.5. The set of hop-ports in each carrier is mapped to the set of subcarriers in this carrier through the hop-permutation.

9.3.2.5.1 Hopping sequence generation

The hop-permutation is used to map the set of hop-ports to the set of subcarriers. The hopping sequence will be described separately for SymbolRateHopping and BlockHopping modes.

In SymbolRateHopping mode, a new pseudorandom hop-permutation is generated every two OFDM symbols.

In BlockHopping mode, the set of non-guard hop-ports is divided into groups of N_{BLOCK} consecutive hop-ports, each of which is denoted as a block. The hop-permutation will map a block of hop-ports to a group of subcarriers with consecutive indices. This group of subcarriers will also be referred to as a block. Furthermore, the hop-permutation will remain constant for the duration of the FL PHY Frame. In this design, therefore, a group of hop-ports spanning a FL PHY Frame worth of OFDM symbols in time and N_{BLOCK} hop-ports in hop-port space are mapped to neighboring subcarriers in the time-frequency grid. This group of $N_{BLOCK}N_{FRAME,F}$ hop-ports shall be referred to as a tile.

[46] All the subcarriers in the OFDM symbol with index 6 are modulated with the energy 1 because this OFDM symbol serves as a power reference for the forward link transmission. The power density to be used for all the other physical layer channels will be specified in reference to this OFDM symbol.

9.3.2.5.1.1 Common permutation generation algorithm

All permutations used for FL hopping shall be generated using a common permutation generation algorithm. The algorithm takes a 20-bit seed and a permutation size M as inputs and outputs a permutation of the set {0, 1, …, M-1}. The algorithm uses a linear feedback shift register to generate pseudorandom numbers, which in turn are used to generate pseudorandom permutations.

The 20-tap linear feedback shift register shall have a generator sequence of $h(D) = 1 + D^{17} + D^{20}$, as shown in Figure 98. The j'th output s(j) of this shift register shall satisfy $s(j) = s(j-17) \oplus s(j-20)$. The initial state of the register shall generate the first output bit. A pseudorandom number x in $\{0,1,\ldots, 2^n-1\}$ for any n<17 can be generated by clocking the register n times, with the initial output bit being the LSB of x and the final (n'th) output bit being the MSB of x.

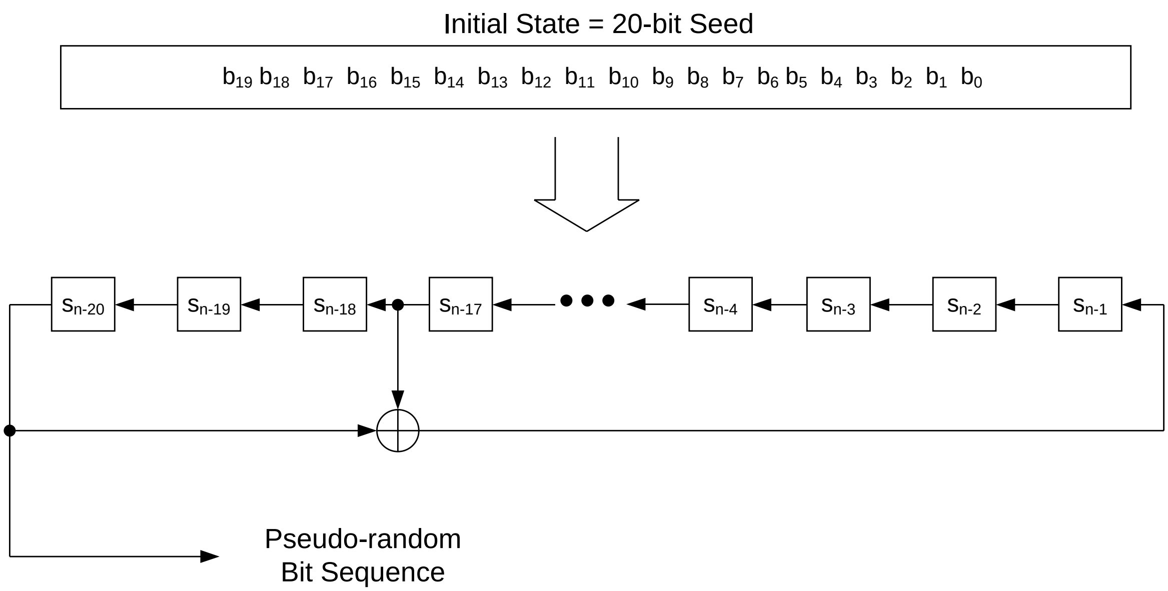

Figure 98 PN Register for generating pseudorandom bits

The common permutation generation algorithm shall generate a permutation of size M as follows:

1. Initialization Steps:
   a. Let n be the smallest integer such that $M \le 2^n$.
   b. Initialize an array A of size M with the numbers 0, 1, 2, …, M-1 (i.e., A[0]=0, A[1]=1 …, A[M-1]= M-1)
   c. Initialize the PN register with the 20-bit seed.
   d. Initialize counter i to M-1.
2. Repeat the following steps until i=0.
   a. Find the smallest p such that $i < 2^p$.

b. Initialize counter j to 0 and an output x to i+1.

c. Repeat the following steps until j=3 or until $x \leq i$.
   i. Clock the PN register n times to obtain an n-bit pseudorandom number. Set x to be the p LSBs of that number.
   ii. Increment j by 1.

d. If x > i, set x = x-i.

e. Swap the i'th and the x'th elements in the array A (i.e., tmp = A[x], A[x] = A[i], A[i] = tmp.)

f. Decrement counter i by 1.

The resulting array A is the output permutation P i.e., P(x) is the location of x in array A. For example, if A reads 345201, then P(0)=4, P(1)=5, P(2)=3, P(3)=0, P(4)=1, and P(5)=2.

9.3.2.5.1.2 FL Hop Permutation Generation

FL Hop Permutation Generation is described in this section for both MultiCarrierOff and MultiCarrierOn modes. In MultiCarrierOff mode, the hop permutation depends on several parameters which are obtained from the Overhead Messages Protocol. In MultiCarrierOn mode, the hop permutation on carrier c, where c is in {0, 1, … $N_{CARRIERS}$ -1} depends on several parameters obtained from the Overhead Messages Protocol for carrier c. These parameters may vary from carrier to carrier.[47]

Space Division Multiple Access (SDMA) is supported on the Forward Link. There are a total of $N_{CARRIER\_SIZE}Q_{SDMA}$ hop-ports on carrier c, which are mapped to the $N_{CARRIER\_SIZE}$ subcarriers corresponding to carrier c. Here Q_{SDMA} is equal to FLNumSDMADimensions, which is part of the public data of the Overhead Messages Protocol for carrier c. The set of hop ports shall be divided into Q_{SDMA} groups, each of which has $N_{CARRIER\_SIZE}$ hop-ports and shall be referred to as an SDMA sub-tree[48]. The sub-trees shall be numbered {0, 1, …, Q – 1} where Q = Q_{SDMA}. The hop port with index p[49] shall belong to sub-tree with index q, where $q = \lfloor p / N_{CARRIER\_SIZE} \rfloor$. Note that hop-ports in different SDMA sub-trees can get mapped to the same subcarrier.

The set of $N_{CARRIER\_SIZE}$ hop-ports in each carrier in each SDMA sub-tree is divided into S subbands, where S shall be determined by FLNumSubbands, which is part of the public data of the Overhead Messages Protocol for carrier c. The subbands shall be numbered {0, 1, … S -1} and each subband

---

[47] A parameter that can vary from carrier to carrier should be indexed by the carrier index c. However, for convenience of notation, the index c is dropped and the parameter is assumed to correspond to the carrier of interest. For example, Q_{SDMA} should be interpreted as $Q_{SDMA}(c)$ when generating the hop permutation for hop-ports in carrier c, and should be obtained from the Overhead Messages Protocol for carrier c.

[48] The term "sub-tree" is used since the $Q_{SDMA}N_{CARRIER\_SIZE}$ hop-ports are part of a "channel tree" defined by the FTC MAC protocol.

[49] Here "hop-port p" should be interpreted as "hop-port p on carrier c." The phrase "on carrier c" will be omitted for convenience of notation.

shall have $N_{SUBBAND}$ hop-ports, where $N_{SUBBAND} = N_{CARRIER\_SIZE} / S$. The hop-port with index p shall belong to subband with index s, where $s = \lfloor (p \bmod N_{CARRIER\_SIZE}) / N_{SUBBAND} \rfloor$.

Furthermore, as mentioned previously, the forward link may implement block hopping. For this reason, the set of $N_{SUBBAND}$ hop-ports in each subband is divided into a number of blocks[50], each of which has N_{BLOCK}[51] hop-ports. The blocks shall be numbered {0, 1, …B -1} where $B = N_{SUBBAND} / N_{BLOCK}$. The hop-port with index p shall belong to block with index b, where $b = \lfloor (p \bmod N_{SUBBAND}) / N_{BLOCK} \rfloor$. The index of the hop port p within the block which it belongs to shall be denoted as r, where $r = p \bmod N_{BLOCK}$. Thus, there is a one-to-one correspondence between hop-port p and the tuple (c, q, s, b, r). For the rest of this document, the two notations are used interchangeably and "hop-port (c, q, s, b, r)" shall be used to refer to hop-port p on carrier c, where $p = qN_{CARRIER\_SIZE} + sN_{SUBBAND} + bN_{BLOCK} + r$.

The hop-ports within each subband shall be divided into two groups: non-guard hop-ports and guard hop-ports. The guard hop-ports shall be mapped to either the guard subcarriers or the quasi-guard subcarriers. The individual elements of this mapping are not specified since these hop-ports shall not be modulated.

In BlockHopping mode, a hop-port (c, q, s, b, r) shall be mapped to a guard subcarrier or a quasi-guard subcarrier either if:[52]

$$b > B - 1 - \left\lfloor \frac{N_{GUARD} / N_{BLOCK}}{S} \right\rfloor$$

or if:

$$b = B - 1 - \left\lfloor \frac{N_{GUARD} / N_{BLOCK}}{S} \right\rfloor \text{ and } \left| \frac{S}{2} - \frac{1}{4} - s \right| > \frac{S - [(N_{GUARD} / N_{BLOCK}) \bmod S]}{2}$$

The set of guard hop-ports in SymbolRateHopping mode shall be identical to the set of guard hop-ports in BlockHopping mode. In other words, a hop port p on carrier c shall be mapped to a guard or a quasi-guard subcarrier when its equivalent representation (c, q, s, b, r) using the value of N_{BLOCK} in BlockHopping mode satisfies the above equations.[53]

---

[50] In SymbolRateHopping mode, contiguous hop-ports are not mapped to contiguous subcarriers. However, the term "block" can still be used if individual hop-ports and subcarriers are thought of as blocks of size 1.

[51] Note that the value of N_{BLOCK} used here corresponds to the Forward Link and this value may be different from the value of N_{BLOCK} used in section 9.4.1.3 which describes hop permutation generation for the Reverse Link.

[52] The idea behind these equations is that all subbands have approximately the same number of non-guard hop-ports. When (N_{GUARD} / N_{BLOCK}) is a multiple of S, the first equation ensures that the highest numbered blocks in each subband are mapped to the guard subcarriers In an asymmetric situation when (N_{GUARD} / N_{BLOCK}) is not a multiple of S, the second equation ensures that the subbands most distant from the center of the carrier have one additional guard hop-port block.

[53] Note that the representation of a hop-port p as (c, q, s, b, r) is different for SymbolRateHopping and BlockHopping modes since the value of N_{BLOCK} is different. Strictly speaking, a hop-port p can be represented

The hop-ports that are not guard hop-ports shall be referred to as non-guard hop-ports. Note that hop-ports in a block are either all guard hop-ports or all non-guard hop-ports. A hop-port block consisting of only non-guard hop-ports shall be referred to as a non-guard hop-port block. The number of non-guard hop-port blocks in subband s shall be denoted as $B_{NON\text{-}GUARD}(s)$. Note that $B_{NON\text{-}GUARD}(s) \leq B$ and a hop-port (c, q, s, b, r) is non-guard if $0 \leq b \leq B_{NON\text{-}GUARD}(s)$ -1.

Furthermore, some non-guard hop-ports may be allocated to the ReservedFLBandwidth segment (as described in 9.3.2.5.1.2.1) in any given interlace. The non-guard hop-ports not allocated to the ReservedFLBandwidth segment in a given interlace shall be referred to as usable hop-ports[54] for that interlace.

Let $H^{ijt}(c, q, s, b, r)$ denote the subcarrier allocated to non-guard hop-port (c, q, s, b, r) in the OFDM Symbol with index t in FL PHY Frame j in superframe i. H^{ijt} is referred to as the hop permutation and shall be given by the following equation:

$$H^{ijt}(c,q,s,b,r) = cN_{CARRIER\_SIZE} + \frac{N_{GUARD}}{2} + N_{BLOCK} H^{ijt}_{GLOBAL}\left(c,q,s,H^{ijt}_{SECTOR}(c,q,s,b)\right) + r$$

Here H^{ijt}_{SECTOR} (c, q, s, b) is a permutation of non-guard hop-port blocks b in the SDMA sub-tree q, carrier c and subband s. The generation of this permutation is described in 9.3.2.5.1.2.4.

H^{ijt}_{GLOBAL}(c, q, s, b') is a permutation of all non-guard hop-port blocks in all subbands in carrier c and SDMA sub-tree q. The generation of H^{ijt} is different for different values of FLDiversityHoppingMode, which is part of the public data of the Overhead Messages Protocol for carrier c. The generation of this permutation is described in 9.3.2.5.1.2.2 and 9.3.2.5.1.2.3.

9.3.2.5.1.2.1 ReservedFLBandwidth Segment

In some FL PHY Frames, a certain set of subbands may be reserved for other uses and therefore shall not be modulated. This set of subbands is known as the ReservedFLBandwidth segment. The number of subbands allocated to the ReservedFLBandwidth segment on a given carrier in a given FL PHY Frame depends only on the interlace index of the FL PHY Frame. The interlace index of PHY Frame j in superframe i shall be equal to $(iN_{FL\text{-}PHY\text{-}FRAMES\text{-}IN\text{-}SUPERFRAME} + j) \bmod N_{FL\text{-}INTERLACES}$, where $N_{FL\text{-}INTERLACES}$ is the number of FL interlaces as specified by the lower MAC sublayer and $N_{FL\text{-}PHY\text{-}FRAMES\text{-}IN\text{-}SUPERFRAME}$ is equal to $N_{FDD,FLPHYFrames}$ in FDD mode and $N_{TDD,FLPHYFrames}$ in TDD mode. The set of interlaces in which the ReservedFLBandwidth segment shall be present is specified by FLReservedInterlaces, which is part of the public data of the Overhead Messages Protocol for carrier c. The number of subbands allocated to the ReservedFLBandwidth segment in these interlaces shall be denoted as $N_{RESERVED\text{-}SUBBANDS}$, which shall be equal to NumFLReservedSubbands, which is part of the public data of the Overhead Messages Protocol for carrier c.

---

as (c, q, s, b_{SRH}, r_{SRH}) using the value of N_{BLOCK} in SymbolRateHopping mode and as (c, q, s, b_{BH}, r_{BH}) using the value of N_{BLOCK} in BlockHopping mode. For the purpose of computing guard hop-ports, (c, q, s, b_{BH}, r_{BH}) is used in both SymbolRateHopping and BlockHopping modes. For all other purposes, the notation (c, q, s, b, r) will refer to (c, q, s, b_{SRH}, r_{SRH}) in SymbolRateHopping mode and to (c, q, s, b_{BH}, r_{BH}) in BlockHopping mode.

[54] Note that "usable hop-ports" refer to hop-ports that can be used by the data segment. Some hop-ports which are not usable are actually used by the ReservedFLBandwidth segment. Contrast this with the definition of "usable subcarriers," which are defined as subcarriers that can be used either by the data segment or ReservedFLBandwidth segment.

A subband s in carrier c shall be allocated to the ReservedFLBandwidth segment if it satisfies the following condition

$$s > S - 1 - N_{RESERVED-SUBBANDS}$$

Let $S_{MIN\text{-}RESERVED\text{-}SUBBAND}$ be the subband with the lowest index allocated to the ReservedFLBandwidth segment in carrier c. (When $N_{RESERVED\text{-}SUBBANDS} = 0$, $S_{MIN\text{-}RESERVED\text{-}SUBBAND}$ shall be set to S.) Thus all non-guard hop-ports in subbands $\{S_{MIN\text{-}RESERVED\text{-}SUBBAND}, S_{MIN\text{-}RESERVED\text{-}SUBBAND} +1, \ldots, S -1\}$ shall be allocated to the ReservedFLBandwidth segment. The number of hop-port-blocks allocated to the ReservedFLBandwidth segment in carrier c equals $N_{RESERVED\text{-}HOP\text{-}PORT\text{-}BLOCKS}$, where

$$N_{RESERVED-HOP-PORT-BLOCKS} = \sum_{k=0}^{N_{RESERVED-SUBBANDS}-1} B_{NON-GUARD}(S_{MIN-RESERVED-SUBBAND} + k).$$

The contiguous set of $N_{BLOCK}N_{RESERVED\text{-}HOP\text{-}PORT\text{-}BLOCKS}$ subcarriers indexed $f_{MIN\text{-}RESERVED}$ to ($f_{MIN\text{-}RESERVED} + N_{BLOCK}N_{RESERVED\text{-}HOP\text{-}PORT\text{-}BLOCKS}$ -1) shall be allocated to the ReservedFLBandwidth segment, where

$$f_{MIN-RESERVED} = cN_{CARRIER\_SIZE} + \frac{N_{GUARD}}{2} + N_{BLOCK} \sum_{k=0}^{S_{MIN-RESERVED-SUBBAND}-1} B_{NON-GUARD}(k)$$

These subcarriers shall not be modulated i.e., zero power shall be transmitted on them.

9.3.2.5.1.2.2 Generation of H^{ijt}_{GLOBAL} when FLDiversityHoppingMode is off

When FLDiversityHoppingMode is off[55], the permutation $H^{ijt}_{GLOBAL}(c, q, s, b)$ shall be given by

$$H^{ijt}_{GLOBAL}(c,q,s,b) = \left[\sum_{P(k)<P(s)} B_{NON-GUARD}(P(k)) \right] + b$$

in both SymbolRateHopping and BlockHopping modes, where $B_{NON\text{-}GUARD}(k)$ is the number of non-guard hop-port blocks in subband k. Here P(.) is a permutation of size $S_{MIN\text{-}RESERVED\text{-}SUBBAND}$ generated as follows:

1. Let $[p_{11}\ p_{10}\ p_9\ p_8\ p_7\ p_6\ p_5\ p_4\ p_3\ p_2\ p_1\ p_0]$ denote the 12-bit PilotPN of the sector, with p_{11} being the MSB and p_0 being the LSB. Define an integer p to be equal to $[p_{11}\ p_{10}\ p_9\ p_8\ p_7\ p_6\ p_5\ p_4\ p_3\ p_2\ p_1\ p_0]$ if FLIntraCellCommonHopping is off, and to be $[p_{11}\ p_{10}\ p_9\ p_8\ 0\ 0\ 0\ p_4\ p_3\ p_2\ p_1\ p_0]$ if FLIntraCellCommonHopping is on. Here, FLIntraCellCommonHopping is part of the public data of the Overhead Messages Protocol for carrier c.

2. Set TMP = [(c*4096+ p) *2654435761] mod 2^{32}.

[55] When FLDiversityHoppingMode is off, the subbands are not permuted. Therefore the hop permutation only permutes the different blocks within each subband.

3. Set $SEED_{GLOBAL}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{GLOBAL}$ = [Bit-Reverse(TMP)] mod 2^{20}.

4. P is the permutation of size $S_{MIN-RESERVED-SUBBAND}$ generated using the common permutation generation algorithm described in 9.3.2.5.1.1 with seed $SEED_{GLOBAL}$.

9.3.2.5.1.2.3 Generation of H^{ijt}_{GLOBAL} in BlockHopping mode when FLDiversityHoppingMode is on

When FLDiversityHoppingMode is on in BlockHopping mode, H^{ijt}_{GLOBAL}(c, q, s, b) is a permutation of the non-guard-hop-port blocks in all subbands in carrier c of SDMA sub-tree q. The generation of H^{ijt}_{GLOBAL} will be different for different values of FLSectorHopSeed, which is part of the public data of the Overhead Messages Protocol on carrier c. H^{ijt}_{GLOBAL} shall be generated as follows:

1. When FLSectorHopSeed has a value not equal to 1111 (in binary notation), set TMP = [(FLSectorHopSeed *4*64*4096+c*64*4096+j*4096 + (i mod 4096)) *2654435761] mod 2^{32}.

2. When FLSectorHopSeed is equal to 1111 (in binary notation), set TMP = [(FLSectorHopSeed *4*64*4096+c*64*4096+j*4096 + P_{SECTOR}) *2654435761] mod 2^{32}, where the 12-bit quantity P_{SECTOR} shall be computed as described in 9.3.2.5.1.2.5.

3. Set $SEED_{GLOBAL}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{GLOBAL}$ = [Bit-Reverse(TMP)] mod 2^{20}.

4. Determine $S_{MIN-RESERVED-SUBBAND}$ and $N_{RESERVED-HOP-PORT-BLOCKS}$ for carrier c as described in 9.3.2.5.1.2.1. Determine $B_{MIN-RESERVED-SUBBAND}$, where

$$B_{MIN-RESERVED-SUBBAND} = \sum_{k=0}^{S_{MIN-RESERVED-SUBBAND}-1} B_{NON-GUARD}(k)$$

5. Generate a permutation π of size $B_{MIN-RESERVED-SUBBAND}$ using the common permutation generation algorithm described in 9.3.2.5.1.1 with seed $SEED_{GLOBAL}$.

6. H^{ijt}_{GLOBAL}(c, q, s, b) = P(β)[56], where $\beta = b + \sum_{k=0}^{s-1} B_{NON-GUARD}(k)$ and

   a. If $\beta \geq B_{MIN-RESERVED-SUBBAND}$ then P(β) = β.

   b. If $\beta < B_{MIN-RESERVED-SUBBAND}$, then P(β) = π (β)

---

[56] P(β) first maps the hop port blocks allocated to the ReservedFLBandwidth segment to a contiguous set of subcarriers. The non-ReservedFLBandwidth hop port blocks are then assigned to the non-ReservedFLBandwidth subcarriers using a pseudo-random permutation π(.)

9.3.2.5.1.2.4 Generation of H^{ijt}_{GLOBAL} in SymbolRateHopping mode when FLDiversityHoppingMode is on

When FLDiversityHoppingMode is on in SymbolRateHopping mode, $H^{ijt}_{GLOBAL}(c, q, s, b)$ is a permutation of the non-guard-hop-port blocks in all subbands in carrier c of SDMA sub-tree q. The generation of H^{ijt}_{GLOBAL} will be different for different values of FLSectorHopSeed, which is part of the public data of the Overhead Messages Protocol on carrier c. H^{ijt}_{GLOBAL} shall be generated as follows:

1. When FLSectorHopSeed has a value not equal to 1111 (in binary notation), set TMP = [(FLSectorHopSeed *4*64*4096+c*64*4096+ (i mod 4096)) *2654435761] mod 2^{32}.

2. When FLSectorHopSeed is equal to 1111 (in binary notation), set TMP = [(FLSectorHopSeed *4*64*4096+c*64*4096+ P_{SECTOR}) *2654435761] mod 2^{32}, where the 12-bit quantity P_{SECTOR} shall be computed as described in 9.3.2.5.1.2.6.

3. Set $SEED_{GLOBAL}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{GLOBAL}$ = [Bit-Reverse(TMP)] mod 2^{20}.

4. Determine $S_{MIN\text{-}RESERVED\text{-}SUBBAND}$ and $N_{RESERVED\text{-}HOP\text{-}PORT\text{-}BLOCKS}$ for carrier c as described in 9.3.2.5.1.2.1. Determine $B_{MIN\text{-}RESERVED\text{-}SUBBAND}$, where

$$B_{MIN-RESERVED-SUBBAND} = \sum_{k=0}^{S_{MIN-RESERVED-SUBBAND}-1} B_{NON-GUARD}(k)$$

5. Generate a permutation $\pi_{INITIAL}$ of size $B_{MIN\text{-}RESERVED\text{-}SUBBAND}$ using the common permutation generation algorithm described in 9.3.2.5.1.1 with seed $SEED_{GLOBAL}$.

6. Set k = $N_{FRAME, F}$ *(j* $N_{FRAME, F}$ /2 + $\lfloor t/2 \rfloor$). Generate a permutation π[57], where π(x) = $\pi_{INITIAL}(k + \pi_{INITIAL}(k+x))$ for all x. Here both additions are performed modulo $B_{MIN\text{-}RESERVED\text{-}SUBBAND}$.

7. $H^{ijt}_{GLOBAL}(c, q, s, b)$ = P(β), where $\beta = b + \sum_{k=0}^{s-1} B_{NON-GUARD}(k)$ and

   a. If $\beta \geq B_{MIN\text{-}RESERVED\text{-}SUBBAND}$ then P(β) = β.

   b. If $\beta < B_{MIN\text{-}RESERVED\text{-}SUBBAND}$, then P(β) = π (β)

[57] In SymbolRateHopping mode, a pseudo-random permutation $\pi_{INITIAL}$ is generated at the beginning of each superframe. This permutation is then used to generate a different pseudo-random permutation π every two OFDM symbols.

9.3.2.5.1.2.5 Generation of H^{ijt}_{SECTOR} in BlockHopping mode

$H^{ijt}_{SECTOR}(c, q, s, .)$ is a permutation of the non-guard hop-port blocks in subband s of carrier c of SDMA sub-tree q. The generation of H^{ijt}_{SECTOR} will be different for different values of FLIntraCellCommonHopping, which is part of the public data of the Overhead Messages Protocol on carrier c.[58]

The PilotPN of the sector of interest is XORed bitwise with the 12 LSBs of the superframe index i to obtain a 12-bit number $[b_{11}\, b_{10}\, b_9\, b_8\, b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0]$ denoted as P_{off}. The 12-bit number $[b_{11}\, b_{10}\, b_9\, b_8\, i_7\, i_6\, i_5\, b_4\, b_3\, b_2\, b_1\, b_0]$, where $i_7\, i_6\, i_5$ are the bits with indices 7,6 and 5 respectively in the superframe index i, is denoted as P_{on}.

The PilotPN is also XORed bitwise with the 12 LSBs of the superframe index (i-1) to obtain a 12-number $[d_{11}\, d_{10}\, d_9\, b_8\, d_7\, d_6\, d_5\, d_4\, d_3\, d_2\, d_1\, d_0]$ denoted as $P_{off,\,MINUS}$. The 12-bit number $[d_{11}\, d_{10}\, d_9\, b_8\, k_7\, k_6\, k_5\, d_4\, d_3\, d_2\, d_1\, d_0]$ where $k_7\, k_6\, k_5$ are the bits with indices 7,6 and 5 respectively in the superframe index (i-1), is denoted as $P_{on,\,MINUS}$. [59]

The permutation shall be generated as follows in FDD mode or in TDD mode when FLSectorHopSeed is not equal to 1110:

1. If FLIntraCellCommonHopping is off, set $P_{SECTOR} = P_{off}$. Otherwise, set $P_{SECTOR} = P_{on}$.
2. Find TMP = $[(c*16*64*4096 + s*64*4096 + j*4096 + P_{SECTOR})* 2654435761] \bmod 2^{32}$. Set $SEED_{SECTOR}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{SECTOR}$ = [Bit-Reverse(TMP)] mod 2^{20}.
3. $H^{ijt}_{SECTOR}(c, q, s, .)$ is the permutation of size $B_{NON\text{-}GUARD}(s)$ generated using the common permutation generation algorithm described in 9.3.2.5.1.1 with seed $SEED_{SECTOR}$.

The permutation shall be generated as follows in TDD mode when FLSectorHopSeed is equal to 1110:[60]

1. Compute $j_{FL,\,BURST} = \lfloor j / N_{FL\_BURST} \rfloor$.

---

[58] When FLIntraCellCommonHopping is off, two sectors with different values of PilotPN have different hopping sequences. When FLIntraCellCommonHopping is on, sectors within the same cell have the same hopping sequences. For proper use of this mode, the operator should ensure that the PilotPNs of two sectors in the same cell differ only in the bits indexed 5,6, and 7.

[59] Note that $P_{off,\,MINUS}$ and $P_{on,\,MINUS}$ computed during superframe i are respectively equal to $P_{off,}$ and P_{on} computed during superframe (i-1).

[60] When RLSectorHopSeed and FLSectorHopSeed are both set to 1110, the FL hop permutations are "slaved" to the RL hop permutations for hop-ports on SDMA sub-tree 0 i.e., if a hop-port p is mapped to a subcarrier f on the RL in a "burst" of RL PHY Frames, then that hop port is mapped to the same subcarrier f on the FL in the subsequent burst of FL PHY Frames as well.

2. If $j_{FL, BURST}$ is not equal to zero

   a. If FLIntraCellCommonHopping is off, set $P_{SECTOR} = P_{off}$. Otherwise, set $P_{SECTOR} = P_{on}$.

   b. Set $j_{RL, BURST} = j_{FL, BURST} - 1$.

3. If $j_{FL, BURST}$ is equal to zero

   a. If FLIntraCellCommonHopping is off, set $P_{SECTOR} = P_{off, MINUS}$. Otherwise, set $P_{SECTOR} = P_{on, MINUS}$.

   b. Set $j_{RL, BURST} = (N_{TDD,FLPHYFrames} / N_{FL\_BURST}) - 1$. Here $N_{TDD, FL PHYFrames}$ and N_{FL_BURST} are as defined in 9.3.2.1.

4. Find TMP = $[(c*16*64*4096 + s*64*4096 + j_{RL, BURST}*4096 + P_{SECTOR})* 2654435761] \bmod 2^{32}$. Set $SEED_{SECTOR}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{SECTOR}$ = [Bit-Reverse(TMP)] mod 2^{20}.

5. $H^{ijt}_{SECTOR}(c, q, s, .)$ is the permutation of size $B_{NON-GUARD}(s)$ generated using the common permutation generation algorithm described in 9.3.2.5.1.1 with seed $SEED_{SECTOR}$.

9.3.2.5.1.2.6 Generation of H^{ijt}_{SECTOR} in SymbolRateHopping mode

$H^{ijt}_{SECTOR}(c, q, s, .)$ is a permutation of the non-guard hop-port blocks in subband s of carrier c of SDMA sub-tree q. The generation of H^{ijt}_{SECTOR} will be different for different values of FLIntraCellCommonHopping, which is part of the public data of the Overhead Messages Protocol on carrier c.

The PilotPN of the sector of interest is XORed bitwise with the 12 LSBs of the superframe index i to obtain a 12-bit number $[b_{11}\, b_{10}\, b_9\, b_8\, b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0]$ denoted as P_{off}. The 12-bit number $[b_{11}\, b_{10}\, b_9\, b_8\, i_7\, i_6\, i_5\, b_4\, b_3\, b_2\, b_1\, b_0]$, where $i_7\, i_6\, i_5$ are the bits with indices 7,6 and 5 respectively in the superframe index i, is denoted as P_{on}. The permutation shall be generated as follows:

1. If FLIntraCellCommonHopping is off, set $P_{SECTOR} = P_{off}$. Otherwise, set $P_{SECTOR} = P_{on}$.

2. Find TMP = $[(c*16*64*4096 + s*64*4096 + P_{SECTOR})* 2654435761] \bmod 2^{32}$. Set $SEED_{SECTOR}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{SECTOR}$ = [Bit-Reverse(TMP)] mod 2^{20}.

3. Generate a permutation $\pi_{INITIAL}$ of size $B_{NON-GUARD}(s)$ using the common permutation generation algorithm described in 9.3.2.5.1.1 with seed $SEED_{SECTOR}$.

4. Set $k = N_{FRAME, F}*(j*N_{FRAME, F}/2 + \lfloor t/2 \rfloor)$. $H^{ijt}_{SECTOR}(c, q, s, b) = \pi_{INITIAL}(k + \pi_{INITIAL}(k+b))$. Here both additions are performed modulo $B_{NON-GUARD}(s)$.

9.3.2.5.2 Pilot channels

The pilot channels consist of the Common Pilot Channel (F-CPICH), the Auxiliary Pilot Channel (F-AuxPICH), and the Dedicated Pilot Channel (F-DPICH). A subcarrier occupied by the F-CPICH or the F-AuxPICH will be referred to as a pilot subcarrier. A hop-port occupied by the F-DPICH will be referred to as a DPICH hop-port.[61]

In SymbolRateHopping Mode, the F-CPICH shall be transmitted such that it can be used as an amplitude and phase reference for all the SISO channels. Moreover, the F-CPICH and F-AuxPICH shall be transmitted such that they can together be used as an amplitude and phase reference for demodulating F-DCH transmissions in MIMO and STTD modes.[62]

In BlockHopping Mode, all channels in a FL PHY Frame except the F-CPICH (namely the F-DPICH, the F-SSCH and the F-DCH) are transmitted using tile-antennas, where a tile-antenna is as defined in 9.3.2.3. These channels shall be transmitted in such a manner that the F-DPICH in each tile (where a tile is as defined in 9.3.2.5.1) can be used as an amplitude and phase reference for the modulation symbols (of the F-SSCH and the F-DCH) within that tile. Note that different precoding matrices may be used to construct the tile-antennas in different tiles, and hence the F-DPICH symbols in one tile should not be used as an amplitude or phase reference for demodulating modulation symbols in another tile. The F-CPICH modulation in BlockHopping mode is described using effective antennas instead of tile-antennas.

Modulation of all physical layers in SymbolRateHopping mode is described using effective antennas. Tile-antennas are not used in this mode.

9.3.2.5.2.1 F-CPICH

9.3.2.5.2.1.1 SymbolRateHopping mode

In SymbolRateHopping mode, the Common Pilot Channel (F-CPICH) shall be present in every OFDM symbol of every FL PHY Frame. The set of subcarriers occupied by the F-CPICH shall be spaced evenly in each carrier. This section describes the F-CPICH modulation procedure for a single carrier, and this procedure shall be repeated across all carriers if $N_{CARRIERS} > 1$.

The F-CPICH configuration is specified in terms of the following quantities, each of which can take a different value for each carrier if $N_{CARRIERS} > 1$.

1. N_p: This is the nominal number of pilot subcarriers in each OFDM symbol in the carrier of interest.[63] The value of N_p is given by the NumPilots parameter, which is part of the public data of the Overhead Messages Protocol.

---

[61] "Pilot subcarriers" and "DPICH hop-ports" are defined separately because F-CPICH and F-AuxPICH modulation is defined in the subcarrier domain, whereas F-DPICH modulation is defined in the hop-port domain.

[62] This requirement ensures that the AN uses similar processing on the pilot and data channels, i.e., if cyclic delay diversity is used for the data channels, it is also used for the pilot channels.

[63] The actual number of pilot subcarriers may be different due to the presence of guard subcarriers.

2. D_p: This is the spacing between neighboring pilot subcarriers, and is given by $N_{CARRIER\_SIZE}/N_p$.

3. $Offset_p$: This is defined using the procedure described in 9.3.2.4.1.

4. N_t: This is the number of transmit antennas, given by the EffectiveNumAntennas parameter, which is part of the public data of the Overhead Messages Protocol.

5. $N_{t,CP}$: This is the number of transmit antennas to be multiplexed on the F-CPICH. This is given by the NumCommonPilotTransmitAntennas parameter, which is part of the public data of the Overhead Messages Protocol.

For each OFDM symbol in an FL PHY Frame, let j be the index of the OFDM symbol within the superframe (starting with index 0). The subcarrier with index i_{sc} in this OFDM symbol, where i_{sc} is such that this subcarrier lies in the carrier of interest, shall be occupied by the F-CPICH if it is a usable subcarrier, if it is not allocated to the ReservedFLBandwidth segment, and if one of the following two conditions is satisfied:

- j is even and i_{sc} mod D_P = $Offset_p$ mod D_p.
- j is odd and i_{sc} mod D_p = ($Offset_p$ + D_p/2) mod D_p.

Here, the ReservedFLBandwidth segment and the subcarriers allocated to it are as defined in 9.3.2.5.1.2.1.

Each subcarrier occupied by the F-CPICH shall be modulated with the complex value $(\sqrt{P},0)$ on only one of the N_t transmit antennas. No power shall be transmitted from the remaining antennas on these subcarriers. Here, P is the ratio of the power spectral density of the F-CPICH to the power spectral density of the second OFDM symbol in the F-ACQCH. This ratio is given by the CommonPilotPower parameter, which is public data of the Overhead Messages Protocol. The antenna index used to transmit an F-CPICH subcarrier in OFDM symbol j in the carrier of interest, denoted by ant_j, is given by the following procedure :

1. If EnableCommonPilotStaggering is set to 0, then ant_j = j mod $N_{t,CP.}$

2. If EnableCommonPilotStaggering is set to 1, then $\mathrm{ant}_j = \lfloor j/2 \rfloor \bmod \mathrm{N}_{t,CP}$.

Here EnableCommonPilotStaggering is part of the public data of the Overhead Messages Protocol.

9.3.2.5.2.1.2 BlockHopping mode

In BlockHopping mode, the Common Pilot Channel (F-CPICH) shall be present in the OFDM symbols with indices 2 and 3 (where the indexing within the FL PHY Frame starts from 0) of some of the FL PHY Frames.

In FDD mode, an FL PHY Frame with index j within the superframe with index i shall carry the F-CPICH if the RL PHY Frame with index j+2 within the superframe with index i contains a Control Segment.

In TDD mode, for each RL PHY Frame with index j within the superframe with index i containing a Control Segment, the FL PHY Frame with index j' within the superframe with index i' shall carry the F-CPICH, where i' and j' are computed as follows:

1. Define $k = i*N_{TDD,RLPHYFrames} + j$.
2. Define $k' = \left(\lfloor k / N_{RL\_BURST} \rfloor + 1\right) N_{FL\_BURST} - 2$.
3. Define $i' = \lfloor k' / N_{TDD,FLPHYFrames} \rfloor$ and $j' = k' \bmod N_{TDD,FLPHYFrames}$.

RL PHY Frame indexing and RL PHY Frame indices containing a control segment are as defined in 9.4.1.1 and 9.4.1.2.5.

This section describes the F-CPICH modulation procedure for a single carrier, and this procedure shall be repeated across all carriers if $N_{CARRIERS} > 1$.

The F-CPICH transmission from each transmit antenna consists of a set of evenly spaced subcarriers in each carrier, with a spacing of $D_p = 16$. Let N_t be the number of antennas in the carrier of interest. N_t is given by the EffectiveNumAntennas parameter, which is part of the public data of the Overhead Messages Protocol. (N_t can take a different value for each carrier.) The antenna with index k, $0 \leq k < N_t$, is associated with two numbers a_k and b_k, which are as defined in columns 2 and 3 respectively of Table 100. A subcarrier with index i_{sc}, where i_{sc} is such that this subcarrier belongs to the carrier of interest, is used to transmit the F-CPICH from antenna k if it is a usable subcarrier, if it is not allocated to the ReservedFLBandwidth segment, and if it satisfies the following condition:

- In the OFDM symbol with index 2 in the FL PHY Frame, i_{sc} satisfies $i_{sc} \bmod D_p = a_k$.
- In the OFDM symbol with index 3 in the FL PHY Frame, i_{sc} satisfies $i_{sc} \bmod D_p = b_k$.

Here, the ReservedFLBandwidth segment and the subcarriers allocated to it are as defined in 9.3.2.5.1.2.1. No power is transmitted on the remaining antennas on this subcarrier.

Each subcarrier occupied by the F-CPICH shall be modulated on the appropriate antenna with the complex value $(\sqrt{P}, 0)$, where P is the ratio of the power spectral density of the F-CPICH to the power spectral density of the second OFDM symbol in the F-ACQCH. This ratio is given by the CommonPilotPower parameter, which is part of the public data of the Overhead Messages Protocol.

Table 100 Values of the parameters a_k and b_k[64]

Antenna Index	Frequency Interlace in OFDM Symbol 2 (a_k)	Frequency Interlace in OFDM Symbol 3 (b_k)
0	0	7
1	2	9
2	3	10

[64] The values of a_k and b_k are chosen so that the F-CPICH in BlockHopping mode does not collide with the F-DPICH.

Antenna Index	Frequency Interlace in OFDM Symbol 2 (a_k)	Frequency Interlace in OFDM Symbol 3 (b_k)
3	4	11
4	5	12
5	6	13
6	7	14
7	9	2

9.3.2.5.2.2 F-AuxPICH

The F-AuxPICH shall not be present in BlockHoppingMode. In SymbolRateHopping mode, the Auxiliary Pilot Channel (F-AuxPICH) shall be present in every OFDM symbol of every FL PHY Frame. The set of subcarriers occupied by the F-AuxPICH shall be spaced evenly in each carrier. This section describes the F-AuxPICH modulation procedure for a single carrier, and this procedure shall be repeated across all carriers if $N_{CARRIERS} > 1$.

The F-AuxPICH configuration is specified in terms of the following quantities, each of which can take a different value for each carrier if $N_{CARRIERS} > 1$.

1. N_p: This is the nominal number of pilot subcarriers in each OFDM symbol in the carrier or interest. The value of N_p is given by NumPilots, which is part of the public data of the Overhead Messages Protocol.

2. D_p: This is the spacing between neighboring pilot subcarriers, and is given by $N_{CARRIER\_SIZE}/N_p$.

3. $Offset_p$: This is defined using the procedure described in 9.3.2.4.1.

4. N_t: This is the number of transmit antennas, given by the EffectiveNumAntennas parameter, which is part of the public data of the Overhead Messages Protocol.

5. $N_{t,CP}$: This is the number of transmit antennas to be multiplexed on the F-CPICH. This is given by the NumCommonPilotTransmitAntennas parameter, which is part of the public data of the Overhead Messages Protocol

6. $N_{t,Aux}$: This is given by $N_t – N_{t,CP}$. No subcarriers are occupied by the F-AuxPICH if $N_{t,Aux} = 0$.

For each OFDM symbol in a FL PHY Frame, let j be the index of the OFDM symbol within the superframe (starting with index 0). The subcarrier with index i_{sc} in this OFDM symbol, where i_{sc} is such that this subcarrier lies in the carrier of interest, shall be occupied by the F-AuxPICH if it is a usable subcarrier, if it is not allocated to the ReservedFLBandwidth segment, if $N_{t,Aux} > 0$ for this carrier, and if one of the following two conditions is satisfied:

- j is odd and $i_{sc} \bmod D_P = Offset_p \bmod D_p$.
- j is even and $i_{sc} \bmod D_p = (Offset_p + D_p/2) \bmod D_p$.

Each subcarrier occupied by the F-AuxPICH shall be modulated with the complex value $(\sqrt{P}, 0)$ on only one of the N_t transmit antennas. No power shall be transmitted from the remaining antennas on these subcarriers. Here, P is the ratio of the power spectral density of the F-AuxPICH to the power spectral density of the second OFDM symbol in the F-ACQCH. This ratio is given by the AuxPilotPower parameter, which is part of the public data of the Overhead Messages Protocol. The antenna index used to transmit an F-AuxPICH subcarrier in OFDM symbol j in the carrier of interest, denoted by ant_j, is given by the following procedure:

1. If EnableAuxPilotStaggering is set to 0, then $ant_j = (j \bmod N_{t,Aux}) + N_{t,CP}$.

2. If EnableAuxPilotStaggering is set to 1, then $ant_j = (\lfloor j/2 \rfloor \bmod N_{t,Aux}) + N_{t,CP}$.

Here, EnableAuxPilotStaggering is part of the public data of the Overhead Messages Protocol. The ReservedFLBandwidth segment and the subcarriers allocated to it are as defined in 9.3.2.5.1.2.1.

9.3.2.5.2.3 F-DPICH

The Dedicated Pilot Channel (F-DPICH) shall be present in the BlockHopping mode only. The modulation of this channel is described for a single tile, where a tile is as described in 9.3.2.5.1. The modulation procedure shall then be repeated for each such tile in each FL PHY Frame. If multiple tiles are mapped to the same set of subcarriers, the AN shall superimpose the F-DPICH waveforms corresponding to these tiles. To elaborate, the AN shall compute the complex value to be transmitted on each subcarrier on each effective transmit antenna corresponding to the different tiles. Note that these complex values may be computed using a different precoding matrix for each data packet, where a precoding matrix is as defined in 9.3.2.3. The AN shall then add the complex values assigned to the same subcarrier corresponding to different data packets.

Each tile in each FL PHY Frame is assigned either to the F-SSCH to the F-DCH by the appropriate MAC protocol (SS MAC in the case of SSCH and FTC MAC in the case of the F-DCH), or is not assigned to any channel. The configuration of the F-DPICH is determined by which of these channels this tile is assigned to and the configuration of this channel. If this tile is not assigned to either the F-SSCH or the F-DCH, then no F-DPICH symbols shall be transmitted in this tile.

The F-DPICH configuration in each tile consists of the following parameters:

1. The number of tile-antennas n_t using which it is transmitted: This shall be equal to 1 if this tile is assigned to the F-SSCH. If this tile is assigned to the F-DCH, n_t shall be equal to the number of tile-antennas used for transmitting the F-DCH over this tile.

2. The energy per modulation symbol per tile-antenna: All the F-DPICH modulation symbols in a given tile shall have the same energy from a given tile-antenna (this energy may be different for different tile-antennas). The F-DPICH power density over a tile is not specified if this tile is assigned to the F-SSCH. If the tile is assigned to the F-DCH, the energy of each F-DPICH modulation symbol on a given tile-antenna shall be equal to the power density used for the F-DCH on that tile for that tile-antenna.

3. F-DPICH format: The F-DPICH in a tile can have three different formats, labeled Format 0, Format 1 and Format 2. Format 0 shall always be used for tiles assigned to the F-SSCH. For F-DCH tiles, the F-DPICH format to be used is determined by the FTC MAC protocol, with the following constraints:

   a. Format 0 is defined only for tiles that are transmitted over up-to 3 antennas, i.e., $n_t \leq 3$.

   b. Format 1 is defined only for tiles that are transmitted over up-to 2 antennas, i.e., $n_t \leq 2$.

4. FLDPISectorOffset: This is part of the public data of the Overhead Messages Protocol, and takes on integer values between 0 and 3. The value used shall correspond to the carrier containing the tile of interest.

5. FLDPIUserOffset: This is an integer that depends on the hop-ports contained in the tile of interest. Let p_{min} be the smallest hop-port index contained in the tile of interest. FLDPIUserOffset is then given by $\lfloor p_{\min} / N_{CARRIER\_SIZE} \rfloor$.

In order to aid the description of the F-DPICH formats, the hop-ports in each tile are numbered from 0 to N_{BLOCK}-1 in increasing order, and the OFDM symbols in each tile from 0 to $N_{FRAME,F}$-1 in increasing order.

9.3.2.5.2.3.1 F-DPICH format 0

In this format, the F-DPICH occupies 18 modulation symbols in each tile in each FL PHY Frame. In this format, a hop-port with index i_{hp} of the OFDM symbol with index t (both measured within the tile) is occupied by the F-DPICH if i is in the set {1,8,15} and if t is in the set {0,1,2,5,6,7}. The set of hop-ports occupied by the F-DPICH for this format is illustrated in Figure 99.

The complex value of the F-DPICH modulation symbol at this location on the tile-antenna with index k is given by

$$S_{i_{hp},t,k} = \sqrt{P}\exp\left(\frac{j2\pi}{3}(k + FLDPISectorOffset + FLDPIUserOffset)t\right) \text{ if } t < 4, \text{ and}$$

$$S_{i_{hp},t,k} = \sqrt{P}\exp\left(\frac{j2\pi}{3}(k + FLDPISectorOffset + FLDPIUserOffset)(t-2)\right) \text{ if } t \geq 4.$$

where j denotes the complex number (0,1), and P denotes the energy per modulation symbol on tile-antenna k used by the F-DPICH. Note that the value of this modulation symbol is the same for all values of i_{hp}.

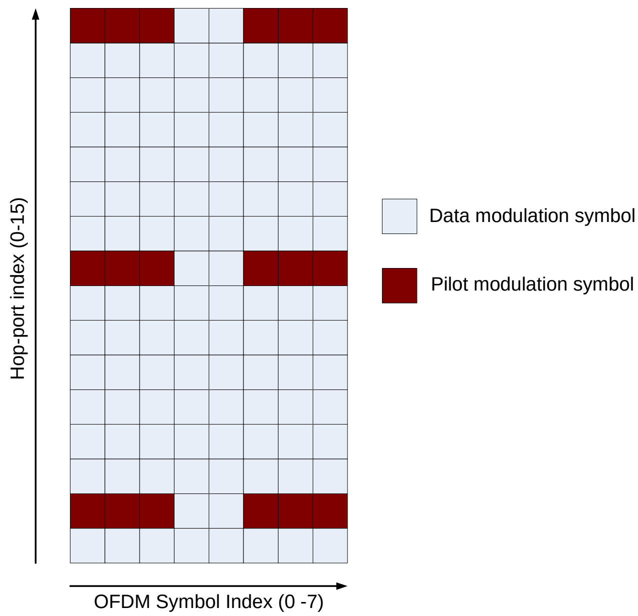

Figure 99 F-DPICH Format 0

9.3.2.5.2.3.2 F-DPICH Format 1

In this format, the F-DPICH occupies 24 modulation symbols in each tile in each FL PHY Frame. In this format, a hop-port with index i_{hp} of the OFDM symbol with index t (both measured within the tile) is occupied by the F-DPICH if i_{hp} is in the set {0,3,6,9,12,15} and if t is in the set {0,1,6,7}. The set of hop-ports occupied by the F-DPICH for this format is illustrated in Figure 100.

The complex value of the F-DPICH modulation symbol at this location on the tile-antenna with index k is given by

$$S_{i_{hp},t,k} = \sqrt{P}\exp\left(j\pi(k + FLDPISectorOffset + FLDPIUserOffset)t\right),$$

where j denotes the complex number (0,1), and P denotes the energy per modulation symbol on tile-antenna k used by the F-DPICH. Note that the value of this modulation symbol is the same for all values of i_{hp}.

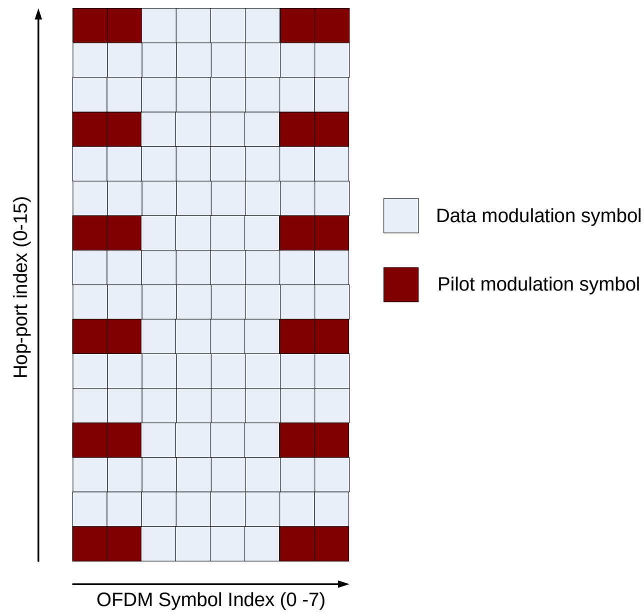

Figure 100 F-DPICH Format 1

9.3.2.5.2.3.3 F-DPICH Format 2

In this format, the F-DPICH occupies 24 modulation symbols in each tile in each FL PHY Frame. In this format, a hop-port with index i_{hp} of the OFDM symbol with index t (both measured within the tile) is occupied by the F-DPICH if i_{hp} is in the set {1,8,15} and for all values of t. The set of hop-ports occupied by the F-DPICH for this format is illustrated in Figure 101.

The complex value of the F-DPICH modulation symbol at this location on the tile-antenna with index k is given by

$$S_{i_{hp},t,k} = \sqrt{P}\exp\left(\frac{j\pi}{2}(k + FLDPISectorOffset + FLDPIUserOffset)t\right),$$

where j denotes the complex number (0,1), and P denotes the energy per modulation symbol on tile-antenna k used by the F-DPICH. Note that the value of this modulation symbol is the same for all values of i_{hp}.

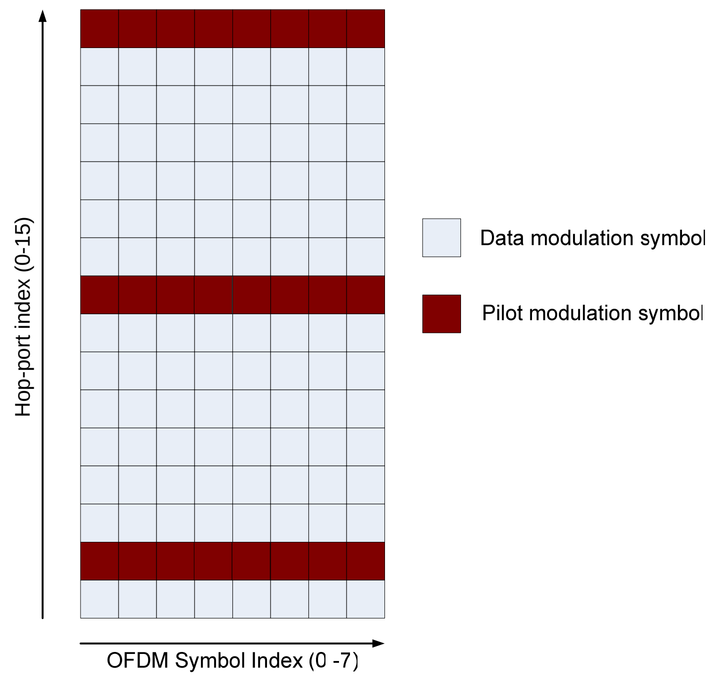

Figure 101 F-DPICH Format 2

9.3.2.5.3 F-SSCH

The Shared Signaling Channel (F-SSCH) shall be present in each FL PHY Frame and shall include four segments populated over hop-ports that are specified by the SS MAC Protocol. We denote by $N_{SSCH\text{-}HP}$ the number of F-SSCH hop-ports. In the BlockHopping mode, $N_{SSCH\text{-}HP}$ shall be an integer multiple of $N_{BLOCK.}$ In MultiCarrierOn mode, there will be an F-SSCH (including all four segments) on every carrier. This section describes the F-SSCH modulation procedure for a single carrier.

9.3.2.5.3.1 Encoded block segment

This segment shall occupy $N_{SSCH\text{-}S1}$ modulation symbols and shall carry $N_{SSCH\text{-}M}$ packets generated by the SS MAC Protocol. The i-th packet of size k_i (before CRC insertion) will be transmitted at power density P_i specified by the SS MAC Protocol. The number $N_{SSCH\text{-}S1}$ is given by

$$N_{SSCH\text{-}S1} = \text{SSCHNumBlocks*SSCHModulationSymbolsPerBlock},$$

where SSCHNumBlocks is the maximum number of packets in the encoded block segment of F-SSCH ($N_{SSCH\text{-}M} \leq$ SSCHNumBlocks) and SSCHModulationSymbolsPerBlock is the number of modulation symbols per packet. SSCHNumBlocks and SSCHModulationSymbolsPerBlock are integers and are part of public data of the Overhead Messages Protocol. The i-th packet is appended with CRC, encoded, channel interleaved, repeated, data-scrambled, and modulated using the procedure described in 9.2 with SSCHModulationSymbolsPerBlock modulation symbols per packet. A MACID of 0 and a packet format index of 0 shall be used to generate the initial state of the data-scrambler. A CRC length of $N_{CRC,SSCH}$ shall be used for this packet.

If i-th packet to be transmitted on the encoded block segment of F-SSCH is an Access Grant block, as specified by the SS MAC Protocol, the sequence of modulation symbols of this block shall be scrambled based on the 10-bit value of the AccessSequenceID which shall be defined by the AC

MAC Protocol. Let n be the number of modulation symbols corresponding to the i-th packet which is an Access Grant block with the AccessSequenceID given by a 10-bit integer m. n is given by n = SSCHModulationSymbolsPerBlock.

First, a sequence $\left[B_{0,i},\ldots,B_{2n-1,i}\right]$ of 2n bits shall be generated using 20-bit shift register which shall have a generator polynomial h(D) = $D^{20}+D^{17}+D^{12}+D^{10}+1$, i.e., the k-th output $B_{k_i,i}$ of the register shall satisfy $B_{k,i}=B_{k-20,i}\oplus B_{k-17,i}\oplus B_{k-12,i}\oplus B_{k-10,i}$. The initial state $\left[B_{-1,i},B_{-2,i},\ldots,B_{-20,i}\right]$ shall be given by the $\left[0\ 0\ 0\ 0\ f_5 f_4 f_3 f_2 f_1 f_0 a_9 a_8 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0\right]$. Here $\left[a_9 a_8 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0\right]$ is the 10-bit AccessSequenceID, with LSB a_0 and MSB a_9 and $\left[f_5 f_4 f_3 f_2 f_1 f_0\right]$ is the 6-bit index of the FL PHY Frame within the current superframe, with LSB f_0 and MSB f_5. Next, the scrambling QPSK sequence $\left[A_{0,i},\ldots,A_{n-1,i}\right]$ will be generated as follows:

$$A_{k,i}=e^{j\pi\left(2*B_{2k,i}+B_{2k+1,i}\right)/2}$$

The k-th modulation symbol of the i-th packet which is an Access Grant block shall be multiplied by the QPSK symbol $A_{k,i}$, $0\le k<n$.

Modulation symbols of all N_{SSCH-M} packets shall be scaled and concatenated in a sequence so that the set of modulation symbols of the i-th packet (in the order of symbols generated by the modulator) shall be scaled by $\sqrt{P_i}$ and followed by the set of modulation symbols of the (i+1)-th packet (in the order of symbols generated by the modulator) scaled by $\sqrt{P_{i+1}}$, $0\le i < N_{SSCH-M}$. To this sequence, a sequence of symbols with complex value (0,0) shall be appended so that the resulting sequence denoted S_M is of length $N_{SSCH-S1}$.

9.3.2.5.3.2 Acknowledgement segment

This segment shall occupy $N_{SSCH-S2} = 3N_{SSCH-ACK}$ modulation symbols where $N_{SSCH-ACK}$ denotes the length of the sequence of acknowledgement bits provided by the SS MAC Protocol. The SS MAC Protocol shall provide a power density P_i ($0\le i < N_{SSCH-ACK}$) and a valid MACID for every non-zero bit. If the i-th bit of the sequence is non-zero, then the MACID corresponding to i-th bit shall be converted into a scrambling sequence of three symbols ($c_{i,0}\ c_{i,1}\ c_{i,2}$) as follows:

1. Obtain an 9-bit sequence ($b_{i,0}\ b_{i,1,\ldots,}\ b_{i,8}$) as 9 LSBs (starting with the LSB) of a binary representation (from LSB to MSB) of the number (k*2654435761) mod 2^{20} where k has 11-bit with representation [$m_{10}\ m_9\ m_8\ m_7\ m_6\ m_5\ m_4\ m_3\ m_2\ m_1\ m_0$] with LSB m_0 and MSB m_{10} which is the 11-bit MACID. If the length of the MACID is less than 11 bits, then the MACID shall be padded with 0's on the left (i.e., in the MSBs) to generate a 11-bit MACID.

2. Map sets ($b_{i,3j}\ b_{i,3j+1,}b_{i,3j+2}$) onto modulation symbols $c_{i,j}$ according to Table 97 (8-PSK mapping), $0\le j < 3$.

The i-th acknowledgement bit shall be mapped to a sequence ($s_{i,0}$ $s_{i,1}$ $s_{i,2}$) of three modulation symbols as follows:

If the i-th acknowledgement bit is 0 then $s_{i,0} = s_{i,1} = s_{i,2} = (0,0)$;

If the i-th acknowledgement bit is 1 then $s_{i,j} = \sqrt{P_i}\, c_{i,j},\ 0 \le j \le 2$.

The resulting modulation symbols shall be concatenated into a sequence of modulation symbols

$$S_{ACK} = (s_{0,0}, s_{0,1}, s_{0,2}, s_{1,0}, s_{1,1}, s_{1,2}, \ldots, s_{i,0}, s_{i,1}, s_{i,2}, \ldots, s_{N_{SSCH\text{-}ACK}-1,0}, s_{N_{SSCH\text{-}ACK}-1,1}, s_{N_{SSCH\text{-}ACK}-1,2})$$

of size $N_{SSCH\text{-}S2}$.

9.3.2.5.3.3 Power Control segment

This segment shall carry power control bits and CQI erasure indication bits for every access terminal with MACID k satisfying the following equation:

k mod FLPCReportInterval = i mod FLPCReportInterval

where k is less or equal to MACIDRange, $i = s \cdot N_F + f$, with N_F the number of FL PHY Frames in a superframe[65], s the Superframe index and f the index of the current FL PHY Frame in the current Superframe starting from 0, and FLPCReportInterval is an integer and is part of the public data of the Overhead Messages Protocol. MACIDRange is an integer which is part of the public data of the Overhead Messages Protocol. The properly ordered sequence [a_0 a_1 $a_{2\ldots}$ $a_{N_{SSCH-S3}-1}$] of power control bits and the corresponding sequence [b_0 b_1 $b_{2\ldots}$ $b_{N_{SSCH-S3}-1}$] of CQI erasure indication bits are provided by the SS MAC protocol. Here $N_{SSCH\text{-}S3} = \lfloor \text{MACIDRange / FLPCReportInterval} \rfloor + 1$ if (i mod FLPCReportInterval) < (MACIDRange mod FLPCReportInterval) and $N_{SSCH\text{-}S3} = \lfloor \text{MACIDRange / FLPCReportInterval} \rfloor$ otherwise. These sequences are converted into a sequence [c_0 c_1 $c_{2\ldots}$ $c_{N_{SSCH-S3}-1}$] of $N_{SSCH\text{-}S3}$ modulation symbols as follows:

$$c_k = \sqrt{\frac{P_{k,0}}{2}}(-1)^{a_k} + j\sqrt{\frac{P_{k,1}}{2}}(-1)^{b_k}, \qquad 0 \le k < N_{SSCH\text{-}S3}$$

where the power densities $P_{k,0}$ and $P_{k,1}$ assigned by the SS MAC Protocol.

This sequence is permuted to a sequence [s_0 s_1 $s_{2\ldots}$ $s_{N_{SSCH-S3}-1}$] according to the following procedure:

1. Compute a 12-bit quantity B_{SECTOR} which is the bit-wise XOR of the 12-bit PilotPN and the 12 LSBs of the superframe index s.

2. Find TMP = [(7*64*4096 + f*4096 + B_{SECTOR})* 2654435761] mod 2^{32}. Set $SEED_{PCS}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{PCS}$ = [Bit-Reverse(TMP)] mod 2^{20}.

[65] N_F is equal to $N_{FDD,FLPHYFrames}$in FDD mode and is equal to $N_{TDD,FLPHYFrames}$ in TDD mode.

3. Generate a permutation H of size $N_{SSCH\text{-}S3}$ using the common permutation generation algorithm described in 9.3.2.5.1.1 with seed $SEED_{PCS}$.

4. $s_{H(k)} = c_k$ for all k satisfying $0 \le k < N_{SSCH\text{-}S3}$.

The sequence S_{PC} of these modulation symbols can be written as follows:

$$S_{\text{PC}} = \left(s_0, s_1, \ldots, s_k, \ldots, s_{N_{SSCH\text{-}S3}-1}\right)$$

9.3.2.5.3.4 Fast OSI Segment

The Fast OSI segment is present in each FL PHY Frame if FastOSIEnabled is set to 1. The Fast OSI segment is not present if FastOSIEnabled is set to 0. Here, FastOSIEnabled is a binary variable and is part of the public data of the Overhead Messages Protocol.

If enabled, the Fast OSI Segment carries a single three-state quantity x in each FL PHYFrame. This quantity takes the values 0, 1 and 2, and is generated by the SS MAC Protocol. The quantity x shall be mapped to complex modulation symbol s as follows:

- If x = 0, then s shall take the value (1,0).
- If x = 1, then s shall take the value (0,1).
- If x = 2, then s shall take the value (-1,0).

The modulation symbol s shall be scaled to the appropriate power P determined by the SS MAC protocol and repeated $N_{FastOSI}$ times to get the sequence $S_{FastOSI}$. $S_{FastOSI}$ is therefore given by the $N_{FastOSI}$ length sequence

$$S_{FastOSI} = (\sqrt{P}s, \sqrt{P}s, ..., \sqrt{P}s)\ .$$

If the Fast OSI Segment is not enabled, then $S_{FastOSI}$ shall be set to the empty sequence.

9.3.2.5.3.5 SSCH Modulation

The sequences $S_{M,}$ S_{ACK}, S_{PC}, and $S_{FastOSI}$ shall be concatenated into a single sequence

$$S_{SSCH} = (S_{M,}, S_{ACK}, S_{PC}, S_{FastOSI})$$

of scaled modulation symbols of size N_{SSCH}. N_{SSCH} is given by $N_{SSCH\text{-}S1} + N_{SSCH\text{-}S2} + N_{SSCH\text{-}S3} + N_{FastOSI}$ if the Fast OSI segment is enabled, and is given by $N_{SSCH\text{-}S1} + N_{SSCH\text{-}S2} + N_{SSCH\text{-}S3}$ otherwise. The sequence S_{SSCH} shall be mapped onto the $N_{SSCH\text{-}HP}$ hop-ports assigned to F-SSCH according to the following procedure:

1. Let $p_0, p_1, \ldots, p_{N_{\text{SSCH-HP}}-1}$ be the set of $N_{SSCH\text{-}HP}$ hop-ports specified by the SS MAC Protocol. In the BlockHopping mode: generate a sequence $q_0, q_1, \ldots, q_{N_{\text{SSCH-HP}}-1}$ so that $q_k = p_{n(k)}$, where $n(k) = \left(k \bmod M\right) \cdot N_{BLOCK} + \lfloor k / M \rfloor$ and $M = N_{\text{SSCH-HP}} / N_{BLOCK}$. In the SymbolRateHopping mode: $q_k = p_k$.

2. Initialize a port counter i to 0, an OFDM symbol counter n to 0, and an F-SSCH sequence index[66] k to 0.

3. Define $m = (n - (i \bmod M)\lceil N_{FRAEME,F} / M \rceil) \bmod N_{FRAME,F}$.[67] Let n_{sc} be the subcarrier index corresponding to hop-port q_i for the m-th OFDM symbol in the FL PHY Frame. If the subcarrier with index n_{sc} is not a pilot subcarrier and if q_i is not a DPICH hop-port, then

    a. modulate this subcarrier with the complex value of the k-th entry of S_M if $k < N_{SSCH}$ and modulate this subcarrier with the complex value (0,0) otherwise. The modulation shall be done on the antenna with index 0 in SymbolRateHopping mode, and on the tile-antenna with index 0 in BlockHopping mode.

    b. increment k.

4. Increment i.

5. If i mod M = 0 then

    a. if $n < N_{FRAME,F}$ then reduce i by M and increment n;

    b. otherwise set n to 0.

6. If $i = N_{SSCH\text{-}HP}$, then stop. Otherwise, repeat steps 3 through 6.

9.3.2.5.4 F-DCH

The Data Channel (F-DCH) shall be present in each FL PHY Frame. The F-DCH consists of one or more packets with different target access terminals, as well as erasure sequences. Each data packet as well as erasure sequence transmission spans one or more FL PHY Frames. The set of FL PHY Frames on which this packet is transmitted is determined by the FTC MAC Protocol. Each data packet and erasure sequence is also assigned a set of hop-ports in each FL PHY Frame of transmission by the FTC MAC Protocol. Note that these hop-ports may span more than one carrier. Each data packet is further associated with a packet format index, which is also assigned by the FTC MAC Protocol.

If some of the hop-ports assigned to two or more data packets or erasure sequences are mapped to the same subcarriers by the hopping sequence, then the AN shall superimpose the waveforms of these data packets. To elaborate, the AN shall compute the complex value to be transmitted on each subcarrier on each effective transmit antenna corresponding to the different data packets. Note that these complex values may be computed using a different precoding matrix for each data packet, where a precoding matrix is as defined in 9.3.2.3. The AN shall then add the complex values assigned to the same subcarrier corresponding to different data packets.

---

[66] F-SSCH sequence index refers to the current position within the sequence $S_{SSCH.}$

[67] This equation provides a different offset for each tile in BlockHopping mode. This ensures that neighboring modulation symbols are mapped to different OFDM symbols.

In the following, power shall not be transmitted on an antenna (in SymbolRateHopping mode) or a tile-antenna (in BlockHopping mode) unless specifically specified.

9.3.2.5.4.1 SISO mode

In this mode, each F-DCH packet is generated by the FTC MAC Protocol, and is split, appended with CRC, encoded, channel interleaved, repeated, data-scrambled and modulated according to the procedure described in 9.2. A CRC length of $N_{CRC,Data}$ is used for this packet. The MACID of the target access terminal, and the packet format index assigned to this packet, shall be used to generate the initial state of the data-scrambler described in 9.2.5. The size of the input packet generated by the FTC MAC Protocol shall be equal to $8\lfloor \rho n_0 N_f / 8 \rfloor - N_{CRC,Data}$, where ρ denotes the spectral efficiency at the first transmission corresponding to the packet format of the packet (defined by the FTC MAC Protocol), n_0 denotes the number of usable hop-ports assigned to this packet in the first FL PHY Frame of transmission, and N_f is equal to $6N_{FRAME,F}$ if this packet is part of an extended duration transmission and is equal to $N_{FRAME,F}$ otherwise. The FTC MAC protocol determines whether or not a packet is part of an extended duration transmission. Here, a usable hop-port is as defined in 9.3.2.5.1.2. This packet shall be modulated on to the hop-ports assigned to this packet according to the following procedure:

1. Initialize a port counter i to 0, a frame counter f to 0, and an OFDM symbol counter j to 0.

2. Arrange the set of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission in increasing order, where the ordering of hop-ports is as defined in 9.3.2.2.5. Let the resulting sequence be denoted by (k_0,p_0), (k_1,p_1), …, (k_{n-1},p_{n-1}), where n is the total number of hop-ports assigned to this packet in the f'th PHY Frame of transmission. The notation for a hop-port is as in 9.3.2.2.5, i.e., k_i denotes the CarrierIndex and p_i denotes the hop-port index in that carrier.

3. Let n_{sc} be the subcarrier index corresponding to the hop-port (k_i,p_i) in the j'th OFDM symbol in the f'th PHY Frame of transmission. Let q be the modulation order to be used for the f'th PHY Frame of transmission, which is a function of the packet format. If n_{sc} is not a pilot subcarrier and if (k_i,p_i) is not a DPICH hop-port, then a modulation symbol s with modulation order q is generated by the modulator according to the procedure described in 9.2.6. This modulation symbol shall be modulated with energy P on hop-port (k_i,p_i), i.e., the value of the corresponding subcarrier shall be $\sqrt{P}\,s$, where P is the power density used for this assignment in the f'th PHY Frame of transmission. The modulation shall be done on the antenna with index 0 in SymbolRateHopping mode, and on the tile-antenna with index 0 in BlockHopping mode. In SymbolRateHopping mode, the power density P is constant over all hop-ports assigned to this packet, while in BlockHopping mode, a different power density P may be used for each tile in the assignment. Determining the value of P is out of the scope of this specification.

4. Increment i. If i = n, increment j and set i = 0.

5. If j = $N_{FRAME,F}$, increment f and set j = 0.

6. If the last PHY Frame of transmission has been completed (as determined by the FTC MAC Protocol), then stop. Else repeat steps 2 through 6.

9.3.2.5.4.2 STTD mode

In the STTD mode, each F-DCH packet is generated by the FTC MAC Protocol, and is split, appended with CRC, encoded, channel interleaved, repeated, data-scrambled and modulated according to the procedure described in 9.2. A CRC length of $N_{CRC,Data}$ is used for this packet. The MACID of the target access terminal, and the packet format index assigned to this packet, shall be used to generate the initial state of the data-scrambler described in 9.2.5. The size of the input packet generated by the FTC MAC Protocol shall be equal to $8\lfloor \rho n_0 N_f / 8 \rfloor - N_{CRC,Data}$, where ρ denotes the spectral efficiency at the first transmission corresponding to the packet format of the packet (defined by the FTC MAC Protocol), n_0 denotes the number of usable hop-ports assigned to this packet in the first FL PHY Frame of transmission, and N_f is equal to $6N_{FRAME,F}$ if this packet is part of an extended duration transmission and is equal to $N_{FRAME,F}$ otherwise. The FTC MAC protocol determines whether or not a packet is part of an extended duration transmission. Here, a usable hop-port is as defined in 9.3.2.5.1.2. This packet shall be modulated on to the hop-ports assigned to this packet according to the following procedure:

1. Initialize a port counter i to 0, a frame counter f to 0, and an OFDM symbol counter j to 0.

2. Arrange the set of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission in increasing order, where the ordering of hop-ports is as defined in 9.3.2.2.5. Let the resulting sequence be denoted by (k_0,p_0), (k_1,p_1), …, (k_{n-1},p_{n-1}), where n is the total number of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission. The notation for a hop-port is as in 9.3.2.2.5, i.e., k_i denotes the CarrierIndex and p_i denotes the hop-port index in that carrier.

3. Let n_{sc} be the subcarrier index corresponding to the hop-port (k_i,p_i) in the j'th OFDM symbol in the f'th PHY Frame of transmission. Let q be the modulation order to be used for the f'th PHY Frame of transmission, which is a function of the packet format. If n_{sc} is not a pilot subcarrier and if (k_i,p_i) is not a DPICH hop-port, then a modulation symbol s with modulation order q is generated by the modulator according to the procedure described in 9.2.6. Label this modulation symbol $s_{i,j}$.

4. Increment i. If i = n, increment j and set i = 0.

5. If j = $N_{FRAME,F}$, set j = 0. Else repeat steps 3 through 5.

6. Collect the set of hop-ports in the FL PHY Frame into a list of pairs as follows:[68]

    a. Start with an empty list.

    b. For each value of i', 0 ≤ i' < n, and for each even value of j', 0 ≤ j' < $N_{FRAME,F}$, add the pair ((i',j'),(i',j'+1)) to the list if the hop-port $(k_{i'},p_{i'})$ is a DPICH hop-port in neither OFDM symbol j' nor OFDM symbol j'+1 OFDM symbol in the FL PHY Frame.

[68] This list is grouping hop-ports that are adjacent to each other into pairs. The different conditions are necessary to account for F-DPICH format 0, which uses up an odd number of adjacent hop-ports.

c. For each value of i', $0 \leq i' < n$ and for each even value of j', $0 \leq j' < N_{FRAME,F}$, add the pair ((i',j'+1),(i',j'+2)) to the list if the hop-port $(k_{i'},p_{i'})$ is a DPICH hop-port in OFDM symbol j' in the FL PHY Frame but is not a DPICH hop-port in OFDM symbol j'+1 in the FL PHY Frame.

7. In the following, let P be the power density to be used for this packet in the f'th frame of transmission. In SymbolRateHopping mode, the power density P is constant over all hop-ports assigned to this packet, while in BlockHopping mode, a different power density P may be used for each tile in the assignment. Determining the value of P is out of the scope of this specification.

   Each pair in the list formed in step 6 is of the form ((i',j'),(i',j'+1)). For each such pair in the list, do the following:[69]

   a. If the hop-port $(k_{i'},p_{i'})$ does not correspond to a pilot subcarrier in either of OFDM symbols j' and j'+1 in the FL PHY Frame, let $s_{i',j'}$ and $s_{i',j'+1}$ denote the modulation symbols allocated in step 3 to this hop-port in OFDM symbols j' and j'+1 respectively. For this case, the following steps shall be carried out:

      i. In SymbolRateHopping mode, $\sqrt{P/2}s_{i',j'}$ shall be transmitted on antenna index 0 on hop-port $(k_{i'},p_{i'})$ of OFDM symbol j'. In BlockHopping mode, $\sqrt{P/2}s_{i',j'}$ shall be transmitted on tile-antenna index 0 on hop-port $(k_{i'},p_{i'})$ of OFDM symbol j'.

      ii. In SymbolRateHopping mode, $\sqrt{P/2}s_{i',j'+1}$ shall be transmitted on antenna index 1 on hop-port $(k_{i'},p_{i'})$ of OFDM symbol j'. In BlockHopping mode, $\sqrt{P/2}s_{i',j'+1}$ shall be transmitted on tile-antenna index 1 on hop-port $(k_{i'},p_{i'})$ of OFDM symbol j'.

      iii. In SymbolRateHopping mode, $-\sqrt{P/2}s_{i',j'+1}*$ shall be transmitted on antenna index 0 of hop-port $(k_{i'},p_{i'})$ of OFDM symbol j'+1. In BlockHopping mode, $-\sqrt{P/2}s_{i',j'+1}*$ shall be transmitted on tile-antenna index 0 of hop-port $(k_{i'},p_{i'})$ of OFDM symbol j'+1.[70]

      iv. In SymbolRateHopping mode, $\sqrt{P/2}s_{i',j'}*$ shall be transmitted on the antenna index 1 of hop-port $(k_{i'},p_{i'})$ of OFDM symbol j'+1. In BlockHopping mode, $\sqrt{P/2}s_{i',j'}*$ shall be transmitted on the tile-antenna index 1 of hop-port $(k_{i'},p_{i'})$ of OFDM symbol j'+1.

---

[69] The different hop-port pairs in the list formed in step 6 now undergo an STTD transformation, which maps two input symbols into two output symbols. However, in some cases, one of the hop-port pairs may map to a pilot (F-CPICH or F-AuxPICH) subcarrier. In this case, that pair of hop-ports does not undergo the STTD transformation.

[70] Here, *s** denotes the complex conjugate of symbol *s*.

b. If the hop-port ($k_{i'}$,$p_{i'}$) corresponds to a pilot subcarrier in OFDM symbols j' but not in OFDM symbol j'+1 in the FL PHY Frame, let $s_{i',j'+1}$ denote the modulation symbols allocated in step 3 to this hop-port in OFDM symbol j'+1. In SymbolRateHopping mode, $\sqrt{P}s_{i',j'+1}$ shall be transmitted on antenna index 0 of hop-port ($k_{i'}$,$p_{i'}$) of OFDM symbol j'+1. In BlockHopping mode, $\sqrt{P}s_{i',j'+1}$ shall be transmitted on tile-antenna index 0 of hop-port ($k_{i'}$,$p_{i'}$) of OFDM symbol j'+1. The F-DCH shall not modulate hop-port ($k_{i'}$,$p_{i'}$) in OFDM symbol j'.

c. If the hop-port ($k_{i'}$,$p_{i'}$) corresponds to a pilot subcarrier in OFDM symbols j'+1 but not in OFDM symbol j' in the FL PHY Frame, let $s_{i',j'}$ denote the modulation symbols allocated in step 3 to this hop-port in OFDM symbols j'. In SymbolRateHopping mode, $\sqrt{P}s_{i',j'}$ shall be transmitted on antenna index 0 of hop-port ($k_{i'}$,$p_{i'}$) of OFDM symbol j'. In BlockHopping mode, $\sqrt{P}s_{i',j'}$ shall be transmitted on tile-antenna index 0 of hop-port ($k_{i'}$,$p_{i'}$) of OFDM symbol j'. The F-DCH shall not modulate hop-port ($k_{i'}$,$p_{i'}$) in OFDM symbol j'+1.

8. Increment f. If the last PHY Frame of transmission has been completed (as determined by the FTC MAC Protocol), then stop. Else repeat steps 2 through 8.

9.3.2.5.4.3 MIMO MCW mode

In the MCW mode, multiple codewords with, in general, different packet formats are transmitted simultaneously on the same set of hop-ports. Each codeword (as well as the associated modulation symbols) is denoted by a data layer. The modulation symbols corresponding to these codewords are transmitted on the different antennas via a scheme called Cyclic Spatial Multiplexing. This scheme is illustrated in Figure 102 for the case of $N_t = 4$ transmit antennas, $N_l = 2$ data layers. The figure shows transmission on 8 consecutive hop-ports. The antennas are indexed from 0 to 3 while the hop-ports are indexed from 0 to 7. The N_l layers are mapped to the space frequency domain in a cyclic fashion such that each layer is transmitted from all antennas, i.e., the first modulation symbol of the first layer is transmitted on the (antenna, hop-port) pair (0,0), the second modulation symbol on (1,1), third on (2,2), and so on. Similarly, the first modulation symbol of the second layer is transmitted on (1,0), the second on (2,1), the third on (3,2), and so on. The total assigned power is initially distributed equally among the different layers. This power shall be split equally among the different antennas in SymbolRateHopping mode. Different tile-antennas may have different powers in BlockHopping mode.

The MIMO MCW mode supports early termination for each layer, i.e., after each FL PHY Frame. The access terminal may acknowledge the first N_{dec} layers, where N_{dec} ranges from 0 to N_l. If the first N_{dec} layers have been acknowledged, then these layers are no longer transmitted and the total assigned power is distributed equally among the remaining layers.

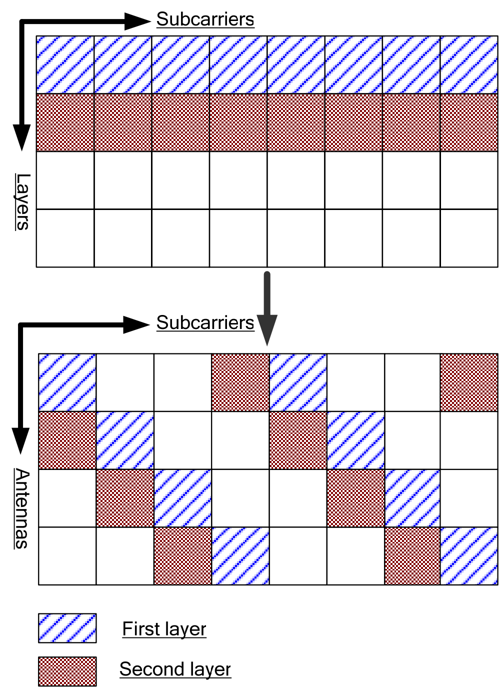

Figure 102 Cyclic spatial multiplexing

9.3.2.5.4.3.1 Transmitter structure

Let N_l denote the number of layers in the MCW transmission, as determined by the FTC MAC protocol. Let n_t denote the number of antennas used for the MCW transmission in SymbolRateHopping mode, and the number of tile-antennas used in BlockHopping mode. In SymbolRateHopping mode, n_t is given by the EffectiveNumAntennas parameter, which is part of the public data of the Overhead Messages Protocol. In BlockHopping mode, n_t shall be equal to N_l.

Each layer consists of a packet that is generated by the FTC MAC Protocol, and is split, appended with CRC, encoded, channel interleaved, repeated, data-scrambled and modulated according to the procedure described in 9.2. A CRC length of $N_{CRC,Data}$ is used for this packet. The MACID of the target access terminal, and the packet format index assigned to this packet, shall be used to generate the initial state of the data-scrambler described in 9.2.5. The size of the input packet generated by the FTC MAC Protocol in each layer shall be equal to $8\lfloor \rho n_0 N_f / 8 \rfloor - N_{CRC,Data}$, where ρ denotes the spectral efficiency at the first transmission corresponding to the packet format for that layer (defined by the FTC MAC Protocol), n_0 denotes the number of usable hop-ports assigned to this packet in the first FL PHY Frame of transmission and N_f is equal to $6N_{FRAME,F}$ if this packet is part of an extended duration transmission and is equal to $N_{FRAME,F}$ otherwise. The FTC MAC protocol determines whether or not a packet is part of an extended duration transmission. Here, a usable hop-port is as defined in 9.3.2.5.1.2. The packet in the l^{th} layer, l ranging from 0 to N_l-1, shall be modulated on to the hop-ports assigned to the access terminal according to the following procedure:

1. Initialize a port counter i to 0, a frame counter f to 0, and an OFDM symbol counter j to 0. Also initialize a modulation counter m to 0.

2. Arrange the set of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission in increasing order, where the ordering of hop-ports is as defined in

9.3.2.2.5. Let the resulting sequence be denoted by (k_0,p_0), (k_1,p_1), …, (k_{n-1},p_{n-1}), where n is the total number of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission. The notation for a hop-port is as in 9.3.2.2.5, i.e., k_i denotes the CarrierIndex and p_i denotes the hop-port index in that carrier.

3. Let n_{sc} be the subcarrier index corresponding to the hop-port (k_i,p_i) in the j'th OFDM symbol in the f'th PHY Frame of transmission. Let q be the modulation order to be used for the f'th PHY Frame of transmission, which is a function of the packet format. If n_{sc} is not a pilot subcarrier and if (k_i,p_i) is not a DPICH hop-port, then a modulation symbol s with modulation order q is generated by the modulator for this packet according to the procedure described in 9.2.6. In SymbolRateHopping mode, this modulation symbol shall be modulated with energy P on hop-port (k_i,p_i) on the antenna with index $(m+l) \bmod n_t$, i.e., the value of the corresponding subcarrier on this antenna shall be $\sqrt{P}\,s$. Here P is the power density per antenna used for this assignment in the f'th PHY Frame of transmission. The same power density shall be used on all antennas. In BlockHopping mode, this modulation symbol shall be modulated with energy P on hop-port (k_i,p_i) on the tile-antenna with index $(m+l) \bmod n_t$, i.e., the value of the corresponding subcarrier on this tile-antenna shall be $\sqrt{P}\,s$. Here P is the power density used for this assignment on tile-antenna $(m+l) \bmod n_t$ in the f'th PHY Frame of transmission. Different power densities may be used for different tile-antennas. Also, in SymbolRateHopping mode, the power density P is constant over all hop-ports assigned to this packet, while in BlockHopping mode, a different power density P may be used for each tile in the assignment. Determining the value of P is out of the scope of this specification.

4. Increment i. If n_{sc} is not a pilot, increment m.

5. If i = n, increment j and set i = 0.

6. If j = $N_{FRAME,F}$, increment f and set j = 0. Also set m = 0.

7. If the last PHY Frame of transmission for this layer has been completed (as determined by the FTC MAC Protocol), then stop. Else repeat steps 2 through 7.

9.3.2.5.4.4 MIMO SCW mode

In SCW mode, the access network does not transmit multiple coded streams on the same set of subcarriers (i.e., two different packets occupy disjoint sets of subcarriers). The overall spectral efficiency is determined by the packet format and the number of layers N_l, which is defined in this case as the number of simultaneously transmitted modulation symbols (on antennas or tile-antennas) and is determined by the FTC MAC Protocol. The Cyclic Spatial Multiplexing structure described in 9.3.2.5.4.3 and illustrated in Figure 102 is also used for this mode, i.e., the set of antennas used to transmit the modulation symbols changes cyclically from subcarrier to subcarrier.

Let n_t denote the number antennas used for the SCW transmission in SymbolRateHopping mode, and the number of tile-antennas used in BlockHopping mode. In SymbolRateHopping mode, n_t shall be given by the EffectiveNumAntennas parameter, which is part of the public data of the Overhead Messages Protocol. In BlockHopping mode, n_t shall be equal to N_l.

Each F-DCH packet is generated by the FTC MAC Protocol, and is split, appended with CRC, encoded, channel interleaved, repeated, data-scrambled, and modulated according to the procedure described in 9.2. A CRC length of $N_{CRC,Data}$ is used for this packet. The MACID of the target access terminal, and the packet format index assigned to this packet, shall be used to generate the initial state of the data-scrambler described in 9.2.5. The size of the input packet generated by the FTC MAC Protocol shall be equal to $8\lfloor \rho n_0 N_f / 8 \rfloor N_l - N_{CRC,Data}$, where ρ denotes the spectral efficiency at the first transmission corresponding to the packet format of the packet (defined by the FTC MAC Protocol), n_0 denotes the number of usable hop-ports assigned to this packet in the first FL PHY Frame of transmission, and N_f is equal to $6N_{FRAME,F}$ if this packet is part of an extended duration transmission and is equal to $N_{FRAME,F}$ otherwise. The FTC MAC protocol determines whether or not a packet is part of an extended duration transmission. Here, a usable hop-port is as defined in 9.3.2.5.1.2. This packet shall be modulated on to the hop-ports assigned to the packet according to the following procedure:

1. Initialize a port counter i to 0, a frame counter f to 0, and an OFDM symbol counter j to 0. Also initialize a modulation symbol counter m to 0.

2. Arrange the set of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission in increasing order, where the ordering of hop-ports is as defined in 9.3.2.2.5. Let the resulting sequence be denoted by (k_0,p_0), (k_1,p_1), …, (k_{n-1},p_{n-1}), where n is the total number of usable hop-ports assigned to this packet in the f'th PHYFrame of transmission. The notation for a hop-port is as in 9.3.2.2.5, i.e., k_i denotes the CarrierIndex and p_i denotes the hop-port index in that carrier.

3. Let n_{sc} be the subcarrier index corresponding to the hop-port (k_i,p_i) in the j'th OFDM symbol in the f'th PHYFrame of transmission. Let q be the modulation order to be used for the f'th PHY Frame of transmission, which is a function of the packet format. If n_{sc} is not a pilot subcarrier and if (k_i,p_i) is not a DPICH hop-port, then generate N_l modulation symbols $s_0, s_1, ..., s_{N_l-1}$ with modulation order q according to the procedure described in 9.2.6.

4. In SymbolRateHopping mode, the modulation symbol s_l, l ranging from 0 to N_l-1, is transmitted with energy P on hop-port (k_i,p_i) on the antenna with index $(m+l)$ mod n_t, i.e., the value of the corresponding subcarrier shall be $\sqrt{P}\,s$. Here P is the power density per antenna used for this assignment in the f'th PHY Frame of transmission. The same power density shall be used on all antennas. In BlockHopping mode, the modulation symbol s_l, l ranging from 0 to N_l-1, is transmitted with energy P on hop-port (k_i,p_i) on the tile-antenna with index $(m+l)$ mod n_t, , i.e., the value of the corresponding subcarrier shall be $\sqrt{P}\,s$. Here P is the power density used for this assignment on tile-antenna $(m+l)$ mod n_t in the f'th PHY Frame of transmission (generated by the FTC MAC Protocol). Different power densities may be used for different tile-antennas. Also, in SymbolRateHopping mode, the power density P is constant over all hop-ports assigned to this packet, while in BlockHopping mode, a different power density P may be used for each tile in the assignment. Determining the value of P is out of the scope of this specification.

5. Increment i. If n_{sc} is not a pilot, increment m.

6. If i = n, increment j and set i = 0.

7. If $j = N_{FRAME,F}$, increment f. Also set j = m = 0.

8. If the last PHY Frame of transmission has been completed (as determined by the FTC MAC Protocol), then stop. Else repeat steps 2 through 8.

9.3.2.5.4.5 Erasure sequence

An erasure sequence spans one or more consecutive FL PHY Frames of transmission on a set of hop-ports determined by the FTC MAC Protocol. The erasure sequence shall be modulated on to the hop-ports assigned to this sequence according to the following procedure:

1. Construct a one-bit packet, with the bit in the packet being set to zero. This packet is encoded, channel interleaved, repeated, scrambled, and modulated according to the procedure described in 9.2.[71] The MAC ID of the target access terminal, and a packet format index of 0, shall be used to generate the initial seed of the scrambler. QPSK modulation shall be used for all of the modulation symbols in the packet.

2. Initialize a port counter i to 0, a frame counter f to 0, and an OFDM symbol counter j to 0.

3. Arrange the set of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission in increasing order, where the ordering of hop-ports is as defined in 9.3.2.2.5. Here, a usable hop-port is as defined in 9.3.2.5.1.2. Let the resulting sequence be denoted by (k_0,p_0), (k_1,p_1), …, (k_{n-1},p_{n-1}), where n is the total number of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission. The notation for a hop-port is as in 9.3.2.2.5, i.e., k_i denotes the CarrierIndex and p_i denotes the hop-port index in that carrier.

4. Let n_{sc} be the subcarrier index corresponding to the hop-port (k_i,p_i) in the j'th OFDM symbol in the PHY Frame. If n_{sc} is not a pilot subcarrier and if (k_i,p_i) is not a DPICH hop-port, then a QPSK modulation symbol s is generated by the modulator according to the procedure described in 9.2.6. This modulation symbol shall be modulated with energy P on hop-port (k_i,p_i), i.e., the value of the corresponding subcarrier shall be $\sqrt{P}s$, where P is the power density used for this erasure sequence. Determining the value of P is out of the scope of this specification. The modulation shall be done on the antenna with index 0 in SymbolRateHopping mode, and on the tile-antenna with index 0 in BlockHopping mode.

5. Increment i. If i = n, or if i = $N_{MaxErasureHopports,F}$, increment j and set i = 0.

6. If j = $N_{FRAME,F}$, increment f and set j = 0.

7. If the last PHY Frame of transmission has been completed (as determined by the FTC MAC Protocol), then stop. Else repeat steps 3 through 7.

[71] The operations before scrambling and modulation are all trivial operations; i.e., they result in an all-zeros sequence. The erasure sequence is equivalent to scrambling an all-zeros sequence of the required length, followed by QPSK modulation.

9.3.2.6 Sector-specific scrambling

Each OFDM symbol in the superframe preamble as well as in every FL PHY Frame shall be scrambled by a sector-specific scrambling sequence. The scrambling operation shall be performed independently on each carrier. The rest of this section describes the scrambling operation for the carrier k, where k=0,1,…, $N_{CARRIERS}$ -1. The scrambling sequence for the carrier k consists of a complex number for every subcarrier in the carrier k in every OFDM symbol in the superframe. The scrambling operation shall consist of multiplying the unscrambled complex symbol on each subcarrier by the corresponding entry in the scrambling sequence, unless both conditions (a) and (b) are true: (a) The subcarrier corresponds to a F-DPICH hop-port (via the hop-permutation), and (b) FLDPISectorScramble, which is part of the public data of the Overhead Messages Protocol for carrier k, is set to 0. For subcarriers for which these conditions (a) and (b) are true, the scrambling operation shall consist of leaving the subcarrier unchanged; and a cell-specific scrambling sequence, as described in 9.3.2.7, shall be used to scramble the subcarrier.

Each complex number in the sector-specific scrambling sequence is generated from two bits, denoted by s_I and s_Q, using the following mapping:

1. The bit combination (s_I, s_Q) = (0,0) is mapped to the complex number $\left(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$.
2. The bit combination (s_I, s_Q) = (0,1) is mapped to the complex number $\left(\frac{1}{\sqrt{2}},\frac{-1}{\sqrt{2}}\right)$.
3. The bit combination (s_I, s_Q) = (1,0) is mapped to the complex number $\left(\frac{-1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$.
4. The bit combination (s_I, s_Q) = (1,1) is mapped to the complex number $\left(\frac{-1}{\sqrt{2}},\frac{-1}{\sqrt{2}}\right)$.

The sector-specific scrambling sequence for the carrier k shall be generated using two 20-bit registers, called the I-register and the Q-register, as shown in Figure 103 and Figure 104, respectively. The I-register shall have a generator polynomial $h_I(D) = D^{20} + D^{19} + D^{16} + D^{14} + 1$ i.e., the n'th output I(n) of the register shall satisfy $I(n) = I(n-20) \oplus I(n-19) \oplus I(n-16) \oplus I(n-14)$. The Q-register shall have a generator polynomial $h_Q(D) = D^{20} + D^{18} + D^{15} + D^{14} + 1$ i.e., the n'th output Q(n) of the register shall satisfy $Q(n) = Q(n-20) \oplus Q(n-18) \oplus Q(n-15) \oplus Q(n-14)$. Each entry in the sector-specific scrambling sequence shall be generated using s_I and s_Q bits which are respectively the outputs of the I-register and the Q-register after they have been appropriately initialized and clocked as in the following description.

At the start of every superframe, define PilotPNSectorScramb to be equal to PilotPhase in SemiSynchronous mode and equal to PilotPN in Asynchronous mode. (Thus, for a given sector, PilotPNSectorScramb is fixed in Asynchronous mode, but changes every superframe in SemiSynchronous mode.) Let p_{11}, p_{10}, …, p_0 be the 12 bits of (PilotPNSectorScramb+k) mod 4096 for a given superframe, with p_{11} being the MSB and p_0 being the LSB. At the beginning of each superframe, the I and Q registers shall both be initialized to the state $[11111111p_{11}p_{10}p_9p_8p_7p_6p_5p_4p_3p_2p_1p_0]$. The I and Q registers shall then be clocked $5N_{CARRIER\_SIZE}$ times in the superframe to generate the s_I and s_Q bits for all of the subcarriers belonging to the carrier k in the OFDM symbols with indices 0, 1, 2, 3, 4 in the superframe. The i'th entry in the scrambling sequence (generated after i clock periods) is used to scramble the subcarrier with index i mod $N_{CARRIER\_SIZE}$ in the carrier k, in the OFDM symbol with index $\lfloor i / N_{CARRIER\_SIZE} \rfloor$ in the superframe. The outputs of the I and Q registers immediately after their state has been initialized (before they are clocked) shall be used to generate the scrambling sequence entry corresponding to the subcarrier with index 0 in the carrier k in the OFDM symbol with index 0.

At the start of each of the OFDM symbols with indices 5,6,7 in the superframe, both the I and the Q registers shall be initialized. The initialization state shall be the same for both the I and the Q registers. The initialization state shall be the state [$11111101p_1p_0p_1p_0p_1p_0p_1p_0p_1p_0p_1p_0$], [$11111110p_3p_2p_1p_0p_7p_6p_5p_4p_3p_2p_1p_0$], and [$11111100p_{11}p_{10}p_9p_8p_7p_6p_5p_4p_3p_2p_1p_0$], at the start of the OFDM symbols indexed 5,6 and 7 respectively.[72] For each of these OFDM symbols, the entry in the scrambling sequence corresponding to the subcarrier with index i in the carrier k (i varying from 0 to $N_{CARRIER\_SIZE}$ -1) shall be generated by clocking the I and Q registers i times, following their initialization. The s_I and s_Q bits shall be, respectively, the outputs of the I and Q-registers. For each of these OFDM symbols indexed 5,6,7 in the superframe, the outputs of the I and Q registers immediately after their state has been initialized (before they are clocked) at the start of the OFDM symbol shall be used to generate the scrambling sequence entry corresponding to the subcarrier with index 0 in the carrier k.

At the start of OFDM symbol with index 8 in the superframe, the I and Q registers shall both be initialized to the state [$11111000p_{11}p_{10}p_9p_8p_7p_6p_5p_4p_3p_2p_1p_0$]. The I and Q registers shall then be clocked $N_{CARRIER\_SIZE}$ times for each remaining OFDM symbol in the superframe to generate the s_I and s_Q entries for all the subcarriers belonging to the carrier k in all the remaining OFDM symbols. The s_I and s_Q entries generated from the I and Q registers after i clock periods, are used to scramble the subcarrier with index i mod $N_{CARRIER\_SIZE}$ in the carrier k, in the OFDM symbol with index $\lfloor i / N_{CARRIER\_SIZE} \rfloor + 8$ in the superframe. The outputs of the I and Q registers immediately after their state has been initialized (before they are clocked) shall be used to generate the scrambling sequence entry corresponding to the subcarrier with index 0 in the carrier k, in the OFDM symbol with index 8.

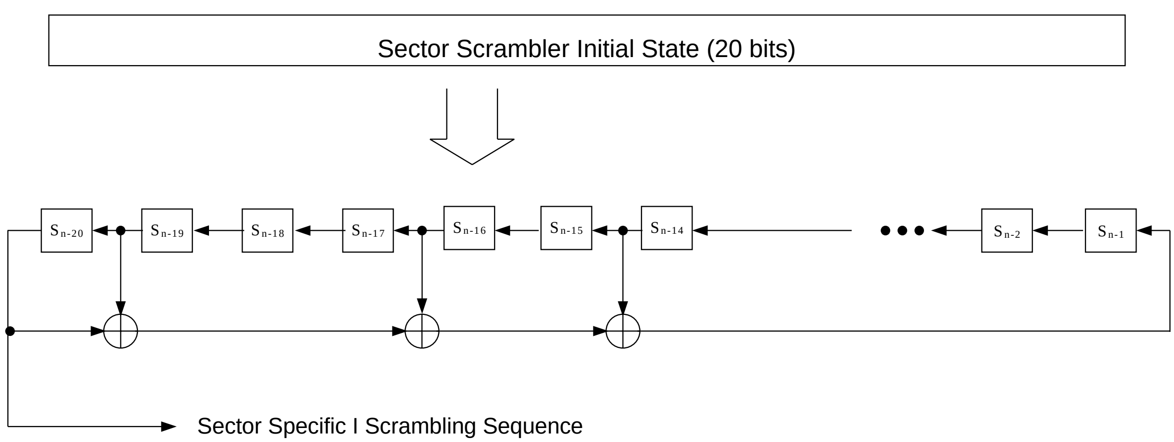

Figure 103 Sector-specific scrambler – I sequence

[72] The OFDM Symbols with indices 5 and 6 are TDM pilots used in acquisition.

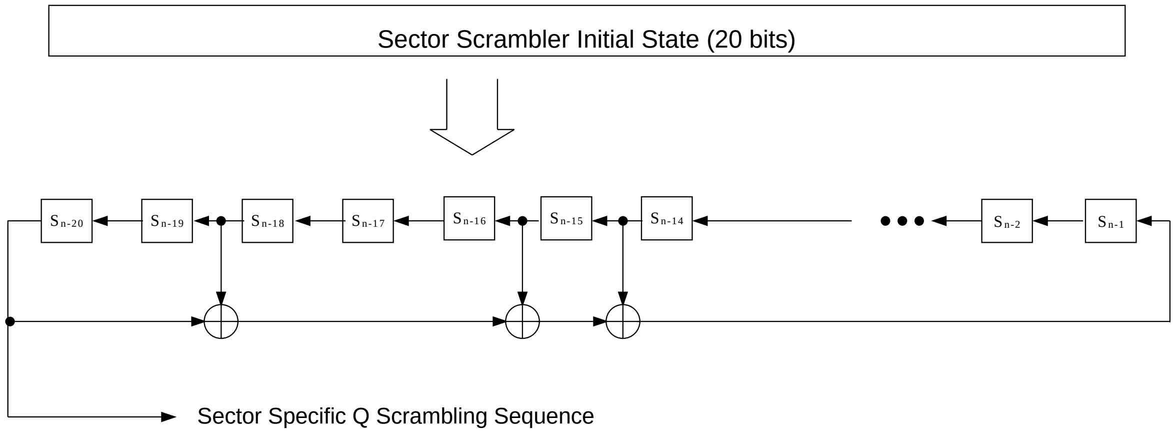

Figure 104 Sector-specific scrambler – Q sequence

9.3.2.7 Cell-specific scrambling for F-DPICH

The operations in this section shall be carried out independently for each carrier, and are described for the carrier with index k, where k=0,1,…, $N_{CARRIERS}$ -1. The operations in this section shall be carried out for the carrier with index k if and only if FLDPISectorScramble, which is part of the public data of the Overhead Messages Protocol for carrier k, is set to 0. A cell-specific scrambling symbol shall be generated for each subcarrier, but only some of the generated scrambling symbols shall be used and the rest shall be discarded. The scrambling symbols that shall be used shall be those generated for subcarriers that correspond to F-DPICH hop-ports (via the hop-permutation), as defined in 9.3.2.5.2.3. These subcarriers are henceforth referred to as F-DPICH subcarriers. The cell-specific scrambling sequence consists of a complex number for every subcarrier. The scrambling operation shall consist of multiplying the unscrambled complex symbol on each F-DPICH subcarrier by the corresponding entry in the scrambling sequence. Each complex number in the cell-specific scrambling sequence is generated from two bits, denoted by s_I and s_Q, using the following mapping:

1. The bit combination (s_I, s_Q) = (0,0) is mapped to the complex number $\left(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$.
2. The bit combination (s_I, s_Q) = (0,1) is mapped to the complex number $\left(\frac{1}{\sqrt{2}},\frac{-1}{\sqrt{2}}\right)$.
3. The bit combination (s_I, s_Q) = (1,0) is mapped to the complex number $\left(\frac{-1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$.
4. The bit combination (s_I, s_Q) = (1,1) is mapped to the complex number $\left(\frac{-1}{\sqrt{2}},\frac{-1}{\sqrt{2}}\right)$.

The cell-specific scrambling sequence for the carrier k shall be generated using two 20-bit registers, called the I-register and the Q-register, as shown in Figure 103 and Figure 104, respectively. The I-register shall have a generator polynomial $h_I(D) = D^{20} + D^{19} + D^{16} + D^{14} + 1$ i.e., the n'th output I(n) of the register shall satisfy $I(n) = I(n-20) \oplus I(n-19) \oplus I(n-16) \oplus I(n-14)$. The Q-register shall have a generator polynomial $h_Q(D) = D^{20} + D^{18} + D^{15} + D^{14} + 1$ i.e., the n'th output Q(n) of the register shall satisfy $Q(n) = Q(n\text{-}20) \oplus Q(n\text{-}18) \oplus Q(n\text{-}15) \oplus Q(n\text{-}14)$. Each entry in the cell-specific scrambling sequence shall be generated using s_I and s_Q bits which are respectively the outputs of the I-register and the Q-register after they have been appropriately initialized and clocked as in the following description.

Let CellPilotPN be the 12 bit number obtained from the PilotPN by setting its 5^{th},6^{th} and 7^{th} bits to zero (where the bits are numbered starting from 0, with the 0^{th} bit denoting the LSB). For the superframe with index s, let SFInd be set equal to s in SemiSynchronous mode and set equal to zero in Asynchronous mode. For the superframe with index s, let $b_{11}, b_{10}, \ldots, b_0$ be the 12 bits of (CellPilotPN+SFInd+k) mod 4096, with b_{11} being the MSB and b_0 being the LSB. At the start of the OFDM symbol with index 0 in the superframe, both the I and the Q registers shall be initialized to the state $[11110000b_{11}b_{10}b_9b_8b_7b_6b_5b_4b_3b_2b_1b_0]$. The outputs of the I and Q registers after they are both clocked i times, shall respectively be the s_I and s_Q bits used to generate a symbol c(i) in the scrambling sequence. This symbol c(i) shall be used to scramble the subcarrier with index i mod $N_{CARRIER\_SIZE}$ in the carrier k in the OFDM Symbol with index $\lfloor i / N_{CARRIER\_SIZE} \rfloor$ in the superframe, provided this subcarrier is an F-DPICH subcarrier

9.3.2.8 Time-domain processing

The sequence of OFDM symbols at the output of the sector scrambler shall be converted to a complex baseband waveform according to the procedure described in Figure 105. This procedure consists of an Inverse Fourier Transform (IFT) operation, a windowing operation, and an overlap-and-add operation.

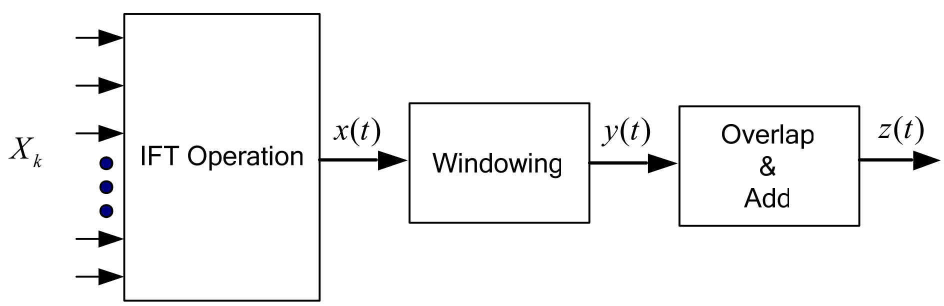

Figure 105 Time-domain processing

9.3.2.8.1 Symbol start time

The start time $T_{START,SF}$ of the superframe with index i is given by the product of i with the superframe duration $T_{SUPERFRAME}$, where $T_{SUPERFRAME}$ is as defined in 9.3.2.2.4.

In FDD, the start time of the k-th OFDM symbol in the superframe, k ranging from 0 to $N_{PREAMBLE} + N_{FDD,FLPHYFrames}N_{FRAME,F} - 1$, is given by $T_{START,SF} + kT_{s,PR}$ if k is less than $N_{PREAMBLE}$, and is given by $T_{START,SF} + N_{PREAMBLE}T_{s,PR} + (k - N_{PREAMBLE})T_s$, otherwise. Here $T_{START,SF}$ is the start time of the superframe, and $N_{FDD,FLPHYFrames}$ is defined by the Lower MAC sublayer.

In TDD mode, the start time of the k-th OFDM symbol in the superframe, k ranging from 0 to $N_{PREAMBLE} + N_{TDD,FLPHYFrames} * N_{FRAME,F} - 1$ is given by $T_{START,SF} + kT_{s,PR}$ if k is less than $N_{PREAMBLE}$, and is given by $T_{START,SF} + N_{PREAMBLE}T_{s,PR} + (k - N_{PREAMBLE})T_s + \lfloor(k - N_{PREAMBLE})/ (N_{FL\_BURST}$

$N_{FRAME,F})\rfloor$ * (N_{RL_BURST} $N_{FRAME,R}T_s$ + $T_{G,TDD,F}$ + $T_{G,TDD,R}$), otherwise. Here $T_{START,SF}$ is the start time of the superframe, while $T_{s,PR}$ and T_s are as defined in 9.3.2.2.3, and $N_{TDD,FLPHYFrames}$ is defined by the Lower MAC sublayer.

9.3.2.8.2 IFT operation

Let X_k be the value of the complex modulation symbol on the k'th subcarrier of an OFDM symbol, k ranging from 0 to N_{FFT}-1. The IFT of the OFDM symbol is given by the infinite duration signal:

$$x(t) = \frac{1}{\sqrt{N_{FFT}}} \sum_{k=0}^{N_{FFT}-1} X_k e^{j2\pi(k-N_{FFT}/2)(t-T_{CP,PR}-T_{START})/(N_{FFT}T_{CHIP})}$$

during the superframe preamble and by:

$$x(t) = \frac{1}{\sqrt{N_{FFT}}} \sum_{k=0}^{N_{FFT}-1} X_k e^{j2\pi(k-N_{FFT}/2)(t-T_{CP}-T_{START})/(N_{FFT}T_{CHIP})}$$

during each FL PHY Frame, where T_{START} denotes the start time of the OFDM symbol, $T_{CP,PR}$ and T_{CP} are as defined in 9.3.2.2.3, and j denotes the complex number (0,1).

9.3.2.8.3 Windowing

The signal x(t) at the output of the IFT shall be multiplied by the window function w(t), where w(t) is given by the equation:

$$w(t) = \begin{cases} 0 & , t < T_{START} - T_{WGI} \\ 0.5 - 0.5\cos\left(\frac{\pi(t + T_{WGI} - T_{START})}{T_{WGI}}\right) & , T_{START} - T_{WGI} \le t < T_{START} \\ 1 & , T_{START} \le t < T_{START} + T_{CP,PR} + T_{FFT} \\ 0.5 + 0.5\cos\left(\frac{\pi(t - T_{START} - T_{CP,PR} - T_{FFT})}{T_{WGI}}\right) & , T_{START} + T_{CP,PR} + T_{FFT} \le t < T'_{s,PR} \\ 0 & , t \ge T'_{s,PR} \end{cases}$$

during the superframe preamble, and by the equation:

$$w(t) = \begin{cases} 0 & , t < T_{START} - T_{WGI} \\ 0.5 - 0.5\cos\left(\dfrac{\pi(t + T_{WGI} - T_{START})}{T_{WGI}}\right) & , T_{START} - T_{WGI} \le t < T_{START} \\ 1 & , T_{START} \le t < T_{START} + T_{CP} + T_{FFT} \\ 0.5 + 0.5\cos\left(\dfrac{\pi(t - T_{START} - T_{CP} - T_{FFT})}{T_{WGI}}\right) & , T_{START} + T_{CP} + T_{FFT} \le t < T'_s \\ 0 & , t \ge T'_s \end{cases}$$

during each FL PHY Frame, where T_{START} denotes the start time of the OFDM symbol. The quantities T_{FFT}, T_s, T_s', $T_{s,PR}$ and $T_{s,PR}$' are as defined in 9.3.2.2.3.

The windowed signal y(t) is given by y(t) = x(t)w(t).

9.3.2.8.4 Overlap and add operation

The windowed IFTs y(t) corresponding to all of the OFDM symbols shall be added together to create the final complex baseband waveform z(t). In this procedure, neighboring OFDM symbols overlap for duration T_{WGI}, as illustrated in Figure 106.

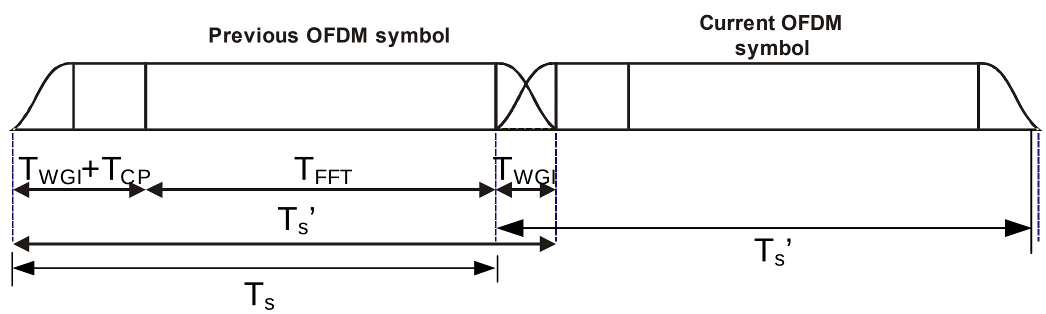

Figure 106 Overlap and add operation

9.3.3 Synchronization and timing

9.3.3.1 Timing reference source

Each sector shall use a time-base reference from which all time-critical transmission components, including superframe boundaries, PHY Frame boundaries, and superframe indices, shall be derived. In SemiSynchronous mode, the time-base reference of any two sectors shall be time-aligned to each other with a maximum error of PilotIncrement times $T_{SUPERFRAME}/4$, where PilotIncrement is a configuration attribute of the Active Set Management Protocol.

There is also a notion of two sectors being synchronous with each other. If two sectors are referred to as being synchronous with each other, their time-base references shall be time-aligned to each other with a maximum error of $10\mu s$.

In Asynchronous mode, there is no requirement for the alignment of the time-base references of two sectors.

9.3.3.2 Sector transmission time

Each sector shall radiate the superframe boundary aligned to its time-base reference. Time measurements are made at the sector antenna connector. If a sector has multiple radiating antenna connectors for the same channel, time measurements are made at the antenna connector having the earliest radiated signal.

The rate of change for timing corrections shall not exceed 102 nanoseconds (ns) per 200 milliseconds (ms).

9.4 Access terminal requirements

9.4.1 Modulation characteristics

This section describes the transmission from an access terminal (AT) to a subset of the set of sectors in its active set, where the Active Set is public data of the Active Set Management Protocol. This subset consists of sectors that are synchronous with each other. Moreover, the subset is a maximal subset, i.e., all sectors that are synchronous with the sectors in this subset are contained in this subset. This subset will be referred to as AS_{SYNCH}. The different synchronous subsets AS_{SYNCH} can be constructed using the last instance of the Active Set Add Message. Transmission from the access terminal to two different synchronous subsets of the active set shall be independent of each other, and shall each follow the procedures specified in this section.

9.4.1.1 Superframe timing

The reverse link transmission shall be divided into units of superframes. The duration of a reverse link superframe shall be the same as the duration of a forward link superframe, and the reverse link superframe shall be time-aligned with the forward link superframe as described in 9.4.2. Each reverse-link superframe is identified by a superframe index that is the same as the index of the time-aligned forward link superframe.

The structure of a reverse link superframe shall be as shown in Figure 107 for FDD and as shown in Figure 108, Figure 109, and Figure 110 for TDD with different values of N_{FL_BURST} and N_{RL_BURST}. Each superframe consists of $N_{FDD,RLPHYFrames}$ RL PHY Frames in FDD and $N_{TDD,RLPHYFrames}$ RL PHY Frames in TDD. Here $N_{FDD,RLPHYFrames}$ and $N_{TDD,RLPHYFrames}$[73] are defined by the Lower MAC sublayer. The structure of the superframe preamble and each RL PHY Frame shall be as shown in Figure 111.

The PHY layer chapter of this specification uses a RL PHY Frame indexing scheme that is convenient for the descriptions herein, but is not necessarily consistent with indexing schemes used in other layers and sublayers in the specification. In this indexing scheme, the RL PHY Frames in a given superframe shall be indexed sequentially from 0 through $N_{FDD,RLPHYFrames}$ -1 in FDD mode and from 0 through $N_{TDD,RLPHYFrames}$ -1 in TDD mode. The RL PHY Frame index is sometimes also referred to using its 6-bit binary representation.

[73] Note that $N_{TDD,RLPHYFrames}$ is a function of N_{FL_BURST} and N_{RL_BURST}.

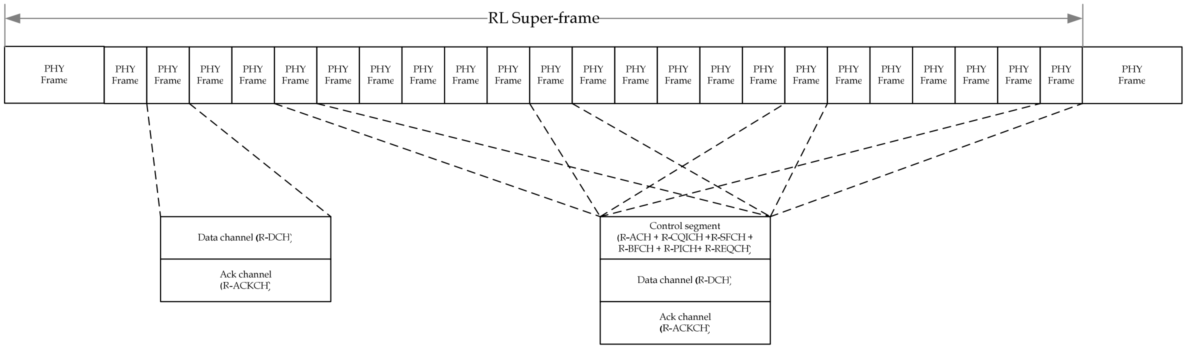

Figure 107 Reverse link superframe structure: FDD

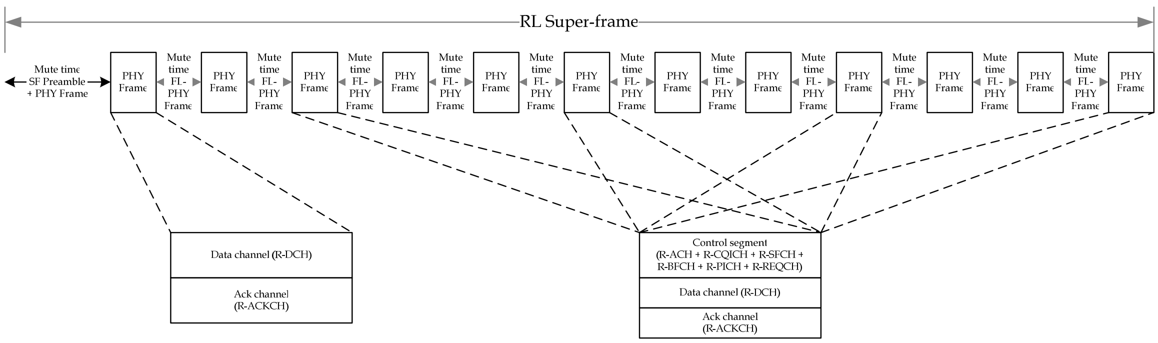

Figure 108 Reverse link superframe structure: TDD (N_{FL_BURST} =1, N_{RL_BURST} =1)

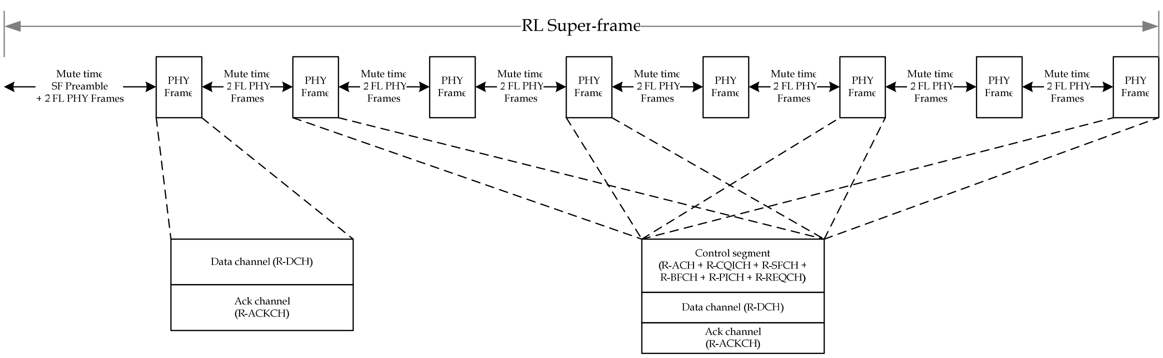

Figure 109 Reverse link superframe structure: TDD (N_{FL_BURST} =2, N_{RL_BURST} =1)

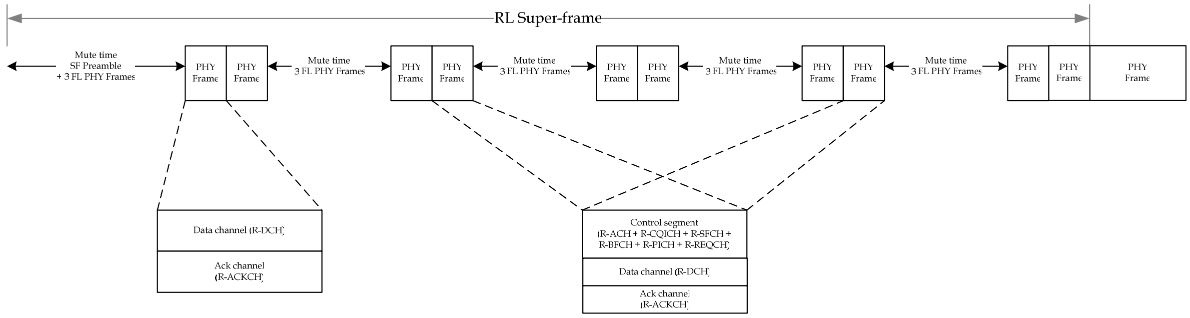

Figure 110 Reverse link superframe structure (N_{FL_BURST} =3, N_{RL_BURST} =2)

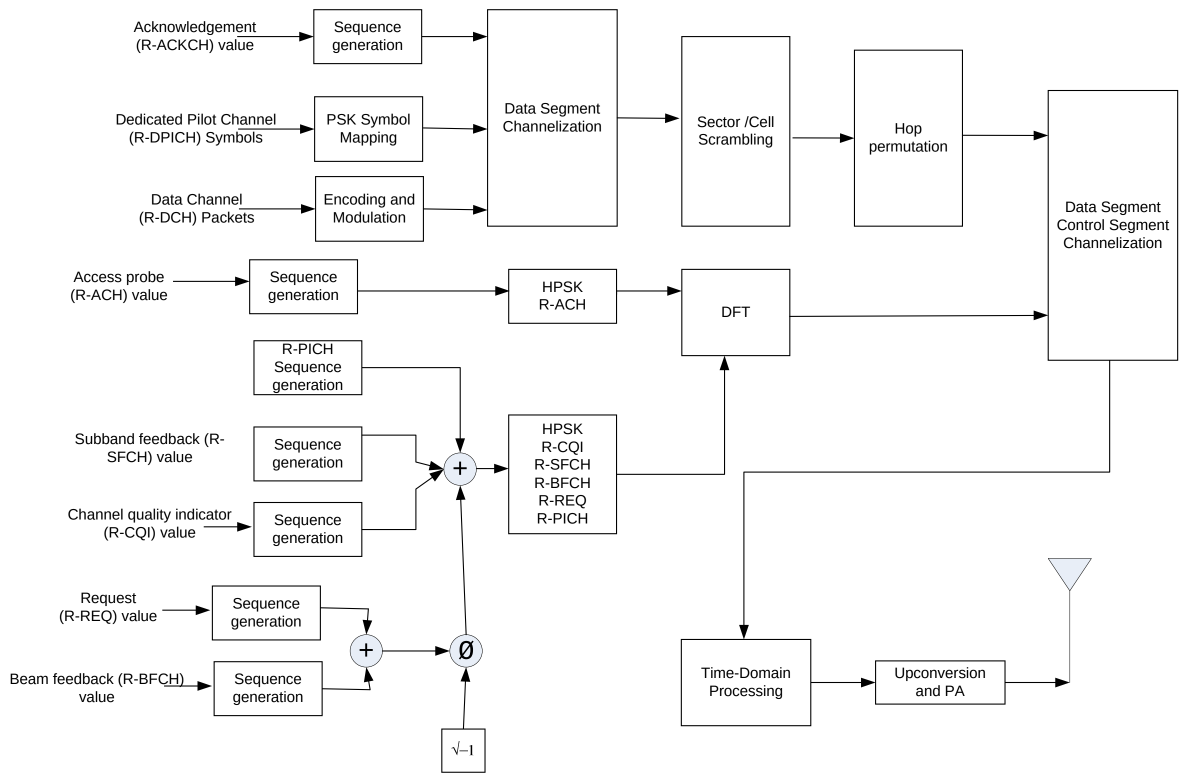

Figure 111 Reverse channel structure

9.4.1.2 OFDM symbol characteristics

The modulation used on the reverse link is Orthogonal Frequency Division Multiplexing (OFDM); i.e., each RL PHY Frame is subdivided into units of OFDM symbols. An OFDM symbol consists of N_{FFT} individually modulated subcarriers that carry complex-valued data. Complex-valued data is represented in the form $d = (d_{re},d_{im})$, where d_{re} and d_{im} represent the real and imaginary components respectively. The subcarriers in each OFDM symbol shall be numbered 0 through N_{FFT}-1.

An additional indexing scheme may be used in MultiCarrierOn mode. The N_{FFT} subcarriers are split into $N_{CARRIERS}$ contiguous groups, each of which is referred to as a carrier. Each carrier consists of $N_{CARRIER\_SIZE}$ subcarriers, where $N_{CARRIER\_SIZE} = N_{FFT} / N_{CARRIERS}$. Each carrier has an associated index, sometimes referred to as CarrierIndex, that ranges from 0 through $N_{CARRIERS}$ -1. The carrier with index c consists of subcarriers indexed $cN_{CARRIER\_SIZE}$ through $(c+1)N_{CARRIER\_SIZE}$ -1. In MultiCarrierOff mode, all N_{FFT} subcarriers belong to a single carrier having CarrierIndex 0. Furthermore, the subcarriers within each carrier may be indexed from 0 to $N_{CARRIER\_SIZE}$ -1 and the phrases “subcarrier f in carrier c” and “subcarrier with index f within carrier with index c” shall be equivalent to “subcarrier $cN_{CARRIER\_SIZE} + f$.” These two subcarrier indexing schemes are used interchangeably in the Physical Layer chapter of this specification.

9.4.1.2.1 Guard subcarriers

Some of the available subcarriers in an OFDM symbol are designated as guard subcarriers and shall not be modulated; i.e., no energy shall be transmitted on these subcarriers. The number of guard subcarriers in each OFDM symbol shall be N_{GUARD}, and the set of guard subcarriers shall be the subcarriers numbered 0 through $N_{GUARD}/2 – 1$ and the subcarriers numbered N_{FFT}-$N_{GUARD}/2$ through N_{FFT} - 1.

The number of guard subcarriers N_{GUARD} for the reverse link shall be the same as the number of guard subcarriers on the forward link, and is given by NumGuardSubcarriers, which is part of the public data of the Overhead Messages Protocol for any sector. The AT shall use the value corresponding to any sector in AS_{SYNCH}. (All sectors in AS_{SYNCH} have the same value of N_{GUARD}.)

9.4.1.2.2 Quasi-guard subcarriers

In multi-carrier mode, additional sub-carriers within each OFDM symbol are designated as quasi-guard subcarriers and shall not be modulated, i.e., no energy shall be transmitted on these subcarriers. The set of quasi-guard subcarriers in each RL PHY Frame shall be the subcarriers numbered $N_{CARRIER\_SIZE}$*m- $N_{GUARD}/2$ through $N_{CARRIER\_SIZE}$ *m + $N_{GUARD}/2 – 1$ where m = 1, …,$N_{CARRIERS}$-1.

Any subcarrier that is not a guard or a quasi-guard subcarrier is referred to as a usable subcarrier.

9.4.1.2.3 OFDM symbol duration

The total OFDM symbol duration, denoted by T_s, consists of four parts:

- A data part with duration T_{FFT}, where $T_{FFT} = N_{FFT} T_{CHIP}$.
- A flat guard interval, also known as a cyclic prefix. The duration of this interval shall be given by T_{CP} for all the OFDM symbols in TDD mode, and in all but the first $N_{PREAMBLE}$ OFDM symbols of each superframe in FDD mode. In FDD mode, the duration of the flat guard interval in the first $N_{PREAMBLE}$ OFDM symbols of each superframe shall be given by $T_{CP,PR}$. Here, T_{CP} and $T_{CP,PR}$ are as defined in 9.3.2.2.3, while $N_{PREAMBLE}$ is as defined in 9.3.2.2.4.
- Two windowed guard intervals, of duration T_{WGI} each., on the two sides of the OFDM symbol. The windowed guard interval duration is the same as on the forward link. There is an overlap of T_{WGI} between consecutive OFDM symbols (see Figure 118).

The effective OFDM symbol duration is given by $T_{s,PR} = T_{FFT} + T_{CP,PR} + T_{WGI}$ for the first $N_{PREAMBLE}$ OFDM symbols of each superframe in FDD mode, and by $T_s = T_{FFT} + T_{CP} + T_{WGI}$ in all other cases. This effective OFDM symbol duration will henceforth be referred to as the OFDM symbol duration.

9.4.1.2.4 Superframe duration

In FDD mode, each RL PHY Frame consists of $N_{FRAME,R}$ OFDM symbols, with the exception of the RL PHY Frame with index 0 in a superframe. The RL PHY Frame with index 0 in a superframe consists of $N_{PREAMBLE}$ additional OFDM symbols, each having a duration of $T_{s,PR}$. The superframe duration on the reverse link is therefore equal to

$$T_{SUPERFRAME} = N_{PREAMBLE}T_{s,PR} + N_{FDD,RLPHYFrames}N_{FRAME,R}T_s.$$

This duration is exactly equal to the forward link superframe duration.

In TDD mode, each RL PHY Frame consists of $N_{FRAME,R}$ OFDM symbols. The mute time between adjacent sets of contiguous RL PHY Frames within a superframe in TDD equals the duration of $N_{FL\_BURST}N_{FRAME,F}$ OFDM symbols plus guard time $T_{G,TDD,F}$ and $T_{G,TDD,R}$. The mute time at the beginning of the reverse superframe equals the duration of the superframe preamble $N_{PREAMBLE}T_{s,PR}$ plus $N_{FRAME,F}$ OFDM symbols and guard time $T_{G,TDD,F}$. The total superframe duration is given by

$$T_{SUPERFRAME} = N_{PREAMBLE}T_{s,PR} + N_{TDD,FLPHYFrames} *N_{FRAME,F}T_s + N_{TDD,RLPHYFrames} *N_{FRAME,R}T_s +$$

$$(T_{G,TDD,F} + T_{G,TDD,R}) * (N_{TDD,FLPHYFrames}/N_{FL\_BURST}).$$

Here T_s is as defined in 9.3.2.2.4 while $N_{TDD,FLPHYFrames}$ and $N_{TDD,RLPHYFrames}$ are as defined by the Lower MAC sublayer.

9.4.1.2.5 Control and data segments

Each RL PHY Frame in both FDD and TDD modes shall contain the Data Segment.

As shown in Figure 107, in FDD mode an RL PHY Frame with index j within the superframe shall contain the Control Segment in every carrier if j mod 6 = 5.

In TDD mode, an RL PHY Frame with index j within the superframe with index i shall contain the Control Segment in every carrier if

$$(i*N_{TDD,RLPHYFrames} + j) \bmod (k*N_{RL\_BURST}) = k*N_{RL\_BURST}-1$$

where k is the smallest integer such that $k*(N_{FL\_BURST}+N_{RL\_BURST}) \geq 6$. The RL PHY Frames containing the control segment for TDD mode with different sets of parameters N_{FL_BURST} and N_{RL_BURST} are shown in Figure 108 and Figure 109. In both TDD and FDD, on the m^{th} carrier, the Control Segment occupies $N_{CTRL\text{-}SUBBANDS}$ subbands where $N_{CTRL\text{-}SUBBANDS}$ is equal to NumRLControlSubbands, which is part of the public data of the Overhead Message Protocol on carrier m.

The Control Segment Period is defined as 6 RL PHY Frames in FDD mode and defined as $k*N_{RL\_BURST}$ in TDD mode.

The Data Segment carries the R-DCH, R-DPICH, and R-ACKCH while the Control Segment carries the R-CQICH, the R-SFCH, the R-BFCH, the R-REQCH, the R-PICH, and the R-ACH.

9.4.1.2.6 Hop-port indexing

The subcarriers in each carrier of each OFDM symbol will also use a second indexing scheme known as hop-port indexing. In this scheme, each carrier in each OFDM symbol consists of $Q_{SDMA}N_{CARRIER\_SIZE}$ individually-modulated hop-ports. Here Q_{SDMA} is equal to RLNumSDMADimensions, which is part of the public data of the Overhead Messages Protocol on that carrier. The hop-ports in each carrier are indexed from 0 through $Q_{SDMA}N_{CARRIER\_SIZE}$ -1. The hop-port with index p in the carrier with CarrierIndex k is sometimes represented by the pair (k,p). An order is defined on the set of such pairs by saying that $(k_0,p_0) < (k_1,p_1)$ if either of the following two conditions is satisfied:

1. $k_0 < k_1$, or
2. $k_0 = k_1$ and $p_0 < p_1$.

There is a mapping between the $Q_{SDMA}N_{CARRIER\_SIZE}$ hop-ports and the $N_{CARRIER\_SIZE}$ subcarriers in each carrier, called a hop-permutation, which changes every RL PHY Frame and is different for different sectors. The sequence of hop-permutations, also called the hopping sequence, is described in 9.4.1.3. The hopping permutation used for different physical layer channels may correspond to different sectors, and the sector to be used is specified individually in the description of each physical layer channel.

9.4.1.3 Hopping sequence generation

The hopping sequence will be described as a mapping from the set of hop-ports to the set of subcarriers. The reverse link implements block hopping. In this scheme, the set of non-guard hop-ports is divided into groups of N_{BLOCK} consecutive hop-ports, each of which is denoted as a block. The hop-permutation will map a block of hop-ports to a group of subcarriers with consecutive indices. This group of subcarriers will also be referred to as a block. Furthermore, the hop permutation will remain constant for the duration of the RL PHY Frame. In this design, therefore, a group of hop-ports spanning a RL PHY Frame worth of OFDM symbols in time and N_{BLOCK} hop-ports in hop-port space are mapped to neighboring subcarriers in the time-frequency grid. This group of $N_{BLOCK}N_{FRAME,R}$ hop-ports shall be referred to as a tile for all RL PHY Frames except those with index 0 in FDD mode. RL PHY Frames with index 0 within a superframe span ($N_{FRAME,R}$ + $N_{PREAMBLE}$) OFDM symbols in FDD mode. For these RL PHY Frames, the group of $N_{BLOCK}(N_{FRAME,R} + N_{PREAMBLE})$ hop-ports shall be referred to as a tile.

9.4.1.3.1 Common permutation generation algorithm

Some of the permutations used for RL hopping shall be generated using a common permutation generation algorithm. The algorithm takes a 20-bit seed and a permutation size M as inputs and outputs a permutation of the set {0, 1,…, M-1}. The algorithm uses a linear feedback shift register to generate pseudorandom numbers, which in turn are used to generate pseudorandom permutations.

The 20-tap linear feedback shift register shall have a generator sequence of $h(D) = 1 + D^{17} + D^{20}$, as shown in Figure 112. The j'th output s(j) of this shift register shall satisfy $s(j) = s(j-17) \oplus s(j-20)$. The initial state of the register shall generate the first output bit. A pseudorandom number x in $\{0,1,\ldots, 2^n-1\}$ for any n<17 can be generated by clocking the register n times, with the initial output bit being the LSB of x and the final (n'th) output bit being the MSB of x.

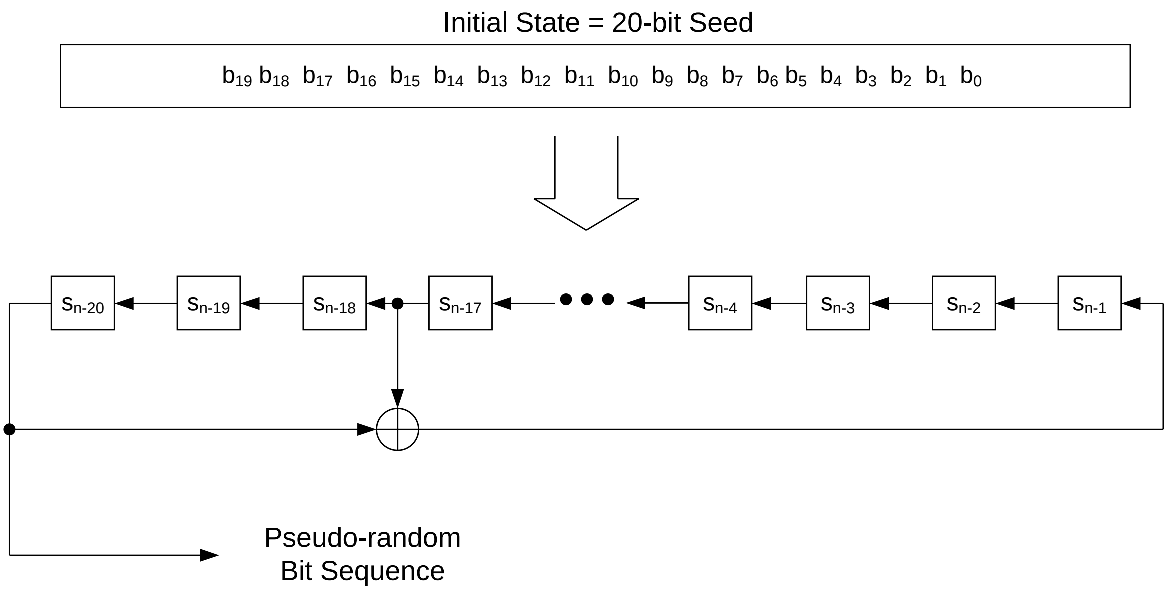

Figure 112 PN Register for generating pseudorandom bits

The common permutation generation algorithm shall generate a permutation of size M as follows:

1. Initialization Steps:
    a. Let n be the smallest integer such that $M \leq 2^n$.
    b. Initialize an array A of size M with the numbers 0, 1, 2, …, M-1 (i.e., A[0]=0, A[1]=1 …, A[M-1]= M-1)
    c. Initialize the PN register with the 20-bit seed.
    d. Initialize counter i to M-1.
2. Repeat the following steps until i=0.
    a. Find the smallest p such that $i < 2^p$.
    b. Initialize counter j to 0 and an output x to i+1.
    c. Repeat the following steps until j=3 or until $x \leq i$.
    d. Clock the PN register n times to obtain an n-bit pseudorandom number. Set x to be the p LSBs of that number.
    e. Increment j by 1.
    f. If x > i, set x = x-i.
    g. Swap the i'th and the x'th elements in the array A (i.e., tmp = A[x], A[x] = A[i], A[i] = tmp.)

h. Decrement counter i by 1.

3. The resulting array A is the output permutation P; i.e., P(x) is the location of x in array A. For example, if A reads 345201, then P(0)=4, P(1)=5, P(2)=3, P(3)=0, P(4)=1, and P(5)=2.

9.4.1.3.2 RL Hop Permutation Generation

RL Hop Permutation Generation is described in this section for both MultiCarrierOff and MultiCarrierOn modes. In MultiCarrierOff mode, the hop permutation depends on several parameters which are obtained from the Overhead Messages Protocol. In MultiCarrierOn mode, the hop permutation on carrier c, where c is in {0, 1, … $N_{CARRIERS}$ -1} depends on several parameters obtained from the Overhead Messages Protocol for carrier c. These parameters may vary from carrier to carrier.[74]

Space Division Multiple Access (SDMA) is supported on the Reverse Link. There are a total of $N_{CARRIER\_SIZE}Q_{SDMA}$ hop-ports on carrier c, which are mapped to the $N_{CARRIER\_SIZE}$ subcarriers corresponding to carrier c. Here Q_{SDMA} is equal to RLNumSDMADimensions, which is part of the public data of the Overhead Messages Protocol for carrier c. The set of hop ports shall be divided into Q_{SDMA} groups, each of which has $N_{CARRIER\_SIZE}$ hop-ports and shall be referred to as an SDMA sub-tree[75]. The sub-trees shall be numbered {0, 1, …, Q – 1} where Q = Q_{SDMA}. The hop port with index p[76] shall belong to sub-tree with index q, where $q = \lfloor p / N_{CARRIER\_SIZE} \rfloor$. Note that hop-ports in different SDMA sub-trees can get mapped to the same subcarrier.

The set of $N_{CARRIER\_SIZE}$ hop-ports in each carrier in each SDMA sub-tree is divided into S subbands, where S shall be equal to RLNumSubbands, which is part of the public data of the Overhead Messages Protocol for carrier c. The subbands shall be numbered {0, 1, … S -1} and each subband shall have $N_{SUBBAND}$ hop-ports, where $N_{SUBBAND} = N_{CARRIER\_SIZE} / S$. The hop-port with index p shall belong to subband with index s, where $s = \lfloor (p \bmod N_{CARRIER\_SIZE}) / N_{SUBBAND} \rfloor$.

Furthermore, as mentioned previously, the reverse link implements block hopping. For this reason, the set of $N_{SUBBAND}$ hop-ports in each subband is divided into a number of blocks, each of which has N_{BLOCK}[77] hop-ports. The blocks shall be numbered {0, 1, …B -1} where B = $N_{SUBBAND} / N_{BLOCK}$. The hop-port with index p shall belong to block with index b, where $b = \lfloor (p \bmod N_{SUBBAND}) / N_{BLOCK} \rfloor$.

The index of the hop port p within the block which it belongs to shall be denoted as r, where $r = p \bmod N_{BLOCK}$. Thus, there is a one-to-one correspondence between hop-port p and the tuple (c,

[74] A parameter that can vary from carrier to carrier should be indexed by the carrier index c. However, for convenience of notation, the index c is dropped and the parameter is assumed to correspond to the carrier of interest. For example, Q_{SDMA} should be interpreted as $Q_{SDMA}(c)$ when generating the hop permutation for hop-ports in carrier c, and should be obtained from the Overhead Messages Protocol for carrier c.

[75] The term "sub-tree" is used since the $Q_{SDMA}N_{CARRIER\_SIZE}$ hop-ports are part of a "channel tree" defined by the RTC MAC protocol.

[76] Here "hop-port p" should be interpreted as "hop-port p on carrier c." The phrase "on carrier c" will be omitted for convenience of notation.

[77] Note that the value of N_{BLOCK} used here corresponds to the Reverse Link and this value may be different from the value of N_{BLOCK} used in section 9.3.2.5.1 which describes hop permutation generation for the Forward Link.

q, s, b, r). For the rest of this document, the two notations are used interchangeably and "hop-port (c, q, s, b, r)" shall be used to refer to hop-port p on carrier c, where

$p = qN_{CARRIER\_SIZE} + sN_{SUBBAND} + bN_{BLOCK} + r$.

The hop-ports within each subband shall be divided into two groups: non-guard hop-ports and guard hop-ports. The guard hop-ports shall be mapped to either the guard subcarriers or the quasi-guard subcarriers. The individual elements of this mapping are not specified since these hop-ports shall not be modulated.

A hop-port (c, q, s, b, r) shall be mapped to a guard subcarrier or a quasi-guard subcarrier either if:[78]

$$b > B - 1 - \left\lfloor \frac{N_{GUARD} / N_{BLOCK}}{S} \right\rfloor$$

or if:

$$b = B - 1 - \left\lfloor \frac{N_{GUARD} / N_{BLOCK}}{S} \right\rfloor \text{ and } \left| \frac{S}{2} - \frac{1}{4} - s \right| > \frac{S - [(N_{GUARD} / N_{BLOCK}) \bmod S]}{2}$$

The hop-ports that are not guard hop-ports shall be referred to as non-guard hop-ports. Note that hop-ports in a block are either all guard hop-ports or all non-guard hop-ports. A hop-port block consisting of only non-guard hop-ports shall be referred to as a non-guard hop-port block. The number of non-guard hop-port blocks in subband s shall be denoted as $B_{NON\text{-}GUARD}(s)$. Note that $B_{NON\text{-}GUARD}(s) \leq B$ and a hop-port (c, q, s, b, r) is non-guard if $0 \leq b \leq B_{NON\text{-}GUARD}(s) - 1$. Also note that $B_{NON\text{-}GUARD}(s)$ does not depend on the carrier index c.

Furthermore, some non-guard hop-ports may be allocated to the control segment (as described in 9.4.1.3.2.1) in any given interlace. The non-guard hop-ports not allocated to the control segment in a given interlace shall be referred to as usable hop-ports[79] for that interlace.

Let $H^{ij}(c, q, s, b, r)$ denote the subcarrier allocated to non-guard hop-port (c, q, s, b, r) in RL PHY Frame j in superframe i. H^{ij} is referred to as the hop permutation and shall be given by the following equation:

$$H^{ij}(c,q,s,b,r) = cN_{CARRIER\_SIZE} + \frac{N_{GUARD}}{2} + N_{BLOCK} H^{ij}_{GLOBAL}\left(c,q,s,H^{ij}_{SECTOR}(c,q,s,b)\right) + r$$

---

[78] The idea behind these equations is that all subbands have approximately the same number of non-guard hop-ports. When (N_{GUARD} / N_{BLOCK}) is a multiple of S, the first equation ensures that the highest numbered blocks in each subband are mapped to the guard subcarriers In an asymmetric situation when (N_{GUARD} / N_{BLOCK}) is not a multiple of S, the second equation ensures that the subbands most distant from the center of the carrier have one additional guard hop-port block.

[79] Note that "usable hop-ports" refer to hop-ports that can be used by the data segment. Some hop-ports which are not usable are actually used by the control segment. Contrast this with the definition of "usable subcarriers," which are defined as subcarriers that can be used either by the data segment or control segment.

Here H^{ij}_{SECTOR} (c, q, s, b) is a permutation of non-guard hop-port blocks b in the SDMA sub-tree q, carrier c and subband s. The generation of this permutation is described in 9.4.1.3.2.4.

H^{ij}_{GLOBAL}(c, q, s, b') is a permutation of all non-guard hop-port blocks in all subbands in carrier c and SDMA sub-tree q. The generation of H^{ij} is different for different values of RLDiversityHoppingMode, which is part of the public data of the Overhead Messages Protocol for carrier c. The generation of this permutation is described in 9.4.1.3.2.2 and 9.4.1.3.2.3.

9.4.1.3.2.1 Control Segment Hopping

The generation of the permutation H^{ij}_{GLOBAL} depends on whether the CDM control segment is present in RL PHY Frame j. The RL PHY Frames which contain a CDM control segment are specified in 9.4.1.2.5.

If the control segment is present in a RL PHY Frame, then an integer number of hop-port subbands shall be allocated to the control segment in each carrier. This number, denoted $N_{CTRL\text{-}SUBBANDS}$ shall be equal to NumRLControlSubbands, which is part of the public data of the Overhead Messages Protocol for carrier c. All non-guard hop-ports in the subbands that satisfy

$$\left|\frac{S}{2}-\frac{1}{4}-s\right| < \frac{N_{CTRL-SUBBANDS}}{2}$$

shall be allocated to the control segment. If the control segment is absent in a RL PHY Frame, then $N_{CTRL\text{-}SUBBANDS}$ shall be equal to zero, and consequently no hop-ports shall be allocated to the control segment.

Let $S_{MIN\text{-}CTRL\text{-}SUBBAND}$ be the subband with the lowest index allocated to the control segment in carrier c. (When $N_{CTRL\text{-}SUBBANDS} = 0$, $S_{MIN\text{-}CTRL\text{-}SUBBAND}$ shall be set to S/2.) Thus all non-guard hop-ports in subbands $\{S_{MIN\text{-}CTRL\text{-}SUBBAND}, S_{MIN\text{-}CTRL\text{-}SUBBAND} +1, \ldots, S_{MIN\text{-}CTRL\text{-}SUBBAND} + N_{CTRL\text{-}SUBBANDS} -1\}$ shall be allocated to the control segment. The number of hop-port-blocks allocated to the control segment in carrier c equals $N_{CTRL\text{-}HOP\text{-}PORT\text{-}BLOCKS}$, where

$$N_{CTRL-HOP-PORT-BLOCKS} = \sum_{k=0}^{N_{CTRL-SUBBANDS}-1} B_{NON-GUARD}(S_{MIN-CTRL-SUBBAND} + k)\ .$$

These hop-ports shall be mapped to a contiguous set of subcarriers as follows.

1. Find TMP = [(RLSectorHopSeed *4*64*4096+c*64*4096+j*4096 + (i mod 4096)) *2654435761] mod 2^{32}. Set $SEED_{CONTROL}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{CONTROL}$ = [Bit-Reverse(TMP)] mod 2^{20}. Here RLSectorHopSeed is part of the public data of the Overhead Messages Protocol for carrier c.

2. Set $S_{SWAP\text{-}LOCATION}$ = min($SEED_{CONTROL}$ mod S, S - $N_{CTRL\text{-}SUBBANDS}$).

3. Generate a permutation $H^{ij}_{SUBBAND}$ as described in 9.4.1.3.2.2. The contiguous set of $N_{CTRL\text{-}SUBCARRIERS}$ subcarriers indexed $f_{MIN\text{-}CTRL}$ to ($f_{MIN\text{-}CTRL}$ + $N_{CTRL\text{-}SUBCARRIERS}$ -1) shall be allocated to the control segment, where $N_{CTRL\text{-}SUBCARRIERS} = N_{BLOCK}N_{CTRL\text{-}HOP\text{-}PORT\text{-}BLOCKS}$ and

$$f_{MIN-CTRL} = cN_{CARRIER\_SIZE} + \frac{N_{GUARD}}{2} + N_{BLOCK} \sum_{k:H^{ij}_{SUBBAND}(k)<S_{SWAP-LOCATION}} B_{NON-GUARD}(k)$$

9.4.1.3.2.2 Generation of H^{ij}_{GLOBAL} when RLDiversityHoppingMode is off

The permutation $H^{ij}_{GLOBAL}(c, q, s, b)$ shall be generated for carrier c as follows when RLDiversityHoppingMode is off:

1. Determine $S_{SWAP\text{-}LOCATION}$ and $S_{MIN\text{-}CTRL\text{-}SUBBAND}$ for carrier c as described in 9.4.1.3.2.1.

2. When the control segment is absent in RL PHY Frame j of superframe i, generate a permutation $H^{ij}_{SUBBAND}$[80] according to the following procedure:

   a. Let [p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0] denote the 12-bit PilotPN of the sector, with p_{11} being the MSB and p_0 being the LSB. Define an integer p to be equal to [p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0] if RLIntraCellCommonHopping is off, and to be [p_{11} p_{10} p_9 p_8 0 0 0 p_4 p_3 p_2 p_1 p_0] if RLIntraCellCommonHopping is on. Here, RLIntraCellCommonHopping is part of the public data of the Overhead Messages Protocol for carrier c.

   b. Set TMP = [(c*4096+ p) *2654435761] mod 2^{32}.

   c. Set $SEED_{GLOBAL}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{GLOBAL}$ = [Bit-Reverse(TMP)] mod 2^{20}.

   d. $H^{ij}_{SUBBAND}$ is the permutation of size S generated using the common permutation generation algorithm described in 9.4.1.3.1 with seed $SEED_{GLOBAL}$.

3. When the control segment is present in RL PHY Frame j of superframe i , generate a permutation $H^{ij}_{SUBBAND}$[81] which shall satisfy $H^{ij}_{SUBBAND}(s) = s$ except for the following subbands:

   a. $H^{ij}_{SUBBAND}(S_{MIN\text{-}CTRL\text{-}SUBBAND} + k) = S_{SWAP\text{-}LOCATION} + k$ for $0 \le k \le N_{CTRL\text{-}SUBBANDS} - 1$

   b. If $| S_{SWAP\text{-}LOCATION} - S_{MIN\text{-}CTRL\text{-}SUBBAND} | \ge N_{CTRL\text{-}SUBBANDS}$, then $H^{ij}_{SUBBAND}(S_{SWAP\text{-}LOCATION} + k) = S_{MIN\text{-}CTRL\text{-}SUBBAND} + k$ for $0 \le k \le N_{CTRL\text{-}SUBBANDS} - 1$.

---

[80] When the control segment is absent $H^{ij}_{SUBBAND}$ is a time-invariant sector-dependent pseudorandom permutation of subbands.

[81] When the control segment is present, $H^{ij}_{SUBBAND}$ maps the control segment to a contiguous pseudo-random set of subbands. The data subbands displaced by the control segment are then mapped to the center of the carrier. All other subbands are left unchanged.

c. If $0 \le S_{MIN\text{-}CTRL\text{-}SUBBAND} - S_{SWAP\text{-}LOCATION} < N_{CTRL\text{-}SUBBANDS}$, then $H^{ij}_{SUBBAND}(S_{SWAP\text{-}LOCATION} + k) = S_{SWAP\text{-}LOCATION} + N_{CTRL\text{-}SUBBANDS} + k$ for $0 \le k < S_{MIN\text{-}CTRL\text{-}SUBBAND} - S_{SWAP\text{-}LOCATION}$

d. If $0 \le S_{SWAP\text{-}LOCATION} - S_{MIN\text{-}CTRL\text{-}SUBBAND} < N_{CTRL\text{-}SUBBANDS}$, then $H^{ij}_{SUBBAND}(S_{SWAP\text{-}LOCATION} + N_{CTRL\text{-}SUBBANDS} - 1 - k) = S_{SWAP\text{-}LOCATION} - 1 - k$ for $0 \le k < S_{SWAP\text{-}LOCATION} - S_{MIN\text{-}CTRL\text{-}SUBBAND}$

4. $$H^{ij}_{GLOBAL}(c,q,s,b) = \left[\sum_{k:H^{ij}_{SUBBAND}(k) < H^{ij}_{SUBBAND}(s)} B_{NON-GUARD}(k) \right] + b$$

9.4.1.3.2.3 Generation of H^{ij}_{GLOBAL} when RLDiversityHoppingMode is on

When RLDiversityHoppingMode is on, $H^{ij}_{GLOBAL}(c, q, s, b)$ is a permutation of the non-guard-hop-port blocks in all subbands in carrier c of SDMA sub-tree q. The generation of H^{ij}_{GLOBAL} will be different for different values of RLSectorHopSeed, which is part of the public data of the Overhead Messages Protocol for carrier c. H^{ij}_{GLOBAL} shall be generated as follows:

1. Determine $S_{SWAP\text{-}LOCATION}$, $S_{MIN\text{-}CTRL\text{-}SUBBAND}$ and $N_{CTRL\text{-}HOP\text{-}PORT\text{-}BLOCKS}$ for carrier c as described in 9.4.1.3.2.1. Generate a permutation $H^{ij}_{SUBBAND}$ as described in 9.4.1.3.2.2. Determine $B_{SWAP\text{-}LOCATION}$ and $B_{MIN\text{-}CTRL\text{-}SUBBAND}$, where

$$B_{SWAP-LOCATION} = \sum_{k:H^{ij}_{SUBBAND}(k) < H^{ij}_{SUBBAND}(s)} B_{NON-GUARD}(k) \text{ and}$$

$$B_{MIN-CTRL-SUBBAND} = \sum_{k=0}^{S_{MIN-CTRL-SUBBAND}-1} B_{NON-GUARD}(k)$$

2. If RLSectorHopSeed is not equal to 1111 (in binary notation), set TMP = [(RLSectorHopSeed *4*64*4096+c*64*4096+j*4096 + (i mod 4096)) *2654435761] mod 2^{32}.

3. When RLSectorHopSeed is equal to 1111, set TMP = [(RLSectorHopSeed *4*64*4096+ c*64*4096+j*4096 + P_{SECTOR}) *2654435761] mod 2^{32}, where the 12-bit quantity P_{SECTOR} shall be computed as described in 9.4.1.3.2.4.

4. Set $SEED_{GLOBAL}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{GLOBAL}$ = [Bit-Reverse(TMP)] mod 2^{20}.

5. Generate a permutation π of size $\left[\sum_{k=0}^{S-1} B_{NON-GUARD}(k) \right] - N_{CTRL-HOP-PORT-BLOCKS}$ using the common permutation generation algorithm described in 9.4.1.3.1 with seed $SEED_{GLOBAL}$.

6. $H^{ij}_{GLOBAL}(c, q, s, b) = P(\beta)$[82], where $\beta = b + \sum_{k=0}^{s-1} B_{NON-GUARD}(k)$ and

   a. If $B_{MIN-CTRL-SUBBAND} \le \beta < B_{MIN-CTRL-SUBBAND} + N_{CTRL-HOP-PORT-BLOCKS}$, then $P(\beta) = (\beta - B_{MIN-CTRL-SUBBAND}) + B_{SWAP-LOCATION}$

   b. If $\beta < B_{MIN-CTRL-SUBBAND}$, then
      i. $P(\beta) = \pi(\beta)$ if $\pi(\beta) < B_{SWAP-LOCATION}$
      ii. $P(\beta) = \pi(\beta) + N_{CTRL-HOP-PORT-BLOCKS}$ if $\pi(\beta) \ge B_{SWAP-LOCATION}$

   c. If $\beta \ge B_{MIN-CTRL-SUBBAND} + N_{CTRL-HOP-PORT-BLOCKS}$, then
      i. $P(\beta) = \pi(\beta - N_{CTRL-HOP-PORT-BLOCKS})$ if $\pi(\beta - N_{CTRL-HOP-PORT-BLOCKS}) < B_{SWAP-LOCATION}$
      ii. $P(\beta) = \pi(\beta - N_{CTRL-HOP-PORT-BLOCKS}) + N_{CTRL-HOP-PORT-BLOCKS}$ if $\pi(\beta - N_{CTRL-HOP-PORT-BLOCKS}) \ge B_{SWAP-LOCATION}$

9.4.1.3.2.4 Generation of H^{ij}_{SECTOR}

$H^{ij}_{SECTOR}(c, q, s, .)$ is a permutation of the non-guard hop-port blocks in subband s of carrier c of SDMA sub-tree q. The generation of H^{ij}_{SECTOR} will be different for different values of RLIntraCellCommonHopping, which is part of the public data of the Overhead Messages Protocol for carrier c.[83]

The PilotPN of the sector of interest is XORed bitwise with the 12 LSBs of the superframe index i to obtain a 12-bit number $[b_{11}\, b_{10}\, b_9\, b_8\, b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0]$ denoted as P_{off}. The 12-bit number $[b_{11}\, b_{10}\, b_9\, b_8\, i_7\, i_6\, i_5\, b_4\, b_3\, b_2\, b_1\, b_0]$, where $i_7\, i_6\, i_5$ are the bits with indices 7,6 and 5 respectively in the superframe index i, is denoted as P_{on}. The permutation shall be generated as follows:

1. If RLIntraCellCommonHopping is off, set $P_{SECTOR} = P_{off}$. Otherwise, set $P_{SECTOR} = P_{on}$.

2. In TDD mode when RLSectorHopSeed is not equal to 1110, and in FDD mode, set TMP = [(4 *4*16*64*4096 + q*4*16*64*4096 + c*16*64*4096 + s*64*4096 + j*4096 + P_{SECTOR})* 2654435761] mod 2^{32}.

3. In TDD mode when RLSectorHopSeed is equal to 1110[84], set TMP = [(q*4*16*64*4096 + c*16*64*4096 + s*64*4096 + $\lfloor j / N_{RL\_BURST} \rfloor$*4096 + P_{SECTOR})* 2654435761] mod 2^{32}.

---

[82] P(β) first maps the hop port blocks allocated to the control segment to a contiguous set of subcarriers. The non-control hop port blocks are then assigned to the non-control subcarriers using a pseudo-random permutation π(.)

[83] When RLIntraCellCommonHopping is off, two sectors with different values of PilotPN have different hopping sequences. When RLIntraCellCommonHopping is on, sectors within the same cell have the same hopping sequences. For proper use of this mode, the operator should ensure that the PilotPNs of two sectors in the same cell differ only in the bits indexed 5,6, and 7.

[84] In TDD mode when RLSectorHopSeed and FLSectorHopSeed are both set to 1110, the FL hop permutations are "slaved" to the RL hop permutations for hop-ports on SDMA sub-tree 0 i.e., if a hop-port p is mapped to a

4. Set $SEED_{SECTOR}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{SECTOR}$ = [Bit-Reverse(TMP)] mod 2^{20}.

5. H^{ij}_{SECTOR}(c, q, s, .) is the permutation of size $B_{NON-GUARD}$(s) generated using the common permutation generation algorithm described in 9.4.1.3.1 with seed $SEED_{SECTOR}$.

9.4.1.4 R-ACKCH

The R-ACKCH is used to acknowledge FL PHY Frames transmitted on the F-DCH. For the purpose of this section, the sector of interest is the Forward Link Serving Sector (FLSS), which may or may not be different from the Reverse Link Serving Sector (RLSS). For convenience of notation, the phrase “of the Forward Link Serving Sector” shall be omitted. Sector-dependent quantities such as PilotPN, hop-permutations etc., used in this section shall be interpreted as “PilotPN of the FLSS,” “hop-permutations of the FLSS” etc.

9.4.1.4.1 R-ACKCH subcarrier allocation

In each RL PHY Frame, the R-ACKCH shall be allocated a number of contiguous subcarrier groups for the duration for the RL PHY Frame. Each such group of contiguous subcarriers shall comprise of $N_{BLOCK,\ R-ACKCH}$ subcarriers and shall be referred to as an R-ACKCH block. The group of $N_{BLOCK,\ R-ACKCH}N_{FRAME,R}$ subcarriers spanning $N_{BLOCK\ R-ACKCH}$ subcarriers in frequency and $N_{FRAME,R}$ OFDM Symbols in time shall be referred to as an R-ACKCH tile.

The number of R-ACKCH tiles allocated to the R-ACKCH in carrier c shall be N_{TILES}, where N_{TILES} = 0 if NumRACKBaseNodes = 0 and

$$N_{TILES} = \max\left(\frac{2NumRACKBaseNodes}{N_{R-ACKCH-SUBTILE-DURATION}N_{BLOCK,R-ACKCH}}, \frac{N_{FRAME,R}}{N_{R-ACKCH-SUBTILE-DURATION}}\right)$$

otherwise. Here NumRACKBaseNodes shall be specified by the RCC MAC protocol. The constant $N_{R-ACKCH-SUBTILE-DURATION}$ is the number of OFDM symbols allocated to each subtile, where the definition of subtile is as described in 9.4.1.4.2.

The set of R-ACKCH tiles in each carrier c of RL PHY Frame j in superframe with index i shall be indexed from 0 to N_{TILES} -1 and shall be determined according to the following procedure:

1. Compute P_{off} as described in 9.4.1.3.2.4. Compute TMP = [(4*4*64*4096 + c*64*4096 + j*4096 + P_{off}) * 2654435761] mod 2^{32}. Set $SEED_{RACKCH}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{RACKCH}$ = [Bit-Reverse(TMP)] mod 2^{20}.

2. Generate a permutation $H^{ij}_{RACKCH-SUBBANDS}$ of size S using the common permutation generation algorithm described in 9.4.1.3.1 with seed $SEED_{RACKCH}$. Here S = $N_{CARRIER\_SIZE}$ / $N_{SUBBAND}$ is the number of subbands in the carrier.

---

subcarrier f on the RL in a “burst” of RL PHY Frames, then that hop port is mapped to the same subcarrier f on the FL in the subsequent burst of FL PHY Frames as well.

3. Generate a permutation $H^{ij}_{RACKCH-BLOCKS}$ of size B using the common permutation generation algorithm described in 9.4.1.3.1 with seed $SEED_{RACKCH}$. Here $B = N_{SUBBAND} / N_{BLOCK}$ is the number of hop-port blocks in a subband.

4. Initialize counters i and t_{TILE} to 0. Also initialize counters $j_0, j_1, \ldots, j_{S-1}$ to 0.

5. Repeat the following steps till $t_{TILE} = N_{TILES}$.

    a. Set $s = H^{ij}_{RACKCH-SUBBANDS}(i)$ and $b = H^{ij}_{RACKCH-BLOCKS}(j_s)$. Set $FLAG_{RACKCH}$ to TRUE.

    b. Increment i and j_s by 1. If i = S, then set i to 0.

    c. Set $FLAG_{RACKCH}$ to FALSE if any one of the following three conditions is satisfied:

        i. Hop port (c, 0, s, b, 0) is a guard hop-port.

        ii. Subband s is allocated to the control segment.

        iii. The bit with index s of the RLRestrictedSetBitmap is set to 1, Here RLRestrictedSetBitmap is part of the public data of the Overhead Messages Protocol for carrier c.

    d. If $FLAG_{RACKCH}$ is TRUE, then

        i. Allocate the set of subcarriers $H^{ij}(c, 0, s, b, 0)$ to $[H^{ij}(c, 0, s, b, 0) + N_{BLOCK,RACKCH} -1]$ to the R-ACKCH for the duration of the RL PHY Frame for all RL PHY Frames other than those with index 0 in FDD mode. Here H^{ij} is the hop permutation for RL PHY Frame j in superframe with index i, as described in 9.4.1.3.2.

        ii. For RL PHY Frames with index 0 within the superframe in FDD mode, allocate the set of subcarriers $H^{ij}(c, 0, s, b, 0)$ to $[H^{ij}(c, 0, s, b, 0) + N_{BLOCK, RACKCH} -1]$ only for the OFDM symbols indexed $N_{PREAMBLE}$ through $N_{PREAMBLE} + N_{FRAME,R} -1$ in the PHY Frame.

        iii. The R-ACKCH tile index of this tile shall be t_{TILE}.

        iv. Increment t_{TILE} by 1.

9.4.1.4.2 R-ACKCH indexing

Each R-ACKCH tile shall further be divided into a number of R-ACKCH subtiles, each of which spans $N_{BLOCK, R-ACKCH}$ subcarriers in frequency and $N_{R-ACKCH\ SUBTILE-DURATION}$ OFDM symbols in time. The subtiles in each R-ACKCH tile shall be indexed from 0 to $N_{SUBTILES} - 1$, where $N_{SUBTILES} = (N_{FRAME, R} / N_{R-ACKCH-SUBTILE-DURATION})$. The OFDM Symbol with index t shall belong to subtile with index $k_{SUBTILE}$, where $k_{SUBTILE} = \lfloor t / N_{R-ACKCH-SUBTILE-DURATION} \rfloor$. Each subtile allocated to the R-ACKCH is thus indexed by the tuple (c, t_{TILE}, $k_{SUBTILE}$) where c is the carrier containing the tile, t_{TILE} is the R-ACKCH tile index within the carrier and $k_{SUBTILE}$ is the subtile index within that tile.

Each sub-tile has a total of $L= N_{BLOCK, R-ACKCH} N_{R-ACKCH-SUBTILE-DURATION}$ subcarriers. Exponential sequences of length L, may be used to modulate these subcarriers. The combination of a sub-tile and a specific exponential sequence of length L shall be referred to as an "R-ACKCH resource." An R-ACKCH resource indexed by the tuple (c, t_{TILE}, $k_{SUBTILE}$, ω) shall correspond to the usage of exponential sequence E_ω^L to modulate the subtile (c, t_{TILE}, $k_{SUBTILE}$). Here the sequence E_ω^L is a sequence of length L, whose i'th element $E_\omega^L(i)$ is given by $E_\omega^L(i) = e^{-2\pi j \omega i / L}$ where $0 \le i < L$, and j denotes the complex number (0,1).

9.4.1.4.3 R-ACKCH resource assignment

R-ACKCH transmissions are determined by an RACKBaseNodeIndex and a corresponding RACKVal specified by the RCC MAC protocol. An AT may be assigned zero, one or more RACKBaseNodeIndices in any RL PHY Frame.

An AT which is assigned an RACKBaseNodeIndex $D_{R\text{-}ACKCH}$ shall be assigned $N_{SUBTILES}$ R-ACKCH resources according to the following procedure:

1. Set g = $\lfloor D_{R-ACKCH} / N_{TILES} \rfloor$ and u = $D_{R\text{-}ACKCH}$ mod N_{TILES}.

2. Compute TMP = [(3*64*4*64*4096 + (g mod 64)*4*64*4096 + c*64*4096 + j*4096 + (i mod 4096)) * 2654435761] mod 2^{32}. Set $SEED_{RACKCH\text{-}ROWS}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{RACKCH\text{-}ROWS}$ = [Bit-Reverse(TMP)] mod 2^{20}.

3. Generate a permutation $H^{ij}_{RACKCH\text{-}ROWS}$ of size N_{TILES} using the common permutation generation algorithm described in 9.4.1.3.1 with seed $SEED_{RACKCH\text{-}ROWS}$.

4. Compute TMP = [(2*64*4*64*4096 + (g mod 64)*4*64*4096 + c*64*4096 + j*4096 + (i mod 4096)) * 2654435761] mod 2^{32}. Set $SEED_{RACKCH\text{-}COLS}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{RACKCH\text{-}COLS}$ = [Bit-Reverse(TMP)] mod 2^{20}.

5. Generate a permutation $H^{ij}_{RACKCH\text{-}COLS}$ of size $N_{SUBTILES}$ using the common permutation generation algorithm described in 9.4.1.3.1 with seed $SEED_{RACKCH\text{-}COLS}$.

6. Compute TMP = [(1*4*64*4096 + c*64*4096 + j*4096 + (i mod 4096)) * 2654435761] mod 2^{32}. Set $SEED_{RACKCH\text{-}CODES}$ to be the 20 LSBs of the bit-reversed value of TMP in a 32-bit representation, i.e., $SEED_{RACKCH\text{-}CODES}$ = [Bit-Reverse(TMP)] mod 2^{20}.

7. Generate a permutation $H^{ij}_{RACKCH\text{-}CODES}$ of size L/2 using the common permutation generation algorithm described in 9.4.1.3.1 with seed $SEED_{RACKCH\text{-}CODES}$.

8. Initialize a counter k to 0. Repeat the following steps until k = $N_{SUBTILES}$.

    a. Compute t = (u-k) mod N_{TILES}

    b. Set $t_{TILE} = H^{ij}_{RACKCH\text{-}ROWS}(t)$, $k_{SUBTILE} = H^{ij}_{RACKCH\text{-}COLS}(k)$ and $\omega = 2 * H^{ij}_{RACKCH\text{-}CODES}((g + t_{TILE}N_{SUBTILES} + k_{SUBTILE}) \bmod (L/2))$.

    c. Assign the R-ACKCH resource (c, t_{TILE}, $k_{SUBTILE}$, ω) to the AT.

    d. Increment k by 1.

9.4.1.4.4 R-ACKCH modulation

An AT shall transmit a sequence $X_{ACK}(c, t_{TILE}, k_{SUBTILE}, \omega)$ on each R-ACKCH resource $(c, t_{TILE}, k_{SUBTILE}, \omega)$ assigned to it. The sequence $X_{ACK}(c, t_{TILE}, k_{SUBTILE}, \omega)$ is an ON-OFF transmission specified by a bit RACKVal defined by the RCC MAC Protocol for each RACKBaseNodeIndex assigned to the AT. When RACKVal is equal to 1, the sequence $X_{ACK}(c, t_{TILE}, k_{SUBTILE}, \omega)$ shall be

$$X_{ACK}(c, t_{TILE}, k_{SUBTILE}, \omega) = \sqrt{P_{RACKCH} N_{FFT} / N_{BLOCK-RACKCH}}\, E_{\omega}^{L}$$

where E_{ω}^{L} is the exponential sequence of length L as defined in 9.4.1.4.2. and P_{RACKCH} is the power allocated to the R-ACKCH by the RCC MAC Protocol. When RACKVal is equal to 0, the sequence $X_{ACK}(c, t_{TILE}, k_{SUBTILE}, \omega)$ shall be a sequence of L zeros.

The sequence $X_{ACK}(c, t_{TILE}, k_{SUBTILE}, \omega)$ shall be used to modulate the L subcarriers in the subtile $(c, t_{TILE}, k_{SUBTILE})$ according to the following procedure:

1. Initialize an OFDM symbol counter t to t_{START}, where t_{START} is the lowest indexed OFDM Symbol in the subtile. Initialize a subcarrier counter f to f_{START}, where f_{START} is the lowest indexed subcarrier in the subtile. Initialize a modulation symbol counter i to 0.

2. Repeat the following steps till i = L.

   a. Modulate the subcarrier f in OFDM Symbol t with modulation symbol $X^{i}_{ACK}(c, t_{TILE}, k_{SUBTILE}, \omega)$. Here $X^{i}_{ACK}(c, t_{TILE}, k_{SUBTILE}, \omega)$ is the i^{th} element in the sequence $X_{ACK}(c, t_{TILE}, k_{SUBTILE}, \omega)$.

   b. Increment f by 1. If $f = f_{START} + N_{BLOCK,\ R\text{-}ACKCH}$, set f to f_{START} and increment t by 1.

   c. Increment i by 1.

9.4.1.5 Control segment modulation

The Control Segment carries the Access Channel (R-ACH), the Channel Quality Indicator Channel (R-CQICH), the Subband Feedback Channel (R-SFCH), the Beam Feedback Channel (R-BFCH), the Request Channel (R-REQCH) and the Pilot Channel (R-PICH).

The Control Segment is modulated in a Code Division Multiple Access (CDMA) fashion; i.e., transmissions from different access terminals are not orthogonal to each other. In MultiCarrierOff mode various channels in the Control Segment are generated in time domain, then added up and are converted to the frequency domain using a Discrete Fourier Transform (DFT) operation. The frequency domain sequence is then mapped to the subcarriers in the Control Segment assigned to the access terminal. In MultiCarrierOn mode various channels in the Control Segment are generated in time domain per carrier. These are then added up per carrier and are converted to the frequency domain using a Discrete Fourier Transform (DFT) operation, also per carrier. The frequency domain sequence is then mapped to the subcarriers in the Control Segment assigned to the access terminal.

In the frequency domain, in the c^{th} carrier, the Control Segment corresponding to each sector consists of NumRLControlSubbands contiguous subbands, where NumRLControlSubbands is public data of the Overhead Messages Protocol for carrier c for that sector. The AT uses the Control Segment of the sector in AS_{SYNCH} which has the smallest value of NumRLControlSubbands, where AS_{SYNCH} is as defined in 9.4.1. This value (the smallest value of NumRLControlSubbands) will henceforth be refered to as $N_{CTRL\_SUBBANDS}$. A hopping sequence for the Control Segment, described in 9.4.1.3.2.1, is used while mapping the frequency-domain sequence to subcarriers. The Control Segment hopping sequence maps R-ACH, R-CQICH, R-BFCH, R-SFCH, R-REQCH and R-PICH to the Control Segment per carrier as specified by the RCC MAC Protocol. The MAC protocol also specifies the carrier on which the above channels are modulated and the power allocated per channel.

9.4.1.5.1 Time-domain sequence generation

The following description is applicable to the time domain sequence generation in MultiCarrierOff mode and to the time domain sequence per carrier in MultiCarrierOn mode. Define $N_{CTRL\_FFT,}$ such that $N_{CTRL\_FFT} = A*2^k$ where $A = N_{SUBBAND}/128$ and k is the smallest integer such that $2^k \geq N_{CTRL\_SUBBANDS\_AT}$. Here, $N_{CTRL\_SUBBANDS\_AT}$ is given by NumRLControlSubbandsUser, which public data of the Active Set Management Protocol. Also define two integers $f_{MIN\text{-}CTRL\text{-}AT}$ and $N_{CTRL\text{-}SUBCARRIERS\text{-}AT}$ according to the following procedure:

1. Compute an integer TMP = $(i_{SF} + i_f*4096) * 2654435761 \bmod 2^{32}$, where i_{SF} denotes the superframe index, and i_f denotes the index of the RL PHY Frame within the superframe.

2. Compute another integer TMP2, which is the bit-reversed value of TMP in a 32-bit representation.

3. Define $N_{MIN\text{-}CTRL\text{-}SUBBAND\text{-}AT}$ as TMP2 + RLControlSubbandUserOffset mod ($N_{CTRL\_SUBBANDS}$ - $N_{CTRL\_SUBBANDS\_AT}$), where RLControlSubbandUserOffset is public data of the Active Set Update Protocol.

4. Compute $f_{MIN\text{-}CTRL\text{-}AT}$ as

$$f_{MIN-CTRL-AT} = f_{MIN-CTRL} + N_{BLOCK} \sum_{k=0}^{N_{MIN-CTRL-SUBBAND-AT}-1} B_{NON-GUARD}(S_{MIN-CTRL-SUBBAND} + k).$$

   Here, $f_{MIN\text{-}CTRL}$ and $S_{MIN\text{-}CTRL\text{-}SUBBAND}$ are as defined in 9.4.1.3.2.1 and $B_{NON\text{-}GUARD}(.)$ is as defined in 9.4.1.3.2. The values of these parameters used correspond to the sector whose Control Segment is being used by the AT.

5. Compute $N_{CTRL\text{-}SUBCARRIERS\text{-}AT}$ as

$$N_{CTRL-SUBCARRIERS-AT} = N_{BLOCK} \sum_{k=N_{MIN-CTRL-SUBBAND-AT}}^{N_{MIN-CTRL-SUBBAND-AT}+N_{CTRL\_SUBBANDS\_AT}} B_{NON-GUARD}\ (S_{MIN-CTRL-SUBBAND} + k).$$

The following sections discuss the modulation procedure for a specific carrier and for an individual instance of the channels specified by the AC MAC and RCC MAC Protocol. There could be more than one instance of the channels R-CQICH, R-BFCH and R-SFCH as specified by the RCC MAC Protocol. The described modulation procedure is repeated for every instance using the corresponding 10-bit value, MAC ID and PilotPN provided by the RCC MAC Protocol for these channels. In the following we assume an 11 bit MAC ID. In case the RCC MAC Protocol provides a MAC ID consisting of fewer than 11 bits, then the MAC ID is augmented to 11 bits by adding the requisite number of bits as MSBs. These added bits are set to 0.

9.4.1.5.1.1 Walsh sequence definition

Walsh sequences are used in the generation of the time-domain sequences for several physical layer channels carried in the Control Segment. A Walsh sequence W_i^N, where N is a power of 2 and i is a non-negative integer less than N, is a length N binary sequence taking on {-1,1} which is given by the i-th column of the N×N Hadamard matrix W^N. The N×N Hadamard matrix W^N is conventionally defined by the following recursive relationship:

$$W^2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad W^{2N} = \begin{bmatrix} W^N & W^N \\ W^N & -W^N \end{bmatrix}$$

9.4.1.5.1.2 R-ACH binary sequence

The sequence $X_{ACH,}$ of R-ACH is obtained from the i-th Walsh sequence W_i^{1024} of length 1024, by repeating each entry of this sequence N_{CTRL_FFT} times and scaling it to achieve the appropriate power P. The index i of the sequence is defined by the AC MAC Protocol. The resulting sequence is scrambled with a binary sequence S_{ACH} of length $1024*N_{CTRL\_FFT}$. Let $W_{k,i}^{1024}$ be the k-th element of the Walsh sequence W_i^{1024} with k within 0 and 1023, then define the sequence Z_{ACH} given by

$$Z_{ACH} = \sqrt{\frac{PN_{FFT}}{N_{CTRL\_SUBCARRIER\_RAT}}} \underbrace{\left(\underbrace{(-1)^{S_{ACH}^{0}} * W_{0,i}^{1024} \ldots (-1)^{S_{ACH}^{N_{CTRL\_FFT}-1}} * W_{0,i}^{1024}}_{N_{CTRL\_FFT}} \ldots \underbrace{(-1)^{S_{ACH}^{1023*N_{CTRL\_FFT}}} * W_{1023,i}^{1024} \ldots (-1)^{S_{ACH}^{1024*N_{CTRL\_FFT}-1}} * W_{1023,i}^{1024}}_{N_{CTRL\_FFT}} \right)}_{1024*N_{CTRL\_FFT}}$$

where P is the total power allocated by the AC MAC Protocol to transmit this sequence. The sequence X_{ACH} is obtained from Z_{ACH} by setting its last $N_{CTRL\_FFT}*N_{SUBBAND}$ elements to zero.

$$X_{ACH} = \sqrt{\frac{PN_{FFT}}{N_{CTRL\_SUBCARRIER\_RAT}}} \underbrace{\left(\underbrace{(-1)^{S_{ACH}^{0}} * W_{0,i}^{1024} \ldots (-1)^{S_{ACH}^{N_{CTRL\_FFT}-1}} * W_{0,i}^{1024}}_{N_{CTRL\_FFT}} \ldots \underbrace{0 \ldots\ldots\ldots 0}_{N_{SUBBAND}*N_{CTRL\_FFT}} \right)}_{1024*N_{CTRL\_FFT}}$$

The binary scrambling sequence S_{ACH} is generated as follows.

First, a binary sequence F_{ACH} of the length $n = 512$ shall be generated using a 20-bit shift register that shall have a generator polynomial $h(D) = D^{20} + D^{17} + D^{12} + D^{10} + 1$; i.e., the k-th output F_{ACH}^k of the register shall satisfy $F_{ACH}^k = F_{ACH}^{k-20} \oplus F_{ACH}^{k-17} \oplus F_{ACH}^{k-12} \oplus F_{ACH}^{k-10}$, for k=0,1,…, n -1. Here the initial state $\left[F_{ACH}^{-1}, \ldots, F_{ACH}^{-20}\right]$ is given $\left[1\,0\,0\,f_4\ f_3 f_2 f_1 f_0 p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$. Here $\left[p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$ is the 12-bit PilotPhase of the sector which is the target of the access probe for a given superframe, with LSB p_0 and MSB p_{11} and $\left[f_4 f_3 f_2 f_1 f_0\right]$ are the 5 LSBs of the binary representation for the index of the RL PHY Frame within the current superframe, with LSB f_0. Finally, the sequence S_{ACH} is obtained by repeating the n elements of F_{ACH} $2*N_{CTRL\_FFT}$ times:

$$S_{\mathrm{ACH}} = \left[\underbrace{F_{\mathrm{ACH}}^{0} F_{\mathrm{ACH}}^{0} \ldots F_{\mathrm{ACH}}^{0} F_{\mathrm{ACH}}^{0}}_{2*N_{CTRL\_FFT}} F_{\mathrm{ACH}}^{1} \ldots F_{\mathrm{ACH}}^{1} F_{\mathrm{ACH}}^{1} \ldots\ldots\ldots \underbrace{F_{\mathrm{ACH}}^{n-1} F_{\mathrm{ACH}}^{n-1} \ldots F_{\mathrm{ACH}}^{n-1} F_{\mathrm{ACH}}^{n-1}}_{2*N_{CTRL\_FFT}} \right]$$

In the absence of a R-ACH transmission request by AC MAC Protocol, X_{ACH} shall be an all-zero sequence of the same length.

9.4.1.5.1.3 R-CQICH binary sequence

The sequence $X_{CQI,}$ of R-CQICH is obtained from the i-th Walsh sequence W_i^{1024} of length 1024, by repeating each entry of this sequence N_{CTRL_FFT} times and scaling it to achieve the appropriate power P. The index i of the sequence takes on values from 0 to 1023 and is defined by ten-bit CQI value $(b_9,b_8,b_7,b_6, \ldots,b_1,b_0)$:

$$\mathrm{i} = \sum_{l=0}^{9} \mathrm{b}_l 2^l$$

The CQI value $(b_9,b_8,b_7,b_6, \ldots,b_1,b_0)$ is defined by the RCC MAC Protocol. The resulting sequence is scrambled with a binary sequence $S_{CQI.}$ Let $W_{k,i}^{1024}$ be the k-th element of the Walsh sequence W_i^{1024} with k within 0 and 1023, then X_{CQI} is given by

$$X_{CQI} = \sqrt{\frac{PN_{FFT}}{N_{CTRL-SUBCARRIER-RAT}}} \left(\underbrace{(-1)^{S_{CQI}^{0}} * W_{0,i}^{1024} \ldots (-1)^{S_{CQI}^{N_{CTRL\_FFT}-1}} * W_{0,i}^{1024}}_{N_{CTRL\_FFT}} \ldots \underbrace{(-1)^{S_{CQI}^{1023*N_{CTRL\_FFT}}} * W_{1023,i}^{1024} \ldots (-1)^{S_{CQI}^{1024*N_{CTRL\_FFT}-1}} * W_{1023,i}^{1024}}_{N_{CTRL\_FFT}} \right)$$

where P is the total power allocated by the RCC MAC Protocol to transmit this sequence. The binary scrambling sequence S_{CQI} is generated as follows. First, a binary sequence F_{CQI} of the length $n = 512$ shall be generated using a 20-bit shift register that shall have a generator polynomial $h(D) = D^{20} + D^{17} + D^{12} + D^{10} + 1$; i.e., the k-th output F_{CQI}^{k} of the register shall satisfy $F_{CQI}^{k} = F_{CQI}^{k-20} \oplus F_{CQI}^{k-17} \oplus F_{CQI}^{k-12} \oplus F_{CQI}^{k-10}$, for k=0,1,…, n-1. Here the initial state $\left[F_{CQI}^{-1}, \ldots, F_{CQI}^{-20}\right]$ is given by $\left[0\,0\,0\,\mathrm{f}_4\ \mathrm{f}_3\mathrm{f}_2\mathrm{f}_1\mathrm{f}_0\mathrm{p}_{11}\mathrm{p}_{10}\mathrm{p}_9\mathrm{p}_8\mathrm{p}_7\mathrm{p}_6\mathrm{p}_5\mathrm{p}_4\mathrm{p}_3\mathrm{p}_2\mathrm{p}_1\mathrm{p}_0\right]$. Here $\left[\mathrm{p}_{11}\mathrm{p}_{10}\mathrm{p}_9\mathrm{p}_8\mathrm{p}_7\mathrm{p}_6\mathrm{p}_5\mathrm{p}_4\mathrm{p}_3\mathrm{p}_2\mathrm{p}_1\mathrm{p}_0\right]$ is the 12-bit PilotPhase for the current superframe, corresponding to the PilotPN specified by the RCC MAC for this channel, with LSB p_0 and MSB p_{11}. Also, $\left[\mathrm{f}_4\mathrm{f}_3\mathrm{f}_2\mathrm{f}_1\mathrm{f}_0\right]$ are the 5 LSBs of the binary representation for the index of the RL PHY Frame within the current superframe, with LSB f_0. Finally, the sequence S_{CQI} is obtained by repeating the n elements of F_{CQI} $2*N_{CTRL\_FFT}$ times:

$$S_{\mathrm{CQI}} = \left[\underbrace{F_{\mathrm{CQI}}^{0} F_{\mathrm{CQI}}^{0} \ldots F_{\mathrm{CQI}}^{0} F_{\mathrm{CQI}}^{0}}_{2*N_{CTRL\_FFT}} F_{\mathrm{CQI}}^{1} F_{\mathrm{CQI}}^{1} \ldots F_{\mathrm{CQI}}^{1} F_{\mathrm{CQI}}^{1} \ldots\ldots\ldots \underbrace{F_{\mathrm{CQI}}^{n-1} F_{\mathrm{CQI}}^{n-1} \ldots F_{\mathrm{CQI}}^{n-1} F_{\mathrm{CQI}}^{n-1}}_{2*N_{CTRL\_FFT}} \right]$$

In the absence of a R-CQICH transmission request by the RCC MAC protocol, the sequence X_{CQI} shall be the all-zero sequence of the same length.

9.4.1.5.1.4 R-BFCH binary sequence

The sequence $X_{BFCH,}$ of R-BFCH is obtained from the i-th Walsh sequence W_i^{1024} of length 1024, by repeating each entry of this sequence N_{CTRL_FFT} times and scaling it to achieve the appropriate power P. The index i of the sequence takes on values from 0 to 1023 and is defined by ten-bit value $(b_9,b_8,b_7,b_6, \ldots,b_1,b_0)$:

$$i = \sum_{l=0}^{9} b_l 2^l$$

The value $(b_9,b_8,b_7,b_6, \ldots,b_1,b_0)$ is defined by the RCC MAC Protocol. The resulting sequence is scrambled with a binary sequence $S_{BFCH.}$ Let $W_{k,i}^{1024}$ be the k-th element of the Walsh sequence W_i^{1024} with k within 0 and 1023, then X_{BFCH} is given by

$$X_{BFCH}=\sqrt{\frac{PN_{FFT}}{N_{CTRL-SUBCARRIERS-RAT}}}\left(\underbrace{(-1)^{S_{BFCH}^{0}}*W_{0,i}^{1024}\ldots(-1)^{S_{BFCH}^{N_{CTRL\_FFT}-1}}*W_{0,i}^{1024}}_{N_{CTRL\_FFT}}\ldots\underbrace{(-1)^{S_{BFCH}^{1023*N_{CTRL\_FFT}}}*W_{1023,i}^{1024}\ldots(-1)^{S_{BFCH}^{1024*N_{CTRL\_FFT}-1}}*W_{1023,i}^{1024}}_{N_{CTRL\_FFT}}\right)$$

where P is the total power allocated by the RCC MAC Protocol to transmit this sequence. The binary scrambling sequence S_{BFCH} is generated as follows. First, a binary sequence F_{BFCH} of the length $n = 512$ shall be generated using a 20-bit shift register that shall have a generator polynomial h(D) = $D^{20} + D^{17} + D^{12} + D^{10} + 1$; i.e., the k-th output F_{BFCH}^{k} of the register shall satisfy $F_{\mathrm{BFCH}}^{k}=F_{\mathrm{BFCH}}^{k-20}\oplus F_{\mathrm{BFCH}}^{k-17}\oplus F_{\mathrm{BFCH}}^{k-12}\oplus F_{\mathrm{BFCH}}^{k-10}$, for k=0,1,…, n-1. Here the initial state $\left[F_{\mathrm{BFCH}}^{-1},\ldots,F_{\mathrm{BFCH}}^{-20}\right]$ is given by

$\left[0\,0\,1\,f_4\ f_3 f_2 f_1 f_0 p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$ if the BFCHReportType is 1 and

$\left[0\,1\,1\,f_4\ f_3 f_2 f_1 f_0 p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$ if the BFCHReportType is 2. Here, the BFCHReportType is a variable passed by the RCC MAC Protocol with each instance of the R-BFCH.

Here $\left[p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$ is the 12-bit PilotPhase for the current superframe, corresponding to the PilotPN specified by the RCC MAC for this channel, with LSB p_0 and MSB p_{11}. Also, $\left[f_4 f_3 f_2 f_1 f_0\right]$ are the 5 LSBs of the binary representation for the index of the RL PHY Frame within the current superframe, with LSB f_0. Finally, the sequence S_{BFCH} is obtained by repeating the n elements of F_{BFCH} $2*N_{CTRL\_FFT}$ times:

$$S_{\mathrm{BFCH}} = \left[\underbrace{F_{\mathrm{BFCH}}^{0}\,F_{\mathrm{BFCH}}^{0}\ldots F_{\mathrm{BFCH}}^{0}\,F_{\mathrm{BFCH}}^{0}}_{2*N_{CTRL\_FFT}}\,F_{\mathrm{BFCH}}^{1}\,F_{\mathrm{BFCH}}^{1}\ldots F_{\mathrm{BFCH}}^{1}\,F_{\mathrm{BFCH}}^{1}\,\ldots\ldots\,\underbrace{F_{\mathrm{BFCH}}^{n-1}\,F_{\mathrm{BFCH}}^{n-1}\ldots F_{\mathrm{BFCH}}^{n-1}\,F_{\mathrm{BFCH}}^{n-1}}_{2*N_{CTRL\_FFT}}\right]$$

In the absence of a R-BFCH transmission request by the RCC MAC protocol, the sequence X_{BFCH} shall be the all-zero sequence of the same length.

9.4.1.5.1.5 R-SFCH binary sequence

The sequence $X_{SFCH,}$ of R-SFCH is obtained from the i-th Walsh sequence W_i^{1024} of length 1024, by repeating each entry of this sequence N_{CTRL_FFT} times and scaling it to achieve the appropriate power P. The index i of the sequence takes on values from 0 to 1023 and is defined by ten-bit value $(b_9,b_8,b_7,b_6, \ldots,b_1,b_0)$:

$$i = \sum_{l=0}^{9} b_l 2^l$$

The value $(b_9,b_8,b_7,b_6, \ldots,b_1,b_0)$ is defined by the RCC MAC Protocol. The resulting sequence is scrambled with a binary sequence $S_{SFCH.}$ Let $W_{k,i}^{1024}$ be the k-th element of the Walsh sequence W_i^{1024} with k within 0 and 1023, then X_{SFCH} is given by

$$X_{SFCH} = \sqrt{\frac{PN_{FFT}}{N_{CTRL-SUBCARRIERS\,RAT}}} \left(\underbrace{(-1)^{S_{SFCH}^{0}} * W_{0,i}^{1024} \ldots (-1)^{S_{SFCH}^{N_{CTRL\_FFT}-1}} * W_{0,i}^{1024}}_{N_{CTRL\_FFT}} \ldots \underbrace{(-1)^{S_{SFCH}^{1023*N_{CTRL\_FFT}}} * W_{1023,i}^{1024} \ldots (-1)^{S_{SFCH}^{1024*N_{CTRL\_FFT}-1}} * W_{1023,i}^{1024}}_{N_{CTRL\_FFT}} \right)$$

where P is the total power allocated by the RCC MAC Protocol to transmit this sequence. The binary scrambling sequence S_{SFCH} are generated as follows. First, a binary sequence F_{SFCH} of the length $n = 512$ shall be generated using a 20-bit shift register that shall have a generator polynomial h(D) = $D^{20} + D^{17} + D^{12} + D^{10} + 1$; i.e., the k-th output F_{SFCH}^{k} of the register shall satisfy $F_{\mathrm{SFCH}}^{k} = F_{\mathrm{SFCH}}^{k-20} \oplus F_{\mathrm{SFCH}}^{k-17} \oplus F_{\mathrm{SFCH}}^{k-12} \oplus F_{\mathrm{SFCH}}^{k-10}$, for k=0,1,…, n -1. Here the initial state $\left[F_{SFCH}^{-1}, \ldots, F_{SFCH}^{-20}\right]$ is given by $\left[0\,1\,0\,f_4\ f_3 f_2 f_1 f_0 p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$. Here $\left[p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$ is the 12-bit PilotPhase for the current superframe, corresponding to the PilotPN specified by the RCC MAC for this channel, with LSB p_0 and MSB p_{11}. Also, $\left[f_4 f_3 f_2 f_1 f_0\right]$ are the 5 LSBs of the binary representation for the index of the RL PHY Frame within the current superframe, with LSB f_0. Finally, the sequence S_{SFCH} is obtained by repeating the n elements of F_{SFCH} $2*N_{CTRL\_FFT}$ times:

$$S_{\mathrm{SFCH}} = \left[\underbrace{F_{\mathrm{SFCH}}^{0}\ F_{\mathrm{SFCH}}^{0} \ldots F_{\mathrm{SFCH}}^{0}\ F_{\mathrm{SFCH}}^{0}}_{2*N_{CTRL\_FFT}}\ F_{\mathrm{SFCH}}^{1}\ F_{\mathrm{SFCH}}^{1} \ldots F_{\mathrm{SFCH}}^{1}\ F_{\mathrm{SFCH}}^{1} \ldots\ldots\ \underbrace{F_{\mathrm{SFCH}}^{n-1}\ F_{\mathrm{SFCH}}^{n-1} \ldots F_{\mathrm{SFCH}}^{n-1}\ F_{\mathrm{SFCH}}^{n-1}}_{2*N_{CTRL\_FFT}} \right]$$

In the absence of an R-SFCH transmission request by the RCC MAC protocol, the sequence X_{SFCH} shall be all-zero sequence of the same length.

9.4.1.5.1.6 R-REQCH binary sequence

The sequence $X_{REQ,}$ of R-REQCH is obtained from the i-th Walsh sequence W_i^{1024} of length 1024, by repeating each entry of this sequence $N_{CTRL-FFT}$ times and scaling it to achieve the appropriate power P. The index i of the sequence takes on values from 0 to 1023 and is defined by the ten-bit vector $(b_9,b_8,b_7,b_6, \ldots,b_1,b_0)$:

$$i = \sum_{l=0}^{9} b_l 2^l$$

The ten-bit vector ($b_9,b_8,b_7,b_6, \ldots,b_1,b_0$) is defined by the RCC MAC Protocol. The resulting sequence is scrambled with a binary sequence S_{REQ}. Let $W_{k,i}^{1024}$ be the k-th element of the Walsh sequence W_i^{1024} with k within 0 and 1023, then X_{REQ} is given by

$$X_{REQ}=\sqrt{\frac{PN_{FFT}}{N_{CTRL-SUBCARRIER-RAT}}}\left(\underbrace{(-1)^{S_{REQ}^{0}}*W_{0,i}^{1024}\ldots(-1)^{S_{REQ}^{N_{CTRL\_FFT}-1}}*W_{0,i}^{1024}}_{N_{CTRL\_FFT}}\ldots\underbrace{(-1)^{S_{REQ}^{1023N_{CTRL\_FFT}}}*W_{1023,i}^{1024}\ldots(-1)^{S_{REQ}^{1023N_{CTRL\_FFT}-1}}*W_{1023,i}^{1024}}_{N_{CTRL\_FFT}}\right)$$

where P is the total power allocated by the RCC MAC Protocol to transmit this sequence. The binary scrambling sequence S_{REQ} are generated as follows. First, a binary sequence F_{REQ} of the length $n = 512$ shall be generated using a 20-bit shift register shall have a generator polynomial $h(D) = D^{20} + D^{17} + D^{12} + D^{10} + 1$, i.e., the k-th output F_{REQ}^{k} of the register shall satisfy $F_{REQ}^{k}=F_{REQ}^{k-20}\oplus F_{REQ}^{k-17}\oplus F_{REQ}^{k-12}\oplus F_{REQ}^{k-10}$, for k=0,1,…, n -1. Here the initial state $\left[F_{REQ}^{-1},\ldots,F_{REQ}^{-20}\right]$ is given by $\left[1\,0\,1\,f_4\ f_3 f_2 f_1 f_0 p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$.

Here $\left[p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$ is the 12-bit PilotPhase for the current superframe, corresponding to the PilotPN specified by the RCC MAC for this channel, with LSB p_0 and MSB p_{11}. Also, $\left[f_4 f_3 f_2 f_1 f_0\right]$ are the 5 LSBs of the binary representation for the index of the RL PHY Frame within the current superframe, with LSB f_0. Finally, the sequence S_{REQ} is obtained by repeating the n elements of F_{REQ} 2*N_{CTRL_FFT} times:

$$S_{\mathrm{REQ}} = \left[\underbrace{F_{\mathrm{REQ}}^{0}\,F_{\mathrm{REQ}}^{0}\ldots F_{\mathrm{REQ}}^{0}\,F_{\mathrm{REQ}}^{0}}_{2*N_{CTRL\_FFT}}\,F_{\mathrm{REQ}}^{1}\,F_{\mathrm{REQ}}^{1}\ldots F_{\mathrm{REQ}}^{1}\,F_{\mathrm{REQ}}^{1}\,\ldots\ldots\,\underbrace{F_{\mathrm{REQ}}^{n-1}\,F_{\mathrm{REQ}}^{n-1}\ldots F_{\mathrm{REQ}}^{n-1}\,F_{\mathrm{REQ}}^{n-1}}_{2*N_{CTRL\_FFT}}\right]$$

In the absence of an R-REQCH transmission request by the RCC MAC protocol, the sequence X_{REQ} shall be all-zero sequence of the same length.

9.4.1.5.1.7 R-PICH binary sequence

The sequence $X_{PICH,}$ of R-PICH is obtained from a binary sequence of length 1024* N_{CTRL_FFT}, which takes values on {-1,1}. The binary sequence is further scaled to achieve the appropriate power P, according to the following equation:

$$X_{PICH}=\sqrt{\frac{PN_{FFT}}{N_{CTRL-SUBCARRIER-RAT}}}\left((-1)^{S_{PICH}^{0}},(-1)^{S_{PICH}^{1}}\ldots(-1)^{S_{PICH}^{1024N_{CTRL\_FFT}-2}},(-1)^{S_{PICH}^{1024N_{CTRL\_FFT}-1}}\right),$$

where P is the total power allocated by the RCC MAC Protocol to transmit this sequence and wherein the binary sequence S_{PICH} is generated as follows. First, a binary sequence F_{PICH} of the length $n = 512$ which shall be generated using a 20-bit shift register which shall have a generator polynomial $h(D) = D^{20} + D^{17} + D^{12} + D^{10} + 1$; i.e., the k-th output F_{PICH}^{k} of the register shall satisfy $F_{PICH}^{k}=F_{PICH}^{k-20}\oplus F_{PICH}^{k-17}\oplus F_{PICH}^{k-12}\oplus F_{PICH}^{k-10}$, for k=0,1,…, n -1. Here the initial state $\left[F_{PICH}^{-1},\ldots,F_{PICH}^{-20}\right]$ is given by $\left[1\,1\,0\,f_4\ f_3 f_2 f_1 f_0 p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$. Here $\left[p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0\right]$ is the 12-bit PilotPhase for the current superframe, corresponding to the PilotPN specified by the RCC

MAC for this channel, with LSB p_0 and MSB p_{11}. Also, $[f_4 f_3 f_2 f_1 f_0]$ are the 5 LSBs of the binary representation for the index of the RL PHY Frame within the current superframe, with LSB f_0. Finally, the sequence S_{PICH} is obtained by repeating the n elements of F_{PICH} $2*N_{CTRL\_FFT}$ times:

$$S_{\text{PICH}} = \left[\underbrace{F^0_{\text{PICH}}\, F^0_{\text{PICH}} \ldots F^0_{\text{PICH}}\, F^0_{\text{PICH}}}_{2*N_{CTRL\_FFT}}\, F^1_{\text{PICH}}\, F^1_{\text{PICH}} \ldots F^1_{\text{PICH}}\, F^1_{\text{PICH}} \ldots\ldots \underbrace{F^{n-1}_{\text{PICH}}\, F^{n-1}_{\text{PICH}} \ldots F^{n-1}_{\text{PICH}}\, F^{n-1}_{\text{PICH}}}_{2*N_{CTRL\_FFT}} \right]$$

9.4.1.5.2 Multiplexing of R-CQICH, R-BFCH, R-SFCH, R-REQCH and R-PICH

If multiple instances of the channels R-CQICH, R-BFCH and R-SFCH are defined by the RCC MAC Protocol, then the resulting modulation sequences are superimposed. For example, X_{CQI} from now on refers to the sequence obtained by superimposing the modulated sequence for every instance of the R-CQICH channel. The same holds for X_{SFCH} and X_{BFCH}.

R-CQICH, R-BFCH, R-SFCH, R-REQCH, and R-PICH are I-Q multiplexed within the Control Segment. The combined complex-valued time domain sequence X_{CTRL} of these channels is given by the following equation:

$$X_{CTRL} = (X_{CQICH} + X_{SFCH} + X_{PICH}) + j*(X_{REQCH} + X_{BFCH})$$

9.4.1.5.3 HPSK scrambling

All the channels within the Control Segment with an assigned MACID undergo sector-specific and MACID scrambling (this refers to R-CQICH, R-BFCH, R-SFCH, R-REQCH, and R-PICH). R-ACH may undergo sector-specific scrambling or sector-specific and MACID scrambling depending on the access sequence ID defined by the AC MAC Protocol.

The sector-specific scrambling sequence $Y_{SS,ACH}$ of the length 1024* N_{CTRL_FFT} is generated according to:

$$Y_{SS,ACH} = \left(Y^0_{SS,ACH} Y^1_{SS,ACH} \ldots Y^{1024*N_{CTRL\_FFT}-1}_{SS,ACH}\right), \quad Y^0_{SS,ACH} = 1, \quad Y^k_{SS,ACH} = Y^{k-1}_{SS,ACH} e^{j\pi/2*\left(2*S^{k-1}_{SS,ACH}-1\right)}$$

where the binary scrambling sequence $S_{SS,ACH}$ of the length (1024* N_{CTRL_FFT} -1) shall be generated using a 20-bit shift register which shall have a generator polynomial $h(D) = D^{20} + D^{17} + D^{12} + D^{10} + 1$, i.e., the k-th output $S^k_{SS,\text{ACH}}$ of the register shall satisfy $S^k_{SS,\text{ACH}} = S^{k-20}_{SS,\text{ACH}} \oplus S^{k-17}_{SS,\text{ACH}} \oplus S^{k-12}_{SS,\text{ACH}} \oplus S^{k-10}_{SS,\text{ACH}}$, where the initial state $\left[S^{-1}_{SS,\text{ACH}}, \ldots, S^{-20}_{SS,\text{ACH}}\right]$ is given by $[0\,0\,1\,0\,f_4 f_3 f_2 f_1 f_0 p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0]$. Here $[p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0]$ is the 12-bit PilotPhase of the target sector for the access probe for the current superframe, with LSB p_0 and MSB p_{11}. Also, $[f_4 f_3 f_2 f_1 f_0]$ are the 5 LSBs of the binary representation for the index of the PHY Frame within the current superframe, with LSB f_0.

The sector-specific and MACID scrambling sequence $Y_{SM,ACH}$ of the length 1024* N_{CTRL_FFT} is given by:

$$Y_{SM,\text{ACH}} = \left(Y^0_{SM,\text{ACH}} Y^1_{SM,\text{ACH}} \ldots Y^{1024*N_{CTRL\_FFT}-1}_{SM,\text{ACH}}\right), \quad Y^0_{SM,\text{ACH}} = 1, \quad Y^k_{SM,\text{ACH}} = Y^{k-1}_{SM,\text{ACH}} e^{j\pi/2*\left(2*S^{k-1}_{SM,\text{ACH}}-1\right)}$$

where the binary scrambling sequence $S_{SM,ACH}$ of the length (1024*N_{CTRL_FFT} -1) shall be generated using a 28-bit shift register which shall have a generator polynomial h(D) = D^{28} +D^{25} +1, i.e., the k-th output $S^{k}_{SM,\mathrm{ACH}}$ of the register shall satisfy $S^{k}_{SM,\mathrm{ACH}}=S^{k-28}_{SM,\mathrm{ACH}}\oplus S^{k-25}_{SM,\mathrm{ACH}}$. Here the initial state $\left[S^{-1}_{SM,ACH},\ldots,S^{-28}_{SM,ACH}\right]$ is given by $\left[f_4f_3f_2f_1f_0m_{10}m_9m_8m_7m_6m_5m_4m_3m_2m_1m_0p_{11}p_{10}p_9p_8p_7p_6p_5p_4p_3p_2p_1p_0\right]$. Here $\left[f_4f_3f_2f_1f_0\right]$ are the 5 LSBs of the binary representation for the index of the RL PHY Frame within the current superframe, with LSB f_0. $\left[p_{11}p_{10}p_9p_8p_7p_6p_5p_4p_3p_2p_1p_0\right]$ is the 12-bit PilotPhase of the target sector of the access probe for the current superframe, with LSB p_0 and MSB p_{11} , and $\left[m_{10}m_9m_8m_7m_6m_5m_4m_3m_2m_1m_0\right]$ is the 11-bit MACID corresponding to the target sector of the access probe.

The sector-specific and MACID scrambling sequence Y_{SM} of the length 1024* N_{CTRL_FFT} is given by:

$$Y_{SM}=\left(Y^{0}_{SM}Y^{1}_{SM}\ldots Y^{1024*N_{CTRL\_FFT}-1}_{SM}\right),\quad Y^{0}_{SM}=1,\quad Y^{k}_{SM}=Y^{k-1}_{SM}e^{j\pi/2*\left(2*S^{k-1}_{SM}-1\right)}$$

where the binary scrambling sequence S_{SM} of the length (1024*N_{CTRL_FFT} -1) shall be generated using a 28-bit shift register which shall have a generator polynomial h(D) = D^{28} +D^{25} +1, i.e., the k-th output S^{k}_{SM} of the register shall satisfy $S^{k}_{SM}=S^{k-28}_{SM}\oplus S^{k-25}_{SM}$. Here the initial state $\left[S^{-1}_{SM},\ldots,S^{-28}_{SM}\right]$ is given by $\left[f_4f_3f_2f_1f_0m_{10}m_9m_8m_7m_6m_5m_4m_3m_2m_1m_0p_{11}p_{10}p_9p_8p_7p_6p_5p_4p_3p_2p_1p_0\right]$. Here $\left[p_{11}p_{10}p_9p_8p_7p_6p_5p_4p_3p_2p_1p_0\right]$ is the 12-bit PilotPhase for the current superframe, corresponding to the PilotPN specified by the RCC MAC for all channels except the R-ACH within this control segment[85], with LSB p_0 and MSB p_{11}. Also, $\left[f_4f_3f_2f_1f_0\right]$ are the 5 LSBs of the binary representation for the index of the RL PHY Frame within the current superframe with LSB f_0 and $\left[m_{10}m_9m_8m_7m_6m_5m_4m_3m_2m_1m_0\right]$ is the 11-bit MACID specified by the RCC MAC.

9.4.1.5.4 Control Segment complex-valued signal in the time domain

The time domain sequence Z of length 1024 * N_{CTRL_FFT} transmitted over the Control Channel segment is given by the following equation:

$$\begin{aligned}Z=&\left(Y^{0}_{SM}*X^{0}_{CTRL},Y^{1}_{SM}*X^{1}_{CTRL},\ldots,Y^{1024*N_{CTRL\_FFT}-1}_{SM}*X^{1024*N_{CTRL\_FFT}-1}_{CTRL}\right)\\&+\left(Y^{0}_{SS,ACH}*X^{0}_{ACH},Y^{1}_{SS,ACH}*X^{1}_{ACH},\ldots,Y^{1024*N_{CTRL\_FFT}-1}_{SS,ACH}*X^{1024*N_{CTRL\_FFT}-1}_{ACH}\right)\end{aligned}$$

if the access sequence ID > $N_{ACMPSpecialSequences}$ and the equation

$$\begin{aligned}Z=&\left(Y^{0}_{SM}*X^{0}_{CTRL},Y^{1}_{SM}*X^{1}_{CTRL},\ldots,Y^{1024*N_{CTRL\_FFT}-1}_{SM}*X^{1024*N_{CTRL\_FFT}-1}_{CTRL}\right)\\&+\left(Y^{0}_{SM,ACH}*X^{0}_{ACH},Y^{1}_{SM,ACH}*X^{1}_{ACH},\ldots,Y^{1024*N_{CTRL\_FFT}-1}_{SM,ACH}*X^{1024*N_{CTRL\_FFT}-1}_{ACH}\right)\end{aligned}$$

[85] Note that the RCC MAC protocol specifies the same PilotPN and MACID for all channels in the control segment except possibly the R-ACH.

if the access sequence ID ≤ $N_{ACMPSpecialSequences}$ where the access sequence ID is as defined by the AC MAC Protocol and used to modulate R-ACH, and $N_{ACMPSpecialSequences}$ is a constant of the AC MAC protocol.

In the above equations, X_{CTRL}^{l} and X_{ACH}^{l} refer to the *l*-th entry of the vectors X_{CTRL} and X_{ACH} respectively, with *l* taking values from 0 to 1024* N_{CTRL_FFT}-1.

9.4.1.5.5 DFT operation

The following description is applicable per carrier in MultiCarrierOn mode. The scrambled sequence Z of length 1024*N_{CTRL_FFT} generated in the previous section shall be broken up into $N_{FRAME,R}$ different subsequences of length R = (N_{CTRL_FFT}/A)*$N_{SUBBAND}$ where A = ($N_{SUBBAND}$/128). The first R elements of the sequence Z form the first subsequence Z_0, the next R elements form the second sequence Z_1, etc. Each of these subsequences shall be converted to a frequency domain sequence through a Discrete Fourier Transform (DFT) operation, modulated and transmitted on the corresponding OFDM symbol. The DFT of an N-length sequence X with elements $x_0, x_1, \ldots, x_{N-1}$ is given by another N-length sequence Y with elements $y_0, y_1, \ldots, y_{N-1}$. The elements of Y are related to the elements of X via the relationship

$$y_i = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} x_k e^{-j2\pi(k-\frac{N}{2})i/N} .$$

Let F_l denote the DFT of the R -length subsequence Z_l, $0 \le l < N_{FRAME,R} - 1$. The first $N_{CTRL\text{-}SUBCARRIERS\text{-}AT}$ elements of the sequence F_l are allocated to hop ports in the Control Segment. More precisely, the t elements of the sequence F_l, t ranging from 0 to $N_{CTRL\text{-}SUBCARRIERS\text{-}AT}$-1, shall be modulated onto subcarriers with indices ranging from $f_{MIN\text{-}CTRL\text{-}AT}$ to $f_{MIN\text{-}CTRL\text{-}AT} + N_{CTRL\text{-}SUBCARRIERS\text{-}AT}$ -1, of the *l*'th OFDM symbol in the RL PHY Frame. Here, $f_{MIN\text{-}CTRL\text{-}AT}$ and $N_{CTRL\text{-}SUBCARRIERS\text{-}AT}$ are as defined in 9.4.1.5.1. The remaining elements of the sequence F_l are discarded.

9.4.1.6 Data segment modulation

The Data Segment carries the R-DPICH and the R-DCH. A subcarrier occupied by the R-DPICH shall be referred to as a pilot subcarrier. For the purpose of this section, the sector of interest is the Reverse Link Serving Sector (RLSS), which may or may not be different from the Forward Link Serving Sector (FLSS). For convenience of notation, the phrase "of the Rorward Link Serving Sector" shall be omitted. Sector-dependent quantities such as PilotPN, hop-permutations etc., used in this section shall be interpreted as "PilotPN of the RLSS," "hop-permutations of the RLSS" etc.

9.4.1.6.1 R-DPICH

The Dedicated Pilot Channel (R-DPICH) shall be present in each tile that is assigned to the R-DCH (for this AT). The modulation of this channel is described for a single tile, where a tile is as described in 9.4.1.3. The modulation procedure shall then be repeated for each such tile that is assigned to the R-DCH (for this AT) in every RL PHY Frame. The configuration of the R-DPICH in a given tile is determined by the configuration of the R-DCH in this tile. If there is no R-DCH present in a tile, R-DPICH shall also not be transmitted.

The R-DPICH configuration in each tile consists of the following parameters:

1. The energy per modulation symbol: All the R-DPICH modulation symbols in a given tile shall have the same energy, which shall be the same as the energy used to transmit R-DCH modulation symbols in this tile. This energy is assigned by the RTC MAC protocol.

2. R-DPICH format: The R-DPICH in a tile can have two different formats, labeled Format 0 and Format 1. The R-DPICH format to be used is determined by the RTC MAC protocol.

3. RLDPISectorOffset: This is part of the public data of the Overhead Messages Protocol, and takes on integer values between 0 and 3. The value used shall correspond to the carrier containing the tile of interest.

4. RLDPIUserOffset: This is an integer that depends on the hop-ports contained in the tile of interest. Let p_{min} be the smallest hop-port index (within the carrier) contained in the tile of interest. RLDPIUserOffset is then given by $\lfloor p_{\min} / N_{CARRIER\_SIZE} \rfloor$.

In order to aid the description of the R-DPICH formats, the hop-ports in each tile are numbered from 0 to N_{BLOCK}-1 in increasing order. Also, for all RL PHY Frames, except for the RL PHY Frame with index 0 in the superframe in FDD mode, the OFDM symbols in each tile are numbered from 0 to $N_{FRAME,R}$-1 in increasing order. For the RL PHY Frame with index 0 in FDD mode, the OFDM symbols in each tile are numbered from 0 to $N_{FRAME,R} + N_{PREAMBLE} - 1$ in increasing order.

9.4.1.6.1.1.1 R-DPICH Format 0

In this format, the R-DPICH occupies 18 modulation symbols in a tile in all RL PHY Frames, except those with index 0 within the superframe in FDD mode. In an RL PHY Frame with index 0 within the superframe in FDD mode, the R-DPICH occupies 36 modulation symbols within each tile. A hop-port with index i_{hp} of the OFDM symbol with index t (both measured within the tile) is occupied by the R-DPICH if i_{hp} is in the set {1,8,15}, if t' = t mod $N_{FRAME,R}$ is in the set {0,1,2,5,6,7}, and if this hop-port is not mapped to a subcarrier assigned to the R-ACKCH of the RLSS. The set of hop-ports occupied by the R-DPICH for this format is illustrated in Figure 113, for the case when none of the hop-ports in the tile are assigned to the R-ACKCH.

The complex value of the R-DPICH modulation symbol at this location is given by

$$S_{i_{hp},t} = \sqrt{P}\exp\left(\frac{j2\pi}{3}(RLDPISectorOffset + RLDPIUserOffset)\,t'\right) \text{ if } t' < 4, \text{ and}$$

$$S_{i_{hp},t} = \sqrt{P}\exp\left(\frac{j2\pi}{3}(RLDPISectorOffset + RLDPIUserOffset)(t'-2)\right) \text{ if } t' \geq 4.$$

where j denotes the complex number (0,1) and P is the energy per modulation symbol used by the R-DPICH. Note that the value of this modulation symbol is the same for all values of i_{hp}.

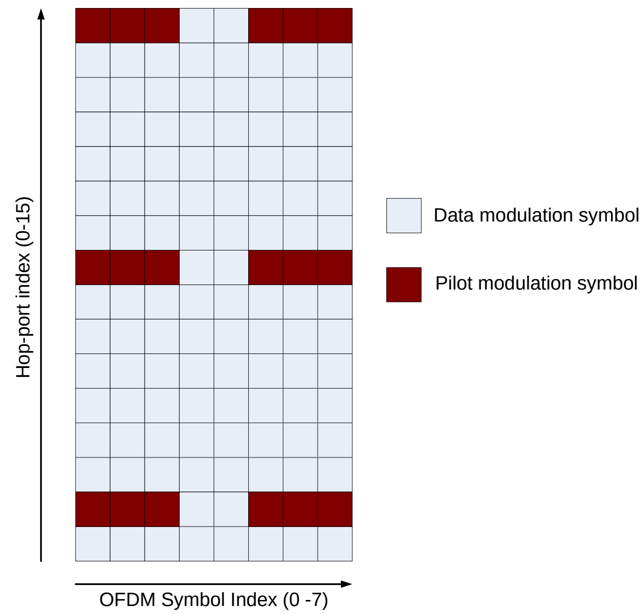

Figure 113 R-DPICH Format 0

9.4.1.6.1.1.2 R-DPICH Format 1

In this format, the R-DPICH occupies 24 modulation symbols in a tile in all RL PHY Frames, except those with index 0 within the superframe in FDD mode. In an RL PHY Frame with index 0 within the superframe in FDD mode, the R-DPICH occupies 48 modulation symbols in each tile. In this format, a hop-port with index i_{hp} of the OFDM symbol with index t (both measured within the tile) is occupied by the R-DPICH if i_{hp} is in the set {0,3,6,9,12,15}, if t' = t mod $N_{FRAME,R}$ is in the set {0,1,6,7}, and if this hop-port is not mapped to a subcarrier assigned to the R-ACKCH of the RLSS. The set of hop-ports occupied by the R-DPICH for this format is illustrated in Figure 114, for the case when none of the hop-ports in the tile are assigned to the R-ACKCH.

The complex value of the R-DPICH modulation symbol at this location is given by

$$S_{i_{hp},t} = \sqrt{P}\exp\left(j\pi(RLDPISectorOffset + RLDPIUserOffset)\,t'\right),$$

where j denotes the complex number (0,1) and P is the energy per modulation symbol used by the R-DPICH. Note that the value of this modulation symbol is the same for all values of i_{hp}.

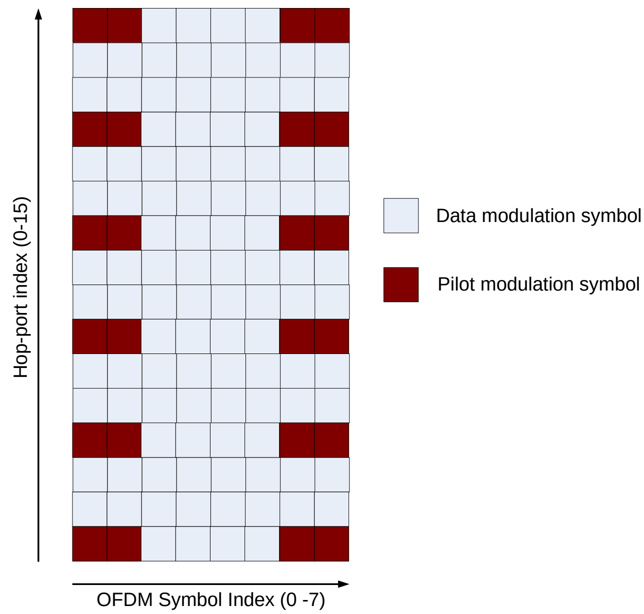

Figure 114 R-DPICH Format 1

9.4.1.6.2 R-DCH

The R-DCH consists of either a data packet or an erasure sequence, both of which can span one or more RL PHY Frames. The set of RL PHY Frames on which this packet or erasure sequence is transmitted is determined by the RTC MAC Protocol. Each data packet and erasure sequence is also assigned a set of hop-ports in each PHY Frame of transmission by the RTC MAC Protocol. Note that this set of hop-ports can span multiple carriers. Each data packet is further associated with a packet format index, which is also assigned by the RTC MAC Protocol.

9.4.1.6.2.1 Data packet transmission

Each R-DCH packet is generated by the RTC MAC Protocol, and is split, appended with CRC, encoded, channel interleaved, repeated, data-scrambled and modulated according to the procedure described in 9.2. A CRC length of $N_{CRC,Data}$ is used for this packet. The MACID of the access terminal corresponding to its Reverse Link Serving Sector (RLSS), and the packet format index assigned to this packet, shall be used to generate the initial state of the data-scrambler described in 9.2.5. The size of the input packet generated by the RTC MAC Protocol shall be equal to $8\lfloor \rho n_0 N_f / 8 \rfloor - N_{CRC,Data}$, where ρ denotes the spectral efficiency at the first transmission corresponding to the packet format of the packet (defined by the RTC MAC Protocol), n_0 denotes the number of usable hop-ports assigned to this packet in the first RL PHY Frame of transmission, and N_f is equal to $6N_{FRAME,R}$ if this packet is part of an extended duration transmission and is equal to $N_{FRAME,R}$ otherwise. The RTC MAC protocol determines whether or not a packet is part of an extended duration transmission. Here, a usable hop-port is as defined in 9.4.1.3.2. This packet shall be modulated on to the hop-ports assigned to this packet according to the following procedure:

1. Initialize a port counter i to 0, a frame counter f to 0, and an OFDM symbol counter j to 0.

2. Arrange the set of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission in increasing order, where the ordering of hop-ports is as defined in 9.4.1.2.6. Let the resulting sequence be denoted by (k_0,p_0), (k_1,p_1), …, (k_{n-1},p_{n-1}), where n is the total number of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission. The notation for a hop-port is as in 9.3.2.2.5, i.e., k_i denotes the CarrierIndex and (k_i,p_i) denotes the hop-port index in that carrier.

3. Let n_{sc} be the subcarrier index corresponding to the hop-port (k_i,p_i) in the j'th OFDM symbol in the f'th RL PHY Frame of transmission. Let q be the modulation order to be used for the f'th PHY Frame of transmission, which is a function of the packet format. If n_{sc} is not assigned to the R-ACKCH of the RLSS and if (k_i,p_i) is not a DPICH hop-port, then a modulation symbol s with modulation order q is generated by the modulator according to the procedure described in 9.2.6. This modulation symbol shall be modulated with energy PN_{FFT}/n on hop-port (k_i,p_i), i.e., the value of the corresponding subcarrier shall be $\sqrt{PN_{FFT}/n}\, s$, where P is the power specified for this assignment in the f'th PHY Frame of transmission (generated by the RTC MAC Protocol).

4. Increment i. If i = n, increment j and set i = 0

5. Increment f and set j = 0 if any of the following two conditions is satisfied:
   - If this is an RL PHY Frame with index 0 within the superframe and the duplexing mode is FDD, and if $j = N_{FRAME,R} + N_{PREAMBLE}$.
   - For any other RL PHY Frame (including all RL PHY Frames in TDD mode), if $j = N_{FRAME,R}$.

6. If the last RL PHY Frame of transmission has been completed (as determined by the RTC MAC Protocol), then stop. Else repeat steps 2 through 6.

9.4.1.6.2.2 Erasure sequence

An erasure sequence spans one or more consecutive RL PHY Frames of transmission on a set of hop-ports determined by the RTC MAC Protocol. The erasure sequence shall be modulated on to the hop-ports assigned to this sequence according to the following procedure:

1. Construct a one-bit packet, with the bit in the packet being set to zero. This packet is encoded, channel interleaved, repeated, scrambled, and modulated according to the procedure described in 9.2[86]. The MAC ID of the access terminal corresponding to its RL Serving Sector, and a packet format index of 0 shall be used to generate the initial seed of the scrambler. QPSK modulation shall be used for all of the modulation symbols in the packet.

2. Initialize a port counter i to 0, an OFDM symbol counter j to 0, and a PHY Frame counter f to 0.

[86] The operations before scrambling and modulation are all trivial operations, i.e., they result in an all-zeros sequence. The erasure sequence is equivalent to scrambling an all-zeros sequence of the required length, followed by QPSK modulation.

3. Arrange the set of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission in increasing order, where the ordering of hop-ports is as defined in 9.3.2.2.5. Let the resulting sequence be denoted by (k_0,p_0), (k_1,p_1), …, (k_{n-1},p_{n-1}), where n is the total number of usable hop-ports assigned to this packet in the f'th PHY Frame of transmission. The notation for a hop-port is as in 9.3.2.2.5, i.e., k_i denotes the CarrierIndex and p_i denotes the hop-port index in that carrier.

4. Let n_{sc} be the subcarrier index corresponding to the hop-port p_i in the j'th OFDM symbol in the f'th RL PHY Frame of transmission. If n_{sc} is not assigned to the R-ACKCH of the RLSS and if (k_i,p_i) is not a DPICH hop-port, then a QPSK modulation symbol s is generated by the modulator according to the procedure described in 9.2.6. This modulation symbol shall be modulated with energy PN_{FFT}/n on hop-port (k_i,p_i). That is, the value of the corresponding subcarrier shall be $\sqrt{PN_{FFT}/n}\,s$, where P is the assigned power to this erasure sequence (generated by the RTC MAC Protocol).

5. Increment i. If i = n, or if i = $N_{MaxErasureHopports,R}$, increment j and set i = 0.

6. Increment f and set j = 0 if any of the following two conditions is satisfied:
   - If this is an RL PHY Frame with index 0 within the superframe and the duplexing mode is FDD, and if j = $N_{FRAME,R} + N_{PREAMBLE}$.
   - For any other RL PHY Frame (including all RL PHY Frames in TDD mode), if j = $N_{FRAME,R}$.

7. If the last RL PHY Frame of transmission has been completed (as determined by the RTC MAC Protocol), then stop. Else repeat steps 3 through 7.

9.4.1.6.3 Sector-specific scrambling

Each OFDM symbol in the superframe shall be scrambled by a sector-specific scrambling sequence. The scrambling operation shall be performed independently on each carrier. The rest of this section describes the scrambling operation for the carrier k, where k=0,1,…, $N_{CARRIERS}$ -1. The scrambling sequence for the carrier k consists of a complex number for every subcarrier in the carrier k in every OFDM symbol in the superframe. The scrambling operation shall consist of multiplying the unscrambled complex symbol on each subcarrier by the corresponding entry in the scrambling sequence, unless the subcarrier is allocated to the control segment, or both conditions (a) and (b) are true: (a) The subcarrier corresponds to a R-DPICH hop-port (via the hop-permutation), and (b) RLDPISectorScramble, which is part of the public data of the Overhead Messages Protocol for carrier k, is set to 0. For subcarriers for which these conditions (a) and (b) are true, the scrambling operation shall consist of leaving the subcarrier unchanged; and a cell-specific scrambling sequence, as described in 9.4.1.6.4, shall be used to scramble the subcarrier. If the subcarrier is allocated to the control segment, the scrambling operation shall consist of leaving the subcarrier unchanged.

Each complex number in the sector-specific scrambling sequence is generated from two bits, denoted by s_I and s_Q, using the following mapping:

1. The bit combination (s_I, s_Q) = (0,0) is mapped to the complex number $\left(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$.
2. The bit combination (s_I, s_Q) = (0,1) is mapped to the complex number $\left(\frac{1}{\sqrt{2}},\frac{-1}{\sqrt{2}}\right)$.
3. The bit combination (s_I, s_Q) = (1,0) is mapped to the complex number $\left(\frac{-1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$.
4. The bit combination (s_I, s_Q) = (1,1) is mapped to the complex number $\left(\frac{-1}{\sqrt{2}},\frac{-1}{\sqrt{2}}\right)$.

The sector-specific scrambling sequence for the carrier k shall be generated using two 20-bit registers, called the I-register and the Q-register, as shown in Figure 115 and Figure 116, respectively. The I-register shall have a generator polynomial $h_I(D) = D^{20} + D^{19} + D^{16} + D^{14} + 1$ i.e., the n'th output I(n) of the register shall satisfy $I(n) = I(n-20) \oplus I(n-19) \oplus I(n-16) \oplus I(n-14)$. The Q-register shall have a generator polynomial $h_Q(D) = D^{20} + D^{18} + D^{15} + D^{14} + 1$ i.e., the n'th output Q(n) of the register shall satisfy $Q(n) = Q(n-20) \oplus Q(n-18) \oplus Q(n-15) \oplus Q(n-14)$. Each entry in the sector-specific scrambling sequence shall be generated using s_I and s_Q bits which are respectively the outputs of the I-register and the Q-register after they have been appropriately initialized and clocked as in the following description.

At the start of every superframe, define PilotPNSectorScramb to be equal to PilotPhase in SemiSynchronous mode and equal to PilotPN in Asynchronous mode. (Thus, for a given sector, PilotPNSectorScramb is fixed in Asynchronous mode, but changes every superframe in SemiSynchronous mode.) Let $p_{11}, p_{10}, \ldots, p_0$ be the 12 bits of (PilotPNSectorScramb+k) mod 4096 for a given superframe, with p_{11} being the MSB and p_0 being the LSB. At the beginning of each superframe, the I and Q registers shall both be initialized to the state $[11111111p_{11}p_{10}p_9p_8p_7p_6p_5p_4p_3p_2p_1p_0]$. The I and Q registers shall then be clocked $N_{CARRIER\_SIZE}$ times for each OFDM symbol in the superframe to generate the s_I and s_Q bits for all of the subcarriers belonging to the carrier k in every OFDM symbol. The i'th entry in the scrambling sequence (generated after i clock periods) is used to scramble the subcarrier with index i mod $N_{CARRIER\_SIZE}$ in the carrier k, in the OFDM symbol with index $\lfloor i / N_{CARRIER\_SIZE} \rfloor$ in the superframe. The outputs of the I and Q registers immediately after their state has been initialized (before they are clocked) shall be used to generate the scrambling sequence entry corresponding to the subcarrier with index 0 in the carrier k in the OFDM symbol with index 0.

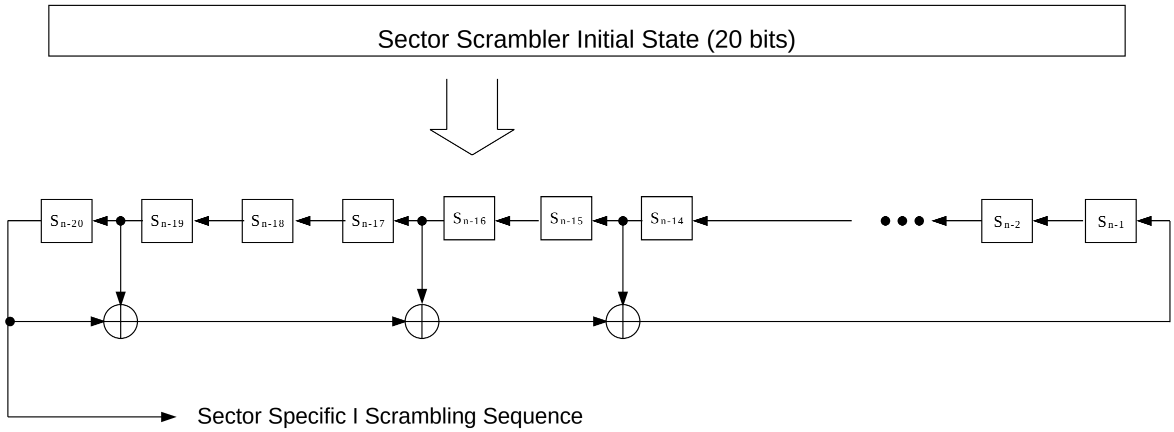

Figure 115 Sector-specific scrambler for the data segments – I sequence

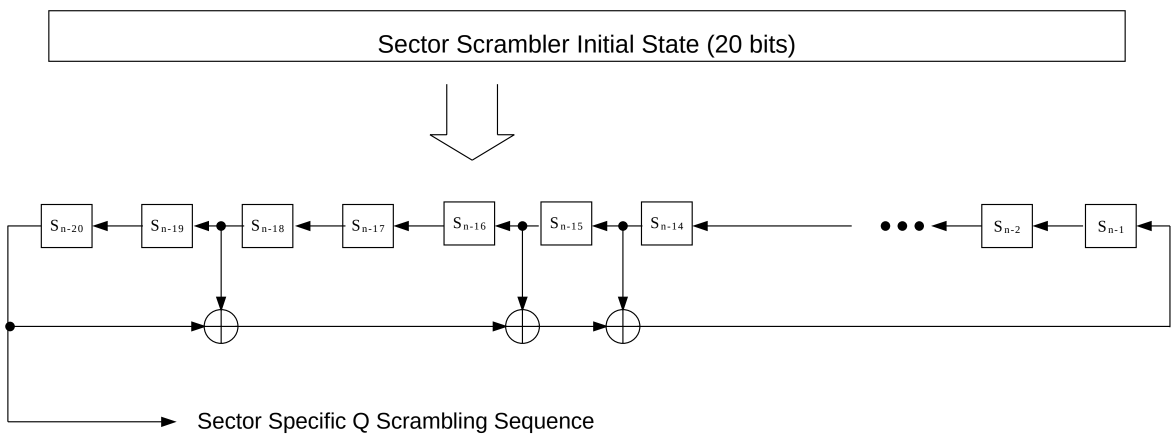

Figure 116 Sector-specific scrambler for the data segments – Q sequence

9.4.1.6.4 Cell-specific scrambling for R-DPICH

The operations in this section shall be carried out independently for each carrier, and shall be described for the carrier with index k, where k=0,1,…, $N_{CARRIERS}$ -1. The operations in this section shall be carried out for the carrier with index k if and only if RLDPISectorScramble, which is part of the public data of the Overhead Messages Protocol for carrier k, is set to 0. A cell-specific scrambling symbol shall be generated for each subcarrier, but only some of the generated scrambling symbols shall be used and the rest shall be discarded. The scrambling symbols that shall be used shall be those generated for subcarriers that correspond to R-DPICH hop-ports (via the hop-permutation), as defined in 9.4.1.6.1. These subcarriers are henceforth referred to as R-DPICH subcarriers. The cell-specific scrambling sequence consists of a complex number for every subcarrier. The scrambling operation shall consist of multiplying the unscrambled complex symbol on each R-DPICH subcarrier by the corresponding entry in the scrambling sequence. Each complex number in the cell-specific scrambling sequence is generated from two bits, denoted by s_I and s_Q, using the following mapping:

1. The bit combination $(s_I, s_Q) = (0,0)$ is mapped to the complex number $\left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$.

2. The bit combination (s_I, s_Q) = (0,1) is mapped to the complex number $\left(\frac{1}{\sqrt{2}},\frac{-1}{\sqrt{2}}\right)$.

3. The bit combination (s_I, s_Q) = (1,0) is mapped to the complex number $\left(\frac{-1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$.

4. The bit combination (s_I, s_Q) = (1,1) is mapped to the complex number $\left(\frac{-1}{\sqrt{2}},\frac{-1}{\sqrt{2}}\right)$.

The cell-specific scrambling sequence for the carrier k shall be generated using two 20-bit registers, called the I-register and the Q-register, as shown in Figure 115 and Figure 116, respectively. The I-register shall have a generator polynomial $h_I(D) = D^{20} + D^{19} + D^{16} + D^{14} + 1$ i.e., the n'th output I(n) of the register shall satisfy $I(n) = I(n-20) \oplus I(n-19) \oplus I(n-16) \oplus I(n-14)$. The Q-register shall have a generator polynomial $h_Q(D) = D^{20} + D^{18} + D^{15} + D^{14} + 1$ i.e., the n'th output Q(n) of the register shall satisfy $Q(n) = Q(n-20) \oplus Q(n-18) \oplus Q(n-15) \oplus Q(n-14)$. Each entry in the cell-specific scrambling sequence shall be generated using s_I and s_Q bits which are respectively the outputs of the I-register and the Q-register after they have been appropriately initialized and clocked as in the following description.

Let CellPilotPN be the 12 bit number obtained from the PilotPN by setting its 5^{th},6^{th} and 7^{th} bits to zero (where the bits are numbered starting from 0, with the 0^{th} bit denoting the LSB). For the superframe with index s, let SFInd be set equal to s in SemiSynchronous mode and set equal to zero in Asynchronous mode. For the superframe with index s, let $b_{11}, b_{10}, \ldots, b_0$ be the 12 bits of (CellPilotPN+SFInd+k) mod 4096, with b_{11} being the MSB and b_0 being the LSB. At the start of the OFDM symbol with index 0 in the superframe, both the I and the Q registers shall be initialized to the state $[11110000b_{11}b_{10}b_9b_8b_7b_6b_5b_4b_3b_2b_1b_0]$. The outputs of the I and Q registers after they are both clocked i times, shall respectively be the s_I and s_Q bits used to generate a symbol c(i) in the scrambling sequence. This symbol c(i) shall be used to scramble the subcarrier with index i mod $N_{CARRIER\_SIZE}$ in the carrier k in the OFDM Symbol with index $\lfloor i / N_{CARRIER\_SIZE} \rfloor$ in the superframe, provided this subcarrier is an R-DPICH subcarrier.

9.4.1.7 Time-domain processing

The sequence of OFDM symbols at the output of the sector scrambler shall be converted to a baseband waveform according to the procedure described in Figure 117. This procedure consists of an Inverse Fourier Transform (IFT) operation, a windowing operation, and an overlap-and-add operation.

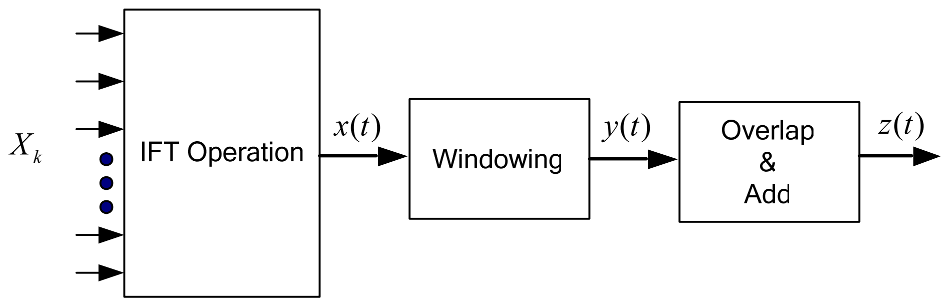

Figure 117 Time-domain processing

9.4.1.7.1 Symbol start time

The start time of the superframe with index i is given by the product of i with the superframe duration $T_{SUPERFRAME}$, where $T_{SUPERFRAME}$ is as defined in 9.4.1.2.4.

In FDD, the start time of the k-th OFDM symbol in the superframe, k ranging from 0 to $N_{PREAMBLE}$ + $N_{FDD,RLPHYFrames}N_{FRAME,R}$ - 1, is given by $T_{START,SF} + kT_{s,PR}$ if k is less than $N_{PREAMBLE}$, and is given by $T_{START,SF} + N_{PREAMBLE}T_{s,PR} + (k - N_{PREAMBLE})T_s$, otherwise. Here $T_{START,SF}$ is the start time of the superframe, T_s and $T_{s,PR}$ are as defined in 9.4.1.2.3, and $N_{FDD,RLPHYFrames}$ is defined by the Lower MAC sublayer.

In TDD, the start time of the k-th OFDM symbol in the superframe, k ranging from 0 to $N_{TDD,RLPHYFrames} * N_{FRAME,R}$ - 1, is given by $T_{START,SF} + N_{PREAMBLE}T_{s,PR} + N_{FL\_BURST}\, N_{FRAME,F}T_s + T_{G,TDD,F} + kT_s + \lfloor k/(N_{RL\_BURST}\, N_{FRAME,R})\rfloor * (N_{FL\_BURST}\, N_{FRAME,F}T_s + T_{G,TDD,F} + T_{G,TDD,R})$, where $T_{START,SF}$ is the start time of the superframe, T_s is as defined in 9.4.1.2.3, and $N_{TDD,RLPHYFrames}$ is defined by the Lower MAC sublayer.

9.4.1.7.2 IFT operation

Let X_k be the value of the complex modulation symbol on the k'th subcarrier of an OFDM symbol, k ranging from 0 to N_{FFT}-1. The IFT of the OFDM symbol is given by the infinite duration signal:

$$x(t) = \frac{1}{\sqrt{N_{FFT}}} \sum_{k=0}^{N_{FFT}-1} X_k e^{j2\pi(k - N_{FFT}/2)(t - T_{CP,PR} - T_{START})/(N_{FFT}T_{CHIP})}$$

during the first $N_{PREAMBLE}$ OFDM symbols in the superframe preamble in FDD, and in all other cases (including all the OFDM symbols for TDD) by the equation:

$$x(t) = \frac{1}{\sqrt{N_{FFT}}} \sum_{k=0}^{N_{FFT}-1} X_k e^{j2\pi(k - N_{FFT}/2)(t - T_{CP} - T_{START})/(N_{FFT}T_{CHIP})}$$

where T_{START} denotes the start time of the OFDM symbol, T_{CP} and $T_{CP,R}$ are as defined in 9.4.1.2.2, and j denotes the complex number (0,1).

9.4.1.7.3 Windowing

The signal x(t) at the output of the IFT shall be multiplied by the window function w(t), where w(t) is given by the equation:

$$w(t) = \begin{cases} 0 & , t < T_{START} - T_{WGI} \\ 0.5 - 0.5\cos\left(\dfrac{\pi(t + T_{WGI} - T_{START})}{T_{WGI}}\right) & , T_{START} - T_{WGI} \le t < T_{START} \\ 1 & , T_{START} \le t < T_{START} + T_{CP,PR} + T_{FFT} \\ 0.5 + 0.5\cos\left(\dfrac{\pi(t - T_{START} - T_{CP,PR} - T_{FFT})}{T_{WGI}}\right) & , T_{START} + T_{CP,PR} + T_{FFT} \le t < T'_{s,PR} \\ 0 & , t \ge T'_{s,PR} \end{cases}$$

during the first $N_{PREAMBLE}$ OFDM symbols in the superframe preamble in FDD, and in all other cases (including all the OFDM symbols for TDD) by the equation:

$$w(t) = \begin{cases} 0 & , t < T_{START} - T_{WGI} \\ 0.5 - 0.5\cos\left(\dfrac{\pi(t + T_{WGI} - T_{START})}{T_{WGI}}\right) & , T_{START} - T_{WGI} \le t < T_{START} \\ 1 & , T_{START} \le t < T_{START} + T_{CP} + T_{FFT} \\ 0.5 + 0.5\cos\left(\dfrac{\pi(t - T_{START} - T_{CP} - T_{FFT})}{T_{WGI}}\right) & , T_{START} + T_{CP} + T_{FFT} \le t < T'_{s} \\ 0 & , t \ge T'_{s} \end{cases}$$

during each RL PHY Frame, where T_{START} denotes the start time of the OFDM symbol. The quantities T_{FFT}, T_s, T_s', $T_{s,PR}$ and $T_{s,PR}$' are as defined in 9.4.1.2.2.

The windowed signal y(t) is given by y(t) = x(t)w(t).

9.4.1.7.4 Overlap and add operation

The windowed IFTs y(t) corresponding to all of the OFDM symbols shall be added together to create the final complex baseband waveform z(t). In this procedure, neighboring OFDM symbols overlap for a duration T_{WGI} as illustrated in Figure 118.

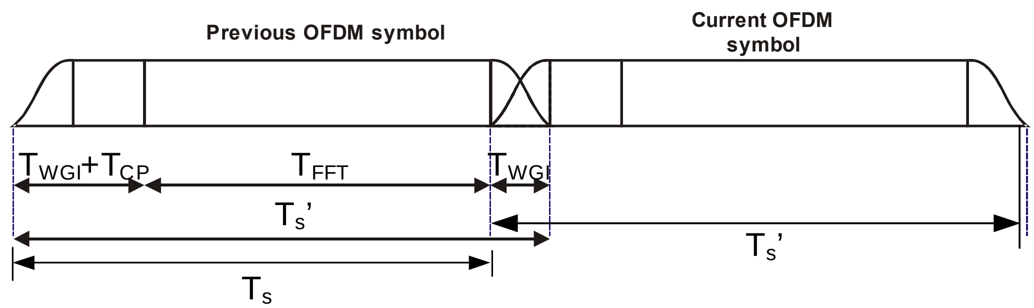

Figure 118 Overlap and add operation

9.4.2 Synchronization and timing

The access terminal shall establish a time reference that is used to derive timing for all time-critical transmission components, including superframe boundaries, PHY Frame boundaries, etc. The access terminal initial time reference shall be established from the acquired Acquisition Channel (F-ACQCH) and from the SystemTimeLSB field transmitted as part of the Primary Broadcast Channel (F-pBCH). The initial access terminal time reference shall coincide with the time of occurrence, as measured at the access terminal antenna connector, of the earliest arriving multipath component of the forward link waveform. To elaborate, the beginning of the reverse link superframe with index i shall coincide with the beginning of the forward link superframe with index i, where the beginning of both superframes are measured at the access terminal antenna connector. The inaccuracy in this time-alignment shall be within $\pm 1 \mu s$.

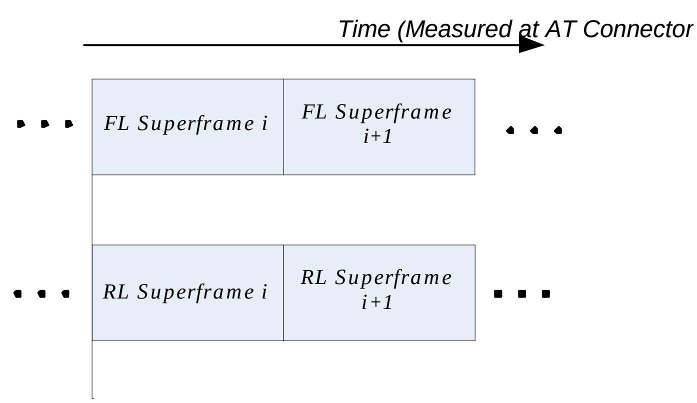

Figure 119 Relationship between forward link and reverse link timings

After the initial time reference has been established, the access terminal shall advance and retard timing in response to the AccessGrant message of the SS MAC Protocol and the TimingCorrection message of this protocol, as specified in the following. The TimingCorrection message shall be declared in error if it contains two SectorIDs within the same synchronous subgroup. Otherwise, the access terminal shall use the TimingCorrection field corresponding to the SectorID in AS_{SYNCH}. If no such SectorID is present, the access terminal shall not retard or advance the timing for that AS_{SYNCH}.

To advance timing by a period of k chips, the access terminal shall move its time reference earlier by a period of kT_c, where T_c is the chip duration. To retard timing by a period of k chips, the access terminal shall move its time reference later by a period of kT_c, where T_c is the chip duration.

If the Reverse Link Serving Sector (RLSS) is contained within AS_{SYNCH}, the access terminal shall also move its time reference when the RLSS changes from one sector to another sector within AS_{SYNCH}. If, at the access terminal connector, the time of arrival of the superframe boundary from the new RLSS is later than the time of arrival of the superframe boundary from the previous RLSS (for the same superframe index), then the access terminal shall move its time reference earlier (i.e., advance its timing) by the difference in the times of arrival between the two sectors. If the time of arrival of the superframe boundary from the new RLSS is earlier than the time of arrival of the superframe boundary from the previous RLSS (for the same superframe index), then the access terminal shall move its time reference later (i.e., retard its timing) by the difference in the times of arrival between the two sectors.[87]

The access terminal shall maintain independent time references for transmission to asynchronous APs.

[87] This ensures that the time of arrival of the access terminal signal at the RLSS remains unchanged, even though the RLSS itself changes. The assumption is that the forward link propagation delay is the same as the reverse link propagation delay.

10 Common Algorithms and Data Structures

10.1 Channel band record

The channel band record defines an access network (AN) channel frequency and the system bandwidth on that frequency. This record contains the following fields:

Field	Length (bits)
ChannelBandRecordType	1
ReverseLinkCenterFrequencyIncluded	1
ForwardLinkSystemBandwidthIncluded	1
ReverseLinkSystemBandwidthIncluded	1
ChannelBandRecordLength	4
ChannelBandRecord	ChannelBandRecordLength× 8

ChannelBandRecordType

The access network shall set this field to an identifier for the channel band record type according to Table 101.

Table 101 ChannelBandRecordType for channel band record

Value	ChannelBandRecordType
‘0’	Band Class
‘1’	Frequency Specified

ReverseLinkCenterFrequencyIncluded

The access network shall set this field to ‘0’ if the value of the ChannelBandRecordType is ‘0’, or if the value of the center frequency of the reverse link channel is not included in the ChannelBandRecord. Otherwise, the access network shall set this field to ‘1’.

ForwardLinkSystemBandwidthIncluded

The access network shall set this field to ‘0’ if the value of the ChannelBandRecordType is ‘0’, or if the value of the system bandwidth of the forward link channel is not included in the ChannelBandRecord. Otherwise, the access network shall set this field to ‘1’.

ReverseLinkSystemBandwidthIncluded

The access network shall set this field to ‘0’ if the value of the ChannelBandRecordType is ‘0’, if the value of the system bandwidth of the reverse link channel is the same as the value of the system bandwidth of the forward link channel, or if the value of the system bandwidth of the reverse link channel is not included in the ChannelBandRecord. Otherwise, the access network shall set this field to ‘1’.

ChannelBandRecordLength
The access network shall set this field to the length of the ChannelBandRecord field in units of octets.

ChannelBandRecord If ChannelBandRecordType is ‘0’, then the access network shall set this record as defined in 10.1.1. If ChannelBandRecordType is ‘1’, then the sender shall set this record as defined in 10.1.2.

10.1.1 Definition of ChannelBandRecord record for Band Class

Field	Length (bits)
BandClass	5
ChannelBandNumber	11

BandClass The access network shall set this field to the band class number corresponding to the frequency assignment of the channel specified by this record.

ChannelBandNumber The access network shall set this field to the channel number corresponding to the frequency assignment of the channel specified by this record.

10.1.2 Definition of ChannelBandRecord record for Frequency Specified

Field	Length (bits)
ForwardLinkCenterFrequency	24
ReverseLinkCenterFrequency	0 or 24
ForwardLinkSystemBandwidth	0 or 16
ReverseLinkSystemBandwidth	0 or 16

ForwardLinkCenterFrequency
The access network shall set this field to the value of the center frequency of the forward link channel in units of kHz.

ReverseLinkCenterFrequency
If ReverseLinkCenterFrequencyIncluded is ‘0’, then the access network shall omit this field. Otherwise, the access network shall set this field to the value of the center frequency of the reverse link channel in units of kHz.

ForwardLinkSystemBandwidth
If ForwardLinkSystemBandwidthIncluded is ‘0’, then the access network shall omit this field. Otherwise, the access network shall set this field to the value of the system bandwidth of the forward link channel in units of kHz.

ReverseLinkSystemBandwidth
If ReverseLinkSystemBandwidthIncluded is ‘0’, then the access network shall omit this field. Otherwise, the access network shall set this field to the value of the system bandwidth of the reverse link channel in units of kHz.

10.2 Access terminal identifier record

The Access Terminal Identifier record provides a unicast, multicast, or broadcast access terminal (AT) address. This record contains the following fields:

Field	Length (bits)
ATIType	2
ATI	0 or 32

ATIType Access Terminal Identifier Type. This field shall be set to the type of the ATI, as shown in Table 102.

ATI Access Terminal Identifier. The field is included only if ATIType is not equal to ‘00’. This field shall be set as shown in Table 102.

Table 102 ATIType field encoding

ATIType	ATIType Description	ATI Length (bits)
‘00’	Broadcast ATI (BATI)	0
‘01’	Multicast ATI (MATI)	32
‘10’	Unicast ATI	32
‘11’	Reserved	N/A

10.3 Attribute record

The attribute record defines a value for a given attribute.

An attribute can be one of the following two types:

- Simple attribute if it contains a single value.
- Complex attribute if it contains multiple values that together form a complex value for a particular attribute identifier.

The type of the attribute is determined by the attribute identifier.

The format of a simple attribute is given by:

Field	Length (bits)
Length	8
AttributeID	Protocol Specific
AttributeValue	Attribute dependent
Reserved	Variable

Length Length in octets of the attribute record, excluding the Length field.

AttributeID	Attribute identifiers are unique in the context of the protocol being configured.
AttributeValue	A suggested value for the attribute. In general, attribute value lengths are an integer number of octets. Attribute values have an explicit or implicit length indication (e.g., fixed length or null terminated strings) so that the recipient can successfully parse the record.
Reserved	The length of this field is the smallest value that will make the attribute record octet aligned. The sender shall set this field to zero. The receiver shall ignore this field.

The format of a complex attribute is given by:

Field	Length (bits)
Length	8
AttributeID	Protocol Specific
An appropriate number of instances of the following record:	
AttributeValue	Attribute dependent
Reserved	Variable

Length	Length in octets of the attribute record, excluding the Length field.
AttributeID	Attribute identifiers are unique in the context of the protocol being configured.
AttributeValue	A suggested value for the attribute. In general, attribute value lengths are an integer number of octets. Attribute values have an explicit or implicit length indication (e.g., fixed length or null terminated strings) so that the recipient can successfully parse the record.
Reserved	The length of this field is the smallest value that will make the attribute record octet aligned. The sender shall set this field to zero. The receiver shall ignore this field.

10.4 Hash function

The hash function takes three arguments:

Key	This argument shall be 32 bits; typically, the access terminal's ATI.
N	The number of resources.
Decorrelate	An argument used to de-correlate values obtained for different applications for the same access terminal.

Define:

- Word *L* to be bits 0-15 of *Key*
- Word *H* to be bits 16-31 of *Key*

where bit 0 is the least significant bit of *Key*.

The hash value is computed as follows[88]:

$$R(L, H, N, \text{Decorrelate}) = \lfloor N \times ((40503 \times (L \oplus H \oplus \text{Decorrelate})) \bmod 2^{16}) / 2^{16} \rfloor.$$

10.5 Computation of the CRC bits

This section describes the computation of CRC bits for a stream of data bits $\{b_{N-1}, b_{N-2}, \ldots b_1, b_0\}$. Here b_0 is the initial bit in the stream and b_{N-1} is the final bit.

The CRC shall be calculated using the standard CRC-CCITT generator polynomial:

$$g(x) = x^{24} + x^{23} + x^6 + x^5 + x + 1$$

The CRC shall be equal to the value computed according to the following procedure as shown in Figure 120:

- All shift-register elements shall be initialized to ‘0’s.
- The register shall be clocked once for each data bit. The bit stream shall be read starting from b_0 and ending in b_{N-1}.
- When all the data bits are exhausted, the values remaining in the shift registers $\{c_0, c_1, \ldots c_{23}\}$ constitute the 24-bit CRC. The data stream padded with a 24-bit CRC shall be $\{c_0, c_1, \ldots c_{22}, c_{23}, b_{N-1}, b_{N-2}, \ldots b_1, b_0\}$ i.e., b_0 is the initial bit and c_0 is the last bit in the padded stream.
- If less than a 24-bit CRC is to be used, the most significant bits of the 24-bit CRC shall be used. The data stream padded with a M-bit CRC (M<24) CRC shall be $\{c_{24-M}, c_{24-M+1}, \ldots c_{23}, b_{N-1}, b_{N-2}, \ldots b_1, b_0\}$ i.e., b_0 is the initial bit and c_{24-M} is the last bit in the padded stream.

[88] This formula is adapted from Knuth, D. N., *Sorting and Searching*, vol. 3 of *The Art of Computer Programming,* 3 vols. (Reading, MA: Addison-Wesley, 1973), pp. 508-513. The symbol ⊕ represents bitwise exclusive-or function (or modulo 2 addition) and the symbol ⌊ ⌋ represents the “largest integer smaller than” function.

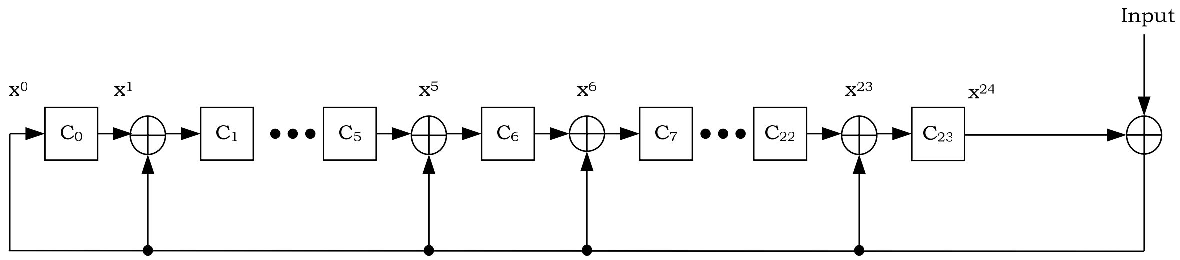

Figure 120 CRC calculation

10.6 Pseudorandom number generator

10.6.1 General procedures

When an access terminal is required to use the pseudo random number generator described in this section, then the access terminal shall implement the linear congruential generator defined by:

$$z_n = a \times z_{n-1} \bmod m$$

where $a = 7^5 = 16807$ and $m = 2^{31} - 1 = 2147483647$. z_n is the output of the generator.[89]

The access terminal shall initialize the random number generator as defined in 10.6.2.

The access terminal shall compute a new z_n for each subsequent use.

The access terminal shall use the value $u_n = z_n / m$ for those applications that require a binary fraction u_n, $0 < u_n < 1$.

The access terminal shall use the value $k_n = \lfloor N \times z_n / m \rfloor$ for those applications that require a small integer k_n, $0 \le k_n \le N-1$.

10.6.2 Initialization

The access terminal shall initialize the random number generator by setting z_0 to

$$z_0 = (\text{HardwareID} \oplus \chi) \bmod m$$

where HardwareID is the least 32 bits of the hardware identifier associated with the access terminal, and χ is a 32-bit, time-varying physical measure available to the access terminal. If the initial value so produced is found to be zero, the access terminal shall repeat the procedure with a different value of χ.

[89] This generator has full period, ranging over all integers from 1 to m-1; the values 0 and m are never produced. Several suitable implementations can be found in Park, Stephen K. and Miller, Keith W., “Random Number Generators: Good Ones are Hard to Find,” *Communications of the ACM*, vol. 31, no. 10, October 1988, pp. 1192-1201.

10.7 Sequence number

When the order in which protocol messages are delivered is important, air interface protocols use a sequence number to verify this order.

The sequence number has *s* bits. The sequence space is 2^S. All operations and comparisons performed on sequence numbers shall be carried out in unsigned modulo 2^S arithmetic. For any message sequence number *N*, the sequence numbers in the range $[N+1, N+2^{S-1}-1]$ shall be considered greater than *N*, and the sequence numbers in the range $[N-2^{S-1}, N-1]$ shall be considered smaller than *N*.

10.7.1 Sequence number initialization

Upon entering into the initial state, the sequence number on the sender side shall be initialized to 0. The sequence number on the receiver, V(R), shall be initialized to 2^s-1.

10.7.2 Sequence number validation

The receiver of the message maintains a receive pointer *V(R)* whose initialization is defined in 10.7.1. When a message arrives, the receiver compares the sequence number of the message with *V(R)*. If the sequence number is greater than *V(R)*, the message is considered a valid message and *V(R)* is set to this sequence number. Otherwise, the message is considered an invalid message.

10.8 TransactionID number

A TransactionID is used in some protocol messages to associate a response message with an initiating message. The initiator sets the value of the TransactionID field in the message. The responder uses the same TransactionID in the response message to associate the response message with the corresponding initiating message. A TransactionID shall not be used to order protocol messages at the responder.

10.8.1 TransactionID initiating procedures

The initiator shall maintain a separate list of the TransactionIDs for each initiating message for which it is expecting a response message. The initiator shall set the TransactionID field for each message sent to a value which shall meet the following requirements:

- The initiator is not expecting the TransactionID in a response message.
- The TransactionID has not been used in the last 8 response messages received for that initiating message.

10.9 Generic Attribute Update Protocol

10.9.1 Introduction

The Generic Attribute Update Protocol provides a means to update and query the values of protocol attributes. The protocol defines an AttributeUpdateRequest message, an AttributeUpdateAccept message, and an AttributeUpdateReject message to negotiate a mutually acceptable configuration. The protocol defines an AttributeValueRequest message and an AttributeValueResponse message to query the value of an attribute.

The initiator uses the AttributeUpdateRequest message to provide the responder with a proposed value for each attribute. The responder uses the AttributeUpdateAccept message to accept the proposed values. If the responder is an access network and if any of the attribute values in the received AttributeUpdateRequest message is not acceptable to it, then the access network sends the AttributeUpdateReject message, and the access terminal and access network continue to use the previously negotiated values for the attributes.

The access terminal shall not send an AttributeUpdateReject message.

After receiving an AttributeValueRequest message, the receiver shall send an AttributeValueResponse message within time $T_{Turnaround}$, unless specified otherwise by the protocol which uses the Generic Attribute Update Protocol.

10.9.2 Procedures

10.9.2.1 Initiator requirements

The initiator shall not send an AttributeUpdateRequest message if the value of the ConfigurationLock public data of the Session Configuration Protocol is Locked.

The initiator shall include one attribute value for each attribute included in the AttributeUpdateRequest message.

After sending an AttributeUpdateRequest message, the initiator should continue to use previously negotiated values for attributes listed in the message until it receives either an AttributeUpdateAccept message or an AttributeUpdateReject message. However, the initiator should be prepared for the responder to begin using attribute values proposed by the initiator in the AttributeUpdateRequest message.

If the initiator receives an AttributeUpdateAccept message, then it shall pair the received message with the associated AttributeUpdateRequest message using the TransactionID field of the messages. The initiator shall use the attribute values in the AttributeUpdateRequest message as the configured attribute values. If the access terminal receives an AttributeUpdateReject message, then it shall use the previously configured values of the attributes included in the corresponding AttributeUpdateRequest message.

If the initiator does not receive the corresponding AttributeUpdateAccept or AttributeUpdateReject message in response to the AttributeUpdateRequest message, it should re-transmit the AttributeUpdateRequest message.

While the initiator is waiting for a response to an AttributeUpdateRequest message, it shall not transmit another AttributeUpdateRequest message with a different TransactionID field that requests reconfiguration of an attribute included in the original AttributeUpdateRequest message.

10.9.2.2 Responder requirements

After receiving an AttributeUpdateRequest message, the responder shall respond within time $T_{Turnaround}$, unless specified otherwise by the protocol which uses the Generic Attribute Update Protocol.

If the responder is an access terminal, then:

- The responder shall send an AttributeUpdateAccept message.
- Upon sending an AttributeUpdateAccept message, the responder shall begin using the accepted attribute values.

If the responder is an access network, then:

- If the responder finds the proposed value for each attribute in the AttributeUpdateRequest message to be acceptable, then the responder shall send an AttributeUpdateAccept message. Upon sending an AttributeUpdateAccept message, the responder shall begin using the accepted attribute values.
- If the responder does not recognize an attribute or does not find a proposed attribute value to be acceptable, then it shall send an AttributeUpdateReject message.
- If the responder sends an AttributeUpdateReject message, then it shall continue to use the previously configured values of the attributes found in the corresponding AttributeUpdateRequest message.

10.9.3 Message formats

No protocol or transport shall define a message with the same MessageID value as the AttributeUpdateRequest, AttributeUpdateAccept, AttributeUpdateReject, AttributeValueRequest, AttributeValueResponse messages of the Generic Attribute Update Protocol.

10.9.3.1 AttributeUpdateRequest

The sender sends an AttributeUpdateRequest message to offer an attribute-value for a given attribute.

Field	Length (bits)
MessageID	8
TransactionID	8
One or more instances of the following record	
AttributeRecord	Attribute dependent

MessageID: The sender shall set this field to 0x52.The value of this field is the same for all protocols using this message.

TransactionID: The sender shall set this field according to 10.8.

AttributeRecord: The format of this record is specified in 10.3.

Channels	FTC RTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

10.9.3.2 AttributeUpdateAccept

The sender sends an AttributeUpdateAccept message in response to an AttributeUpdateRequest message to accept the offered attribute value.

Field	Length (bits)
MessageID	8
TransactionID	8

MessageID — The sender shall set this field to 0x53.The value of this field is the same for all protocols using this message.

TransactionID — The sender shall set this value to the TransactionID field of the corresponding AttributeUpdateRequest message.

Channels	FTC RTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

10.9.3.3 AttributeUpdateReject

The access network sends an AttributeUpdateReject message in response to an AttributeUpdateRequest message to reject the offered attribute value.

Field	Length (bits)
MessageID	Protocol dependent
TransactionID	8

MessageID — The sender shall set this field to 0x54.The value of this field is the same for all protocols using this message.

TransactionID — The sender shall set this value to the TransactionID field of the corresponding AttributeUpdateRequest message.

Channels	FTC		**SLP**	Reliable
Addressing	Unicast		**Security**	Required

10.9.3.4 AttributeValueRequest

The sender sends an AttributeValueRequest message to request the value of an attribute.

Field	Length (bits)
MessageID	8
TransactionID	8
AttributeCount	8

AttributeCount instances of the following field:

{

AttributeID	16

}

MessageID — The sender shall set this field to 0x55.

TransactionID — The sender shall set this field according to 10.8.

AttributeCount — The sender shall set this field to the number of AttributeID fields included in this message.

AttributeID — The sender shall set this field to the AttributeID for which this request is generated.

Channels	FTC RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

10.9.3.5 AttributeValueResponse

The sender sends an AttributeValueResponse message in response to an AttributeValueRequest message.

Field	Length (bits)
MessageID	8
TransactionID	8
AttributeCount	8

AttributeCount instances of the following field:

{

AttributeRecord	Attribute dependent

}

MessageID — The sender shall set this field to 0x56.

TransactionID — The sender shall set this value to the TransactionID field of the corresponding AttributeValueRequest message.

AttributeCount	The sender shall set this field to the number of AttributeID fields included in this message.
AttributeRecord	The format of this record is specified in 10.3.

Channels	FTC RTC	**SLP**	Reliable
Addressing	Unicast	**Security**	Required

10.9.4 Protocol numeric constants

Constant	Meaning	Value
$T_{Turnaround}$	Maximum time to respond to an AttributeUpdateRequest message.	2 sec

10.10 Session state information record

The Session State Information is to be used for transferring the session parameters corresponding to the InUse and Suspended protocol and transport instances from a source access network to a target access network. Session parameters are the attributes and the internal parameters that define the state of each protocol. The format of this record is shown in Table 103. If an attribute is not contained in the Session State Information record, the target access network shall assume that the missing attributes have the default values (specified for each attribute in each protocol). The sender shall include all of the Parameter Records associated with the ProtocolType and ProtocolSubtype in the same Session State Information Record.

Table 103 Format of the session state information record

Field	Length (bits)
FormatID	8
ProtocolType	8
ProtocolSubtype	16
One or more instances of the following Parameter Record:	
ParameterType	8
ParameterType-specific record	Variable

FormatID	This field identifies the format of the rest of the fields in this record and shall be set to 0x00.
ProtocolType	This field shall be set the Type value (see Table 9) for the protocol associated with the encapsulated session parameters.
ProtocolSubtype	This field shall be set to the protocol subtype value (see Table 105) for the protocol associated with the encapsulated session parameters.
ParameterType	This field shall be set according to Table 104.

Table 104 Encoding of the DataType field

Field Value	Meaning
0x00	The ParameterType-specific record consists of a Complex or a Simple Attribute as defined in 10.3.
All other values	ParameterType-specific records are protocol dependent.

ParameterType-specific record

If the ParameterType field is set to 0x00, then this record shall be set to the simple or complex attribute (see 10.3) associated with the protocol identified by the (ProtocolType, ProtocolSubtype) pair. Otherwise, the structure of this record shall be as specified by the protocol which is identified by the (ProtocolType, ProtocolSubtype) pair.

10.11 SectorID provisioning

The SectorID may have the format of an IPv6 address from one of the following four address pools: Global Unicast, Unique-Local Unicast, Link-Local Unicast, and Reserved defined in RFC 3513 [21] and RFC 4193 [23].

10.11.1 SectorID construction

The access network shall construct the SectorID to be either a Globally Unique SectorID or a Locally Unique SectorID as described in the following:

- If a Globally Unique SectorID is used, the SectorID is universally unique by construction.
- If a Locally Unique SectorID is used, it is the responsibility of the access network to ensure the uniqueness of the SectorID throughout the access networks that the access terminal can visit while maintaining the session.

The methods for constructing the SectorID are beyond the scope of this specification.

11 Assigned Names and Numbers

11.1 Protocols

Table 105 shows the Protocol Types and Protocol Subtypes assigned to the protocols defined in this specification.

Table 105 Protocol types and subtypes

Protocol Type		Protocol Subtype	
Name	**ID**	**Name**	**ID**
Physical Layer	0x00	Default Physical Layer	0x0000
Control Channel MAC	0x01	Default Control Channel MAC	0x0000
Access Channel MAC	0x02	Default Access Channel MAC	0x0000
Forward Traffic Channel MAC	0x03	Default Forward Traffic Channel MAC	0x0000
Reverse Traffic Channel MAC	0x04	Default Reverse Traffic Channel MAC	0x0000
Reverse Control Channel MAC	0x05	Default Reverse Control Channel MAC	0x0000
Shared Signaling MAC	0x06	Default Shared Signaling MAC	0x0000
Air Link Management	0x07	Default Air Link Management	0x0000
Initialization State	0x08	Default Initialization State	0x0000
Idle State	0x09	Default Idle State	0x0000
Connected State	0x0a	Default Connected State	0x0000
Active Set Management	0x0b	Default Active Set Management	0x0000
Overhead Messages	0x0c	Overhead Messages	0x0000
Authentication	0x0d	Default Authentication	0x0000
Encryption	0x0e	Default Encryption	0x0000
Encryption	0x0e	Generic Encryption	0x0001
Security	0x0f	Default Security	0x0000
Key Exchange	0x10	Default Key Exchange	0x0000
Session Management	0x11	Default Session Management	0x0000
Address Management	0x12	Default Address Management	0x0000
Session Configuration	0x13	Default Session Configuration	0x0000
Capabilities Discovery	0x14	Default Capabilities Discovery	0x0000
Inter RAT Protocol	0x15	Default Inter RAT Protocol	0x0000
Packet Consolidation	0x16	Default Packet Consolidation	0x0000
Transport 0	0x17	Transport Subtype per 11.2	See 11.2
Transport 1	0x18	Transport Subtype per 11.2	See 11.2
Transport 2	0x19	Transport Subtype per 11.2	See 11.2
Transport 3	0x1a	Transport Subtype per 11.2	See 11.2

Protocol Type		Protocol Subtype	
Name	ID	Name	ID
Transport 4	0x1b	Transport Subtype per 11.2	See 11.2
Transport 5	0x1c	Transport Subtype per 11.2	See 11.2
Transport 6	0x1d	Transport Subtype per 11.2	See 11.2
Transport 7	0x1e	Transport Subtype per 11.2	See 11.2

11.2 Transport subtype assignments

A transport subtype identifies the transport that is bound to a Transport of the Packet Consolidation Protocol. Table 106 shows the transport subtype defined in this specification.

Table 106 Transport subtypes assignments

Value	Name
0x0000	Default Signaling Transport
0x0001	Default Data Transport
0x0002	Test Transport
0xffff	Transport not used
All other values reserved	

11.3 Messages

Protocol/Transport		Message	
Subtype Name	Type ID	Name	ID
Default Data	0x18 – 0x1e	ActivateRoute	0x02
Default Data	0x18 – 0x1e	ActivateRouteAck	0x03
Default Active Set Management	0x0b	ActiveSetAssignment	0x03
Default Active Set Management	0x0b	ActiveSetComplete	0x04
Default Key Exchange	0x10	ANKeyComplete	0x03
Default Key Exchange	0x10	ATKeyComplete	0x04
All subtypes with configurable attributes	N/A	AttributeUpdateAccept	0x53
All subtypes with configurable attributes	N/A	AttributeUpdateReject	0x54
All subtypes with configurable attributes	N/A	AttributeUpdateRequest	0x52
All subtypes with configurable attributes	N/A	AttributeValueRequest	0x55
All subtypes with configurable attributes	N/A	AttributeValueResponse	0x56
Default Capabilities Discovery	0x14	CapabilitiesRequest	0x00
Default Capabilities Discovery	0x14	CapabilitiesResponse	0x01
Default Connected State	0x0a	ChannelMeasurementReport	0x08

Protocol/Transport		Message	
Subtype Name	Type ID	Name	ID
Default Connected State	0x0a	ChannelMeasurementReport Request	0x07
Default Session Configuration	0x13	ConfigurationAccept	0x0a
Default Session Configuration	0x13	ConfigurationReject	0x0b
Default Session Configuration	0x13	ConfigurationRequest	0x09
Default Connected State	0x0a	ConnectionClose	0x00
Default Idle State	0x09	ConnectionOpenRequest	0x00
Default Idle State	0x09	ConnectionOpenResponse	0x01
Overhead Messages	0x0c	EncapsulatedQuickChannelI nfo	0x02
Overhead Messages	0x0c	ExtendedChannelInfo	0x00
Default Data	0x18 – 0x1e	FlowQoSDetect	0x0d
Default Data	0x18 – 0x1e	FwdReservationAck	0x0c
Default Data	0x18 – 0x1e	FwdReservationOff	0x0b
Default Data	0x18 – 0x1e	FwdReservationOn	0x0a
Default Address Management	0x12	HardwareIDRequest	0x03
Default Address Management	0x12	HardwareIDResponse	0x04
Default Inter RAT	0x15	InterRATBlob	0x00
Default Inter RAT	0x15	InterRATIDRequest	0x01
Default Inter RAT	0x15	InterRATIDResponse	0x02
Default Session Management	0x11	KeepAliveRequest	0x02
Default Session Management	0x11	KeepAliveResponse	0x03
Default Key Exchange	0x10	KeyChangeAck	0x07
Default Key Exchange	0x10	KeyChangeInitiateRequest	0x05
Default Key Exchange	0x10	KeyChangeRequest	0x06
Default Key Exchange	0x10	KeyInitiateRequest	0x00
Default Key Exchange	0x10	KeyRequest	0x01
Default Key Exchange	0x10	KeyResponse	0x02
Default Session Configuration	0x13	LockConfiguration	0x05
Default Session Configuration	0x13	LockConfigurationAck	0x06
Default Connected State	0x0a	MIMORequest	0x01
Default Reverse Traffic Channel MAC	0x04	OSIReport	0x01
Default Reverse Traffic Channel MAC	0x04	OSIReportRequest	0x00
Default Idle State	0x09	PageUATI	0x02
Default Active Set Management	0x0b	PilotReport	0x00
Default Active Set Management	0x0b	PilotReportRequest	0x06
Default Idle State	0x09	PreferredChannelRequest	0x03
Default Signaling	0x17	ReceiverStatus	0x04

Protocol/Transport		Message	
Subtype Name	**Type ID**	**Name**	**ID**
Default Air Link Management	0x07	Redirect	0x00
Default Data	0x18 – 0x1e	ReservationAccept	0x06
Default Data	0x18 – 0x1e	ReservationOffRequest	0x05
Default Data	0x18 – 0x1e	ReservationOnRequest	0x04
Default Data	0x18 – 0x1e	ReservationReject	0x07
Default Active Set Management	0x0b	ResetReport	0x05
Default Signaling	0x17	ResetRxAck	0x01
Default Signaling	0x17	ResetRxRequest	0x00
Default Signaling	0x17	ResetTxAck	0x03
Default Signaling	0x17	ResetTxIndication	0x02
Default Data	0x18 – 0x1e	RestartNetworkInterface	0x0e
Default Data	0x18 – 0x1e	RestartNetworkInterfaceAck	0x0f
Default Data	0x18 – 0x1e	RevReservationOff	0x09
Default Data	0x18 – 0x1e	RevReservationOn	0x08
Default Data	0x18 – 0x1e	RouteSelect	0x00
Default Data	0x18 – 0x1e	RouteSelectAck	0x01
Overhead Messages	0x0c	SectorParameters	0x01
Default Connected State	0x0a	SelectedInterlaceAck	0x04
Default Connected State	0x0a	SelectedInterlaceAssignment	0x03
Default Connected State	0x0a	SelectedInterlaceRequest	0x02
Default Session Management	0x11	SessionClose	0x01
Default Session Management	0x11	SessionOpen	0x00
Default Physical Layer	0x00	TimingCorrection	0x02
Default Session Configuration	0x13	TokenAssignment	0x03
Default Session Configuration	0x13	TokenComplete	0x04
Default Session Configuration	0x13	TokensSupportedRequest	0x00
Default Session Configuration	0x13	TokensSupportedResponse	0x01
Default Session Configuration	0x13	TokenUpdateRequest	0x02
Default Connected State	0x0a	TuneAwayRequest	0x05
Default Connected State	0x0a	TuneAwayResponse	0x06
Default Address Management	0x12	UATIAssignment	0x01
Default Address Management	0x12	UATIComplete	0x02
Default Address Management	0x12	UATIUpdateRequest	0x00
Default Session Configuration	0x13	UnlockConfiguration	0x07
Default Session Configuration	0x13	UnlockConfigurationAck	0x08
Default Active Set Management	0x0b	VCQIReportMIMO	0x02
Default Active Set Management	0x0b	VCQIReportSISO	0x01

11.4 Other RAT Types

A RAT Type ID identifies the type of the radio access technology. Table 107 shows the RAT types defined in this specification.

Table 107 RAT Types and IDs

RAT Type ID	RAT Type	RAT ID
0x00	The technology specified in this document	UATI as specifiedin 2.3
0x01	L2TP	N/A
0x02	802.11	48-bit extended unique identifier (EUI-48)
0x03	CDMA2000 1x	IMSI[90]
0x04	CDMA2000 1xEV	UATI[91]
0x05	GSM	IMSI[92]
0x06	WCDMA	IMSI[92]
All other values	Reserved	N/A

11.5 Session Configuration Tokens

11.5.1 SessionConfigurationToken 0x0000

Table 108 shows the Protocol Types and Subtypes for SessionConfigurationToken 0x0000. All attributes for SessionConfigurationToken 0x0000 shall be set to the default values defined by the protocol or transport Subtype.

Table 108 Protocol types and subtypes

Protocol Type		Protocol Subtype	
Name	ID	Name	ID
Physical Layer	0x00	Default Physical Layer	0x0000
Control Channel MAC	0x01	Default Control Channel MAC	0x0000
Access Channel MAC	0x02	Default Access Channel MAC	0x0000
Forward Traffic Channel MAC	0x03	Default Forward Traffic Channel MAC	0x0000
Reverse Traffic Channel MAC	0x04	Default Reverse Traffic Channel MAC	0x0000
Reverse Control Channel MAC	0x05	Default Reverse Control Channel MAC	0x0000
Shared Signaling MAC	0x06	Default Shared Signaling MAC	0x0000
Air Link Management	0x07	Default Air Link Management	0x0000

[90] 3GPP2 C.S0005-D v1.0, Upper Layer (Layer 3) Signaling Specification for cdma2000® Spread Spectrum Systems.
[91] 3GPP2 C.S0024-A v2.0, High Rate Packet Data Air Interface Specification.
[92] 3GPP TS 23.003: Numbering, addressing and identification.

Protocol Type		Protocol Subtype	
Name	**ID**	**Name**	**ID**
Initialization State	0x08	Default Initialization State	0x0000
Idle State	0x09	Default Idle State	0x0000
Connected State	0x0a	Default Connected State	0x0000
Active Set Management	0x0b	Default Active Set Management	0x0000
Overhead Messages	0x0c	Overhead Messages	0x0000
Authentication	0x0d	Default Authentication	0x0000
Encryption	0x0e	Default Encryption	0x0000
Security	0x0f	Default Security	0x0000
Key Exchange	0x10	Default Key Exchange	0x0000
Session Management	0x11	Default Session Management	0x0000
Address Management	0x12	Default Address Management	0x0000
Session Configuration	0x13	Default Session Configuration	0x0000
Capabilities Discovery	0x14	Default Capabilities Discovery	0x0000
Inter RAT Protocol	0x15	Default Inter RAT Protocol	0x0000
Packet Consolidation	0x16	Default Packet Consolidation	0x0000
Transport 0	0x17	Default Signaling Transport	0x0000
Transport 1	0x18	Default Data Transport	0x0001
Transport 2	0x19	Transport not used	0xffff
Transport 3	0x1a	Transport not used	0xffff
Transport 4	0x1b	Transport not used	0xffff
Transport 5	0x1c	Transport not used	0xffff
Transport 6	0x1d	Transport not used	0xffff
Transport 7	0x1e	Transport not used	0xffff

11.5.2 SessionConfigurationToken 0x0001

Table 109 shows the Protocol Types and Subtypes for SessionConfigurationToken 0x0001. All attributes for SessionConfigurationToken 0x0001 shall be set to the default values defined by the protocol or transport Subtype except for the attributes defined in Table 110.

Table 109 Protocol types and subtypes

Protocol Type		Protocol Subtype	
Name	**ID**	**Name**	**ID**
Physical Layer	0x00	Default Physical Layer	0x0000
Control Channel MAC	0x01	Default Control Channel MAC	0x0000
Access Channel MAC	0x02	Default Access Channel MAC	0x0000
Forward Traffic Channel MAC	0x03	Default Forward Traffic Channel MAC	0x0000

Protocol Type		Protocol Subtype	
Name	**ID**	**Name**	**ID**
Reverse Traffic Channel MAC	0x04	Default Reverse Traffic Channel MAC	0x0000
Reverse Control Channel MAC	0x05	Default Reverse Control Channel MAC	0x0000
Shared Signaling MAC	0x06	Default Shared Signaling MAC	0x0000
Air Link Management	0x07	Default Air Link Management	0x0000
Initialization State	0x08	Default Initialization State	0x0000
Idle State	0x09	Default Idle State	0x0000
Connected State	0x0a	Default Connected State	0x0000
Active Set Management	0x0b	Default Active Set Management	0x0000
Overhead Messages	0x0c	Overhead Messages	0x0000
Authentication	0x0d	Default Authentication	0x0000
Encryption	0x0e	Generic Encryption	0x0001
Security	0x0f	Default Security	0x0000
Key Exchange	0x10	Default Key Exchange	0x0000
Session Management	0x11	Default Session Management	0x0000
Address Management	0x12	Default Address Management	0x0000
Session Configuration	0x13	Default Session Configuration	0x0000
Capabilities Discovery	0x14	Default Capabilities Discovery	0x0000
Inter RAT Protocol	0x15	Default Inter RAT Protocol	0x0000
Packet Consolidation	0x16	Default Packet Consolidation	0x0000
Transport 0	0x17	Default Signaling Transport	0x0000
Transport 1	0x18	Default Data Transport	0x0001
Transport 2	0x19	Transport not used	0xffff
Transport 3	0x1a	Transport not used	0xffff
Transport 4	0x1b	Transport not used	0xffff
Transport 5	0x1c	Transport not used	0xffff
Transport 6	0x1d	Transport not used	0xffff
Transport 7	0x1e	Transport not used	0xffff

For this SessionConfigurationToken, the following attributes shall be set to the specified values:

Table 110 Configuration attributes that shall be set to non-default values

Protocol Subtype		Configuration Attribute		
Name	**ID**	**Name**	**ID**	**Value**
Default Security	0x0000	SecurityEnabled	0x00	0x01

11.5.3 SessionConfigurationToken 0x0002

Table 111 shows the Protocol Types and Subtypes for SessionConfigurationToken 0x0002. All attributes for SessionConfigurationToken 0x0002 shall be set to the default values defined by the protocol or transport Subtype except for the attributes defined in Table 112.

Table 111 Protocol types and subtypes

Protocol Type		Protocol Subtype	
Name	**ID**	**Name**	**ID**
Physical Layer	0x00	Default Physical Layer	0x0000
Control Channel MAC	0x01	Default Control Channel MAC	0x0000
Access Channel MAC	0x02	Default Access Channel MAC	0x0000
Forward Traffic Channel MAC	0x03	Default Forward Traffic Channel MAC	0x0000
Reverse Traffic Channel MAC	0x04	Default Reverse Traffic Channel MAC	0x0000
Reverse Control Channel MAC	0x05	Default Reverse Control Channel MAC	0x0000
Shared Signaling MAC	0x06	Default Shared Signaling MAC	0x0000
Air Link Management	0x07	Default Air Link Management	0x0000
Initialization State	0x08	Default Initialization State	0x0000
Idle State	0x09	Default Idle State	0x0000
Connected State	0x0a	Default Connected State	0x0000
Active Set Management	0x0b	Default Active Set Management	0x0000
Overhead Messages	0x0c	Overhead Messages	0x0000
Authentication	0x0d	Default Authentication	0x0000
Encryption	0x0e	Generic Encryption	0x0001
Security	0x0f	Default Security	0x0000
Key Exchange	0x10	Default Key Exchange	0x0000
Session Management	0x11	Default Session Management	0x0000
Address Management	0x12	Default Address Management	0x0000
Session Configuration	0x13	Default Session Configuration	0x0000
Capabilities Discovery	0x14	Default Capabilities Discovery	0x0000
Inter RAT Protocol	0x15	Default Inter RAT Protocol	0x0000
Packet Consolidation	0x16	Default Packet Consolidation	0x0000
Transport 0	0x17	Default Signaling Transport	0x0000
Transport 1	0x18	Default Data Transport	0x0001
Transport 2	0x19	Transport not used	0xffff
Transport 3	0x1a	Transport not used	0xffff
Transport 4	0x1b	Transport not used	0xffff
Transport 5	0x1c	Transport not used	0xffff
Transport 6	0x1d	Transport not used	0xffff
Transport 7	0x1e	Transport not used	0xffff

For this SessionConfigurationToken, the following attributes shall be set to the specified values:

Table 112 Configuration attributes that shall be set to non-default values

Protocol Subtype		Configuration Attribute		
Name	ID	Name	ID	Value
Default Authentication	0x0000	AuthenticationMode	0x00	0x01
Default Security	0x0000	SecurityEnabled	0x00	0x01

11.6 Flow profile identifier assignments

The FlowProfileID identifies the service needs for an application flow. In order to utilize the FlowProfileID as a short hand method for specifying all relevant air interface parameters necessary to support a particular multimedia packet data service, one must explicitly define the service for which flow requirements are being indicated by the identifier.

Each flow profile is identified by a unique FlowProfileID to facilitate proper processing within the access network and access terminals. FlowProfileID may be used for standard services, as well as for proprietary (non-standard) services. The FlowProfileID format is shown in Table 113.

Table 113 FlowProfileID format

Field	Length (bits)
FlowProfileIDType	4
FlowProfileIDLength	4
FlowProfileNum	FlowProfileIDLength × 8

FlowProfileIDType — This field shall be set to an identifier for the FlowProfileIDType according to Table 15.

Table 114 FlowProfileIDType values

Value	Description
0x0	Standard FlowProfileNum according to 11.6.1
0x1	cdma2000 FlowProfileNum according to[24]
0x2	Proprietary FlowProfileNum
All other values	Reserved

FlowProfileIDLength — This field shall be set to the length of the FlowProfileID field in units of octets.

FlowProfileNum — If FlowProfileIDType is 0x0, then this field shall be set as defined in 11.6.1. If FlowProfileIDType is 0x1, then this field shall be set as defined in [24].If FlowProfileIDType is 0x2, the setting of this field is beyond the scope of this specification.

11.6.1 Flow Profile Identifier Assignments

The FlowProfileIDType of standard FlowProfileNum is set to 0x0. The characteristics associated with each FlowProfileNum are beyond the scope of this specification.

11.6.1.1 Generic data service flow profile identifier assignments

Table 9 shows the Generic data FlowProfileNum assignments defined for this specification.

Table 115 Generic data service profile identifier assignments

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
0	0x0000	Best effort
1	0x0001	Streaming 32 kbps
2	0x0002	Streaming 64 kbps
3	0x0003	Streaming 96 kbps
4	0x0004	Streaming 128 kbps
5	0x0005	Minimum Acceptable User Data Rate of 32kbps, max. latency[93] is 100msec, 1% avg. data loss rate[94].
6	0x0006	Minimum Acceptable User Data Rate of 64kbps, max. latency is 100msec, 1% avg. data loss rate.
7	0x0007	Minimum Acceptable User Data Rate of 128kbps, max. latency is 100msec, 1% avg. data loss rate.
8	0x0008	Minimum Acceptable User Data Rate of 256kbps, max. latency is 100msec, 1% avg. data loss rate.
9	0x0009	Minimum Acceptable User Data Rate of 512kbps, max. latency is 100msec, 1% avg. data loss rate.
10	0x000a	Minimum Acceptable User Data Rate of 1024kbps, max. latency is 100msec, 1% avg. data loss rate.
11	0x000b	Minimum Acceptable User Data Rate of 2048kbps, max. latency is 100msec, 1% avg. data loss rate.
12	0x000c	Minimum Acceptable User Data Rate of 4096kbps, max. latency is 100msec, 1% avg. data loss rate.
13	0x000d	Minimum Acceptable User Data Rate of 32kbps, max. latency is 100msec, 0.01% avg. data loss rate.
14	0x000e	Minimum Acceptable User Data Rate of 64kbps, max. latency is 100msec, 0.01% avg. data loss rate.
15	0x000f	Minimum Acceptable User Data Rate of 128kbps, max. latency is 100msec, 0.01% avg. data loss rate.

[93] Maximum latency is defined to be the maximum amount of time allowed from the time that an octet of user data is submitted to the transmitting RLP until the receiving RLP either delivers the octet or aborts its delivery.

[94] Data loss rate is defined as the ratio of the number of lost data octets to the number of transmitted data octets, measured above RLP.

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
16	0x0010	Minimum Acceptable User Data Rate of 256kbps, max. latency is 100msec, 0.01% avg. data loss rate.
17	0x0011	Minimum Acceptable User Data Rate of 512kbps, max. latency is 100msec, 0.01% avg. data loss rate.
18	0x0012	Minimum Acceptable User Data Rate of 1024kbps, max. latency is 100msec, 0.01% avg. data loss rate.
19	0x0013	Minimum Acceptable User Data Rate of 2048kbps, max. latency is 100msec, 0.01% avg. data loss rate.
20	0x0014	Minimum Acceptable User Data Rate of 4096kbps, max. latency is 100msec, 0.01% avg. data loss rate.
21 – 255	0x0015 – 0x00ff	Reserved

11.6.1.2 Speech service flow profile identifier assignments

Table 116 shows the Speech Service FlowProfileNum assignments defined for this specification.

Table 116 Speech service profile identifier assignments

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
256	0x0100	G.711 at 64 kbps[95]
257	0x0101	G.722.1 at 24 kbps[96]
258	0x0102	G.722.1 at 32 kbps[96]
259	0x0103	G.723.1 at 5.3 kbps[97]
260	0x0104	G.723.1 at 6.3 kbps[97]
261	0x0105	G.726 at 16 kbps[98]
262	0x0106	G.726 at 24 kbps[98]
263	0x0107	G.726 at 32 kbps[98]
264	0x0108	G.726 at 40 kbps[98]
265	0x0109	G.728 at 16 kbps[99]

[95] ITU-T Recommendation G.711: "Pulse code modulation (PCM) of voice frequencies".

[96] ITU-T Recommendation G.722.1: "Low-complexity coding at 24 and 32 kbit/s for hands-free operation in systems with low frame loss."

[97] ITU-T Recommendation G.723.1: "Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s."

[98] ITU-T Recommendation G.726: "40, 32, 24, 16 kbit/s adaptive differential pulse code modulation (ADPCM)."

[99] ITU-T Recommendation G.728: "Coding of speech at 16 kbit/s using low-delay code excited linear prediction."

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
266	0x010a	G.729 at 8 kbps[100]
267	0x010b	AMR at 12.2 kbps (Also known as GSM-EFR which is the TIA TDMA-US1 Enhanced speech codec)[101]
268	0x010c	AMR at 10.2 kbps[101]
269	0x010d	AMR at 7.95 kbps[101]
270	0x010e	AMR at 7.40 kbps (Also known as TIA/EIA IS-641 TDMA Enhanced Full Rate Speech Codec) [101]
271	0x010f	AMR at 6.70 kbps (Also known as PDC-EFR which is the ARIB 6.7 kbps Enhanced Full Ratge Speech Codec) [101]
272	0x0110	AMR at 5.90 kbps[101]
273	0x0111	AMR at 5.15 kbps[101]
274	0x0112	AMR at 4.75 kbps[101]
275	0x0113	AMR at 1.80 kbps (SID) [101]
276	0x0114	EVRC[102]
277	0x0115	SMV[103]
278 – 511	0x116 – 0x01ff	Reserved

11.6.1.3 Audio service flow profile identifier assignments

The audio media service is specified using the “Streaming” (high latency, low error rate) traffic class. Table 117 shows the Audio Service FlowProfileNum assignments defined for this specification.

Table 117 Audio service profile identifier assignments

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
512	0x0200	Streaming Audio 16k
513	0x0201	Streaming Audio 24k
514	0x0202	Streaming Audio 32k
515	0x0203	Streaming Audio 48k
516	0x0204	Streaming Audio 64k

[100] ITU-T Recommendation G.729: “Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction (CS-ACELP).”

[101] 3GPP TS 26.071: "Adaptive Multi-Rate (AMR) Speech Codec; General Description."

[102] 3GPP2 C.S0014-A v1.0, Enhanced Variable Rate Codec, Speech Service Option 3 for Wideband Spread Spectrum Digital Systems.

[103] 3GPP2 C.S0030-0 v2.0, Selectable Mode Vocoder Service Option 56 for Wideband Spread Spectrum Communication Systems.

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
517	0x0205	Streaming Audio 128k
518	0x0206	Streaming Audio 256k
519	0x0207	Streaming Audio 512k
520	0x0208	Streaming Audio 1024k
521 – 767	0x0209 – 0x02ff	Reserved

11.6.1.4 Video service flow profile identifier assignments

Video media services are specified using the "Conversational" (low latency, low error rate) and "Streaming" (high latency, low error rate) traffic classes. Table 118 shows the Video Service FlowProfileNum assignments defined for this specification.

Table 118 Video service profile identifier assignments

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
768	0x0300	Conversational Interactive Video 24k
769	0x0301	Conversational Interactive Video 32k
770	0x0302	Conversational Interactive Video 40k
771	0x0303	Conversational Interactive Video 48k
772	0x0304	Conversational Interactive Video 56k
773	0x0305	Conversational Interactive Video 64k
774	0x0306	Conversational Interactive Video 128k
775	0x0307	Conversational Interactive Video 256k
776	0x0308	Conversational PtT Video 24k
777	0x0309	Conversational PtT Video 32k
778	0x030a	Conversational PtT Video 40k
779	0x030b	Conversational PtT Video 48k
780	0x030c	Conversational PtT Video 56k
781	0x030d	Conversational PtT Video 64k
782	0x030e	Conversational PtT Video 128k
783	0x030f	Conversational PtT Video 256k
784	0x0310	Streaming Video 24k
785	0x0311	Streaming Video 48k
786	0x0312	Streaming Video 64k
787	0x0313	Streaming Video 96k
788	0x0314	Streaming Video 120k
789	0x0315	Streaming Video 128k
790	0x0316	Streaming Video 256k

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
791	0x0317	Streaming Video 512k
792	0x0318	Streaming Video 1024k
793	0x319	Streaming Video 2000k
794	0x31a	Streaming Video 4000k
795	0x31b	Streaming Video 8000k
796	0x31c	Streaming Video 16000k
797	0x31d	Streaming Video 20000k
798 – 1023	0x031e – 0x03ff	Reserved

11.6.1.5 Text service flow profile identifier assignments

The text media service is specified using the “Streaming” (high latency, low error rate) traffic class. Table 119 shows the Text Service FlowProfileNum assignments defined for this specification.

Table 119 Text service profile identifier assignments

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
1024	0x0400	Streaming Text
1025 – 1279	0x0400 – 0x04ff	Reserved

11.6.1.6 Signaling service flow profile identifier assignments

The signaling media service is specified using the “Conversational” (low latency, low error rate), “Streaming” (high latency, low error rate), and “Interactive” (low latency, medium error rate) traffic classes. Table 120 shows the Signaling Service FlowProfileNum assignments defined for this specification.

Table 120 Signaling service profile identifier assignments

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
1280	0x0500	Conversational Media Control Signaling
1281	0x0501	Streaming Media Control Signaling
1282	0x0502	Interactive Media Control Signaling
1283 – 1535	0x0503 – 0x05ff	Reserved

11.6.1.7 Gaming Service Flow Profile Identifier Assignments

The gaming media service is specified using the “Interactive” (low latency, medium error rate) traffic class. Table 121 shows the Gaming Service FlowProfileNum assignments defined for this specification.

Table 121 Gaming service profile identifier assignments

FlowProfileNum (Decimal)	FlowProfileNum (Hexadecimal)	Flow Description
1536	0x0600	Interactive Gaming
1537 – 1791	0x0601 – 0x06ff	Reserved

12 Precoding and SDMA Codebooks

Precoding and SDMA are defined in Chapter 9 as a mapping between effective antennas and tile antennas. A particular mapping is defined by a precoding matrix. The columns of the precoding matrix define a set of spatial beams that can be used by AN. The AN uses only one column, of the precoding matrix in SISO transmission, and multiple columns in STTD or MIMO transmissions.

The AT may choose to feedback a preferred precoding matrix to be used by AN for future transmissions. The set of such precoding matrices forms a codebook. In this chapter a number of precoding/SDMA codebooks are listed and the corresponding values of BFCHBeamCodeBookIndex are defined in the Overhead Messages protocol. Some of the precoding matrices in a codebook are grouped into clusters. Matrices in a single cluster typically span only part of the space. If the AT feeds back a beam index within a cluster, the AN treats this as an indication that it may schedule other ATs on different clusters.

The BeamIndex field, in the BFCHBeamIndex report, indexes a beam in a codebook specified by BFCHBeamCodeBookIndex. The BeamIndex may indicate one or more of the following: A no preferred precoding or SDMA matrix, a preferred SISO precoding or SDMA transmission on a spatial beam, a preferred STTD precoding or SDMA transmission on two spatial beams, and a preferred MIMO precoding or SDMA transmission on a set of spatial beams (more than one column of the precoding matrix).

It shall be understood that, if the codebook supports MIMO, only the 0^{th} to $(\text{SpatialOrder-1})^{th}$ columns of the precoding matrix are to be used, where SpatialOrder is defined in the Forward Traffic Channel MAC protocol. If a spatial beam $\mathbf{w} = \left[w_0 \quad w_1 \quad \cdots \quad w_{N_{EFT\_TX\_ANT}-1} \right]^T$, is used to transmit a modulation symbol s, then $w_j s$ is transmitted on effective antenna j, where $N_{EFT\_TX\_ANT}$ is equal to the parameter EffectiveNumAntenna maintained in the overhead messages protocol, and T is the matrix transpose.

In the sequel i indicates the imaginary part in a complex number.

12.1 BFCHBeamCodeBookIndex = 0000

This codebook is only valid for SISO transmission and $N_{EFT\_TX\_ANT}$=4.

BeamIndex=0: The AT does not prefer a specific precoding matrix and rather prefers a random matrix that is chosen by AN. The random matrix may change at a rate chosen by AN.

12.1.1 Cluster 1 :

BeamIndex=1: $[0.5\ 0.5i\ -0.5\ -0.5i]^T$

12.1.2 Cluster 2:

BeamIndex=2: $[0.5\ -0.5i\ -0.5\ 0.5i]^T$

12.2 BFCHBeamCodeBookIndex = 0001

This codebook is valid for SISO and MIMO transmissions, and only if $N_{EFT\_TX\_ANT}$=4.

Let **BeamMat**=

[0.5000 0.5000 0.5000 0.5000

0.3536 + 0.3536i -0.3536 + 0.3536i 0.3536 - 0.3536i -0.3536 - 0.3536i

0.0000 + 0.5000i -0.0000 - 0.5000i 0.0000 - 0.5000i -0.0000 + 0.5000i

-0.3536 + 0.3536i 0.3536 + 0.3536i -0.3536 - 0.3536i 0.3536 - 0.3536i]

The codebook is defined as follows:

BeamIndex=0: The AT does not prefer a specific precoding matrix and rather prefers a random matrix that is chosen by AN. The random matrix may change at a rate chosen by AN.

BeamIndex=1: Zero[th] column in **BeamMat**

BeamIndex=2: First column in **BeamMat**

BeamIndex=3: Second column in **BeamMat**

BeamIndex=4: Third column in **BeamMat**

12.2.1 Precoding Matrices

BeamIndex=5 to BeamIndex=35: If the BeamIndex is equal to j, define $seed_{k,j}$=(2π [BIT_REVERSE([(4*j+k)*2654435761] mod 2^{32})] mod 2^{20})/ 2^{20} , k=0,1,2,3.The corresponding precoding matrix, $\mathbf{U}_j$, is a random unitary matrix defined as $\mathbf{U}_j = \mathbf{\Lambda}_j \mathbf{D}$, where $\mathbf{\Lambda}_j$ is a diagonal matrix on the form $\begin{bmatrix} e^{i\phi_0} & 0 & 0 & 0 \\ 0 & e^{i\phi_1} & 0 & 0 \\ 0 & 0 & e^{i\phi_2} & 0 \\ 0 & 0 & 0 & e^{i\phi_3} \end{bmatrix}$, ϕ_k = $seed_{k,j}$ is a uniform random variable between 0 and 2π, and $\mathbf{D}$ is the 4x4 DFT matrix , i.e.,

$$\mathbf{D} = \left\{ D_{m,n}, m,n = 0,\cdots,3 \right\}, D_{m,n} = \frac{1}{\sqrt{4}} e^{j 2\pi mn/4} .$$

12.2.2 Cluster 1:

BeamIndex=36 to BeamIndex=49: If the BeamIndex is equal to j, define seed$_{k,j}$=(2π [BIT_REVERSE([(2*j+k)*2654435761] mod 2^{32})] mod 2^{20})/ 2^{20} , k=0,1. The corresponding precoding matrix is defined as **BeamMat**(:,0:1)*$\mathbf{U}_j$, where **BeamMat**(:,0:1) is the zeroth and first columns of **BeamMat**, $\mathbf{U}_j = \mathbf{\Lambda}_j \mathbf{D}$, $\mathbf{\Lambda}_j = \begin{bmatrix} e^{j\phi_0} & 0 \\ 0 & e^{j\phi_1} \end{bmatrix}$, ϕ_k = seed$_{k,j}$ is a uniform random variable between 0 and 2π , and **D** is the 2x2 DFT matrix, i.e.,

$$\mathbf{D} = \left\{ D_{m,n}, m, n = 0,1 \right\}, D_{m,n} = \frac{1}{\sqrt{2}} e^{j 2\pi mn/2}.$$

12.2.3 Cluster 2:

BeamIndex=50 to BeamIndex=63: If the BeamIndex is equal to j, define seed$_{k,j}$=(2π [BIT_REVERSE([(2*j+k)*2654435761] mod 2^{32})] mod 2^{20})/ 2^{20} , k=0,1. The corresponding precoding matrix is defined as **BeamMat**(:,2:3)*$\mathbf{U}_j$, where **BeamMat**(:,2:3) is the second and third columns of **BeamMat**, $\mathbf{U}_j = \mathbf{\Lambda}_j \mathbf{D}$, $\mathbf{\Lambda}_j = \begin{bmatrix} e^{j\phi_0} & 0 \\ 0 & e^{j\phi_1} \end{bmatrix}$, ϕ_k = seed$_{k,j}$ is a uniform random variable between 0 and 2π , and **D** is the 2x2 DFT matrix, i.e.,

$$\mathbf{D} = \left\{ D_{m,n}, m, n = 0,1 \right\}, D_{m,n} = \frac{1}{\sqrt{2}} e^{j 2\pi mn/2}.$$

13 MAC and PHY MIB

13.1 Overview

This chapter defines a Management Information Base (MIB) module for managing the MAC and PHY. Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP).The objects in this MIB are defined using the mechanisms specified in the Structure of Management Information (SMI). The MIB module specified is compliant to SMIv2 which is described in RFC 2578 [18], RFC 2579 [19], and RFC 2580 [20].

13.2 MIB Structure

The MIB structure is based on the architecture reference model in Figure 1 and the layering architecture for the air interface in Figure 2 describes the layering architecture for the air interface. The MIB object is composed of two groups:

- dot20An: This group contains managed objects defined for the access network.
- dot20Cmn: This group contains managed objects defined for the access network and the access terminal.

13.3 Definition

```
IEEE802dot20-MIB DEFINITIONS ::= BEGIN

IMPORTS
    ifIndex
        FROM IF-MIB
    MODULE-COMPLIANCE, OBJECT-GROUP
        FROM SNMPv2-CONF
    Counter32, Counter64, Integer32, MODULE-IDENTITY, OBJECT-IDENTITY,
    OBJECT-TYPE, transmission
        FROM SNMPv2-SMI
    RowPointer, RowStatus, TEXTUAL-CONVENTION, TruthValue
        FROM SNMPv2-TC
    ;

ieee802dot20 MODULE-IDENTITY
    LAST-UPDATED  "200512301948Z" -- December 30, 2005
    ORGANIZATION
        "IEEE 802.20"
    CONTACT-INFO
        "Contact: J. Tomcik
         Postal:  5775 Morehouse Dr
         San Diego, CA, 92121, USA
         Tel: 858-658-3231
         Fax: 858-658-3231
         E-mail:  jtomcik@qualcomm.com"
    DESCRIPTION
        "The MIB module for IEEE 802.20 entities.
         (The transmission oid used for this MIB needs to be updated
         when a valid one is obtained from IANA along with the new
         802.20 ifType)"
    ::= { transmission 9999 }
```

```
Dot20AnCarrierConfigEntry ::= SEQUENCE
{
    dot20AnCarrierID                Integer32,
    dot20AnFLReservedInterlaces     INTEGER,
    dot20AnNumFLReservedSubbands    Integer32,
    dot20AnFLFirstRestrSetSubband   Integer32,
    dot20AnFLNumRestrSetSubbands    Integer32,
    dot20AnFLChannelTreeIndex       Integer32,
    dot20AnFLSectorHopSeed          Integer32,
    dot20AnFLIntraCellCommonHopping TruthValue,
    dot20AnFLDPISectorOffset        Integer32,
    dot20AnFLDPISectorScramble      Integer32,
    dot20AnFLNumSDMADimensions      Integer32,
    dot20AnFLNumSubbands            Integer32,
    dot20AnFLDiversityHoppingMode   TruthValue,
    dot20AnNumPilots                Integer32,
    dot20AnEffectiveNumAntennas     Integer32,
    dot20AnNumCmnPilotTxAntennas    Integer32,
    dot20AnEnableCmnPilotStaggering TruthValue,
    dot20AnEnableAuxPilotStaggering TruthValue,
    dot20AnSSCHNumHopports          Integer32,
    dot20AnSSCHNumBlocks            Integer32,
    dot20AnSSCHModSymbolsPerBlock   Integer32,
    dot20AnPilotPN                  Integer32,
    dot20AnRLChannelTreeIndex       Integer32,
    dot20AnRLSectorHopSeed          Integer32,
    dot20AnRLIntraCellCommonHopping TruthValue,
    dot20AnBFCHBeamCodeBookIndex    Integer32,
    dot20AnRPICHEnabled             TruthValue,
    dot20AnRLDPISectorOffset        Integer32,
    dot20AnRLDPISectorScramble      Integer32,
    dot20AnRLNumSDMADimensions      Integer32,
    dot20AnRLRestrictedSetBitmap    Integer32,
    dot20AnRLNumSubbands            Integer32,
    dot20AnRLDiversityHoppingMode   TruthValue,
    dot20AnNumRLControlSubbands     Integer32,
    dot20AnRACKBandwidthFactor      Integer32,
    dot20AnHalfDuplexModeSupported  TruthValue,
    dot20AnRevLinkSilenceDuration   Integer32,
    dot20AnRevLinkSilencePeriod     Integer32,
    dot20AnTransmitPower            Integer32,
    dot20AnCommonPilotPower         Integer32,
    dot20AnAuxPilotPower            Integer32,
    dot20AnPreamblePilotPower       Integer32,
    dot20AnMACIDRange               INTEGER,
    dot20AnFLPCReportInterval       Integer32,
    dot20AnRLCtrlPCMode             INTEGER,
    dot20AnCtrlAccessOffset         Integer32,
    dot20AnBFCHPowerOffset          Integer32,
    dot20AnSFCHPowerOffset          Integer32,
    dot20AnPICHPowerOffset          Integer32,
    dot20AnReqChannelGain0          Integer32,
    dot20AnReqChannelGain1          Integer32,
    dot20AnReqChannelGain2          Integer32,
    dot20AnReqChannelGain3          Integer32,
    dot20AnErasureGain0             Integer32,
    dot20AnErasureGain1             Integer32,
    dot20AnErasureGain2             Integer32,
    dot20AnErasureGain3             Integer32,
    dot20AnAccessCycleDuration      Integer32,
    dot20AnAccessSequencePartition  Integer32,
    dot20AnMaxProbesPerSequence     Integer32,
    dot20AnProbeRampUpStepSize      Integer32,
    dot20AnFastOSIEnabled           TruthValue,
    dot20AnRDCHInitialPacketFormat  Integer32,
    dot20AnPilotThreshold1          Integer32,
    dot20AnPilotThreshold2          Integer32,
```

```
    dot20AnOpenLoopAdjust           Integer32,
    dot20AnAccessRetryPersistance0  Integer32,
    dot20AnAccessRetryPersistence1  Integer32,
    dot20AnAccessRetryPersistence2  Integer32,
    dot20AnAccessRetryPersistence3  Integer32,
    dot20AnSynchGroup               Integer32,
    dot20AnCarrierConfigRowStatus   RowStatus
}

Dot20AnChannelBandsEntry ::= SEQUENCE
{
    dot20AnChannelBandIndex         Integer32,
    dot20AnChannelBandRecordType    INTEGER,
    dot20AnChannelBandClass         Integer32,
    dot20AnChannelNumber            Integer32,
    dot20AnChannelFwdLinkCenterFreq Integer32,
    dot20AnChannelRevLinkCenterFreq Integer32,
    dot20AnChannelFwdLinkSystemBw   Integer32,
    dot20AnChannelRevLinkSystemBw   Integer32,
    dot20AnChannelBandStatus        RowStatus
}

Dot20AnIdleStateStatsEntry ::= SEQUENCE
{
    dot20AnAccessAttemptCounts     Counter32,
    dot20AnAccessAttemptFailCounts Counter32,
    dot20AnPageAttemptCounts       Counter32,
    dot20AnPageFailureCounts       Counter32
}

Dot20AnNeighborCarriersEntry ::= SEQUENCE
{
    dot20AnNeighborSectorIndex   Integer32,
    dot20AnNeighborCarrierID     Integer32,
    dot20AnNeighborSectorID      OCTET STRING,
    dot20AnNeighborPilotPN       Integer32,
    dot20AnNeighborTransmitPower Integer32,
    dot20AnNghbrGloballySynch    TruthValue,
    dot20AnNgbhrSynchGroup       Integer32,
    dot20AnNeighborCarrierStatus RowStatus
}

Dot20AnNeighborListEntry ::= SEQUENCE
{
    dot20AnNeighborIndex          Integer32,
    dot20AnNeighborCarrierPointer RowPointer,
    dot20AnNeighborRowStatus      RowStatus
}

Dot20AnOtherTechNghbrsEntry ::= SEQUENCE
{
    dot20AnOtherTechnologyIndex    Integer32,
    dot20AnTechnologyType          INTEGER,
    dot20AnTechNghbrListLength     Integer32,
    dot20AnTechnologyNeighborList  OCTET STRING,
    dot20AnOtherTechNghbrRowStatus RowStatus
}

Dot20AnSectorConfigEntry ::= SEQUENCE
{
    dot20AnSectorID                 OCTET STRING,
    dot20AnChannelBandRef           Integer32,
    dot20AnMaximumRevision          Integer32,
    dot20AnMinimumRevision          Integer32,
    dot20AnNumCarriers              Integer32,
    dot20AnCPLength                 Integer32,
    dot20AnPrNumGuardSubcarriers    Integer32,
    dot20AnNumGuardSubcarriers      Integer32,
```

```
    dot20AnBlockHoppingEnabled      TruthValue,
    dot20AnNFLBurst                 Integer32,
    dot20AnNRLBurst                 Integer32,
    dot20AnCountryCode              Integer32,
    dot20AnSubnetMask               OCTET STRING,
    dot20AnLatitude                 Integer32,
    dot20AnLongitude                Integer32,
    dot20AnRegistrationRadius       Integer32,
    dot20AnLeapSeconds              Integer32,
    dot20AnLocalTimeOffset          Integer32,
    dot20AnRegistrationZoneIncluded TruthValue,
    dot20AnRegistrationZoneCode     Integer32,
    dot20AnRegistrationZoneMaxAge   Integer32,
    dot20AnSynchronousSystem        TruthValue,
    dot20AnPhysicalMode             INTEGER,
    dot20AnSectorConfigRowStatus    RowStatus
}

Dot20AnSectorToIfIndexEntry ::= SEQUENCE
{
    dot20AnIfChannelBandRef Integer32
}

Dot20CmnAuthStatsEntry ::= SEQUENCE
{
    dot20CmnAuthFailureCounts Counter64,
    dot20CmnAuthSuccessCounts Counter64
}

Dot20CmnDataTransportStatsEntry ::= SEQUENCE
{
    dot20CmnRlpTxBytes               Counter64,
    dot20CmnRlpReTxBytes             Counter64,
    dot20CmnRlpTxDropBytes           Counter64,
    dot20CmnRlpTxStatus              Counter64,
    dot20CmnRlpRxBytes               Counter64,
    dot20CmnRlpRxStatus              Counter64,
    dot20CmnRlpTxPackets             Counter64,
    dot20CmnRlpReTxPackets           Counter64,
    dot20CmnRlpTxrDropPackets        Counter64,
    dot20CmnRlpRxPackets             Counter64,
    dot20CmnRlpTxNAKTimeouts         Counter64,
    dot20CmnActiveReservationsCounts Counter64,
    dot20CmnIdleReservationsCounts   Counter64,
    dot20CmnReservationOpenCounts    Counter64,
    dot20CmnReservationCloseCounts   Counter64,
    dot20CmnReservationFailCounts    Counter64
}

Dot20CmnLMACPacketStatsEntry ::= SEQUENCE
{
    dot20CmnPacketFormatIndex Integer32,
    dot20CmnARQAttemptsIndex  Integer32,
    dot20CmnFwdTxPacketCounts Counter64,
    dot20CmnRevRxPacketCounts Counter64
}

Dot20CmnLMACStatsEntry ::= SEQUENCE
{
    dot20CmnFLABCounts        Counter64,
    dot20CmnRLABCounts        Counter64,
    dot20CmnAccessGrantCounts Counter64
}

Dot20CmnSessionConfigTokenEntry ::= SEQUENCE
{
    dot20CmnSessionTokenIndex      Integer32,
    dot20CmnSessionConfToken       Integer32,
```

```
    dot20CmnSessionConfTokenStatus RowStatus
}

Dot20CmnSigTransportStatsEntry ::= SEQUENCE
{
    dot20CmnSlpTxBytes        Counter64,
    dot20CmnSlpReTxBytes      Counter64,
    dot20CmnSlpTxDroppedBytes Counter64,
    dot20CmnSlpTxStatus       Counter64,
    dot20CmnSlpRxBytes        Counter64,
    dot20CmnSlpRxStatus       Counter64,
    dot20CmnSlpTxPackets      Counter64,
    dot20CmnSlpReTxPackets    Counter64,
    dot20CmnSlpTxDropPackets  Counter64,
    dot20CmnSlpRxPackets      Counter64,
    dot20CmnSlpTxACKTimeouts  Counter64
}

dot20An OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION
        "AN specific configuration and statistics."
    ::= { ieee802dot20 1 }

dot20AnMac OBJECT-IDENTITY
    STATUS        current
    DESCRIPTION
        "MAC layer objects"
    ::= { dot20An 1 }

dot20AnLowerMACControl OBJECT IDENTIFIER ::= { dot20AnMac 3 }

dot20AnIdleState OBJECT IDENTIFIER ::= { dot20AnLowerMACControl 1 }

dot20AnIdleStateStatsTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF Dot20AnIdleStateStatsEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "This table provides one row of Idle State protocol statistics
         per 802.20 interface (i.e. sector for a specific ChannelBand)
         and carrier."
    ::= { dot20AnIdleState 1 }

dot20AnIdleStateStatsEntry OBJECT-TYPE
    SYNTAX        Dot20AnIdleStateStatsEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An Entry (conceptual row) in the IdleStateStats table. This
         table is indexed by ifIndex and CarrierID. ifIndex: Each IEEE
         802.20 interface (uniquely identified by SectorID) is
         represented by an ifEntry. In the case of a multicarrier
         Sector, the carrierID indentifies one specific carrier."
    INDEX
        { ifIndex, dot20AnCarrierID }
    ::= { dot20AnIdleStateStatsTable 1 }

dot20AnAccessAttemptCounts OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of Access Attempts"
    ::= { dot20AnIdleStateStatsEntry 1 }

dot20AnAccessAttemptFailCounts OBJECT-TYPE
    SYNTAX        Counter32
```

```
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of Failed Access Attempts"
    ::= { dot20AnIdleStateStatsEntry 2 }

dot20AnPageAttemptCounts OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of Page Attempts"
    ::= { dot20AnIdleStateStatsEntry 3 }

dot20AnPageFailureCounts OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of Failed Page Attempts"
    ::= { dot20AnIdleStateStatsEntry 4 }

dot20AnOverheadMessages OBJECT IDENTIFIER ::= { dot20AnLowerMACControl 4 }

dot20AnSectorConfigTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF Dot20AnSectorConfigEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "This table provides one row per 802.20 interface, i.e. sector
         for a specific ChannelBand. This table's attributes specify the
         configuration of the corresponding sector, and can be used to
         populate fields in SystemInfo block and SectorParameters
         message."
    ::= { dot20AnOverheadMessages 1 }

dot20AnSectorConfigEntry OBJECT-TYPE
    SYNTAX        Dot20AnSectorConfigEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An Entry (conceptual row) in the SectorConfig table. This
         table is indexed by IfIndex. ifIndex: Each IEEE 802.20
         interface (uniquely identified by SectorID) is represented by
         an ifEntry."
    INDEX
        { ifIndex }
    ::= { dot20AnSectorConfigTable 1 }

dot20AnSectorID OBJECT-TYPE
    SYNTAX        OCTET STRING (SIZE(16))
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "Sector Address Identifier. The access network shall set the
         value of the SectorID according to the rules specified in IEEE
         802.20 AIS. The access terminal shall not assume anything about
         the format of the SectorID other than that it uniquely
         identifies the sector."
    ::= { dot20AnSectorConfigEntry 1 }

dot20AnChannelBandRef OBJECT-TYPE
    SYNTAX        Integer32
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "The reference to the ChannelBand defined in ChannelBands table
         using this value as index (dot20AnChannelBandIndex)"
```

```
    ::= { dot20AnSectorConfigEntry 2 }

dot20AnMaximumRevision OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the maximum revision number
         that the sector can support."
    ::= { dot20AnSectorConfigEntry 3 }

dot20AnMinimumRevision OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the minimum revision number
         that the sector can support."
    ::= { dot20AnSectorConfigEntry 4 }

dot20AnNumCarriers OBJECT-TYPE
    SYNTAX        Integer32 (1..4)
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the number of carriers
         available at this sector."
    ::= { dot20AnSectorConfigEntry 5 }

dot20AnCPLength OBJECT-TYPE
    SYNTAX        Integer32 (0..3)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine the cyclic
         prefix length in units of chips. The cyclic prefix length shall
         take the value NFFT*(1+n)/16."
    ::= { dot20AnSectorConfigEntry 6 }

dot20AnPrNumGuardSubcarriers OBJECT-TYPE
    SYNTAX        Integer32 (0..7)
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the preamble's number of guard
         subcarriers as defined in 802.20 Physical layer
         specification."
    ::= { dot20AnSectorConfigEntry 7 }

dot20AnNumGuardSubcarriers OBJECT-TYPE
    SYNTAX        Integer32 (0..7)
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the number of guard subcarriers
         as defined in 802.20 Physical layer specification."
    ::= { dot20AnSectorConfigEntry 8 }

dot20AnBlockHoppingEnabled OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to true if block hopping is
         enabled. This attribute shall be set to false if symbol rate
         hopping is disabled"
    ::= { dot20AnSectorConfigEntry 9 }
```

```
dot20AnNFLBurst OBJECT-TYPE
    SYNTAX        Integer32 (1..4)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the number of forward link PHY
         Frames that comprise a forward link burst in TDD mode."
    ::= { dot20AnSectorConfigEntry 10 }

dot20AnNRLBurst OBJECT-TYPE
    SYNTAX        Integer32 (1..4)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the number of reverse link PHY
         Frames that comprise a reverse link burst in TDD mode."
    ::= { dot20AnSectorConfigEntry 11 }

dot20AnCountryCode OBJECT-TYPE
    SYNTAX        Integer32 (0..999)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the three digit Mobile Country
         Code associated with this sector (as specified in ITU-T
         Recommendation E.212, Identification Plan for Land Mobile
         Stations)."
    ::= { dot20AnSectorConfigEntry 12 }

dot20AnSubnetMask OBJECT-TYPE
    SYNTAX        OCTET STRING (SIZE(16))
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "Sector Subnet identifier."
    ::= { dot20AnSectorConfigEntry 13 }

dot20AnLatitude OBJECT-TYPE
    SYNTAX        Integer32 (-1296000..1296000)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "The latitude of the sector. This attribute shall be set to
         this sector's latitude in units of 0.25 second, expressed as a
         two's complement signed number with positive numbers signifying
         North latitudes. This attribute shall be set to a value in the
         range 1296000 to 1296000 inclusive (corresponding to a range of
         -90 to +90)."
    ::= { dot20AnSectorConfigEntry 14 }

dot20AnLongitude OBJECT-TYPE
    SYNTAX        Integer32 (-2592000..2592000)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "The longitude of the sector. This attribute shall be set to
         this sectorâ ™s longitude in units of 0.25 second, expressed as a
         two's complement signed number with positive numbers signifying
         East longitude. This attribute shall be set to a value in the
         range 2592000 to 2592000 inclusive (corresponding to a range of
         -180Â° to +180Â°)."
    ::= { dot20AnSectorConfigEntry 15 }

dot20AnRegistrationRadius OBJECT-TYPE
    SYNTAX        Integer32 (0..2047)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
```

```
        "If access terminals are to perform distance based
         registration, this attribute shall be set to the non-zero
         distance beyond which the access terminal is to send a new
         PilotReport message. If access terminals are not to perform
         distance based registration, This attribute shall be set to
         0."
    ::= { dot20AnSectorConfigEntry 16 }

dot20AnLeapSeconds OBJECT-TYPE
    SYNTAX        Integer32 (0..255)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "The number of leap seconds that have occurred since the start
         of system time."
    ::= { dot20AnSectorConfigEntry 17 }

dot20AnLocalTimeOffset OBJECT-TYPE
    SYNTAX        Integer32 (0..2047)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the offset of the local time
         from System Time. This value will be in units of minutes,
         expressed as a two's complement signed number."
    ::= { dot20AnSectorConfigEntry 18 }

dot20AnRegistrationZoneIncluded OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to true if the
         RegistrationZoneCode and RegistrationZoneMaxAge are included."
    ::= { dot20AnSectorConfigEntry 19 }

dot20AnRegistrationZoneCode OBJECT-TYPE
    SYNTAX        Integer32 (0..4095)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "The zone code value for this sector."
    ::= { dot20AnSectorConfigEntry 20 }

dot20AnRegistrationZoneMaxAge OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "The Max Age value for this sector."
    ::= { dot20AnSectorConfigEntry 21 }

dot20AnSynchronousSystem OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to true if all sectors in the
         deployment are synchronous. This attribute shall be set to
         false otherwise."
    ::= { dot20AnSectorConfigEntry 22 }

dot20AnPhysicalMode OBJECT-TYPE
    SYNTAX        INTEGER {
        tdd(1),
        fdd(2)
    }
    MAX-ACCESS    read-create
```

```
    STATUS        current
    DESCRIPTION
        "The Physical mode used for this sector."
    ::= { dot20AnSectorConfigEntry 23 }

dot20AnSectorConfigRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The status column used for creating, modifying, and deleting
         instances of the columnar objects in the SectorConfig Table. If
         the implementor of this MIB has chosen not to implement
         'dynamic assignment' of sectors, this attribute is not useful
         and should return noSuchName upon SNMP request."
    DEFVAL         { active }
    ::= { dot20AnSectorConfigEntry 24 }

dot20AnCarrierConfigTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF Dot20AnCarrierConfigEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "This table provides one row per 802.20 carrier of a sector for
         a specific ChannelBand. This table's attributes specify the
         configuration of the corresponding carrier and can be used to
         populate  fields in SystemInfo block, QuickChannelInfo block
         and ExtendedChannelInfo message."
    ::= { dot20AnOverheadMessages 2 }

dot20AnCarrierConfigEntry OBJECT-TYPE
    SYNTAX        Dot20AnCarrierConfigEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An Entry (conceptual row) in the AnCarrierConfig table. This
         table is indexed by ifIndex and CarrierID. ifIndex: Each IEEE
         802.20 interface (uniquely identified by SectorID) is
         represented by an ifEntry. In the case of a multicarrier
         Sector, the carrierID indentifies one specific carrier."
    INDEX
        { ifIndex, dot20AnCarrierID }
    ::= { dot20AnCarrierConfigTable 1 }

dot20AnCarrierID OBJECT-TYPE
    SYNTAX        Integer32 (0..3)
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the CarrierID of the carrier
         this block is transmitted on. The CarrierID of a carrier is
         unique within the corresponding sector."
    ::= { dot20AnCarrierConfigEntry 1 }

dot20AnFLReservedInterlaces OBJECT-TYPE
    SYNTAX        INTEGER {
        none(1),
        zero(2),
        zeroToOne(3),
        zeroToTwo(4),
        zeroToThree(5),
        zeroToFour(6),
        zeroToFive(7),
        zeroToSix(8),
        zeroToSeven(9),
        zeroToEight(10),
        zeroToNine(11),
        zeroToTen(12),
```

```
            zeroToEleven(13),
            zeroToTwelve(14),
            zeroAndThree(15),
            zeroAndSix(16)
        }
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "This attribute shall determine which interlaces contain
             reserved bandwidth on the forward link."
        ::= { dot20AnCarrierConfigEntry 2 }

    dot20AnNumFLReservedSubbands OBJECT-TYPE
        SYNTAX        Integer32 (0..15)
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "This attribute shall determine the number of subbands
             allocated to the ReservedFLBandwidth segment. The
             interpretation of this field is used by the Physical Layer to
             govern FL PHY Frame Modulaiton."
        ::= { dot20AnCarrierConfigEntry 3 }

    dot20AnFLFirstRestrSetSubband OBJECT-TYPE
        SYNTAX        Integer32 (0..15)
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "This attribute shall be set to the index of the first
             restricted subband on the forward link."
        ::= { dot20AnCarrierConfigEntry 4 }

    dot20AnFLNumRestrSetSubbands OBJECT-TYPE
        SYNTAX        Integer32 (0..3)
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "This attribute shall be set to the number of restricted
             subbands on the forward link. This attribute shall be set to 0
             if no subbands are restricted. Otherwise, subbands
             FLFirstRestrictedSetSubband through
             (FLFistRestrictedSetSubband+FLNumRestrictedSetSubbands-1) shall
             be considered to be restricted subbands, with possible rollover
             at subband zero."
        ::= { dot20AnCarrierConfigEntry 5 }

    dot20AnFLChannelTreeIndex OBJECT-TYPE
        SYNTAX        Integer32 (0..15)
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "FL Channel Tree Index used by the Lower MAC Sublayer for this
             carrier."
        ::= { dot20AnCarrierConfigEntry 6 }

    dot20AnFLSectorHopSeed OBJECT-TYPE
        SYNTAX        Integer32 (0..15)
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "FL Sector Hop seed used by the PHY Layer to determine the
             hopping pattern for this carrier."
        ::= { dot20AnCarrierConfigEntry 7 }

    dot20AnFLIntraCellCommonHopping OBJECT-TYPE
        SYNTAX        TruthValue
        MAX-ACCESS    read-write
        STATUS        current
```

```
    DESCRIPTION
        "This attribute is set to True if FL Intra Cell common hopping
         used by the PHY Layer is enabled. This attribute is set to
         False if FL Intra Cell common hopping is disabled."
    ::= { dot20AnCarrierConfigEntry 8 }

dot20AnFLDPISectorOffset OBJECT-TYPE
    SYNTAX         Integer32 (0..3)
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "This attribute shall be set to the relative offset of F-DPICH
         pilots."
    ::= { dot20AnCarrierConfigEntry 9 }

dot20AnFLDPISectorScramble OBJECT-TYPE
    SYNTAX         Integer32 (0..1)
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "This attribute shall determine the scrambling of pilots as
         defined by the Physical Layer sector and cell specific
         scrambling."
    ::= { dot20AnCarrierConfigEntry 10 }

dot20AnFLNumSDMADimensions OBJECT-TYPE
    SYNTAX         Integer32 (1..4)
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "This attribute shall determine the number of spatial
         dimensions on the forward link."
    ::= { dot20AnCarrierConfigEntry 11 }

dot20AnFLNumSubbands OBJECT-TYPE
    SYNTAX         Integer32 (0..1)
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "This attribute shall determine the number of subbands on the
         forward link. If equal to 0, the number of subbands is equal to
         NCARRIER_SIZE/128 and if equal to 1, the number of subbands is
         equal to NCARRIER_SIZE/256."
    ::= { dot20AnCarrierConfigEntry 12 }

dot20AnFLDiversityHoppingMode OBJECT-TYPE
    SYNTAX         TruthValue
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "This attribute shall be used by the Physical Layer to
         determine the hop pattern for the sector. This attribute shall
         be set to true if DiversityHoppingMode is On, false if not."
    ::= { dot20AnCarrierConfigEntry 13 }

dot20AnNumPilots OBJECT-TYPE
    SYNTAX         Integer32 (0..1)
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "This attribute shall determine the nominal number of pilots in
         F-CPICH as being NCARRIER_SIZE/16 or NCARRIER_SIZE/8, depending
         on whether the attribute is set to 0 or 1, respectively."
    ::= { dot20AnCarrierConfigEntry 14 }

dot20AnEffectiveNumAntennas OBJECT-TYPE
    SYNTAX         Integer32 (1..8)
    MAX-ACCESS     read-write
```

```
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the effective number of
         antennas. This attribute shall be set to three or below when
         the BlockHoppingEnabled field of the SystemInfo block is set to
         0."
    ::= { dot20AnCarrierConfigEntry 15 }

dot20AnNumCmnPilotTxAntennas OBJECT-TYPE
    SYNTAX        Integer32 (1..2)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the number of common pilot
         transmit antennas."
    ::= { dot20AnCarrierConfigEntry 16 }

dot20AnEnableCmnPilotStaggering OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to true if common pilot staggering
         is enabled on this sector, false if not."
    ::= { dot20AnCarrierConfigEntry 17 }

dot20AnEnableAuxPilotStaggering OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to true if auxiliary pilot
         staggering is enabled on this sector, false if not."
    ::= { dot20AnCarrierConfigEntry 18 }

dot20AnSSCHNumHopports OBJECT-TYPE
    SYNTAX        Integer32 (0..7)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the number of hop-ports
         allocated to F-SSCH. This attribute shall be interpreted as
         indicated in the 802.20 AIS spec (QuickChannelInfo block
         description)."
    ::= { dot20AnCarrierConfigEntry 19 }

dot20AnSSCHNumBlocks OBJECT-TYPE
    SYNTAX        Integer32 (0..7)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine the number of
         blocks carried by the F-SSCH. The number of F-SSCH blocks shall
         be equal to 2*(n+1)."
    ::= { dot20AnCarrierConfigEntry 20 }

dot20AnSSCHModSymbolsPerBlock OBJECT-TYPE
    SYNTAX        Integer32 (0..3)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the number of modulation
         symbols for each block carried by the F-SSCH. This attribute
         shall be interpreted as indicated in 802.20 AIS spec
         (QuickChannelInfo block description)."
    ::= { dot20AnCarrierConfigEntry 21 }

dot20AnPilotPN OBJECT-TYPE
```

```
    SYNTAX        Integer32 (0..4095)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the PilotPN of the sector."
    ::= { dot20AnCarrierConfigEntry 22 }

dot20AnRLChannelTreeIndex OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "RL Channel Tree Index used by the Lower MAC Sublayer."
    ::= { dot20AnCarrierConfigEntry 23 }

dot20AnRLSectorHopSeed OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "RL Hop Seed used by the PHY Layer to determine the hopping
         pattern."
    ::= { dot20AnCarrierConfigEntry 24 }

dot20AnRLIntraCellCommonHopping OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute is set to True if RL Intra Cell common hopping
         used by the PHY Layer is enabled. This attribute is set to
         False if RL Intra Cell common hopping is disabled."
    ::= { dot20AnCarrierConfigEntry 25 }

dot20AnBFCHBeamCodeBookIndex OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall refer to the code book index, the code
         book comprising of transmit weights for SDMA and precoding."
    ::= { dot20AnCarrierConfigEntry 26 }

dot20AnRPICHEnabled OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to true if RPICH enabled on this
         sector, false if not."
    ::= { dot20AnCarrierConfigEntry 27 }

dot20AnRLDPISectorOffset OBJECT-TYPE
    SYNTAX        Integer32 (0..3)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "Relative offset of reverse pilots"
    ::= { dot20AnCarrierConfigEntry 28 }

dot20AnRLDPISectorScramble OBJECT-TYPE
    SYNTAX        Integer32 (0..1)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the scrambling of pilots as
         defined by the Physical Layer sector and cell specific
         scrambling."
```

```
    ::= { dot20AnCarrierConfigEntry 29 }

dot20AnRLNumSDMADimensions OBJECT-TYPE
    SYNTAX        Integer32 (1..4)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the number of spatial
         dimensions on the reverse link."
    ::= { dot20AnCarrierConfigEntry 30 }

dot20AnRLRestrictedSetBitmap OBJECT-TYPE
    SYNTAX        Integer32 (0..65535)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "Bit position j in this bitfield shall be set to 1 if subband j
         is restricted on the reverse link."
    ::= { dot20AnCarrierConfigEntry 31 }

dot20AnRLNumSubbands OBJECT-TYPE
    SYNTAX        Integer32 (0..1)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the number of subbands on the
         reverse link. If equal to 0, the number of subbands is equal to
         NCARRIER_SIZE/128 and if equal to 1, the number of subbands is
         equal to NCARRIER_SIZE/256."
    ::= { dot20AnCarrierConfigEntry 32 }

dot20AnRLDiversityHoppingMode OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be used by the Physical Layer to
         determine the hop pattern for the sector. This attribute shall
         be set to true if DiversityHoppingMode is On, and to false if
         DiversityHoppingMode is Off."
    ::= { dot20AnCarrierConfigEntry 33 }

dot20AnNumRLControlSubbands OBJECT-TYPE
    SYNTAX        Integer32 (1..8)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the number of control subbands
         on the reverse link."
    ::= { dot20AnCarrierConfigEntry 34 }

dot20AnRACKBandwidthFactor OBJECT-TYPE
    SYNTAX        Integer32 (1..4)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the bandwidth reduction on the
         R-ACKCH."
    ::= { dot20AnCarrierConfigEntry 35 }

dot20AnHalfDuplexModeSupported OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to True if the access network
         supports half duplex terminals, and shall be set to False
         otherwise. If half-duplex terminals are supported, the access
```

```
         network should assign MAC IDs and channel assignments in a
         manner that enables half-duplex terminal operation. A
         half-duplex access terminal is not required to monitor forward
         link transmissions on a PHY Frame where it is scheduled to make
         a reverse link transmission."
    ::= { dot20AnCarrierConfigEntry 36 }

dot20AnRevLinkSilenceDuration OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine the duration of
         the Reverse Link Silence Interval. The Reverse Link Silence
         duration shall be equal to 2^n PHY Frames. In a region with
         asynchronous sectors, this attribute shall be set to a value
         larger than the timing offset between sectors."
    ::= { dot20AnCarrierConfigEntry 37 }

dot20AnRevLinkSilencePeriod OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine the periodicity
         of occurrence the Reverse Link Silence Interval. The reverse
         link silence interval shall take the value
         ReverseLinkSilencePeriod = (1+n)*144000. The Reverse Link
         Silence Interval is defined as the time interval of duration
         ReverseLinkSilenceDuration RL PHY Frames that starts at
         superframe index m that satisfies the following equation: m mod
         (ReverseLinkSilencePeriod) = 0"
    ::= { dot20AnCarrierConfigEntry 38 }

dot20AnTransmitPower OBJECT-TYPE
    SYNTAX        Integer32 (0..63)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the transmit power of the
         sector in units of dBm"
    ::= { dot20AnCarrierConfigEntry 39 }

dot20AnCommonPilotPower OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "The attribute's value noted n shall determine the power
         spectral density of the F-CPICH during the FL PHY frame
         relative to the F-ACQCH. The pilot power density shall be equal
         to (-4 + n*0.5) dB."
    ::= { dot20AnCarrierConfigEntry 40 }

dot20AnAuxPilotPower OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "The attribute's value noted n shall determine the power
         spectral density of the F-AuxPICH relative to the F-ACQCH. The
         pilot power density shall be equal to (-4 + n*0.5) dB."
    ::= { dot20AnCarrierConfigEntry 41 }

dot20AnPreamblePilotPower OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
```

```
        DESCRIPTION
            "The attribute's value noted n shall determine the power
             spectral density of the F-CPICH during the superframe preamble
             relative to the F-ACQCH. The pilot power density shall be equal
             to (-4 + n*0.5) dB."
        ::= { dot20AnCarrierConfigEntry 42 }

    dot20AnMACIDRange OBJECT-TYPE
        SYNTAX        INTEGER {
            upTo63(1),
            upTo127(2),
            upTo255(3),
            upTo511(4),
            upTo1023(5),
            upTo2047(6)
        }
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "This attribute shall be set to indicate the range of assigned
             MACID values in the sector."
        ::= { dot20AnCarrierConfigEntry 43 }

    dot20AnFLPCReportInterval OBJECT-TYPE
        SYNTAX        Integer32 (1..16)
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "FLPC Report Interval determines the periodicity at which power
             control commands are sent to the access terminal in PHY
             Frames."
        ::= { dot20AnCarrierConfigEntry 44 }

    dot20AnRLCtrlPCMode OBJECT-TYPE
        SYNTAX        INTEGER {
            upDown(1),
            erasureBased(2)
        }
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "This attribute shall determine the closed loop power control
             mode of the sector."
        ::= { dot20AnCarrierConfigEntry 45 }

    dot20AnCtrlAccessOffset OBJECT-TYPE
        SYNTAX        Integer32 (0..7)
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "This attribute shall be set to the initial gain of the R-CQICH
             over the R-ACH in units of dB expressed in 2's complement
             notation."
        ::= { dot20AnCarrierConfigEntry 46 }

    dot20AnBFCHPowerOffset OBJECT-TYPE
        SYNTAX        Integer32 (0..15)
        MAX-ACCESS    read-write
        STATUS        current
        DESCRIPTION
            "This attribute shall be set to power offset of the R-BFCH
             relative to the R-CQICHin units of dB expressed in 2's
             complement notation."
        ::= { dot20AnCarrierConfigEntry 47 }

    dot20AnSFCHPowerOffset OBJECT-TYPE
        SYNTAX        Integer32 (0..15)
        MAX-ACCESS    read-write
```

```
        STATUS         current
        DESCRIPTION
            "This attribute shall be set to power offset of the R-SFCH
             relative to the R-CQICHin units of dB expressed in 2's
             complement notation."
        ::= { dot20AnCarrierConfigEntry 48 }

    dot20AnPICHPowerOffset OBJECT-TYPE
        SYNTAX         Integer32 (0..15)
        MAX-ACCESS     read-write
        STATUS         current
        DESCRIPTION
            "This attribute shall be set to power offset of the R-PICH
             relative to the R-CQICHin units of dB expressed in 2's
             complement notation."
        ::= { dot20AnCarrierConfigEntry 49 }

    dot20AnReqChannelGain0 OBJECT-TYPE
        SYNTAX         Integer32 (0..15)
        MAX-ACCESS     read-write
        STATUS         current
        DESCRIPTION
            "This attribute shall be set to power offset of the R-REQCH
             relative to the R-CQICHin units of dB expressed in 2's
             complement notation."
        ::= { dot20AnCarrierConfigEntry 50 }

    dot20AnReqChannelGain1 OBJECT-TYPE
        SYNTAX         Integer32 (0..15)
        MAX-ACCESS     read-write
        STATUS         current
        DESCRIPTION
            "This attribute shall be set to power offset of the R-REQCH
             relative to the R-CQICHin units of dB expressed in 2's
             complement notation."
        ::= { dot20AnCarrierConfigEntry 51 }

    dot20AnReqChannelGain2 OBJECT-TYPE
        SYNTAX         Integer32 (0..15)
        MAX-ACCESS     read-write
        STATUS         current
        DESCRIPTION
            "This attribute shall be set to power offset of the R-REQCH
             relative to the R-CQICHin units of dB expressed in 2's
             complement notation."
        ::= { dot20AnCarrierConfigEntry 52 }

    dot20AnReqChannelGain3 OBJECT-TYPE
        SYNTAX         Integer32 (0..15)
        MAX-ACCESS     read-write
        STATUS         current
        DESCRIPTION
            "This attribute shall be set to power offset of the R-REQCH
             relative to the R-CQICHin units of dB expressed in 2's
             complement notation."
        ::= { dot20AnCarrierConfigEntry 53 }

    dot20AnErasureGain0 OBJECT-TYPE
        SYNTAX         Integer32 (0..15)
        MAX-ACCESS     read-write
        STATUS         current
        DESCRIPTION
            "This attribute's value noted n shall determine the transmit
             power of erasure sequences for different assignment sizes. The
             transmit power shall be equal to n-4 dB."
        ::= { dot20AnCarrierConfigEntry 54 }

    dot20AnErasureGain1 OBJECT-TYPE
```

```
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine the transmit
         power of erasure sequences for different assignment sizes. The
         transmit power shall be equal to n-4 dB."
    ::= { dot20AnCarrierConfigEntry 55 }

dot20AnErasureGain2 OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine the transmit
         power of erasure sequences for different assignment sizes. The
         transmit power shall be equal to n-4 dB."
    ::= { dot20AnCarrierConfigEntry 56 }

dot20AnErasureGain3 OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine the transmit
         power of erasure sequences for different assignment sizes. The
         transmit power shall be equal to n-4 dB."
    ::= { dot20AnCarrierConfigEntry 57 }

dot20AnAccessCycleDuration OBJECT-TYPE
    SYNTAX        Integer32 (1..4)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the duration of the access
         cycle in units of Control Segment Periods (as defined by the
         802.20 AIS spec Physical Layer)."
    ::= { dot20AnCarrierConfigEntry 59 }

dot20AnAccessSequencePartition OBJECT-TYPE
    SYNTAX        Integer32 (0..31)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall indicate the partition of the access
         sequence space to allow the access terminal to signal pilot
         power and buffer status information with the access sequence."
    ::= { dot20AnCarrierConfigEntry 60 }

dot20AnMaxProbesPerSequence OBJECT-TYPE
    SYNTAX        Integer32 (1..16)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the maximum number of probe
         sequences that can be part of one access sequence."
    ::= { dot20AnCarrierConfigEntry 61 }

dot20AnProbeRampUpStepSize OBJECT-TYPE
    SYNTAX        Integer32 (0..15)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine the power ramp
         up used for probes within a probe sequence and shall indicate a
         ramp up value of 0.5*(1+n) dB."
    ::= { dot20AnCarrierConfigEntry 62 }
```

```
dot20AnFastOSIEnabled OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This field shall be set to true if the F-SSCH transmitted by
         this sector contains a Fast OSI Segment. This field shall be
         set to false if the F-SSCH transmitted by this sector does not
         contain a Fast OSI Segment."
    ::= { dot20AnCarrierConfigEntry 63 }

dot20AnRDCHInitialPacketFormat OBJECT-TYPE
    SYNTAX        Integer32 (0..63)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the packet format that is used
         on the first transmission the access terminal makes on the
         R-DCH after getting an access grant."
    ::= { dot20AnCarrierConfigEntry 64 }

dot20AnPilotThreshold1 OBJECT-TYPE
    SYNTAX        Integer32 (0..3)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine PilotThreshold1
         used by the Access Channel MAC Protocol. The value shall be -2n
         dB."
    ::= { dot20AnCarrierConfigEntry 65 }

dot20AnPilotThreshold2 OBJECT-TYPE
    SYNTAX        Integer32 (0..3)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine PilotThreshold2
         used by the Access Channel MAC Protocol. The value shall be -2n
         dB."
    ::= { dot20AnCarrierConfigEntry 66 }

dot20AnOpenLoopAdjust OBJECT-TYPE
    SYNTAX        Integer32 (0..255)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute's value noted n shall determine the nominal
         power to be used by access terminal in the open loop power
         estimate. The value of nominal power shall be 70+n dB."
    ::= { dot20AnCarrierConfigEntry 67 }

dot20AnAccessRetryPersistance0 OBJECT-TYPE
    SYNTAX        Integer32 (0..7)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the persistence probability for
         determining access sequence backoff. If this attribute's value
         is set to n, the access terminal shall use 2^-n as the retry
         persistence."
    ::= { dot20AnCarrierConfigEntry 68 }

dot20AnAccessRetryPersistence1 OBJECT-TYPE
    SYNTAX        Integer32 (0..7)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the persistence probability for
```

```
          determining access sequence backoff. If this attribute's value
          is set to n, the access terminal shall use 2^-n as the retry
          persistence."
    ::= { dot20AnCarrierConfigEntry 69 }

dot20AnAccessRetryPersistence2 OBJECT-TYPE
    SYNTAX        Integer32 (0..7)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the persistence probability for
         determining access sequence backoff. If this attribute's value
         is set to n, the access terminal shall use 2^-n as the retry
         persistence."
    ::= { dot20AnCarrierConfigEntry 70 }

dot20AnAccessRetryPersistence3 OBJECT-TYPE
    SYNTAX        Integer32 (0..7)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall determine the persistence probability for
         determining access sequence backoff. If this attribute's value
         is set to n, the access terminal shall use 2^-n as the retry
         persistence."
    ::= { dot20AnCarrierConfigEntry 71 }

dot20AnSynchGroup OBJECT-TYPE
    SYNTAX        Integer32 (0..2147483647)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute specifies the synchronization group to which
         this carrier belongs to. All carriers (local, i.e. defined in
         the CarrierConfig table, or remote, i.e. defined in the
         NeighborCarriers table) which are synchronous with this carrier
         should belong to the same group. The value 0 indicates that the
         synchronization for this carrier is unknown."
    ::= { dot20AnCarrierConfigEntry 72 }

dot20AnCarrierConfigRowStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The status column used for creating, modifying, and deleting
         instances of the columnar objects in the CarrierConfig Table.
         If the implementor of this MIB has chosen not to implement
         'dynamic assignment' of carriers, this attribute is not useful
         and should return noSuchName upon SNMP request."
    DEFVAL         { active }
    ::= { dot20AnCarrierConfigEntry 73 }

dot20AnChannelBandsTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF Dot20AnChannelBandsEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "This table provides one row per 802.20 ChannelBand. This
         table's attributes specify the ChannelBand record of a
         particular ChannelBand which may be used for a sector defined
         in the SectorConfig table, or as a neighbor to one sector
         defined in the SectorConfig table."
    ::= { dot20AnOverheadMessages 4 }

dot20AnChannelBandsEntry OBJECT-TYPE
    SYNTAX        Dot20AnChannelBandsEntry
    MAX-ACCESS    not-accessible
```

```
    STATUS        current
    DESCRIPTION
        "An Entry (conceptual row) in the ChannelBands table. This
         table is indexed by ChannelBandIndex."
    INDEX
        { dot20AnChannelBandIndex }
    ::= { dot20AnChannelBandsTable 1 }

dot20AnChannelBandIndex OBJECT-TYPE
    SYNTAX        Integer32 (1..2147483647)
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "Index of the ChannelBand within the ChannelBands table."
    ::= { dot20AnChannelBandsEntry 1 }

dot20AnChannelBandRecordType OBJECT-TYPE
    SYNTAX        INTEGER {
        frequencySpecified(1),
        bandClass(2)
    }
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "ChannelBand Record Type for this ChannelBand. If equal to
         bandclass then ChannelBandClass and ChannelNumber are used to
         specify the Channel record for this channel. If equal to
         frequencySpecified then ChannelFwdLinkCenterFreq,
         ChannelRevLinkCenterFreq, ChannelFwdLinkSystemBw amd
         ChannelRevLinkSystemBw are used to specify the ChannelBand
         record for this ChannelBand."
    ::= { dot20AnChannelBandsEntry 2 }

dot20AnChannelBandClass OBJECT-TYPE
    SYNTAX        Integer32 (0..31)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the band class number
         corresponding to the frequency assignment of the ChannelBand
         specified by this record."
    ::= { dot20AnChannelBandsEntry 3 }

dot20AnChannelNumber OBJECT-TYPE
    SYNTAX        Integer32 (0..2047)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the Channel number
         corresponding to the frequency assignment of the ChannelBand
         specified by this record."
    ::= { dot20AnChannelBandsEntry 4 }

dot20AnChannelFwdLinkCenterFreq OBJECT-TYPE
    SYNTAX        Integer32 (0..16777215)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the value of the center
         frequency of the forward link channel band in units of KHz. The
         value 0 indicates that the center frequency of the forward link
         channel is unspecified."
    ::= { dot20AnChannelBandsEntry 5 }

dot20AnChannelRevLinkCenterFreq OBJECT-TYPE
    SYNTAX        Integer32 (0..16777215)
    MAX-ACCESS    read-write
    STATUS        current
```

```
    DESCRIPTION
        "This attribute shall be set to the value of the center
         frequency of the reverse link channel band in units of KHz. The
         value 0 indicates that the center frequency of the reverse link
         channel is unspecified."
    ::= { dot20AnChannelBandsEntry 6 }

dot20AnChannelFwdLinkSystemBw OBJECT-TYPE
    SYNTAX         Integer32 (0..65535)
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "The access network shall set this field to the value of the
         system bandwidth of the forward link channel in units of kHz.
         If this value is 0, then the forward link channel's system
         bandwidth is unspecified."
    ::= { dot20AnChannelBandsEntry 7 }

dot20AnChannelRevLinkSystemBw OBJECT-TYPE
    SYNTAX         Integer32 (0..65535)
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "The access network shall set this field to the value of the
         system bandwidth of the reverse link channel in units of kHz.
         If this value is 0, then the reverse link channel's system
         bandwidth is unspecified."
    ::= { dot20AnChannelBandsEntry 8 }

dot20AnChannelBandStatus OBJECT-TYPE
    SYNTAX         RowStatus
    MAX-ACCESS     read-create
    STATUS         current
    DESCRIPTION
        "The status column used for creating, modifying, and deleting
         instances of the columnar objects in the  ChannelBands Table.
         If the implementor of this MIB has chosen not to implement
         'dynamic assignment' of ChannelBands, this attribute is not
         useful and should return noSuchName upon SNMP request."
    DEFVAL          { active }
    ::= { dot20AnChannelBandsEntry 9 }

dot20AnNeighborCarriersTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF Dot20AnNeighborCarriersEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "This table provides one row per 802.20 neighbor carrier of a
         neighbor sector. This table's attributes specify the sector and
         carrier parameters of a particular neighbor carrier which may
         be used as a neighbor to one sector defined in the SectorConfig
         table."
    ::= { dot20AnOverheadMessages 5 }

dot20AnNeighborCarriersEntry OBJECT-TYPE
    SYNTAX         Dot20AnNeighborCarriersEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "An Entry (conceptual row) in the AnNeighborCarriers table.
         This table is indexed by ChannelBandIndex, NeighborSectorIndex,
         NeighborCarrierID."
    INDEX
        { dot20AnChannelBandIndex, dot20AnNeighborSectorIndex,
        dot20AnNeighborCarrierID }
    ::= { dot20AnNeighborCarriersTable 1 }

dot20AnNeighborSectorIndex OBJECT-TYPE
```

```
    SYNTAX        Integer32 (1..2147483647)
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "Index of the Neighbor Sector for this Neighbor Carrier within
         the ChannelBand."
    ::= { dot20AnNeighborCarriersEntry 1 }

dot20AnNeighborCarrierID OBJECT-TYPE
    SYNTAX        Integer32 (0..3)
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "CarrierID of the Neighbor Carrier for this Neighbor Sector.
         The CarrierID is unique within the NeighborSector."
    ::= { dot20AnNeighborCarriersEntry 2 }

dot20AnNeighborSectorID OBJECT-TYPE
    SYNTAX        OCTET STRING (SIZE(16))
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "Sector Address Identifier. The access network shall set the
         value of the SectorID according to the rules specified in IEEE
         802.20 AIS. The access terminal shall not assume anything about
         the format of the SectorID other than that it uniquely
         identifies the sector."
    ::= { dot20AnNeighborCarriersEntry 3 }

dot20AnNeighborPilotPN OBJECT-TYPE
    SYNTAX        Integer32 (0..4095)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the PilotPN of a neighboring
         sector that the access terminal should add to its Neighbor
         Set."
    ::= { dot20AnNeighborCarriersEntry 4 }

dot20AnNeighborTransmitPower OBJECT-TYPE
    SYNTAX        Integer32 (0..63)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to the transmit power of the
         sector in units of dBm."
    ::= { dot20AnNeighborCarriersEntry 5 }

dot20AnNghbrGloballySynch OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute shall be set to true if the sector transmitting
         this pilot is synchronous to system time."
    ::= { dot20AnNeighborCarriersEntry 6 }

dot20AnNgbhrSynchGroup OBJECT-TYPE
    SYNTAX        Integer32 (0..2147483647)
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "This attribute specifies the synchronization group to which
         this carrier belongs to. All carriers (local, i.e. defined in
         the CarrierConfig table, or remote, i.e. defined in the
         NeighborCarriers table) which are synchronous with this carrier
         should belong to the same group. The value 0 indicates that the
         synchronization for this carrier is unknown."
```

```
    ::= { dot20AnNeighborCarriersEntry 7 }

dot20AnNeighborCarrierStatus OBJECT-TYPE
    SYNTAX         RowStatus
    MAX-ACCESS     read-create
    STATUS         current
    DESCRIPTION
        "The status column used for creating, modifying, and deleting
         instances of the columnar objects in the  NeighborCarriers
         Table. If the implementor of this MIB has chosen not to
         implement 'dynamic assignment' of neighbor carriers this
         attribute is not useful and should return noSuchName upon SNMP
         request."
    DEFVAL          { active }
    ::= { dot20AnNeighborCarriersEntry 8 }

dot20AnOtherTechNghbrsTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF Dot20AnOtherTechNghbrsEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "This table provides one row per other technology neighbor
         channel. This table's attributes specify the technology type
         and neighborlist of a particular neighbor channel which may be
         used by one sector defined in the SectorConfig table for
         inter-technology handoff."
    ::= { dot20AnOverheadMessages 6 }

dot20AnOtherTechNghbrsEntry OBJECT-TYPE
    SYNTAX         Dot20AnOtherTechNghbrsEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "An Entry (conceptual row) in the AnOtherTechNghbrs table. This
         table is indexed by Sector (ifIndex) and OtherTechnologyIndex"
    INDEX
        { ifIndex, dot20AnOtherTechnologyIndex }
    ::= { dot20AnOtherTechNghbrsTable 1 }

dot20AnOtherTechnologyIndex OBJECT-TYPE
    SYNTAX         Integer32 (1..2147483647)
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "The neighbor other technology entry index"
    ::= { dot20AnOtherTechNghbrsEntry 1 }

dot20AnTechnologyType OBJECT-TYPE
    SYNTAX         INTEGER {
        l2tp(1),
        w802dot11(2),
        cdma20001x(3),
        cdma20001xev(4),
        gsm(5),
        wcdma(6)
    }
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "This attribute shall be set to the type of other technology"
    ::= { dot20AnOtherTechNghbrsEntry 2 }

dot20AnTechNghbrListLength OBJECT-TYPE
    SYNTAX         Integer32 (0..255)
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "This attribute shall be set the length, in bytes, of the
```

```
            neighbor list information for the other technology."
        ::= { dot20AnOtherTechNghbrsEntry 3 }

    dot20AnTechnologyNeighborList OBJECT-TYPE
        SYNTAX          OCTET STRING (SIZE(256))
        MAX-ACCESS      read-write
        STATUS          current
        DESCRIPTION
            "This attribute shall be set to the neighbor list information
             for the other technology."
        ::= { dot20AnOtherTechNghbrsEntry 4 }

    dot20AnOtherTechNghbrRowStatus OBJECT-TYPE
        SYNTAX          RowStatus
        MAX-ACCESS      read-create
        STATUS          current
        DESCRIPTION
            "The status column used for creating, modifying, and deleting
             instances of the columnar objects in the OtherTechNghbrs Table.
             If the implementor of this MIB has chosen not to implement
             'dynamic assignment' of other technology neighbors, this
             attribute is not useful and should return noSuchName upon SNMP
             request."
        DEFVAL           { active }
        ::= { dot20AnOtherTechNghbrsEntry 5 }

    dot20AnNeighborListTable OBJECT-TYPE
        SYNTAX          SEQUENCE OF Dot20AnNeighborListEntry
        MAX-ACCESS      not-accessible
        STATUS          current
        DESCRIPTION
            "This table defines the neighbor lists for the sectors defined
             in the SectorConfig table. Each row in this table indexed per
             sector (ifIndex) specifies a pointer to a neighbor carrier of
             this sector."
        ::= { dot20AnOverheadMessages 7 }

    dot20AnNeighborListEntry OBJECT-TYPE
        SYNTAX          Dot20AnNeighborListEntry
        MAX-ACCESS      not-accessible
        STATUS          current
        DESCRIPTION
            "An Entry (conceptual row) in the AnNeighborList table. This
             table is indexed by Sector (ifIndex) and NeighborIndex indexing
             each neighbor carrier for a particular Sector."
        INDEX
            { ifIndex, dot20AnNeighborIndex }
        ::= { dot20AnNeighborListTable 1 }

    dot20AnNeighborIndex OBJECT-TYPE
        SYNTAX          Integer32 (1..32)
        MAX-ACCESS      not-accessible
        STATUS          current
        DESCRIPTION
            "This index identifies one neighbor carrier for a Sector."
        ::= { dot20AnNeighborListEntry 1 }

    dot20AnNeighborCarrierPointer OBJECT-TYPE
        SYNTAX          RowPointer
        MAX-ACCESS      read-create
        STATUS          current
        DESCRIPTION
            "This attribute points to an instance of carrier in
             CarrierConfig table or in NeighborCarriers table. This carrier
             is defined as a neighbor of the sector identified by the
             ifIndex of this attribute's entry."
        ::= { dot20AnNeighborListEntry 2 }
```

```
dot20AnNeighborRowStatus OBJECT-TYPE
    SYNTAX         RowStatus
    MAX-ACCESS     read-create
    STATUS         current
    DESCRIPTION
        "The status column used for creating, modifying, and deleting
         instances of the columnar objects in the  NeighborList Table.
         If the implementor of this MIB has chosen not to implement
         'dynamic assignment' of neighbor list entries this attribute is
         not useful and should return noSuchName upon SNMP request."
    DEFVAL          { active }
    ::= { dot20AnNeighborListEntry 3 }

dot20AnSectorToIfIndexTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF Dot20AnSectorToIfIndexEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "This table can be used to find the ifIndex of an 802.20
         interface based on its SectorID and ChannelBand information
         (reverse mapping of the Sector Config table)."
    ::= { dot20An 2 }

dot20AnSectorToIfIndexEntry OBJECT-TYPE
    SYNTAX         Dot20AnSectorToIfIndexEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "An Entry (conceptual row) in the AnSectorToIfIndex table."
    INDEX
        { dot20AnSectorID, ifIndex }
    ::= { dot20AnSectorToIfIndexTable 1 }

dot20AnIfChannelBandRef OBJECT-TYPE
    SYNTAX         Integer32
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "The reference to the ChannelBand defined in ChannelBands table
         (dot20AnChannelBandIndex)"
    ::= { dot20AnSectorToIfIndexEntry 1 }

dot20Cmn OBJECT-IDENTITY
    STATUS         current
    DESCRIPTION
        "Common configuration and statistics."
    ::= { ieee802dot20 2 }

dot20CmnMac OBJECT-IDENTITY
    STATUS         current
    DESCRIPTION
        "MAC layer objects"
    ::= { dot20Cmn 1 }

dot20CmnSessionControl OBJECT IDENTIFIER ::= { dot20CmnMac 1 }

dot20CmnSessionMgtProtocol OBJECT IDENTIFIER ::= { dot20CmnSessionControl 1 }

dot20CmnSessionOpenCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of session opened"
    ::= { dot20CmnSessionMgtProtocol 1 }

dot20CmnSessionCloseCounts OBJECT-TYPE
    SYNTAX         Counter64
```

```
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of session closed"
    ::= { dot20CmnSessionMgtProtocol 2 }

dot20CmnSessionFailureCounts OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of session open/close failures"
    ::= { dot20CmnSessionMgtProtocol 3 }

dot20CmnSessionConfigProtocol OBJECT IDENTIFIER ::= { dot20CmnSessionControl 2
}

dot20CmnSessionConfigTokenTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF Dot20CmnSessionConfigTokenEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "This table provides one row per supported session
         configuration token."
    ::= { dot20CmnSessionConfigProtocol 1 }

dot20CmnSessionConfigTokenEntry OBJECT-TYPE
    SYNTAX        Dot20CmnSessionConfigTokenEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An Entry (conceptual row) in the SessionConfigToken table.
         This table is indexed by TokenIndex."
    INDEX
        { dot20CmnSessionTokenIndex }
    ::= { dot20CmnSessionConfigTokenTable 1 }

dot20CmnSessionTokenIndex OBJECT-TYPE
    SYNTAX        Integer32 (1..2147483647)
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "The index of a supported session configuration token."
    ::= { dot20CmnSessionConfigTokenEntry 1 }

dot20CmnSessionConfToken OBJECT-TYPE
    SYNTAX        Integer32 (0..65535)
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The value of a supported session configuration token."
    ::= { dot20CmnSessionConfigTokenEntry 2 }

dot20CmnSessionConfTokenStatus OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
        "The status column used for creating, modifying, and deleting
         instances of the columnar objects in the SessionConfigToken
         Table. If the implementor of this MIB has chosen not to
         implement 'dynamic assignment' of session configuration tokens,
         this attribute is not useful and should return noSuchName upon
         SNMP request."
    DEFVAL         { active }
    ::= { dot20CmnSessionConfigTokenEntry 3 }

dot20CmnSecurityCcontrol OBJECT IDENTIFIER ::= { dot20CmnMac 2 }
```

```
dot20CmnKeyExchangeProtocol OBJECT IDENTIFIER ::= { dot20CmnSecurityCcontrol 1
}

dot20CmnKeyExchangeAttemptCounts OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of key exchanges attempts"
    ::= { dot20CmnKeyExchangeProtocol 1 }

dot20CmnKeyExchangeFailureCounts OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of key exchanges failures"
    ::= { dot20CmnKeyExchangeProtocol 2 }

dot20CmnLowerMACControl OBJECT IDENTIFIER ::= { dot20CmnMac 3 }

dot20CmnConnectedState OBJECT IDENTIFIER ::= { dot20CmnLowerMACControl 1 }

dot20CmnActiveConnectionCounts OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of current active connections"
    ::= { dot20CmnConnectedState 1 }

dot20CmnConnectionAttemptCounts OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of connection attempts"
    ::= { dot20CmnConnectedState 2 }

dot20CmnConnectionFailureCounts OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of connection failures during connection attempt."
    ::= { dot20CmnConnectedState 3 }

dot20CmnConnectionDropCounts OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of dropped connection after a connection has been
         established."
    ::= { dot20CmnConnectedState 4 }

dot20CmnConnectionReleaseCounts OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of connection release after a connection has been
         established."
    ::= { dot20CmnConnectedState 5 }

dot20CmnConvergence OBJECT IDENTIFIER ::= { dot20CmnMac 4 }
```

```
dot20CmnSignalingTransport OBJECT IDENTIFIER ::= { dot20CmnConvergence 1 }

dot20CmnSigTransportStatsTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Dot20CmnSigTransportStatsEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This table provides one row of Signaling Transport statistics
         per 802.20 interface."
    ::= { dot20CmnSignalingTransport 1 }

dot20CmnSigTransportStatsEntry OBJECT-TYPE
    SYNTAX          Dot20CmnSigTransportStatsEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An Entry (conceptual row) in the SigTransportStats table. This
         table is indexed by IfIndex. ifIndex: Each IEEE 802.20
         interface is represented by an ifEntry."
    INDEX
        { ifIndex }
    ::= { dot20CmnSigTransportStatsTable 1 }

dot20CmnSlpTxBytes OBJECT-TYPE
    SYNTAX          Counter64
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Number of SLP bytes transmitted"
    ::= { dot20CmnSigTransportStatsEntry 1 }

dot20CmnSlpReTxBytes OBJECT-TYPE
    SYNTAX          Counter64
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Number of SLP bytes retransmitted"
    ::= { dot20CmnSigTransportStatsEntry 2 }

dot20CmnSlpTxDroppedBytes OBJECT-TYPE
    SYNTAX          Counter64
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Number of SLP bytes dropped before transmission"
    ::= { dot20CmnSigTransportStatsEntry 3 }

dot20CmnSlpTxStatus OBJECT-TYPE
    SYNTAX          Counter64
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Number of ReceiverStatus messages transmitted"
    ::= { dot20CmnSigTransportStatsEntry 4 }

dot20CmnSlpRxBytes OBJECT-TYPE
    SYNTAX          Counter64
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Number of SLP Bytes received"
    ::= { dot20CmnSigTransportStatsEntry 5 }

dot20CmnSlpRxStatus OBJECT-TYPE
    SYNTAX          Counter64
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
```

```
        "Number of ReceiverStatus messages received"
    ::= { dot20CmnSigTransportStatsEntry 6 }

dot20CmnSlpTxPackets OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of SLP Packets transmitted"
    ::= { dot20CmnSigTransportStatsEntry 7 }

dot20CmnSlpReTxPackets OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of SLP Packets retransmitted"
    ::= { dot20CmnSigTransportStatsEntry 8 }

dot20CmnSlpTxDropPackets OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of SLP Packets dropped before transmission"
    ::= { dot20CmnSigTransportStatsEntry 9 }

dot20CmnSlpRxPackets OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of SLP Packets received"
    ::= { dot20CmnSigTransportStatsEntry 10 }

dot20CmnSlpTxACKTimeouts OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of ACK Timeouts"
    ::= { dot20CmnSigTransportStatsEntry 11 }

dot20CmnDataTransport OBJECT IDENTIFIER ::= { dot20CmnConvergence 2 }

dot20CmnDataTransportStatsTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF Dot20CmnDataTransportStatsEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "This table provides one row of Data Transport statistics per
         802.20 interface"
    ::= { dot20CmnDataTransport 1 }

dot20CmnDataTransportStatsEntry OBJECT-TYPE
    SYNTAX        Dot20CmnDataTransportStatsEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An Entry (conceptual row) in the DataTransportStats table.
         This table is indexed by IfIndex. ifIndex: Each IEEE 802.20
         interface is represented by an ifEntry."
    INDEX
        { ifIndex }
    ::= { dot20CmnDataTransportStatsTable 1 }

dot20CmnRlpTxBytes OBJECT-TYPE
    SYNTAX        Counter64
```

```
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of RLP bytes of payload transmitted"
    ::= { dot20CmnDataTransportStatsEntry 1 }

dot20CmnRlpReTxBytes OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of RLP bytes of payload retransmitted"
    ::= { dot20CmnDataTransportStatsEntry 2 }

dot20CmnRlpTxDropBytes OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of RLP bytes of dropped before transmission"
    ::= { dot20CmnDataTransportStatsEntry 3 }

dot20CmnRlpTxStatus OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of RLP ReceiverStatus messages transmitted"
    ::= { dot20CmnDataTransportStatsEntry 4 }

dot20CmnRlpRxBytes OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of RLP bytes of payload received"
    ::= { dot20CmnDataTransportStatsEntry 5 }

dot20CmnRlpRxStatus OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of RLP ReceiverStatus messages received"
    ::= { dot20CmnDataTransportStatsEntry 6 }

dot20CmnRlpTxPackets OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of RLP Packets transmitted"
    ::= { dot20CmnDataTransportStatsEntry 7 }

dot20CmnRlpReTxPackets OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of RLP Packets retransmitted"
    ::= { dot20CmnDataTransportStatsEntry 8 }

dot20CmnRlpTxrDropPackets OBJECT-TYPE
    SYNTAX        Counter64
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Number of RLP Packets dropped before transmission"
```

```
    ::= { dot20CmnDataTransportStatsEntry 9 }

dot20CmnRlpRxPackets OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of RLP Packets received"
    ::= { dot20CmnDataTransportStatsEntry 10 }

dot20CmnRlpTxNAKTimeouts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of NAK Timeouts"
    ::= { dot20CmnDataTransportStatsEntry 11 }

dot20CmnActiveReservationsCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of Active Reservations"
    ::= { dot20CmnDataTransportStatsEntry 12 }

dot20CmnIdleReservationsCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of Idle Reservations"
    ::= { dot20CmnDataTransportStatsEntry 13 }

dot20CmnReservationOpenCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of Reservations Open requests"
    ::= { dot20CmnDataTransportStatsEntry 14 }

dot20CmnReservationCloseCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of Reservations Close requests"
    ::= { dot20CmnDataTransportStatsEntry 15 }

dot20CmnReservationFailCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of Failed Reservations requests"
    ::= { dot20CmnDataTransportStatsEntry 16 }

dot20CmnSecurity OBJECT IDENTIFIER ::= { dot20CmnMac 5 }

dot20CmnAuthProtocol OBJECT IDENTIFIER ::= { dot20CmnSecurity 1 }

dot20CmnAuthStatsTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF Dot20CmnAuthStatsEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "This table provides one row of Authentication protocol
```

```
          statistics per 802.20 interface"
     ::= { dot20CmnAuthProtocol 1 }

dot20CmnAuthStatsEntry OBJECT-TYPE
     SYNTAX        Dot20CmnAuthStatsEntry
     MAX-ACCESS    not-accessible
     STATUS        current
     DESCRIPTION
         "An Entry (conceptual row) in the AuthStats table. This table
          is indexed by IfIndex. ifIndex: Each IEEE 802.20 interface is
          represented by an ifEntry."
     INDEX
         { ifIndex }
     ::= { dot20CmnAuthStatsTable 1 }

dot20CmnAuthFailureCounts OBJECT-TYPE
     SYNTAX        Counter64
     MAX-ACCESS    read-only
     STATUS        current
     DESCRIPTION
         "Number of Authentication failures"
     ::= { dot20CmnAuthStatsEntry 1 }

dot20CmnAuthSuccessCounts OBJECT-TYPE
     SYNTAX        Counter64
     MAX-ACCESS    read-only
     STATUS        current
     DESCRIPTION
         "Number of successful Authentications"
     ::= { dot20CmnAuthStatsEntry 2 }

dot20CmnLowerMAC OBJECT IDENTIFIER ::= { dot20CmnMac 6 }

dot20CmnLMACPacketStatsTable OBJECT-TYPE
     SYNTAX        SEQUENCE OF Dot20CmnLMACPacketStatsEntry
     MAX-ACCESS    not-accessible
     STATUS        current
     DESCRIPTION
         "This table provides one row of Lower MAC protocol statistics
          per 802.20 interface, packet format and nb of ARQ attempts
          needed in order to successfully transmit/receive a packet."
     ::= { dot20CmnLowerMAC 1 }

dot20CmnLMACPacketStatsEntry OBJECT-TYPE
     SYNTAX        Dot20CmnLMACPacketStatsEntry
     MAX-ACCESS    not-accessible
     STATUS        current
     DESCRIPTION
         "An Entry (conceptual row) in the LMACPacketStats table. This
          table is indexed by IfIndex, PacketFormatIndex and
          ARQAttemptsIndex."
     INDEX
         { ifIndex, dot20CmnPacketFormatIndex, dot20CmnARQAttemptsIndex
         }
     ::= { dot20CmnLMACPacketStatsTable 1 }

dot20CmnPacketFormatIndex OBJECT-TYPE
     SYNTAX        Integer32 (0..15)
     MAX-ACCESS    not-accessible
     STATUS        current
     DESCRIPTION
         "The packet format index as defined in 802.20 AIS spec."
     ::= { dot20CmnLMACPacketStatsEntry 1 }

dot20CmnARQAttemptsIndex OBJECT-TYPE
     SYNTAX        Integer32 (0..15)
     MAX-ACCESS    not-accessible
     STATUS        current
```

```
    DESCRIPTION
        "Number of ARQ attempts that were needed in order to transmit
         or receive a packet. Index 0 means that the packets failed to
         be transmitted/received."
    ::= { dot20CmnLMACPacketStatsEntry 2 }

dot20CmnFwdTxPacketCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of transmitted packets"
    ::= { dot20CmnLMACPacketStatsEntry 3 }

dot20CmnRevRxPacketCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of received packets"
    ::= { dot20CmnLMACPacketStatsEntry 4 }

dot20CmnLMACStatsTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF Dot20CmnLMACStatsEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "This table provides one row of Lower MAC protocol statistics
         per 802.20 interface and packet formats."
    ::= { dot20CmnLowerMAC 2 }

dot20CmnLMACStatsEntry OBJECT-TYPE
    SYNTAX         Dot20CmnLMACStatsEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "An Entry (conceptual row) in the LMACStats table. This table
         is indexed by IfIndex, PacketFormatIndex."
    INDEX
        { ifIndex, dot20CmnPacketFormatIndex }
    ::= { dot20CmnLMACStatsTable 1 }

dot20CmnFLABCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of FL SSCH assignments"
    ::= { dot20CmnLMACStatsEntry 1 }

dot20CmnRLABCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of RL SSCH assignments"
    ::= { dot20CmnLMACStatsEntry 2 }

dot20CmnAccessGrantCounts OBJECT-TYPE
    SYNTAX         Counter64
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Number of Access Grants"
    ::= { dot20CmnLMACStatsEntry 3 }

dot20Conformance OBJECT IDENTIFIER ::= { ieee802dot20 4 }
```

```
dot20Groups OBJECT IDENTIFIER ::= { dot20Conformance 1 }

dot20CmnSessionMgtPGroup OBJECT-GROUP
    OBJECTS
        { dot20CmnSessionCloseCounts, dot20CmnSessionFailureCounts,
        dot20CmnSessionOpenCounts }
    STATUS        current
    DESCRIPTION
        "The session management protocol statistics"
    ::= { dot20Groups 1 }

dot20CmnSessionConfigPGroup OBJECT-GROUP
    OBJECTS
        { dot20CmnSessionConfToken }
    STATUS        current
    DESCRIPTION
        "The session configuration protocol configuration"
    ::= { dot20Groups 2 }

dot20CmnSessionConfigPGroup2 OBJECT-GROUP
    OBJECTS
        { dot20CmnSessionConfTokenStatus }
    STATUS        current
    DESCRIPTION
        "This group should be implemented if assignment of tokens is
         performed through snmp."
    ::= { dot20Groups 3 }

dot20CmnKeyExchangePGroup OBJECT-GROUP
    OBJECTS
        { dot20CmnKeyExchangeAttemptCounts,
        dot20CmnKeyExchangeFailureCounts }
    STATUS        current
    DESCRIPTION
        "The key exchange protocol statistics"
    ::= { dot20Groups 4 }

dot20CmnConnectedStatePGroup OBJECT-GROUP
    OBJECTS
        { dot20CmnActiveConnectionCounts,
        dot20CmnConnectionAttemptCounts, dot20CmnConnectionDropCounts,
        dot20CmnConnectionFailureCounts, dot20CmnConnectionReleaseCounts
        }
    STATUS        current
    DESCRIPTION
        "The connected state protocol statistics"
    ::= { dot20Groups 5 }

dot20CmnSigTransportGroup OBJECT-GROUP
    OBJECTS
        { dot20CmnSlpReTxBytes, dot20CmnSlpReTxPackets,
        dot20CmnSlpRxBytes, dot20CmnSlpRxPackets, dot20CmnSlpRxStatus,
        dot20CmnSlpTxACKTimeouts, dot20CmnSlpTxBytes,
        dot20CmnSlpTxDropPackets, dot20CmnSlpTxDroppedBytes,
        dot20CmnSlpTxPackets, dot20CmnSlpTxStatus }
    STATUS        current
    DESCRIPTION
        "The signaling transport statistics"
    ::= { dot20Groups 6 }

dot20CmnDataTransportGroup OBJECT-GROUP
    OBJECTS
        { dot20CmnActiveReservationsCounts,
        dot20CmnIdleReservationsCounts, dot20CmnReservationCloseCounts,
        dot20CmnReservationFailCounts, dot20CmnReservationOpenCounts,
        dot20CmnRevRxPacketCounts, dot20CmnRlpReTxPackets,
        dot20CmnRlpReTxBytes, dot20CmnRlpRxPackets, dot20CmnRlpRxBytes,
        dot20CmnRlpRxStatus, dot20CmnRlpTxNAKTimeouts,
```

```
        dot20CmnRlpTxPackets, dot20CmnRlpTxBytes,
        dot20CmnRlpTxDropBytes, dot20CmnRlpTxStatus,
        dot20CmnRlpTxrDropPackets }
    STATUS        current
    DESCRIPTION
        "The data transport statistics"
    ::= { dot20Groups 7 }

dot20CmnAuthGroup OBJECT-GROUP
    OBJECTS
        { dot20CmnAuthFailureCounts, dot20CmnAuthSuccessCounts }
    STATUS        current
    DESCRIPTION
        "The authentication protocol statistics"
    ::= { dot20Groups 8 }

dot20CmnLowerMACGroup OBJECT-GROUP
    OBJECTS
        { dot20CmnAccessGrantCounts, dot20CmnFLABCounts,
        dot20CmnFwdTxPacketCounts, dot20CmnRLABCounts,
        dot20CmnRevRxPacketCounts }
    STATUS        current
    DESCRIPTION
        "The lower mac sublayer statistics"
    ::= { dot20Groups 9 }

dot20AnIdleStatePGroup OBJECT-GROUP
    OBJECTS
        { dot20AnAccessAttemptCounts, dot20AnAccessAttemptFailCounts,
        dot20AnPageAttemptCounts, dot20AnPageFailureCounts }
    STATUS        current
    DESCRIPTION
        "The An idle state protocol statistics"
    ::= { dot20Groups 10 }

dot20AnOverheadGroup OBJECT-GROUP
    OBJECTS
        { dot20AnAccessCycleDuration, dot20AnAccessRetryPersistance0,
        dot20AnAccessRetryPersistence1, dot20AnAccessRetryPersistence2,
        dot20AnAccessRetryPersistence3, dot20AnAccessSequencePartition,
        dot20AnAuxPilotPower, dot20AnBFCHBeamCodeBookIndex,
        dot20AnBFCHPowerOffset, dot20AnBlockHoppingEnabled,
        dot20AnCPLength, dot20AnCarrierID, dot20AnChannelBandClass,
        dot20AnChannelFwdLinkCenterFreq, dot20AnChannelFwdLinkSystemBw,
        dot20AnChannelNumber, dot20AnChannelBandRecordType,
        dot20AnChannelBandRef, dot20AnChannelRevLinkCenterFreq,
        dot20AnChannelRevLinkSystemBw, dot20AnCommonPilotPower,
        dot20AnCountryCode, dot20AnCtrlAccessOffset,
        dot20AnEffectiveNumAntennas, dot20AnEnableAuxPilotStaggering,
        dot20AnEnableCmnPilotStaggering, dot20AnErasureGain0,
        dot20AnErasureGain1, dot20AnErasureGain2, dot20AnErasureGain3,
        dot20AnFLChannelTreeIndex, dot20AnFLDPISectorOffset,
        dot20AnFLDPISectorScramble, dot20AnFLDiversityHoppingMode,
        dot20AnFLFirstRestrSetSubband, dot20AnFLIntraCellCommonHopping,
        dot20AnFLNumRestrSetSubbands, dot20AnFLNumSDMADimensions,
        dot20AnFLNumSubbands, dot20AnFLPCReportInterval,
        dot20AnFLReservedInterlaces, dot20AnFLSectorHopSeed,
        dot20AnFastOSIEnabled, dot20AnHalfDuplexModeSupported,
        dot20AnIfChannelBandRef, dot20AnLatitude, dot20AnLeapSeconds,
        dot20AnLocalTimeOffset, dot20AnLongitude, dot20AnMACIDRange,
        dot20AnMaxProbesPerSequence, dot20AnMaximumRevision,
        dot20AnMinimumRevision, dot20AnNFLBurst, dot20AnNRLBurst,
        dot20AnNeighborCarrierID, dot20AnNeighborCarrierPointer,
        dot20AnNeighborPilotPN, dot20AnNeighborSectorID,
        dot20AnNeighborTransmitPower, dot20AnNgbhrSynchGroup,
        dot20AnNghbrGloballySynch, dot20AnNumCarriers,
        dot20AnNumCmnPilotTxAntennas, dot20AnNumFLReservedSubbands,
        dot20AnNumGuardSubcarriers, dot20AnNumPilots,
```

```
        dot20AnNumRLControlSubbands, dot20AnRACKBandwidthFactor,
        dot20AnOpenLoopAdjust, dot20AnPICHPowerOffset,
        dot20AnPhysicalMode, dot20AnPilotPN, dot20AnPilotThreshold1,
        dot20AnPilotThreshold2, dot20AnPrNumGuardSubcarriers,
        dot20AnPreamblePilotPower, dot20AnProbeRampUpStepSize,
        dot20AnRDCHInitialPacketFormat, dot20AnRLChannelTreeIndex,
        dot20AnRLCtrlPCMode, dot20AnRLDPISectorOffset,
        dot20AnRLDPISectorScramble, dot20AnRLDiversityHoppingMode,
        dot20AnRLIntraCellCommonHopping, dot20AnRLNumSDMADimensions,
        dot20AnRLNumSubbands, dot20AnRLRestrictedSetBitmap,
        dot20AnRLSectorHopSeed, dot20AnRPICHEnabled,
        dot20AnRegistrationRadius, dot20AnRegistrationZoneCode,
        dot20AnRegistrationZoneIncluded, dot20AnRegistrationZoneMaxAge,
        dot20AnReqChannelGain0, dot20AnReqChannelGain1,
        dot20AnReqChannelGain2, dot20AnReqChannelGain3,
        dot20AnRevLinkSilenceDuration, dot20AnRevLinkSilencePeriod,
        dot20AnSFCHPowerOffset, dot20AnSSCHModSymbolsPerBlock,
        dot20AnSSCHNumBlocks, dot20AnSSCHNumHopports, dot20AnSectorID,
        dot20AnSubnetMask, dot20AnSynchGroup, dot20AnSynchronousSystem,
        dot20AnTechNghbrListLength, dot20AnTechnologyNeighborList,
        dot20AnTechnologyType, dot20AnTransmitPower }
    STATUS        current
    DESCRIPTION
        "The overhead messages protocol configuration"
    ::= { dot20Groups 11 }

dot20AnOverheadGroup2 OBJECT-GROUP
    OBJECTS
        { dot20AnCarrierConfigRowStatus, dot20AnChannelBandStatus,
        dot20AnNeighborCarrierStatus, dot20AnNeighborRowStatus,
        dot20AnOtherTechNghbrRowStatus, dot20AnSectorConfigRowStatus }
    STATUS        current
    DESCRIPTION
        "The overhead messages protocol configuration This group should
         be implemented if assignment of 802.20 interfaces is to be
         preformed through snmp."
    ::= { dot20Groups 12 }

dot20Compliances OBJECT IDENTIFIER ::= { dot20Conformance 2 }

dot20AnCompliance MODULE-COMPLIANCE
    STATUS         current
    DESCRIPTION
        "The compliance statement for SNMPv2 entities that implement
         the IEEE 802.20 MIB for the An."
    MODULE         IEEE802dot20-MIB
        MANDATORY-GROUPS
            { dot20AnIdleStatePGroup, dot20AnOverheadGroup,
            dot20CmnAuthGroup, dot20CmnConnectedStatePGroup,
            dot20CmnDataTransportGroup, dot20CmnKeyExchangePGroup,
            dot20CmnLowerMACGroup, dot20CmnSessionConfigPGroup,
            dot20CmnSessionMgtPGroup, dot20CmnSigTransportGroup }
        GROUP          dot20AnOverheadGroup2
        DESCRIPTION
            "This group is required only if 'dynamic assignment' of
             rows in the OverheadGroup tables is supported."
        GROUP          dot20CmnSessionConfigPGroup2
        DESCRIPTION
            "This group is only implemented if 'dynamic assignment' of
             rows in the SessionConfigP group tables is supported."
    ::= { dot20Compliances 1 }

END
```

14 System Overview of 625k-MC (625kiloHertz-spaced MultiCarrier) Mode

14.1 Scope

This specification defines the physical layer, medium access control, and the signaling layer for 625k-MC, see 1.1 of ATIS-PP-0700004-2005, High Capacity-Spatial Division Multiple Access (HC-SDMA), September 2005. For brevity, references to the baseline specification: ATIS-PP-0700004-2005, High Capacity-Spatial Division Multiple Access (HC-SDMA), September 2005 will be denoted as "HC-SDMA [25]" in this document. Unless otherwise specified in this document, the requirements of HC-SDMA [25] shall apply to the 625k-MC mode of 802.20.

14.2 Architecture Reference Model

The architecture reference model for 625k-MC Mode is presented in Figure 121. The reference model includes the air interface between the user terminal and the access network. While the protocols specified in HC-SDMA [25] are referred to as basic protocols, this document defines only the additional specifications that are needed to supplement the protocols, and are referred to as enhanced specifications with reference to Base-Draft Specifications of HC-SDMA [25].

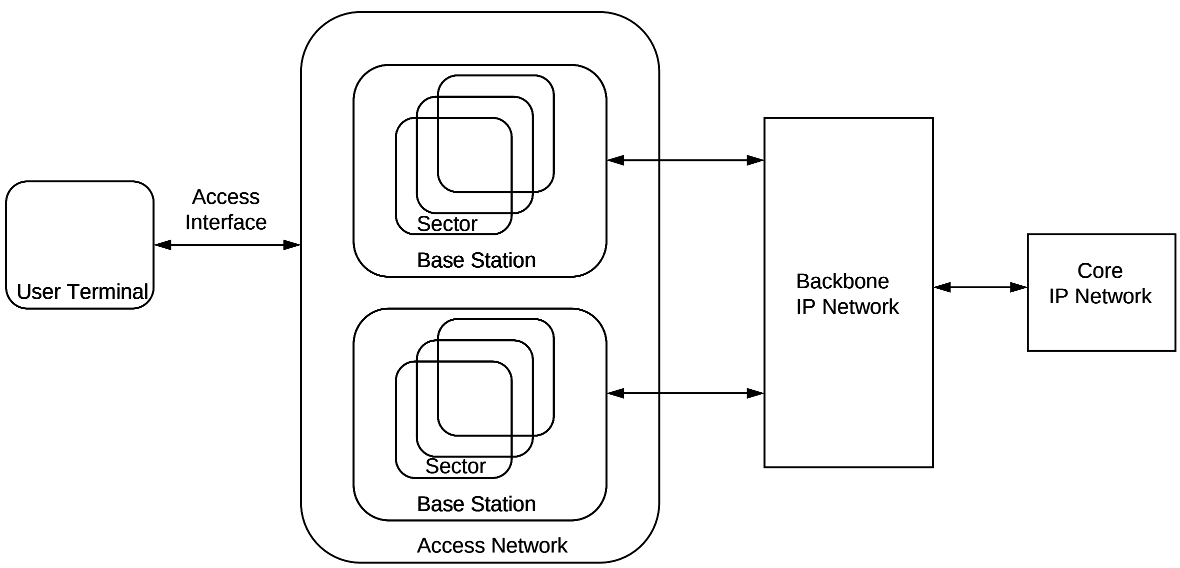

Figure 121 Architecture reference model

The functional units of the reference architecture in Figure 121 are:

Access Network (AN)	The network equipment providing IP layer connectivity between an IP network (typically the Internet) and the user terminals.
Base Station (BS)	The device in the access network that communicates over the air interface, via one or more sectors, with the user terminals. Base Stations coordinate the management of the air interface attributes.

User Terminal (UT)	A device providing data connectivity to an end user device (EUD) user. A user terminal may be connected to a computing device such as a laptop personal computer or it may be a self-contained data device such as a personal digital assistant.
Sector	One set of physical layer channels transmitted between base station and the user terminals within a given frequency assignment. A sector consists of a reverse link radio channel and a forward link radio channel.

14.3 Acronyms

Acronyms as specified in Section 1.2 of HC-SDMA [25] with additional acronyms as underlined below:

BCMCS	**Broadcast and Multicast Services**
SMB	**Short Message Broadcast**
VoIP	**Voice over Internet Protocol**

14.4 Conventions

The terminologies and conventions used in this specification shall be as specified in 1.3 of HC-SDMA [25].

14.5 625k-MC Application Overview

The 625k-MC application overview is as specified in 1.4 of HC-SDMA [25].

14.6 625k-MC Protocol Overview

625k-MC mode's RF/PHY/MAC/LLC specifications are based on the L0/L1/L2/L3 specifications defined in HC-SDMA [25]. The 625k-MC PHY/MAC/LLC Protocol Reference model is as shown in Figure 122.

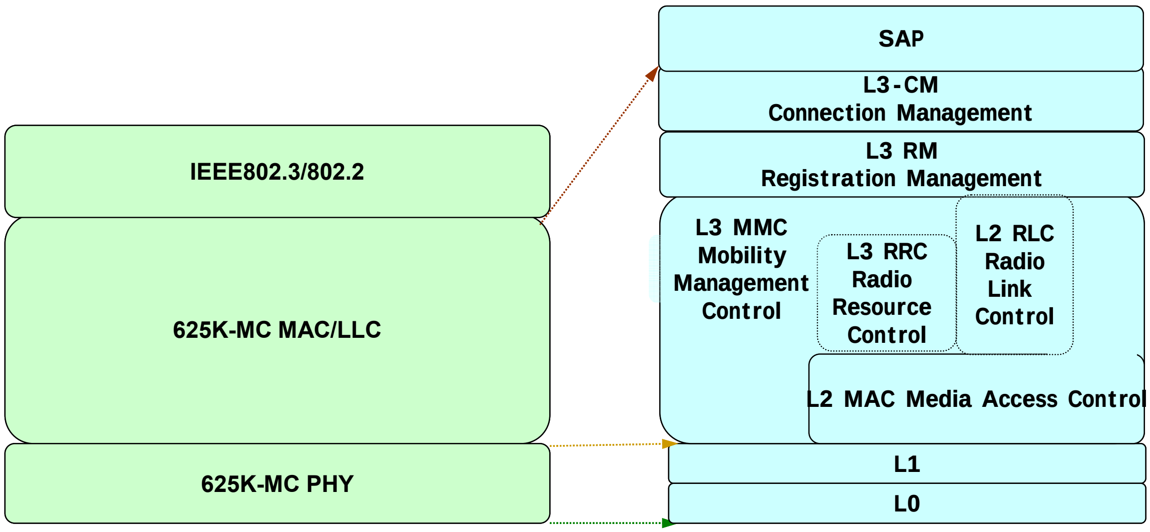

Figure 122 625k-MC PHY/MAC/LLC Protocol Reference

The 625k-MC Physical layer (PHY) specifications are based on the definitions of L1 (Layer 1) and L0 (Layer 0) protocols of HC-SDMA [25]. The 625k-MC Medium Access Control (MAC) specifications are based on the definitions of the L2-MAC (Layer 2-MAC) protocol of HC-SDMA [25]. The 625k-MC Logical Link Control (LLC) specifications are based on the definitions of L2-RLC (Layer 2-Radio Link Control), Layer L3 (L3-RRC, L3-MMC, L3-CM, L3-RM) of HC-SDMA [25].

14.6.1 625k-MC Protocol Features

The basic protocol features of *625k*-MC shall be as defined in 1.5.1 of HC-SDMA [25].

14.6.2 625k-MC Protocol Reference Model and Interfaces

The 625k-MC protocol reference model and communication interface between protocol layers shall be as defined in 1.6 of HC-SDMA [25].

15 625k-MC Spectral Layout Terminology and Requirements

The 625k-MC spectral layout terminology and requirements shall be as defined in Chapter 2 of HC-SDMA [25].

16 625k-MC Slot and Frame Structure

The 625k-MC mode MAC frame and slot structures for each "carrier" whose RF channel bandwidth is 625 kHz, are specified in this chapter. The RF carrier frequency in the carrier allocation shall be a consecutive set of frequencies separated by 625 kHz.

16.1 Overview

As defined in 3.1 of HC-SDMA [25].

16.2 RF Channel and Frame Structure

As defined in 3.2 of HC-SDMA [25].

16.3 Burst Formats

16.3.1 Frequency Synchronization

As defined in 3.3.1 of HC-SDMA [25].

16.3.2 Timing Synchronization

As defined in 3.3.2 of HC-SDMA [25].

16.3.3 Broadcast Burst

As defined in 3.3.3 of HC-SDMA [25].

16.3.4 Page Burst

As defined in 3.3.4 of HC-SDMA [25].

16.3.5 Configuration Request Burst

As defined in 3.3.5 of HC-SDMA [25].

16.3.6 Standard Uplink Burst (FACCH, RACH& TCH)

As defined in 3.3.6 of HC-SDMA [25].

16.3.7 Standard Downlink Burst (CM, SMB, AA & TCH)

As defined in 3.3.7 of HC-SDMA [25] with ***the following underlined text included before the end of this section:***

The 625k-MC Short Message Broadcast (SMB) burst shall adhere to the standard downlink burst format.

17 625k-MC Modulation and Channel Coding

17.1 625k-MC Modulation and Channel Coding Overview

Chapter 4 of HC-SDMA [25] gives the baseline capabilities for 625k-MC Modulation and channel coding. The capability therein described is further enhanced by the requirements of this section as underlined below:

625k-MC supports 11 Modulation Classes (ModClasses). ModClasses 0-8 for the downlink and ModClasses 0-7 for uplink shall be as specified in the 4.2 of HC-SDMA [25]. Table 122 shows the updated Table 4.1 of HC-SDMA [25] and adds ModClasses 9-10 for the downlink and 8-10 for the uplink.

Table 122 User data throughput at various ModClasses

ModClass	Single stream downlink throughput (kbps) per carrier	Aggregated 3 stream downlink throughput (kbps) per carrier	Single stream uplink throughput (kbps) per carrier	Aggregated 3 stream uplink throughput (kbps) per carrier
0	35.2	105.6	6.4	19.2
1	49.6	148.8	12.8	38.4
2	81.6	244.8	25.6	76.8
3	126.4	379.2	43.2	129.6
4	161.6	484.8	57.6	172.8
5	198.4	595.2	72.0	216.0
6	262.4	787.2	97.6	292.8
7	307.2	921.6	115.2	345.6
8	353.6	1060.8	132.8	398.4
9	377.6	1132.8	142.4	427.2
10	497.6	1492.8	190.4	571.2

17.2 Standard Modulation and Coding

As specified in Section 4.2 of HC-SDMA[25] ***with the additional text and tables as underlined below***.

The 625k-MC physical layer supports different data rates by selecting among various coding and modulation schemes. Figure 123 illustrates the coded modulation system that achieves rates from approximately ½ to 5.5 bits/symbol. Table 123 lists the modulation and signal sets together with the associated parameters for puncturing, shaping and block coding for ModClasses 9 and 10, in addition to those for modClasses 0-8 as shown in Table 4.2 of HC-SDMA[25].

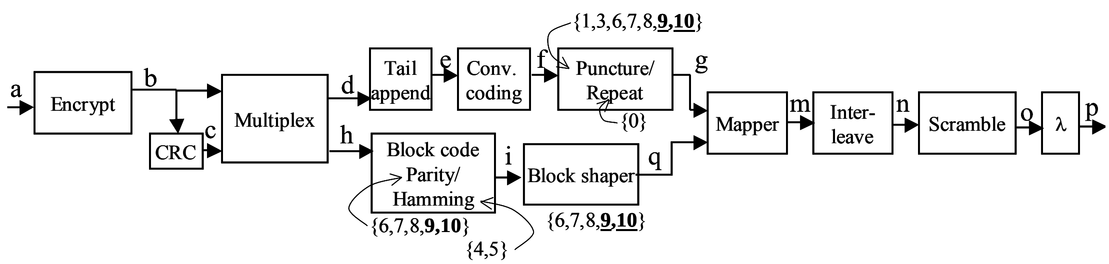

Figure 123 Block diagram for error control coding scheme. The notation {*i*, *j*} indicates that a block is active only for modulation classes *i* and *j*.

Coding is provided by a rate-1/2 convolutional code with 256 states combined in some cases with a block code. In some ModClasses, periodic puncturing is applied to increase the rate of the convolutional code to 2/3, ¾ or 5/6. Repetition is used in ModClass 0 to construct RA, AA, CM and SMB bursts (and is not used in ModClass 0 to construct CR and TCH bursts). ModClasses 6, 7 and 8 employ four-dimensional block shaping to generate 12-, 16- and 24-QAM signal set respectively. ModClasses 9 and 10 employ block shaping to generate 32-, and 64-QAM, respectively.

The operations of each block in Figure 123 are detailed in the subsequent sections. In the descriptions that follow, $\{a_1, a_{2, \ldots,} a_{Na\}}$ denotes the sequence of N_a bits input to the encrypt block, $\{b_1, b_{2, \ldots,} b_{Na}\}$ denotes the N_b bits input to the CRC block. The block sizes $\{N_a, N_b, \ldots N_q\}$ are listed in Table 125 and Table 126. The block sizes for CR, CM, RA and AA are as shown in Table 4.6 to 4.9 of HC-SDMA [25]. The block sizes for SMB are as shown in Table 127.

Table 123 Modulation and Coding Rates

ModClass	Bits/Sym	Signal Set	Puncture	Shaper	Block Code
0	0.5	BPSK	Repeat	-	-
1	0.67	BPSK	1 of 4	-	-
2	1	QPSK	-	-	-
3	1.5	QPSK	2 of 6	-	-
4	2	8-PSK	-	-	(64,57)
5	2.5	8-PSK	-	-	(64,57)
6	3	12-QAM	2 of 6	3/4	(48,47)
7	3.5	16-QAM	2 of 6	4/4	(64,63)
8	4	24-QAM	2 of 6	5/4	(80,79)

ModClass	Bits/Sym	Signal Set	Puncture	Shaper	Block Code
9	4.5	32-QAM	2 of 6	5/5	(80,79)
10	5.5	64-QAM	2 of 5	6/6	(80,79)
11-15	RESERVED				

Table 124 ModClass versus Burst Type

Logical Channel	Burst Type	ModClass
TCH-uplink	Standard uplink	0-10
TCH-downlink	Standard downlink	0-10
RACH-uplink	Standard uplink	0
RACH-downlink	Standard downlink	0
CCH-uplink	Configuration Request	0
CCH-downlink	Standard downlink	0
BCH and PCH	Broadcast and Page	See Section 4.3 Broadcast channel Modulation and Coding on page 4.22 of HC-SDMA [25]

Table 125 Block Lengths in Downlink Traffic Burst

ModClass	Nab	Nc	Nd	Ne	Nf	Ng	Nh	Ni	Nq	Nm	Nmnop
0	206	16	222	230	460	460	0	0	0	460	460
1	282	16	298	306	613	460	0	0	0	460	460
2	436	16	452	460	920	920	0	0	0	920	460
3	666	16	682	690	1380	920	0	0	0	920	460
4	840	16	452	460	920	920	404	460	460	460	460
5	1021	16	222	230	460	460	815	920	920	460	460
6	1341	16	682	690	1380	920	675	690	920	920	460
7	1571	16	682	690	1380	920	905	920	920	920	460
8	1801	16	682	690	1380	920	1135	1150	920	920	460
9	1919	16	1027	1035	2070	1380	908	920	920	920	460
10	2523	16	950	958	1916	1150	1589	1610	1610	920	460

Table 126 Block Lengths in Uplink Traffic Burst

ModClass	Nab	Nc	Nd	Ne	Nf	Ng	Nh	Ni	Nq	Nm	Nmnop
0	67	16	83	91	182	182	0	0	0	182	182
1	67	16	113	121	182	182	0	0	0	182	182
2	158	16	174	182	364	364	0	0	0	364	182
3	249	16	265	273	364	364	0	0	0	364	182
4	319	16	174	182	364	364	161	182	182	182	182
5	389	16	83	91	182	182	322	364	364	182	182
6	516	16	265	273	364	364	267	273	364	364	182
7	607	16	265	273	364	364	358	364	364	364	182
8	698	16	265	273	546	364	449	455	364	364	182
9	744	16	401	409	819	546	359	364	364	364	182
10	984	16	371	379	758	455	629	937	637	364	182

Table 127 Block Lengths in Short Message Broadcast

ModClass	Nab	Nc	Nd	Ne	Nf	Ng	Nh	Ni	Nq	Nm	Nmnop
0	105	16	121	129	258	460	0	0	0	460	460

17.2.1 Encryption

As defined in 4.2.1 of HC-SDMA [25].

17.2.2 Cyclic redundancy check

As defined in 4.2.2 of HC-SDMA [25].

17.2.3 Multiplexing

As defined in 4.2.3 of HC-SDMA [25].

17.2.4 Tail append

As defined in 4.2.4 of HC-SDMA [25].

17.2.5 Convolutional Encoding

As defined in 4.2.5 of HC-SDMA [25].

17.2.6 Puncturing and repeating

Puncturing and repeating for modClasses 0-8 is as specified in 4.2.6 of HC-SDMA[25] ***with the following underlined text included at the end of the section:***

- For modClass 9, the coded outputs are punctured by a periodic puncturing pattern that deletes two bits from every block of six as that for modClass 3, 5, 7 and 8.
- For modulation class 10, the coded outputs are punctured by a periodic puncturing pattern that deletes two bits from every block of five.

Input	f_1 f_3 f_4	f_6 f_8 f_9	f_{11} f_{13} f_{14}	…
Output	g_1 g_2 g_3	g_4 g_5 g_6	g_7 g_8 g_9	…

17.2.7 Block Coding

As defined in 4.2.7 of HC-SDMA [25].

17.2.7.1 Extended Hamming Code

As defined in 4.2.7.1 of HC-SDMA [25].

17.2.7.2 Parity check Code

The parity check code for ModClasses 6, 7 and 8 shall be as defined in 4.2.7.2 of HC-SDMA [25] ***with the following underlined text included at the end of the section:***

The parity check code specified in 4.2.7.2 of HC-SDMA [25] for ModClass 8 with input block size 79 shall be used for ModClasses 9 and 10.

17.2.8 Block Shaper

ModClasses 6-10 employ block shaping. The block shaper processes a binary-valued input sequence into a ternary- valued output sequence. The block shaping for ModClasses 6, 7 and 8 are as defined in 4.2.8 of HC-SDMA [25]. ***The following underlined text and tables are included at the end of the section:***

The input sequence is divided into blocks of size B=5 and 6 bits for ModClasses 9 and 10 respectively. ModClasses 9 and 10 use 5/5 and 6/6 block shaper as shown in Table 128 and Table 129, respectively.

Table 128 Mapping for Rate 5/5 Block shaper

i_{1+5l}, i_{2+5l}, i_{3+5l}, i_{4+5l}, i_{5+5l}	q_{1+5l}, qi_{2+5l}, q_{3+5l}, qi_{4+5l}, qi_{5+5l}
x_1 x_2 x_3 x_4 x_5	x_1 x_2 x_3 x_4 x_5

Table 129 Mapping for Rate 6/6 Block shaper

i_{1+6l}, i_{2+6l}, i_{3+6l}, i_{4+6l}, i_{5+6l}, i_{5+6l}	q_{1+6l}, qi_{2+6l}, q_{3+6l}, qi_{4+6l}, qi_{5+6l} qi_{5+6l}
x_1 x_2 x_3 x_4 x_5 x_6	x_1 x_2 x_3 x_4 x_5 x_6

17.2.9 Symbol Mapping

Symbol mapping for ModClasses 2, 3, 6, 7, and 8 shall be as defined in 4.2.9 of HC-SDMA [25]. ***The following underlined text and tables are included at the end of the section 4.2.9 of HC-SDMA [25]:***

For modclasses 9 and 10, pairs of outputs from the lookup table are multiplexed into a single complex output symbol as specified in 4.2.9 of HC-SDMA [25]. The symbol mapper for modclass 9 is shown in Table 130.

Table 130 Symbol mapper for Modulation classes 9

q_{2k-1}	q_{2k}	q_{3k-2}	q_{3k-1}	q_{3k}	$\tilde{m}$	
					2k-1	2k
0	0	0	0	0	-3	5
0	1	0	0	0	-1	5
1	0	0	0	0	-5	3
1	1	0	0	0	-5	1
0	0	0	0	1	-3	3
0	1	0	0	1	-1	3
1	0	0	0	1	-3	1
1	1	0	0	1	-1	1
0	0	0	1	0	-3	-5
0	1	0	1	0	-1	-5
1	0	0	1	0	-5	-3
1	1	0	1	0	-5	-1
0	0	0	1	1	-3	-3
0	1	0	1	1	-1	-3
1	0	0	1	1	-3	-1
1	1	0	1	1	-1	-1
0	0	1	0	0	3	5
0	1	1	0	0	1	5
1	0	1	0	0	5	3
1	1	1	0	0	5	1
0	0	1	0	1	3	3
0	1	1	0	1	1	3
1	0	1	0	1	3	1
1	1	1	0	1	1	1

q_{2k-1}	q_{2k}	g_{3k-2}	g_{3k-1}	g_{3k}	$\tilde{m}$	
					2k-1	2k
0	0	1	1	0	3	-5
0	1	1	1	0	1	-5
1	0	1	1	0	5	-3
1	1	1	1	0	5	-1
0	0	1	1	1	3	-3
0	1	1	1	1	1	-3
1	0	1	1	1	3	-1
1	1	1	1	1	1	-1

For ModClass10, the mapping rules are:

q_{7k-6} q_{7k-4} g_{5k-4} → X_{12k-11} X_{12k-10} X_{12k-9}

q_{7k-5} q_{7k-3} g_{5k-3} → X_{12k-8} X_{12k-7} X_{12k-6}

q_{7k-2} q_{7k} g_{5k-1} → X_{12k-5} X_{12k-4} X_{12k-3}

q_{7k-1} g_{5k-2} g_{5k} → X_{12k-2} X_{12k-1} X_{12k}

for k=1,2,□□□

Table 131 Symbol mapper for Modulation classes 10

X_{3l-2} X_{3l-1} X_{3l}	$\tilde{m}$
0 0 0	-7
0 0 1	-5
0 1 1	-3
0 1 0	-1
1 1 0	1
1 1 1	3
1 0 1	5
1 0 0	7

17.2.10 Interleaving

As defined in 4.2.10 of HC-SDMA [25].

17.2.11 Scrambling

As defined in 4.2.11 of HC-SDMA [25] ***with the following underlined text included at the end of the section:***

Scrambling for SMB shall be same as that for CM.

17.2.12 π/2 Rotation and Scaling

As defined in 4.2.12 HC-SDMA [25] ***with the following underlined text included at the end of the section:***

The modulation scaling for ModClasses 9 and 10 are as follows:

Table 132 Modulation Scaling

modclass	λ
2	$1/\sqrt{2}$
3	$1/\sqrt{2}$
4	1
5	1
6	$1/\sqrt{7}$
7	$1/\sqrt{10}$
8	$1/\sqrt{14}$
9	$1/\sqrt{20}$
10	$1/\sqrt{42}$

17.3 Broadcast Channel Modulation and Coding

The broadcast logical channels shall be as specified in 4.3 of the HC-SDMA [25].

18 625k-MC User Terminal Radio Transmission and Reception

The UT shall follow all procedures as specified in Chapter 5 of HC-SDMA [25]. The terminology used for both BS and UTs radio compliance specifications are defined in 5.1.1 of HC-SDMA [25].

19 625k-MC Base Station Radio Transmission and Reception

The BS shall follow all procedures as specified in Chapter 6 of HC-SDMA [25]. The terminology used for both BS and UTs radio compliance specifications are defined in 5.1.1 of HC-SDMA [25].

20 625k-MC L2 MAC Protocol Sublayer Specification

The 625k-MC L2 MAC protocol shall be as specified in Chapter 7 of HC-SDMA [25] ***with the supplemental changes outlined herein.***

To support broadcast services, UT and BS L2 MAC protocol defines additional message: *Short Message Broadcast* transported by logical channel CCH.

20.1 Logical Channels

The relationship between logical channels, messages, and burst types is described in Table 7.1 of HC-SDMA [25] ***with the additional text as underlined in Table 133***.

Table 133 Logical Cannels, Messages, and Burst Types

Logical Channel	Message	Burst Types
BCH	Frequency synch message timing synch message broadcast message	frequency synchronization timing synchronization broadcast
PCH	page message	page
RACH	request access message access assignment message	standard uplink standard downlink
CCH	configuration request message configuration message short message broadcast	configuration request standard downlink standard downlink
TCH	uplink traffic downlink traffic	standard uplink standard downlink
FACCH	current modulation class (odd RFN’s) additional available power (even RFN’s) recommended modulation class (odd RFN’s) current modulation class (even RFN’s)	standard uplink standard uplink standard downlink standard downlink

20.1.1 Short Message Broadcast (SMB)

This complete section is added before the end of section 7.3.4 of HC-SDMA [25].

SMB shall conform to Table 134.

Table 134 SMB fields

Field Name	Number of bits
Minimum protocol	4
AFN	10
Network Operator ID	8
Data flag	4
Sequence number	7
Length	13
Null flag	1
Reserve	10
Data	48
Total	105

Short message channel field:

- Minimum protocol: minimal protocol shall be the same as that for CM in 7.3.4. 2 of HC-SDMA [25].
- AFN: The 10 least significant bits of the Absolute Frame Number on which short message channel was transmitted.
- Network Operator ID: Network Operation ID shall be the same as that for CM in 7.3.4. 2 of HC-SDMA [25].
- Data flag: Data flag shall identify the broadcast data among the 16 different sets available
- Sequence number: Sequence number shall specify the sequence number of the broadcast packet where the first packet in the sequence shall be 0.
- Length: Shall be the total length of the broadcast message in bits.
- Null flag: Shows that BS does not have any data to transmit.
- Data: Broadcast Data for transmission.

20.2 625k-MC Minimized RMU header

20.2.1 Message Format for AA-cts and AA-short

The RMU header defined in HC-SDMA shall be referred to as the *basic RMU header*. For small burst sizes, the *basic RMU header presents significant overhead.* For small burst sizes, UT and BS protocols of the 625k-MC mode shall support *minimized RMU header*. The minimized RMU header removes the ARQ acknowledgement field from bursts in which the successful receipt of the reverse-link burst by the peer can implicitly signal correct reception of forward-link data. In addition, the minimized RMU header incorporates only the 7 LSBs of the ARQ sequence number when it is possible to infer the full 13-bit sequence number from the 7 LSBs on the peer. A new ARQ format field is added to the minimized RMU header to indicate whether the header carries an explicit or implicit acknowledgement, and whether the full sequence number or only the 7 LSBs is present. The following sections outline the supplemental changes to the baseline L2 MAC Protocol sublayer specifications of HC-SDMA [25] when either UT or BS use *minimized RMU header*.

The message format for AA-cts and AA-short shall conform to Table 135 which is an updated version of Table 7.14 of HC-SDMA [25] ***with the additional text as underlined below:***

Table 135 Message format for subtypes AA-cts and AA-short

Field	# of Bits	Bit Positions
(AA common fields)	30	0:29
FrameDec	4	30:33
TchIndex	3	34:36
ModClassUp	4	37:40
modClassDown	4	41:44
Resource	6	45:50
rmuHeaderType	1	51
Reserved	53	52:104
Total	105	

AA common fields consist of the first 5 fields on Table 7.11 of HC-SDMA [25].

The definitions of frameDec, tchIndex, modClassUp, modClassDown and resource are as defined in 7.3.5.1.2.3.1 of HC-SDMA [25]. The rmuHeaderType field is defined below.

- rmuHeaderType: Shall indicate whether the basic RMU header or minimized RMU header shall be used for the TCH bursts of the stream opened by the AA message. If rmuHeaderType = 0, the basic RMU header shall be used. Else, the minimized RMU header shall be used. The BS shall set the RMU header type field to 0 unless the BS has selected in the Reg Params registration message use of a protocol version that defines the minimized RMU header.

20.2.2 625k-MC RMU Field Definition

As specified in Section 7.3.7.2.1 of HC-SDMA [25] ***with the additional text as underlined below:***

The structure of AM RMU shall conform to Figure 7.9 of HC-SDMA [25]. The gray portion is the RMU Header. The basic RMU header is shown in Figure 7.9 of HC-SDMA [25]. The minimized RMU header may be used instead, at the discretion of the BS, if both the BS and UT support it. The header shall be present in every AM RMU. The white portion is the RMU Payload.

Note that the payload size can change when octets are retransmitted. For example, suppose that the AM octets of RMU 1 are lost, and suppose further that RMU 2 is to retransmit as many of these octets as possible. Since the UM portions of RMU 1 and RMU 2 can be different sizes, it is possible that the set of AM octets in RMU 1 will not fit into RMU 2.

20.2.3 625k-MC Header Field Insertion

As specified in Section 7.3.7.2.2 of HC-SDMA [25] ***with the additional text and table as underlined below:***

The RMU header fields shall conform to either the basic RMU header shown in Table 136 (Table 7.29 of HC-SDMA [25]), or the minimized RMU header shown in Table 137. The pwrCtrl bit in a downlink burst defines the relative adjustment to apply to transmit power uplink. The adjustment is 1dB higher if the bit is 1 and 1dB lower if the bit is 0. On an uplink burst, the pwrCtrl bit indicates the UT's measurement of SINR relative to the target SINR for a received downlink burst (which is a function of the modulation class of the downlink burst). The measured SINR is lower than the target if the bit is a 1 and higher if the bit is a 0. The seqNum and ack fields are used for the retransmission scheme as defined in 8.5.1 and 8.5.2 of HC-SDMA [25]. The AM RMU definition is defined in Figure 7.9 of HC-SDMA. During each frame, L2 MAC shall set the pwrCtrl bit with the value indicated by an L2MacTchTxInfoGet.resp primitive from L3 RRC. The seqNum and ack fields are filled by L2 RLC.

Table 136 Basic RMU Header Fields (see Table 7.29)

Field	# of Bits	Bit Positions	Interpretation
pwrCtrl	1	0	Power Control (downlink) / SINR (uplink) bit
Type	2	1:2	00: payload is pure AM data 01: payload is mixture of AM and UM data 10, 11: reserved
Seqnum	13	3:15	sequence number of the first AM octet in the payload. If the AM payload size is 0 octets, this value is ignored.
Ack	12	16:27	RLC acknowledgment

Table 137 Minimized RMU Header Fields

Field	# of Bits	Bit Positions	Interpretation
pwrCtrl	1	0	Power Control (downlink) / SINR (uplink) bit
Type	2	1:2	00: payload is pure AM data 01: payload is mixture of AM and UM data 10, 11: reserved
arqFormat	1 or 3	3:	Coded indication of the format of the ARQ seq num and ack fields. 0: 7 bit sequence number, ack field is not present 001: 7 bit sequence number, ack field is present 111: 13 bit seq num, ack field is not present 011: 13 bit seq num, ack field is present 101: reserved

Field	# of Bits	Bit Positions	Interpretation
seqNum	7 or 13		Sequence number of the first AM octet in the payload. If the AM payload size is 0 octets, this value is ignored. If the arqFormat field is 0 or 001, this field contains the 7 LSBs of the full sequence number.
Ack	0 or 12		RLC acknowledgment. This field is only present when the arqFormat is 001 or 011.
Total	11,19,25, or 31		

20.2.4 UM Message Insertion

As specified in Section 7.3.7.2.3 of HC-SDMA [25] ***with the additional text as underlined below:***

The L2 MAC has two priority levels for AM data (see Chapter 8 of HC-SDMA [25]). The L2 MAC may prioritize UM messages above the high priority AM data, or below the high priority AM data and above the low priority

These respective priority levels for UM messages will be referred to as high, medium and low.

The method that L2 MAC does this prioritization shall be as follows:

1. Let N_{tot} be the total size of the RMU in bits (which is a function of modulation class) and N_{hdr} the size of the RLC header in bits (see Figure 7.9 of HC-SDMA [25]). If the basic RMU header is used, then N_{hdr} = 28 (see Table 136). If the minimized RMU header is used, then L2 MAC must ask L2 RLC on each frame whether the RMU header will be 11, 19, 25, or 31 bits (see Table 137). The maximum number of bytes available for UM and AM messages is $N_{rmu} = \lfloor (N_{tot} - N_{hdr})/8 \rfloor$ bytes where the floor function $\lfloor x \rfloor$ equals the largest integer less than or equal to *x*.

The steps 2 to 8 and remaining text shall be as specified in 7.3.7.2.3 of HC-SDMA [25].

21 625k-MC L2 RLC Protocol Sublayer Specification

625k-MC L2 RLC protocol shall be as specified in Chapter 8 of HC-SDMA [25] with the supplemental changes outlined herein.

The RMU header defined in HC-SDMA is referred to as the *basic RMU header*. UT and BS of the 625k-MC mode shall support header compression techniques to minimize the *basic RMU header*. The following sections outline the supplemental changes to the baseline L2 RLC Protocol sublayer specifications of HC-SDMA [25] when either UT or BS use the *minimized RMU header*.

21.1 625k-MC AM RMU

As specified in Section 8.2.1.1 of HC-SDMA [25] ***with the additional text as underlined below:***

The AM RMU shall send L3 SDUs and some control messages generated by L2 MAC using the L2 RLC entity described in 8.1.3 L2 RLC Model of HC-SDMA [25]. Figure 7.9 of HC-SDMA [25] shows the format of the RMU Pay load.

21.1.1 RMU Header

The RMU header fields shall be as defined in Table 136 and Table 137.

For the basic RMU header and for the minimized RMU headers which include an explicit acknowledgement, the ack field shall contain the 12 LSBs of a sequence number.

21.2 625k-MC Transmit Procedure

As specified in Section 8.5.1of HC-SDMA [25] ***with the additional text as underlined below***:

During each frame, L2 MAC shall compute the maximum number of octets n_{max} available for transmitting high priority AM octets (see 8.1.3.3 Transmit ARQ of HC-SDMA [25]), by considering the size of the transmit burst's RMU payload, how many bits are needed for the RMU header, and how many octets are needed for high priority UM messages. The number of bits needed for the RMU header is constant (28) if the basic RMU header is used, and variable if the minimized RMU header is used. Similarly, L2 MAC shall compute $n_{reserve}$, the number of octets desired for medium priority UM messages if L2 RLC has less than n_{max} high priority AM octets. If the number of high priority AM octets is less than n_{max} - $n_{reserve}$, L2 RLC may use the remaining space to transmit any low priority AM octets next to the sequence of high priority AM octets. The Transmit ARQ shall also set seqNum and ack in the RMU header. If the basic RMU header is used, seqNum shall be set to the sequence number of the first octet written to the payload, ack shall be set to the S – 1 LSBs of the value of pAck. If the minimized RMU header is used, L2 RLC shall determine whether to put the full sequence number or only the LSBs into the RMU header, and whether to include or exclude the ack. The L2 RLC shall set arqFormat to indicate whether the RMU header contains the full sequence number or the LSBs, and whether the ack is present or implicit (see Table 137).

If the minimized RMU header is used, the RMU header shall have only the 7 LSBs of the sequence number if pTransmit $-_m$ pAckPeer < 128 and pSweep = pAckPeer. Otherwise the full sequence number shall be used.

If the minimized RMU header is used, then the RMU header shall not include the ack if the following three conditions are met:

1. pAck $=_m$ pReceive
2. the last received RMU was decoded successfully
3. the sequence number of the last byte of the last received RMU payload = pAck $-_m$ 1

Otherwise the ack shall be included in the RMU header.

The ack from the peer receive entity shall be processed and the transmit state variable shall be updated as described in 8.5.2.1 of HC-SDMA [25] prior to the execution of the transmit states in 8.5.1.1 of HC-SDMA [25].

21.2.1 625k-MC Transmit State Execution

Transmit state execution shall be as specified in 8.5.1.1 ***with the additional text as underlined in the last bullet point:***

The execution of the Transmit ARQ shall follow the following steps:

- **If** *nRetransmit* > 0 (**Retransmit state**)

  1. If $nRetransmit > n_{max} - n_{reserve}$, set n:=min($nRetransmit$, n_{max}). Else, set n:=min($nTransmit + pTransmit -_{m} pRetransmit$, $n_{max} - n_{reserve}$).
  2. Send the segment of octets in

     $$[pRetransmit, pRetransmit +_{m} (n-1)]_{m}$$

  3. Update *mapTx* as described in 8.5.1.2 mapTx Update Procedure of HC-SDMA [25].
  4. Set $pRetransmit :=_{m} pRetransmit + n$.
  5. Set $nRetransmit := nRetransmit - n$.

- **Else if** *nTransmit* > 0 and $pTransmit -_{m} pAckPeer < 2^{S-1} - 1$(**Transmit state**).

  1. Set $n := min(nTransmit, n_{max}, 2^{S-1} - (pTransmit -_{m} pAckPeer) - 1)$.
  2. Send the segment of octets in

     $$[pTransmit, pTransmit +_{m} (n-1)]_{m}$$

  3. Update *mapTx* as described in 8.5.1.2 mapTx Update Procedure of HC-SDMA [25].
  4. Set $pTransmit :=_{m} pTransmit + n$.
  5. Set $nTransmit := nTransmit - n$.

Cancel timer $T_{shutdown}$ if it is running.

- **Else if** $pAckPeer \neq pTransmit$(**Sweep state**)

  1. If $pSweep <_{m} pAckPeer$, set $pSweep := pAckPeer$.
  2. If $pSweep \geq_{m} pTransmit$, set $pSweep := pAckPeer$.
  3. Set n :=$min(pTransmit -_{m} pSweep, n_{max})$.
  4. Send the segment of octets in $[pSweep, pSweep +_{m} (n-1)]_{m}$.

5. If the basic RMU is used, update mapTx as described in 8.5.1.2 mapTx Update Procedure of HC-SDMA [25].

6. Set *pSweep* $:=_m$ *pSweep* + *n*.

- **Else nothing to transmit(Idle State)**

21.2.2 mapTx Update Procedure

As defined in 8.5.1.2 of HC-SDMA [25].

21.3 Receive Procedure

As specified in Section 8.5.2of HC-SDMA [25] ***with the additional text as underlined below***:

21.3.1 Receive Task Execution

During each frame, the RMU Demux shall deliver to the Receive ARQ the AM payload (possibly 0 octets) as well as the RMU header fields seqNum and ack unless there is a CRC error. (In case of a CRC error, none of them shall be delivered to the AM unit, and the following tasks are not executed.)

If the minimized RMU header is used, then the receiver shall check the arqFormat RMU header field to determine whether the sequence number and ack fields need full length. If arqFormat indicates that the RMU header contains only the 7 LSBs of the sequence number, then the receiver shall recreate the 13 bit sequence number with the following formulas. In these formulas, the & operator indicates a bitwise AND operation, and the ~ operator indicates bitwise negation.

if seqNum7bits >= (pAck & 0x7f)

seqNum13bits = (pAck & ~0x7f) + seqNum7bits

else

seqNum13bits = (pAck & ~0x7f) + 0x80 + seqNum7bits

If arqFormat indicates that the RMU header has no ack, then the receiver shall recreate the ack as the following steps:

1. Set AckRFN = RFN – $N_{turnaround}$

2. Here, RFN shows the RFN in which the burst was received.

3. Find the range of mapTx elements = AckRFN

4. Set ack to the 12 LSBs of the sequence number of the next mapTx element beyond the range identified in step 2.

The receiver shall perform the following three tasks.

Task 1:receive the data in the AM payload

1. Set *first* := *seqNum*.
2. Set last $:=_m$ *seqNum* +size of payload.
3. If last $\geq_m$ *pAck*+$2^{S-1}$and *first* $\geq_m$ *pAck*.
   - i) Recommend forced shut down (E_{seqNum_OB} error).
   - ii) Go to task 2.
4. If last $\leq_m$ *pAck*, ignore the AM octets in this RMU and go to task 2.
5. Store date in appropriate location.
6. Cancel timer $T_{shutdown}$ if it is running.
7. If last $>_m$ *pReceive*, set *pReceive* $:=_m$ *last*.
8. If *first* $\leq_m$ *pAck*,
   - i) Find p ($0 \leq p < 2^S$), the first sequence number that satisfies $p \geq \leq_m$ last and *mapRx*[*p*] = 0 (that is, there is not a $q <_m p$ such that $q \geq$ *last* and *mapRx*[*q*]=0).
   - ii) Set *mapRx*[*k*] = 0 for $k \in [pAck, p -_m 1]_m$.
   - iii) Deliver and record octets in $[pAck, p -_m 1]_m$ to L3.
   - iv) Set *pAck* := *p*.
   - v) Cancel timer $T_{receive}$.
   - vi) If $pAck \neq pReceive$, restart timer $T_{receive}$,

   Else (that is, *first* $>_m$ *pAck*) Set *mapRx*[*k*] = 1 for $k \in [first, last -_m 1]_m$.

Task 2:Process ack in RMU header (for transmit side)

1. ack is *S*-1 bits only. Convert it to *S* bits:

   If *ack* $<_m$ *pAckPeer*, set *ack* $:=_m$ *ack*+2^{S-1}.
2. If *ack* $>_m$ *pTransmit*
   - i) Recommend forced shut down (error E_{ack_OB}).
   - ii) Go to task 3.
3. If *ack* $>_m$ *pAckPeer*
   - i) Set *mapTx*[*k*] = -1 for $k \in [pAckPeer, ack -_m 1]_m$.

ii) If *pSweep* $<_m$ *ack* Set *pSweep* $:=_m$ *ack*.

iii) Set *pAckPeer* $:=_m$ *ack*.

iv) Cancel timer $T_{transmit}$.

v) If *pAckPeer* $\neq$ *pTransmit*, restart timer $T_{transmit}$

vi) If *nRetransmit* > 0 and *pRetransmit* $<_m$ *ack*

Perform a mapTx retransmit update as described in *8.5.2.2 mapTx Retransmit Update* of HC-SDMA [25].

4. If $RFN - mapTx[pAckPeer] \geq N_{turnaround}$ (here, *RFN* is when the relevant octets were transmitted on the air interface)

   Perform a mapTx retransmit update as outlined in *8.5.2.2 mapTx Retransmit Update* of HC-SDMA [25].

Task 3: check if stream is ready to end

If timer $T_{shutDown}$ is not running

If *nTransmit* =0, *pAckPeer* = *pTransmit*, *pAck* = *pReceive*, a complete packet has been transmitted, a complete packet has been received, and pAck has not changed for at least 1 frame, start timer $T_{shutDown}$

21.3.2 MapTx Retransmit Update

As defined in 8.5.2.2 of HC-SDMA [25].

21.3.3 Reset Procedure

As defined in Sect. 8.5.2.3 of HC-SDMA [25].

22 625k-MC L3 Protocol Specification

The L3 protocol shall be as specified in Chapter 9 of HC-SDMA [25].

23 625k-MC Protocol Layer Primitives (Informative)

The protocol layer primitive shall be as specified in Chapter 10 of HC-SDMA [25]. To support broadcast and multicast services, following primitives are added to individual interfaces in Chapter 10 of HC-SDMA [25].

23.1 Interface list

The interface list shall be as specified in 10.3.1 of HC-SDMA [25].

23.2 Individual Interfaces

The basic individual interface primitives shall be as specified in 10.3.2 of HC-SDMA [25] along with additional primitives as defined in the sections 23.2.1 to 23.2.6 below:

23.2.1 L2 MAC to L3 RM Interface Primitives

The L2 MAC to L3 RM Interface Primitives shall be as specified in 10.3.2.10 of HC-SDMA [25] ***with the following additional primitive described herein***.

Primitive list:

- L2MacUtBCData.ind

L2MacUtBCData.ind	
Applies to:	UT only
Associated primitives:	L2MacBsBCData.req (on BS)
Description: indication from UT L2 MAC to UT L3 RM that incoming broadcast data packets are available	
Parameters: • processID -- ID of the L2 MAC process associated with this stream • L2MacUtBcDataInfoSort -- the information of the broadcast data (struct) • L2MacUtBcDataInfoSort :: MinimumProtocol -- a lowest version of the 625k-MC protocol that the BS can support • L2MacUtBcDataInfoSort :: NetworkOperationID -- a identification of the network operator that owns the BS • L2MacUtBcDataInfoSort :: DataFlag -- a flag categories of broadcast data • L2MacUtBcDataInfoSort :: SequenceNumber -- a sequence number of broadcast data • L2MacUtBcDataInfoSort :: Length -- a length of broadcast delivery data	

23.2.2 L3 RM to L2 MAC Interface Primitives

The L3 RM to L2 MAC Interface Primitives shall be as specified in 10.3.2.11 of HC-SDMA [25] ***with the following additional primitive described herein***.

Primitive list:

- L2MacBsBCData.req

L2MacBsBCData.req	
Applies to:	BS Only
Associated primitives:	L2MacUtBCData.ind (on UT) (on peer)
Description: request from BS L3 RM to BS L2 MAC to deliver broadcast data packets across the air interface to UT L3 RM	
Parameters: • processID -- ID of the L3 RM process requesting the stream • L2MacBsBcDataInfoSort -- the information of the broadcast data (struct) • L2MacBsBcDataInfoSort :: MinimumProtocol -- a lowest version of the 625k-MC protocol that the BS can support • L2MacBsBcDataInfoSort :: NetworkOperationID -- a identification of the network operator that owns the BS • L2MacBsBcDataInfoSort :: DataFlag -- a flag categories of broadcast data • L2MacBsBcDataInfoSort :: SequenceNumber -- a sequence number of broadcast data • L2MacBsBcDataInfoSort :: Length -- a length of broadcast delivery data	

23.2.3 L3 RM to L3 CM Interface Primitives

The L3 RM to L3 CM Interface Primitives shall be as specified in 10.3.2.18 of HC-SDMA [25] ***with the following additional primitives described herein***.

Primitive list:

- L3RmBsMCStart.ind
- L3RmUtMCData.ind
- L3RmUtBCData.ind

L3RmBsMCStart.ind	
Applies to:	BS only
Associated primitives:	L3RmBsMcStart.resp/L3RmBsMCSStart.ind (on peer)
Description: indication from BS L3 RM to BS L3 CM that a new multicast service has been allowed	
Parameters: • MCstrmID – desired multicast information is requested across the L4.	

1

L3RmUtMCData.ind	
Applies to:	UT only
Associated primitives:	L3RmBsMCData.req (on BS) (on peer)
Description: indication from UT L3 RM to UT L3 CM that incoming delivered multicast packets are available	
Parameters: • PktQueue -- a list of multicast data packets received across the air interface from the peer BS L3 CM entity	

2

L3RmUtBCData.ind	
Applies to:	UT only
Associated primitives:	L3RmBsBCData.req (on BS)（on peer）
Description: Indication from UT L3 RM to UT L3 CM that incoming broadcast data packets are available	
Parameters: • PktQueue -- a list of broadcast data packets received across the air interface from the peer UT L3 RM entity • MinimumProtocol -- a lowest version of the 625k-MC protocol that the BS can support • NetworkOperationID -- a identification of the network operator that owns the BS • DataFlag -- a flag categories of broadcast data • Length -- a length of broadcast delivery data	

23.2.4 L3 CM to L3 RM Interface Primitives

The L3 CM to L3 RM Interface Primitives shall be as specified in 10.3.2.19 of HC-SDMA [25] ***with the following additional primitives described herein***.

Primitive list:

- L3RMUtMCStart.req
- L3RmBsMCStart.resp
- L3RmBsMCData.req
- L3RmBsBCData.req

L3RmUtMCStart.req	
Applies to:	UT only
Associated primitives:	L3RmBsMCStart.ind (on BS) (on peer)
Description: request from UT L3 CM to UT L3 RM for a multicast service to allow communication across the air interface	
Parameters: • MCstrmID – desired multicast information is tunneled across the air interface	

L3RmBsMCStart.resp	
Applies to:	BS only
Associated primitives:	L3RmBsMCStart.ind
Description: response from BS L3 CM to BS L3 RM allowance whether BS L3 CM was able to accept a multicast service in response to the L3RmStart.ind from BS L3 RM	
Parameters: • Boolean -- was the connection started	

L3RmBsMCData.req	
Applies to:	BS only
Associated primitives:	L3RmUtMCData.ind (on UT) (on peer)
Description: request to UT L3 RM to deliver multicast data packets across the air interface to BS L3 CM	
Parameters: • PktQueue -- a list of multicast data packets to deliver across the air interface to the BS L3 CM entity	

L3RmBsBCData.req	
Applies to:	BS only
Associated primitives:	L3RmUtBCData.ind (on UT) (on peer)
Description: request from BS L3 CM to BS L3 RM to deliver broadcast data packets across the air interface to UT L3 CM	
Parameters: • PktQueue -- a list of broadcast data packets to send across the air interface to the UT L3 CM entity • MinimumProtocol -- a lowest version of the 625k-MC protocol that the BS can support • NetworkOperationID -- a identification of the network operator that owns the BS • DataFlag -- a flag categories of broadcast data • SequenceNumber -- a sequence number of broadcast data • Length -- a length of broadcast delivery data	

23.2.5 L3 CM to L4 Interface Primitives

The L3 CM to L4 Interface Primitives shall be as specified in 10.3.2.20 of HC-SDMA [25] ***with the following additional primitives described herein***.

Primitive list:

- L3CmBsMCStart.ind
- L3CmUtMCData.ind
- L3CmUtBCData.ind

L3CmBsMCStart.ind	
Applies to:	BS only
Associated primitives:	L3CmBsMCStart.resp
Description: indication from BS L3 CM to BS L4 to confirm a multicast service	
Parameters: • MCstrmID – desired multicast information is requested.	

L3CmUtMCData.ind	
Applies to:	UT only
Associated primitives:	L3CmBsMCData.req (on BS) (on peer)
Description: indication from UT L3 CM to UT L4 that incoming multicast data packets are available	
Parameters: • PktQueue -- a list of data packets received across the air interface from the peer L4 entity	

L3CmUtBCData.ind	
Applies to:	UT only
Associated primitives:	L3CmBsBCData.req (on BS)
Description: indication from UT L3 CM to UT L4 that incoming broadcast data packets are available	
Parameters: • PktQueue -- a list of broadcast data packets received across the air interface from the peer L3 CM entity • DataFlag -- a flag categories of broadcast data • Length -- a length of broadcast delivery data	

23.2.6 L4 to L3 CM Interface Primitives

The L4 to L3 CM Interface Primitives shall be as specified in 10.3.2.21 of HC-SDMA [25] ***with the following additional primitives described herein***.

Primitive list:

- L3CmUtMCStart.req
- L3CmBsMCStart.resp
- L3CmMCData.req
- L3CmBsBCData.req

L3CmUtMCStart.req	
Applies to:	UT only
Associated primitives:	None
Description: request from UT L4 to UT L3 CM to start a multicast streaming across the air interface	
Parameters: • MCstrmID – desired multicast information is tunneled across the air interface	

L3CmBsMCStart.resp	
Applies to:	BS only
Associated primitives:	L3CmBsMCStart.ind
Description: response from BS L4 to BS L3 CM allowance whether BS L4 was able to accept a multicast service in response to the L3CmBsMCStart.ind from BS L3 CM	
Parameters: • Boolean -- was the connection started	

L3CmBsMCData.req	
Applies to:	BS only
Associated primitives:	L3CmUtMCData.ind (on peer)
Description: request from BS L4 to BS L3 CM to deliver multicast data packets across the air interface to UT L4	
Parameters: • PktQueue -- a list of multicast data packets to deliver across the air interface to the UT L4 entity	

L3CmBsBCData.req	
Applies to:	BS only
Associated primitives:	L3CmUtBCData.ind (on UT) (on peer)
Description: request from BS L4 to BS L3 CM to deliver broadcast data packets across the air interface to UT L4	
Parameters: • PktQueue -- a list of broadcast data packets to deliver across the air interface to the UT L4 entity • DataFlag -- a flag categories of broadcast data • Length -- a length of broadcast delivery data	

24 625k-MC QoS Enhancements

This Chapter is in addition to the baseline specification HC-SDMA [25].

This chapter describes the QoS enhancements of 625k-MC mode.

24.1 Classes of Services

The 625k-MC system is optimized for application of Quality of Service (QoS) to connections. QoS L3 AM control message is used between BS and UT to handle the changes in session's QoS requirements. Different sessions (connections) within a UT may have different QoS markings same as DSCP (Differential Service Code Point from RFC2474) markings, which are conveyed by QoS message as specified in Table 24.2. If so, the BS may allot the stream resources available to the UT between its registrations based on the appropriate relative priority. See the description of ibQos (QoS) argument of the L3CmBsData.req primitives in 10.3.2.1 and 10.3.2.19 of HC-SDMA [25].

To support the Standard DiffServ QoS Models, UT and BS of 625k-MC shall support the three classes of services as defined in Table 138 that incorporate forwarding Behaviors in the descending order of priority: Priority 1, Priority 2 and Priority 3. These classes are handled by Expedited Forwarding (EF), Assured Forwarding (AF) and Best Effort (BE) forwarding, respectively. In this table, QoS value refers to 6 MSB (most significant bits) of QoS marking in the QoS message of Table 138.

Table 138 Classes of QoS Service

	Class of Sessions/Users	Class of QoS	QoS Value (as listed in Table 139 Qos Message)	Example of Service
1)	Priority1	Expedited Forwarding (EF) - PHB	101110	VoIP
2)	Priority2	Assured Forwarding (AF) - PHB	001010 / 001100 / 001110 010010 / 010100 / 010110 011010 / 011100 / 011110 100010 / 100100 / 100110	Video Stream, Bandwidth Priority
3)	Priority3	Best Effort	000000	Data

The QoS value (or DSCP) for Priority 1 is EF Code Point 101110 (Refer RFC 2598), the DSCP for Priority 2 are defined with twelve kinds of AF Code Point (same as defined in RFC 2597) and the DSCP for Priority 3 is default Code Point 000000.

24.2 Session QoS Information Exchange Procedures

BS informs any change in the session's QoS by using 'AM-QoS Message". The Formats of "AM-QoS Message" is shown in 9.5.1 in HC-SDMA [25], the sequence is shown in Figure 124 which is inherited from Fig.9.25 of HC-SDMA [25].

"QoS Value" (Table 138) is given by 6 msb of the octet: "QoS Marking" of "AM-QoS Message" in Section 9.5.1.3.12 of HC-SDMA [25]. "QoS marking" which is shown Figure 9.59 of HC-SDMA [25], is defined by 8 bits, therefore "QoS Value" (6bits) is set up as follows.

Table 139 QoS Message

msb				bits			lsb	
8	7	6	5	4	3	2	1	octet
0	0	0	0	1	1	0	0	0
msgID								
0 ext bit	TLVG length (assuming<128bytes)							1
0	0	0	0	1	1	0	0	2
TLVG ID								
QoS marking								3
<u>QoS Value</u>						<u>Reserved</u>		

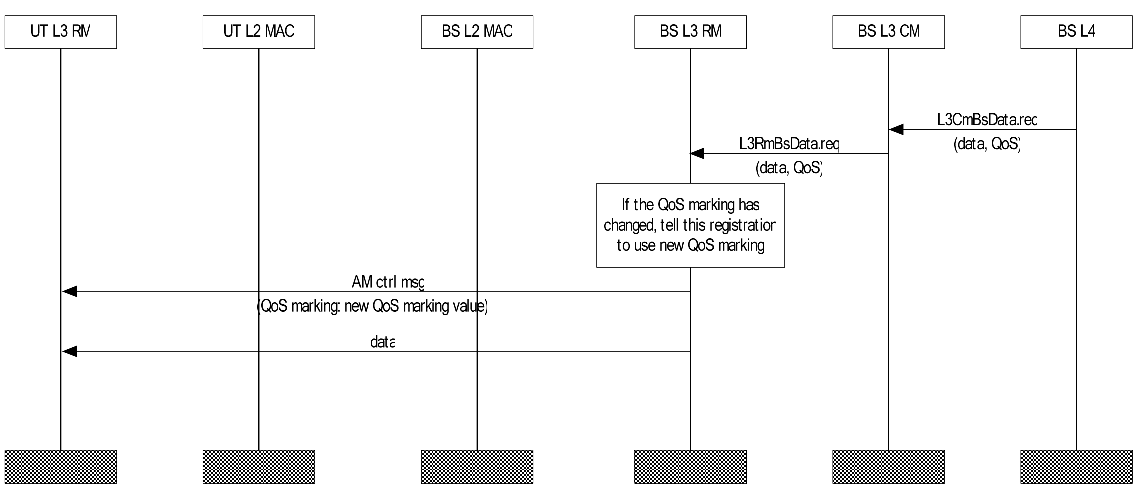

Figure 124 QoS Message sequence

When the BS is told by network to change the session's quality of service, BS delivers the QoS L3 AM Control (AM-QoS) Message and thereby notifies a new QoS value to UT.

24.3 QoS Priority

As listed in Table 138, there are three Classes of services with deferent priorities.

For each class, Shutdown Timer (defined in 8.2.2.3 of HC-SDMA [25]), which can specify the time to hold a stream can be set. Based on the priority, the Shutdown Timer can be set differently for different streams. For high priority data, a large timeout value should be used for the Shutdown Timer. These values are different for different streams.

It can perform differentiation by reservation of the stream according to priority. Generally, longer Shutdown Timer is set up for the sessions with in high class of priority. The values of longer Shutdown Timer may be implementation specific .

24.3.1 QoS Priority 1

Priority 1 of QoS shall be the highest priority of three Classes of Service.

It shall support high quality transmission achieving low packet loss, low latency, and low jitter duration. It is defined as the EF-PHB, which 6bits from MSB of QoS Marking field in the QoS Message is 101110.

The Shutdown Timer shall be specified to a larger value than that of other classes.

24.3.2 QoS Priority 2

Priority 2 of QoS shall be defined to a priority level lower than Priority 1.

It shall support the higher quality transmission than Priority 3.

It is defined as the AF-PHB, and twelve kinds of QoS Marking field in the QoS Message could be defined and used flexibly in each implementation.

The Shutdown Timer shall be set to a value smaller than Priority 1 and larger than Priority 3.

24.3.3 QoS Priority 3

Priority 3 of QoS shall be defined to Best Effort PHB.

It is defined as the BE-PHB, which 6bits from MSB of QoS Marking field in the QoS Message is 000000.

The Shutdown Timer shall be specified to a smaller value than that of other classes.

25 625k-MC Broadcast and Multicast Service (BCMCS) Support Enhancement

This Chapter is in addition to the baseline specification HC-SDMA [25].

25.1 Overview

Broadcast service: Any user can get information from the *BCMCS Contents Server.* Any user can get this information; extra certification is not needed.

Multicast service: Particular user can get various beneficial information from the *BCMCS Content Server*.

25.2 Broadcast Service

A BCH superframe is as defined in Fig. 7.2, Chapter 7, HC-SDMA [25]. Broadcast service shall be sent using CM burst in the superframe. This CM burst is called *Short Message Broadcast* (SMB). There is one SMB broadcast in a superframe and the position is always after the B burst. SMB follows the B depending on the BSCC; for example, if BSCC = 3 then SMB is after B3 as shown in Figure 125. SMB channel is only used for short broadcast service. BS shall inform BCMCS contents controller of Broadcast service request. BS gets Broadcast contents data from BCMCS contents server. Broadcast service can send at most 768 bytes. This message has 4 bit data flag to identify the broadcast data among the 16 different sets available. UT receives a part of the data with each superframe, and combines the data after receiving the entire data. Message flow for short message broadcast is shown in Figure 126.

SMB fields is shown in Table 134.

	AFN k	AFN k+1																		AFN k+19
Uplink		**CR**		**CR**		**CR**		**CR**		**CR**		**CR**		**CR**		**CR**		**CR**		**CR**
Downlink	**F**	**CM**	**T**	**CM**	**B_0**	**CM**	**B_1**	**CM**	**B_2**	**CM**	**B_3**	**<u>SM</u>**	**B_4**	**CM**	**B_5**	**CM**	**B_6**	**CM**	**B_7**	**CM**

Figure 125 BCH superframe with SMB after B_3

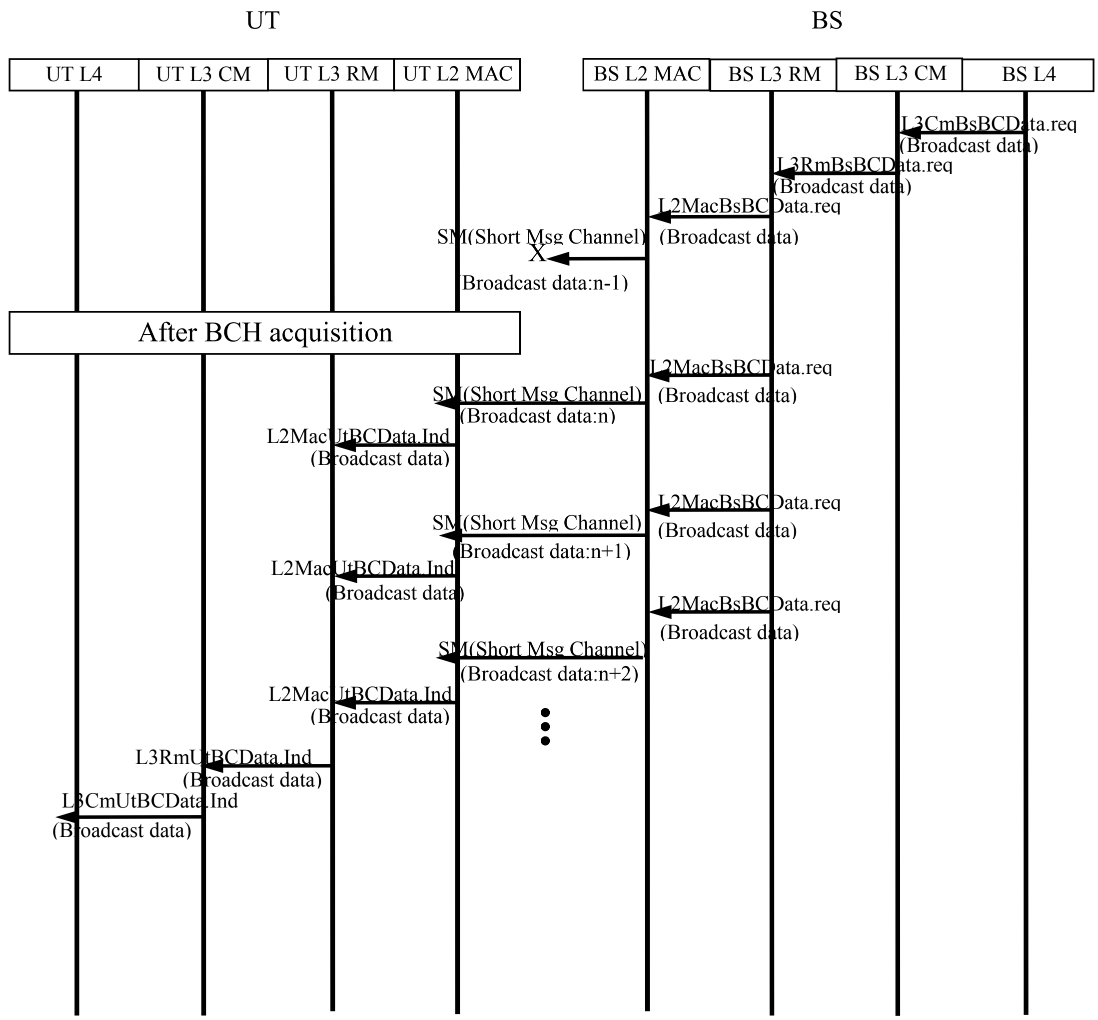

Figure 126 Message Flow for short message channel

25.3 Multicast Service

25.3.1 Overview

Multicast channels are made available to many users, therefore the functions of RRC are necessary for individual TCH of multicast channel.

25.3.2 Multicast handshake

To connect to a multicast channel, UT must send *Multicast stream request* to BS. BS sends *Multicast stream response* which includes *resource map* information. Resource map information tells the UT about the TCH channel over which multicast data is transmitted. This TCH channel shall be called the *Multicast channel*. UT changes the receiving TCH channel to Multicast Channel. The messaging handshake for Multicast stream shall conform to Figure 127.

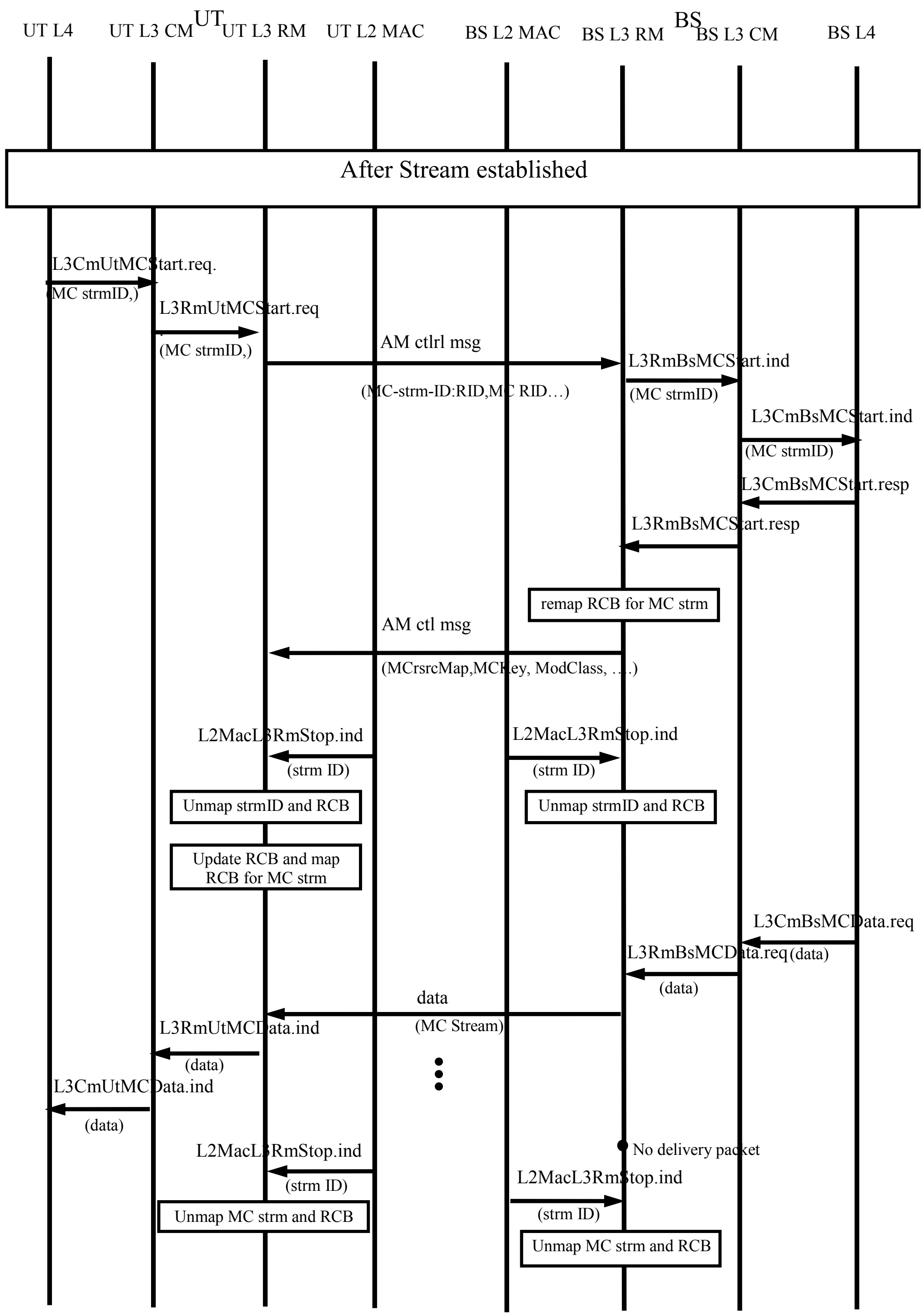

Figure 127 Message Flow for Multicast channel

- To send the MCStream.req to BS, UT opens a stream after TCH connection following the *UT Initiated stream Start* sequence specified in Fig. 9.19 of the HC-SDMA [25].
- UT sends information about StrmID, MC, RID together with the MCStream.req over the connected TCH. If BS allows the stream to open, it shall respond by sending the MCStream.resp with information necessary for Multicast like Mckey, ModClass, and rsrcMap.
- UT and BS shall close the stream following the *Graceful Stream Closure* sequence specified in Fig. 9.27 of HC-SDMA [25].
- BS shall transmit Multicast information over the channel specified using the MCrsrc during MCStream.resp; UT shall receive over the same channel. During Multicast information transmission and reception, RRC functions are not considered. Data shall be transmitted from BS at a constant level with the same ModClass specified during MCStream.resp. The data shall be encrypted at BS and the UT shall decrypt it using the Mckey. Mckey length shall choose between 56 bit and 280 bit for Multicast registration.

In case the UT is unable to receive multicast data or the multicast data finishes, the Multicast session shall be closed following the graceful stream closure sequence as specified in Fig. 9.27 of HC-SDMA [25]. The BS shall continue to transmit until the data finishes.

25.3.3 Message Format

Before the UT receives the multicast data, UT needs to register with the BS using the AM message data, viz., MCStream.req and MCStream.resp. BS uses AM mode also to deliver Multicast data to UT. The RMU header field shall conform to the Table 7.29 in 7.3.7.2.2 of HC-SDMA [25]. BS shall use RMU header type as "01" which indicates "payload is mixture of AM and UM data". During multicast session, BS and UT shall ignore "power control", "FACCH" and "ack" for AM in this BCMCS protocol.

The message format for MCStream.req and MCStream.resp are as shown in Figure 128 and Figure 129, respectively.

25.3.3.1 MCStream.req Message

Message ID: 16

TLVG: MCStream.req

TLVG ID: 16

Information Elements:

- IE: Assigned protocol version, 8 bits

  Description:

  The current protocol version field indicates which revision of the protocol the sending party is currently using. This is used by the UT in the MCStream.req message and by the BS in the MCStream.resp message to inform each other of which protocol version is being used during the multicast stream.

- IE: RID, 15 bits

  Description:

  The registration ID (RID) is used in various of the RA/AA messages that open streams to identify which registration the stream is for.

- IE: MCstrmID, 17 bits

  Description:

  The MCstrmID indicates the UT's desired multicast information which is depended on delivered contents. When the BS delivers only one multicast channel, MCstrmID value is all zero.

msb 1	2	3	4	5	6	7	lsb 8
0	0	0	1	0	0	0	0
msg ID							
0 ext bit	TLVG length (assuming < 128 bytes)						
0	0	0	1	0	0	0	0
msg ID							
Assigned 625K-MC protocol version							
RID							
RID							MCstrmID
MCstrmID							
MCstrmID							
Reserved							

Figure 128 MCStream.req format

25.3.3.2 MCStream.resp Message

Message ID: 17

TLVG: MCStream.resp

TLVG ID: 17

Information Elements:

- IE: Assigned 625k-MC protocol version, 8 bits

  Description:

  The assigned protocol version field indicates which revision of the 625k-MC protocol the sending party is requiring the receiving party to use. This is used by the BS in the MCStream.resp message to tell the UT what protocol version to use throughout the multicast registration.

- IE: McrsrcMap, 48 bits

  Description:

  The McrsrcMap specifies which conventional channels of the BS the UT is allowed to use for Multicast stream. The LSB of this bit map corresponds to conventional channel ID 0. See Section Conventional Channel ID on HC-SDMA [25] page A-3 for a definition of the conventional channel ID.

- IE: reject presence, 1 bit

  Description:

  This reject presence indicates whether the BS chose to reject the multicast registration. When this field is 0, the multicast registration is not rejected. When this field is 1, the multicast registration is rejected by the BS.

- IE: MC reject cause, 5 bit

  Description:

  This MC reject cause is indicated the reason for the multicast rejection. The rejection causes are listed in Table 140 when reject presence field is 1.

Table 140 Multicast connection rejection values

Value	Rejection/Cause
0	Unknown/unspecified
1	The UT’s protocol version is too old
2	The UT’s protocol version is too new
3	The BS’s current traffic load is too high
4	The UT is not registered
5	The BS is not delivered for multicast data
6	The BS is preparing multicast service

Value	Rejection/Cause
7	Invalid MCStreamID value
8-31	Reserved

- IE: MC key size, 5 bits

  Description:

  The K size field MCkey is used by the BS to specify how many bytes of the (decrypted) multicast security stream secret will be used as the i-SEC encryption secret. If this field is non-zero, the key size is the value of this field plus four, in bytes. If the field is zero, this indicates that for testing purposes multicast security is disabled for the registration.

- IE: MC key, 326-640 bits

  Description:

  The MC key consists of i-SEC key (K), i-SEC key(Kr), counter value and AFN. This K, AFN and counter value changes every frame. In fact, i-SEC key and AFN are one-on-one relationship.

  This K is composed in multiples of 8 bits that total size is between 40 bits and 280 bits. The key size depends on the multicast provider. And AFN size is 26 bits, counter value size is 8 bits, and Kr is 326 bits. Total size is 640 bits.

  Multicast delivery data is encrypted to use the part of MC key which is shared secret key by the BS. In contrast, the UT identifies the encrypted data to decrypt the same parts of MC key

- IE: MC Modclass, 4 bits

  Description:

  The MC Modclass indicates multicast delivering modClass from the BS. This delivery modclass shall be set by the BS.

msb							lsb
1	2	3	4	5	6	7	8
0	0	0	1	0	0	0	1
msg ID							
0 ext bit	TLVG length (assuming < 128 bytes)						
0	0	0	1	0	0	0	1
msg ID							
Assigned 625K-MC protocol version							
MCrsrcMap							
MCrsrcMap							
MCrsrcMap							
MCrsrcMap							
MCrsrcMap							
MCrsrcMap							
reject presence	MC reject cause					Reserved	
Reserved			MC key size				
MC key							
MC key							
. . .							
MC Modclass				Reserved			
Reserved							

Figure 129 MCStream.resp format

26 625k-MC Privacy and Authentication Enhancement

As defined in pages 11-1 and 11-2 of Chapter 11 of HC-SDMA [25] ***with the additional text as underlined below***:

The symmetric key encryption engine utilized in the AES for AES algorithm is as described in FIPS PUB 197.

26.1 Overview

As defined in 11.1 of HC-SDMA [25].

26.2 625k-MC Handshake and BS Authentication Protocol, i-HAP

As defined in 11.2 of HC-SDMA [25], ***with the additional definitions and text for the certificate message fields in the section of 11.2.2 f HC-SDMA [25] as underlined below:***

10. *i-SEC* Bulk Encryption Algorithm Choices: Shall be 4 bits. This shall be a bitmap of bulk encryption algorithms supported by the BS. Every supported algorithm shall have a ‘1’ in the relevant bit. For example, if the stream cipher method and the AES modes are supported then this field shall be set to [1,1,0,0].

Bit 0: Stream Cipher (see Section 11.4.2).

Bit 1: AES Cipher (see Section 11.4.3).

Bit 2: Reserved.

Bit 3: Reserved.

11. *i-SEC* Bulk Encryption Parameters: Shall be 20 bits. This field shall contain 4 subfields, each 5 bits. The *k*-th subfield shall contain parameters related to the *k*-th bulk encryption algorithm presented in Field (10) above. If a bulk encryption algorithm is not supported, then the corresponding subfield shall be set to all zeros. The information in each subfield shall depend on the definition and parameters of the corresponding bulk encryption algorithm.

- For the stream cipher:
  - The first subfield shall provide the secret key size in bytes: if this field is all zeros, then the stream cipher is disabled; otherwise, the secret key size is the value represented in first subfield (in little endian format) plus four in bytes.
  - The resulting value (in bytes) shall be defined as the maximum value of encryption key size (L_{key}) supported by the BS.
  - For example, a BS that supports the stream cipher alone, with a maximum encryption secret key size of 56 bits (7 bytes) has:
    - [1,0,0,0] as its Algorithm Choices field (Field 9 above), and

- [1,1,0,0,0;0,0,0,0,0;0,0,0,0,0;0,0,0,0,0] as its Bulk Encryption Parameters field. Note that each subfield represents an integer as a bit string encoded with LSB first.

- For the AES cipher:
  - The first two bits of the second subfield shall be used to indicate the maximum length of secret key size to be used in the AES cipher. The value [10] indicates 128-bit secret key, [01] indicates 192-bit secret key, and [11] indicates a 256-bit secret key for the cipher. The value [0,0] indicates that the AES cipher is disabled.
  - The remaining three bits of the second subfield shall be used to indicate the encryption mode of the AES cipher. If the AES cipher is disabled, then this field must be set to [0,0,0]. The value [1,0,0] indicates that the output feedback (OFB) mode is supported. The remaining three-bit combinations are reserved.
  - For example, a BS that supports the AES cipher alone, with a maximum of 192 bit secret key length and supports the OFB mode has:
    - [0,1,0,0] as its Algorithm Choices Field, and
    - [0,0,0,0,0;**0,1**,**1,0,0**; 0,0,0,0,0; 0,0,0,0,0; 0,0,0,0,0]. Note that each subfield represents an integer as a bit string encoded with LSB first.
  - A BS station that supports **both** the stream cipher (key size 56 bits) and the AES cipher (key size 256 bits) has:
    - [**1,1**,0,0] as its Algorithm Choices Field, and
    - [**1,1,0,0,0;1,1**,**1,0,0**; 0,0,0,0,0; 0,0,0,0,0; 0,0,0,0,0] as its Bulk Encryption Parameters field.

With the above additional definitions, the BS certificate message fields are illustrated in Figure 130.

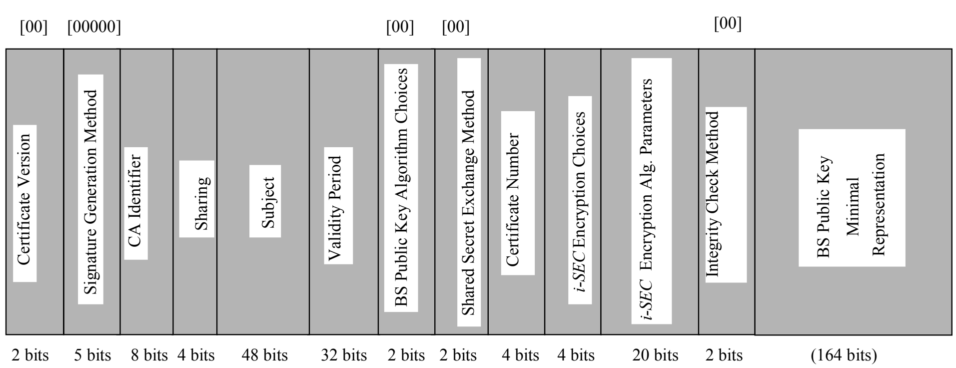

Figure 130 HC-SDMA BS certificate message fields. Mandatory field values are identified in square brackets. Not to scale.

26.3 625k-MC Terminal Authentication Protocol, i-TAP

As defined in 11.3 of HC-SDMA [25].

26.4 625k-MC Secure Communications Protocol, i-SEC

The Secure Communications Protocol, i-SEC shall be as specified in 11.4 of HC-SDMA [25] ***with additional support for minimized RMU header and AES as specified herein***.

In HC-SDMA [25], the *i-SEC* protocol is responsible for symmetric encryption of data and control messages based on the shared secret. In 625k-MC mode, BS and UT shall support the AES cipher algorithm in addition to the stream cipher algorithm used in HC-SDMA [25].

i-SEC protocol is responsible for the encryption of TCH traffic streams that carry AM data and control messages. The BS certificate provides information about the capabilities of the base station in terms of supported algorithms and secret key sizes. In 625k-MC mode, i-SEC shall support the stream cipher algorithm and the AES algorithm explained in this section and shall use a portion[104] of the shared secret established during i-HAP for encryption.

In addition, the BS shall have two thresholds: regThreshBS that limits the number of TCH streams that use the same shared secret and streamThreshBS that limits the number of bursts per TCH stream. The UT shall have its own thresholds regThreshUT and streamThreshUT for the same purpose. The UT shall be responsible for closing TCH streams or initiating new registrations based on the updated values of regThreshUT and streamThreshUT gracefully. The BS shall be responsible for terminating TCH streams and registrations based on its thresholds streamThreshBS and regThreshBS.

The i-SEC encryption key K shall consist of the L_{key} lowest significant bytes of the shared secret (326 bit long shared secret which shall be exchanged with the current BS public key encryption algorithm). The largest value of the L_{key} parameter supported by a BS shall be presented by field (12) of the BS certificate. For the stream cipher, L_{key} shall have a maximum value of 35 bytes, allowing for encryption keys of up to 280 bits. The remaining 46 bits of the shared secret shall be reserved. For the AES cipher, shall have a maximum value of 32 bytes, allowing for encryption keys of up to 256 bits. The next 32 bits of the shared secret shall be be used for initialization purposes (shared secret bits 0-255 for AES encryption key, and shared secret bits 256-287 are used for initialization). The remaining shared secret bits are reserved. can take on only three values: {16,24,32} bytes. The actual length of the i-SEC encryption key is negotiated between the UT and the BS.

26.4.1 TCH Streams

As defined in 11.4.1 of HC-SDMA [25].

26.4.2 625k-MC Symmetric Key Stream Cipher Algorithm

As specified in 11.4.2 of HC-SDMA [25] ***with the following changes to support minimized RMU header.***

26.4.2.1 Initialization Vector Selection

As specified in 11.4.2.1 of HC-SDMA [25].

[104] A disjoint portion of the shared secret shall be used for scrambling in *i-TAP*.

26.4.2.2 Stream Counter Test

As specified in 11.4.2.2 of HC-SDMA [25]

26.4.2.3 Determination of Encryption Key from Shared Secret

As specified in 11.4.2.3 of HC-SDMA [25]

26.4.2.4 Stream Cipher State Initialization

As specified in 11.4.2.4 of HC-SDMA [25]

26.4.2.5 Keystream Generation

As specified in 11.4.2.5 of HC-SDMA [25] ***with the additional text as underlined below:***

- • The maximum uplink burst is 1000 bits, <u>including 16 bits of CRC</u>. The CRC shall not be encrypted in i-SEC.
- • The maximum downlink burst is 2539 bits, <u>including 16 bits of CRC</u>. The CRC shall not be encrypted in *i-SEC*.

26.4.2.6 Burst Counter Test

As specified in 11.4.2.6 of HC-SDMA [25]

26.4.2.7 Encryption and Decryption Using i-SEC Keystream

As specified in Section 11.4.2.7 of HC-SDMA [25] with the additional text as underlined below:

i-SEC encryption shall be performed by XORing the RMU header and payload (see Section 7.3.7.2 RMU MuxFunction on page 7-45 and Figure 7.9 on page 7-47 of HC-SDMA [25] for definitions) by the *i-SEC* keystream before transmission. Therefore, *i-SEC* encryption is not a bandwidth expanding transformation. The 16 bit CRC shall be calculated over the encrypted bits. The CRC shall not be encrypted. *i-SEC* decryption shall be performed by XORing the received encrypted RMU header and payload octets by the *i-SEC* keystream. The 16 bit CRC check shall be performed on the received encrypted bits before decryption.

<u>In streams using the basic RMU header,</u> The RMU header and payload shall be aligned such that the last payload falls exactly at the end of the relevant keystream buffer. <u>In streams using the minimized RMU header, the last burst bit before the CRC bits shall be aligned with the end of the relevant keystream buffer.</u> During this process, some of the keystream bytes <u>or bits</u> are possibly left unused and they shall be discarded.

For example, consider an uplink burst payload that contains 28 bits of <u>basic</u> RMU header and 3 payload octets {ABC} with a <u>basic</u> RMU header H represented by a bit vector $[h_0, h_1, \dots h_{27}]$ where h_0 is transmitted first and the header is partitioned into four bytes: $H_0 = [0,0,0,0,h_0,h_1,h_2,h_3]$, $H_1 = [h_4,h_5,h_6,h_7,h_8,h_9,h_{10},h_{11}]$, $H_2 = [h_{12},h_{13},h_{14},h_{15},h_{16},h_{17},h_{18},h_{19}]$, and $H_3 = [h_{20},h_{21},h_{22},h_{23},h_{24},h_{25},h_{26},h_{27}]$.

26.4.3 625k-MC AES Cipher Algorithm

This complete subsection 26.4.3 is added before end of Section 11.4 of HC-SDMA [25]

i-SEC shall employ the symmetric key encryption algorithm for AES. The AES algorithm is defined in FIPS PUB 197. i-SEC shall support the AES encryption algorithm. The following sections provide the details. Note that the AES cipher can only utilize up to 256 least significant bits of the shared secret (K_r) established during i-HAP as the AES cipher key (K_{AES}).

26.4.3.1 AES Initialization Vector Selection

The *i-SEC* encryption key K_{AES} shall be utilized together with a 128-bit AES initialization vector (IV_{AES}) to initialize the AES cipher operation at the beginning of a stream. IV_{AES} shall be different for each TCH stream over the expected lifetime of the shared secret key.

Let IV denote the 32-bit initialization vector defined in Section 11.3.3 of the HC-SDMA standard for the stream cipher. Let us define the 32 bit vector W:

$$W(n) = K_r(n+256), \quad n=0,1,\ldots,31.$$

Where W (n) denotes the n-th bit of W. The 32-bit vector W will be modified by a component-wise XOR with the IV defined in Section 11.3.3 of the HC-SDMA standard.

$$W(n) = W(n) \text{ XOR } IV(n), \quad n=0,1,\ldots,31.$$

The initialization vector for the AES algorithm is defined as the concatenation of W to obtain a length 128-bit vector:

$$IV_{AES} = [W, W, W, W];$$

As a result the true value of IV_{AES} is unknown to parties other than the BS and the UT.

26.4.3.2 AES Stream Counter Test

As specified in 11.4.2.2 of HC-SDMA [25].

26.4.3.3 Determination of AES Encryption Key from Shared Secret

The length (in bytes) of *i-SEC* encryption key (K) shall be defined as L_{key}. The second subfield of Field (12) of the BS certificate shall specify the maximum value of L_{key} supported by the BS. The value of L_{key} shall be in the set {16, 24, 32} bytes for the AES cipher.

The first ($L_{key} \times 8$) least significant bits of the shared secret K_r shall be defined as the encryption key, K_{AES}:

$$K_{AES}(n) = K_r(n), \quad n=0,1,\ldots,8L_{key}-1.$$

Where $K_{AES}(n)$ is the *n*-th bit of the *i-SEC* encryption key K_{AES}, and $K_r(n)$ is the *n*-th bit of the *i-SEC* encryption key K_r. For maximally secure deployments, L_{key} may be set to 32[1]. The actual key size used by *i-SEC* shall be negotiated during registration. The BS certificate shall set the maximum length on the key size that the BS supports. The UT shall send the key length to be used for the registration in the `UT Params` message. Test UTs may use a method to bypass *i-SEC* encryption.

26.4.3.4 AES Cipher State Initialization

Definition 1: Let Y=***E*** (Z;M) denote the encryption of a 128 bit message vector M, with an AES key named as Z (128, 192 or 256 bits) as defined in the FIPS Standard 197. The cipher output is a 128 bit vector Y.

Definition 2: Let M=***D*** (Z;Y) denote the decryption of a 128 bit vector Y, with an AES key named as Z (128, 192 or 256 bits) as defined in the FIPS Standard 197.

Each stream shall maintain a single AES encryption engine, and a 128-bit initialization vector (IV_{AES}). The computation initialization vector (IV_{AES}) is defined in Section 26.4.3.1 above.

At the beginning of each stream, the AES initialization vector is updated (IV_{AES}) by two successive encryption operations with the AES encryption key (K_{AES}) before encrypting any information bits:

$$IV_{AES} = \boldsymbol{E}(K_{AES}\ ,\boldsymbol{E}(K_{AES}\ ,IV_{AES}));$$

This operation takes place both at the message originator (encryptor) and the message receiver (decryptor).

26.4.3.5 AES Keystream Generation

After the double encryption operation on IV_{AES} (refer to previous section) that precedes the keystream generation procedure, a block of 128 keystream bits $U=[U_0,\dots,U_{127}]$ is generated and the initialization vector IV_{AES} is updated by the following procedure:

$$U= \boldsymbol{E}(K_{AES}\ ,IV_{AES});$$

$$IV_{AES}(n) = U_n \text{ XOR } IV_{AES}(n), n=0,1,\dots 127;$$

Note that each call to the above two-line procedure generates 16 keystream bytes.

On every frame following the RA/AA frame, 448-keystream bytes (28 calls to the above procedure per frame) shall be generated for uplink and downlink encryption. The first 128-keystream bytes (8 calls) shall be allocated for uplink encryption and the remaining 320-keystream bytes (20 calls) are allocated for downlink encryption.

Note that these keystream bytes will be generated regardless of the modulation class employed. The unused keystream bytes will be disregarded. The encryption/decryption procedure using the keystream is explained in Section 26.4.3.7 below.

26.4.3.6 AES Burst Counter Test

The UT shall increment its counter `burstCount` after generating the keystream for a burst. This counter shall be set to zero at every stream start. After incrementing `burstCount`, the UT shall compare the value of the counter to a threshold entitled `streamThreshUT`. The UT application software shall set the value for the stream termination threshold `streamThreshUT`. If `streamThreshUT` is non-zero, and the value of `burstCount` is equal to `streamThreshUT`, then the UT shall close the current stream and shall start a new stream with a new stream secret and IV. If the value of `streamThreshUT` is zero, then the UT shall not force a stream closure.

The BS shall perform the same operations independently by using its own counter `burstCount`, and its own threshold `streamThreshBS`. A nominal value for `streamThreshBS and streamThreshUT` shall be 20,000 bursts. At a combined uplink/downlink approximate data rate of 700 kilobits/sec (single burst/frame, 200 frames/sec), this results in a stream closure approximately after 7.5 megabytes of over the air traffic before the initialization vector is changed and the stream cipher is reinitialized. This selection also limits the stream duration to 100 seconds regardless of the data rate."

26.4.3.7 Encryption and Decryption Using i-SEC AES Keystream

As specified in Section 11.4.2.7 of HC-SDMA [25] ***with the additional text as underlined below:***

i-SEC encryption shall be performed by XORing the RMU header and payload (see Section 7.3.7.2 RMU MuxFunction on page 7-45 and Figure 7.9 on page 7-47 of HC-SDMA [25] for definitions) by the *i-SEC* keystream before transmission. Therefore, *i-SEC* encryption is not a bandwidth expanding transformation. The 16 bit CRC shall be calculated over the encrypted bits. The CRC shall not be encrypted. *i-SEC* decryption shall be performed by XORing the received encrypted RMU header and payload by the *i-SEC* keystream. The 16 bit CRC check shall be performed on the received encrypted bits before decryption.

In streams using the basic RMU header, The RMU header and payload shall be aligned such that the last payload falls exactly at the end of the relevant keystream buffer. In streams using the minimized RMU header, the last burst bit before the CRC bits shall be aligned with the end of the relevant keystream buffer. During this process, some of the keystream bytes or bits are possibly left unused and they shall be discarded.

For example, consider an uplink burst payload that contains 28 bits of basic RMU header and 3 payload octets {ABC} with a basic RMU header H represented by a bit vector $[h_0, h_1, \ldots h_{27}]$ where h_0 is transmitted first and the header is partitioned into four bytes: $H_0 = [0,0,0,0,h_0,h_1,h_2,h_3]$, $H_1 = [h_4,h_5,h_6,h_7,h_8,h_9,h_{10},h_{11}]$, $H_2 = [h_{12},h_{13},h_{14},h_{15},h_{16},h_{17},h_{18},h_{19}]$, and $H_3 = [h_{20},h_{21},h_{22},h_{23},h_{24},h_{25},h_{26},h_{27}]$.

27 625k-MC Sleep Mode Control Protocol

This Chapter is an added to the baseline specification HC-SDMA [25].

In Idle mode UT periodically monitors own paging PCH (Refer to 7.3.3 of HC-SDMA [25]) for wake up requirement. When BS needs to deliver data to UT in Idle Mode, BS shall follow the procedure as specified in 9.3.4.1 of HC-SDMA [25]. The frequency of pages from BS shall conform to five discrete levels of paging activity levels (Paging States) transitions from high level to low level: Page every frame, Page every 8th frame, Page every 64th frame, Page every 512th frame and No paging at all, as specified in 9.3.4.1.2 of HC-SDMA [25]. BS and UT shall agree during the registration phase on the length of duration of paging activity before changing to next lower paging activity, thereby allowing the power conservation feature in 625k-MC mode. This feature may be used to plan the hardware resources at UT at powerdown mode.

28 625k-MC OA & M Radio Network Quality Monitor and Control Enhancement

This Chapter is an added to the baseline specification HC-SDMA [25].

625k-MC network systems provide radio network quality monitoring and control functionality. The MIB of 625k-MC mode comprises of the managed objects, attributes, actions, and notifications required to manage a BS. The definition of these managed objects, attributes, actions, and notifications, as well as their structure, is presented below.

28.1 625k-MC Mode MIB

28.1.1 Overview

This chapter defines a Management Information Base (MIB) module for managing the 625k-MC mode. Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP).The objects in this MIB are defined using the mechanisms specified in the Structure of Management Information (SMI). The MIB module specified is compliant to SMIv2 which is described in RFC 2578 [18], RFC 2579 [19], and RFC 2580 [20].

28.1.2 Definition

```

IEEE802dot20-625k-MC-MIB  DEFINITIONS ::= BEGIN

IMPORTS
    OBJECT-TYPE
       FROM RFC-1212
    enterprises, Gauge, Counter
       FROM RFC1155-SMI
    transmission  FROM RFC1213-MIB
   ;

IEEE802dot20-625k-MC-MIB MODULE-IDENTITY
   LAST-UPDATED        ""
   ORGANIZATION        "IEEE 802.20"
   CONTACT-INFO        ""
    DESCRIPTION
        "The MIB private module for IEEE 802.20 entities"
    ::= { enterprises 9999 }

_625k-MCSystem                          OBJECT IDENTIFIER
    -- DESCRIPTION       "System Elements"
    ::= { IEEE802dot20-625k-MC-MIB 1 }


_625k-MCSysAlarms                       OBJECT IDENTIFIER
    -- DESCRIPTION       "Alarms"
    ::= { _625k-MCSystem 1}



_625k-MCAlarmScalars                    OBJECT IDENTIFIER
    -- DESCRIPTION       "Alarm Scalars"
    ::= { _625k-MCSysAlarms 1 }
```

```
_625k-MCCommonAlarmStatus               OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Common alarm atatus.



        (From mibCtl ElementType 16 CommonAlarmStatus)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCAlarmScalars 1 }



_625k-MCFailReasonForAlarm              OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Fail reason for alarm.



        (From mibCtl ElementType 15 FailReasonForAlarm)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCAlarmScalars 2 }




_625k-MCAlarmSummaryTable               OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCAlarmSummaryTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "Alarm Summary Table"
    ::= { _625k-MCSysAlarms 2 }



_625k-MCAlarmSummaryTableEntry          OBJECT-TYPE
    SYNTAX           _625k-MCAlarmSummaryTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCAlarmSummaryTableIndex }
    ::= { _625k-MCAlarmSummaryTable 1 }

_625k-MCAlarmSummaryTableEntry ::= SEQUENCE {
   _625k-MCAlarmSummaryTableIndex         INTEGER, -- AlarmEventType
   _625k-MCAlarmSummary                   INTEGER -- AlarmStateType
    }
```

```
_625k-MCAlarmSummaryTableIndex          OBJECT-TYPE
    SYNTAX            INTEGER -- AlarmEventType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 85 AlarmEventType :
            Enumeration of alarm event types.

            Defines semantics of events that are also alarms.
            All alarm events are enumerated first in the list of event types.
            The highest alarm event index will never be more than 255.
            [Limits: 0 255 ]
        Type derived from mibCtl Type 3 EventType :
            Enumeration of event types.

            Defines semantics of events.
            An event is re. an event log message.
            [Limits: 0 255 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCAlarmSummaryTableEntry 1 }



_625k-MCAlarmSummary                    OBJECT-TYPE
    SYNTAX            INTEGER -- AlarmStateType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Summary of all alarms generated by the base station.

        Each element contains the summary of a type of alarm (e.g. module
        over temperature).  When the management station sees that alarm
        summary is SET, it can, for example, query AlarmModuleOverTemp
        to see which module(s) is over temperature.

        (From mibCtl ElementType 5210 AlarmSummary)
        Description for mibCtl Type 80 AlarmStateType :
            Current state of an alarm.

            This value is CLEARED when
            the conditions which caused the alarm to occur are taken care of
            and no longer exist.
            The value is SET when due to some conditions, the Base Station
            software decides that an alarm is necessary.
            Typically (though this may not be true for all alarms
            or if the alarm changes state too frequently)
            an event is logged when an alarm is SET and then again when it
            is CLEARED.
            [Limits: 0 1 ]
        Description for mibCtl AlarmStateType 0 CLEARED :
            No alarm.
        Description for mibCtl AlarmStateType 1 SET :
            Alarm is set.
        "
    ::= { _625k-MCAlarmSummaryTableEntry 2 }



_625k-MCSysFiles                        OBJECT IDENTIFIER
    -- DESCRIPTION       "Files"
    ::= { _625k-MCSystem 2}
```

```
_625k-MCStatsFiles                      OBJECT IDENTIFIER
    -- DESCRIPTION       "Statistics file"
    ::= { _625k-MCSysFiles 1 }



_625k-MCStatsUploadURL                  OBJECT-TYPE
    SYNTAX              OCTET STRING (SIZE(0..64)) -- URLType
    ACCESS              read-write
    STATUS              mandatory
    DESCRIPTION
        "EMS location to upload BS statistics file.



        (From mibCtl ElementType 2831 StatsUploadURL)
        Description for mibCtl Type 401 URLType :
            Universal Resource Locator (URL).

            A Universal Resource Locator (URL) is a text string
            that specifies a network location for a file.
            The general format for a URL consists of 2 parts:

            1. Protocol name: lower case letters, followed by a colon.
            See below for supported protocols.
            This field may be omitted, to default to the file: protocol.

            2. Additional information, depending on the protocol.
            For many protocols, a host name is required,
            which consists of a dotted numerical Internet Protocol (IP)
address,
            or a dotted symbolic name with alphanumerical components,
            where supported.

            Supported protocols are:

            tftp: is the Trivial File Transfer Protocol.
            The additional information should begin with two slashes (//)
            followed by a host name, a slash (/) and a file path.
            The file path is interpreted by the host system,
            frequently relative to a special directory set up for this
purpose.

            file: is the plain old file protocol.
            The additional information consists of a file path, which
            should begin with a slash (/).
            This is only useful if Base Station has been configured
            to be an Network File System (NFS) client of the host.
            The filepath is interpreted on the Base Station, so it
            must begin with the mount name specified in the NFS configuration.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCStatsFiles 1 }



_625k-MCStatsUploadStatus               OBJECT-TYPE
    SYNTAX              INTEGER -- FileUploadStatusType
    ACCESS              read-only
    STATUS              mandatory
    DESCRIPTION
        "Stats file upload status.
```

```
        (From mibCtl ElementType 2832 StatsUploadStatus)
        Description for mibCtl Type 403 FileUploadStatusType :
            File upload status.


        Description for mibCtl FileUploadStatusType 0 Unknown :
            File upload status is unknown.
        Description for mibCtl FileUploadStatusType 1 Missing :
            File is missing or invalid.
        Description for mibCtl FileUploadStatusType 2 PartialUpload :
            File is in the process of being upload to EMS.
        Description for mibCtl FileUploadStatusType 4 Complete :
            File is completely uploaded to EMS.
        Description for mibCtl FileUploadStatusType 5 Failure :
            Upload process is failure.
        Description for mibCtl FileUploadStatusType 6 NotManaged :
            File upload is not being managed.
        "
    ::= { _625k-MCStatsFiles 2 }



_625k-MCStatsUploadFailReason           OBJECT-TYPE
    SYNTAX            INTEGER -- FileUploadFailReasonType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Reason for last stats file upload failure.



        (From mibCtl ElementType 2833 StatsUploadFailReason)
        Description for mibCtl Type 406 FileUploadFailReasonType :
            Reason for failure to upload a file..


        Description for mibCtl FileUploadFailReasonType 0 NoFailure :
            File upload in progress or completed without problem.
        Description for mibCtl FileUploadFailReasonType 1 BadPathSpecified :
            File upload failed because network path not found.
        Description for mibCtl FileUploadFailReasonType 2 FlashDiskReadError :
            File upload failed because of flash disk read error.
        Description for mibCtl FileUploadFailReasonType 3 Aborted :
            File upload aborted due to change of specification.
        Description for mibCtl FileUploadFailReasonType 4 WriteError :
            Error in putting a file.
        "
    ::= { _625k-MCStatsFiles 3 }



_625k-MCStatsUploadBytes                OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Upload size of BS stats file in bytes.



        (From mibCtl ElementType 2834 StatsUploadBytes)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
```

```
        "
    ::= { _625k-MCStatsFiles 4 }



_625k-MCStatsUploadDate                  OBJECT-TYPE
    SYNTAX            Gauge -- AbsoluteTimeType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "BS Stats File upload complete time.



        (From mibCtl ElementType 2835 StatsUploadDate)
        Description for mibCtl Type 801 AbsoluteTimeType :
            Absolute time in GPS seconds.

            GPS (Global Positioning System) time in seconds since Jan. 6,
1980.
            Note that this differs from UTC (in addition to a possible
            offset due to starting time) due to leap seconds; see
            the GpsLeapSecond element.
        Type derived from mibCtl Type 18 Gauge32Type :
            32 bits of Gauge data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCStatsFiles 5 }



_625k-MCUploadStatsFile                  OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-write -- REALLY: write-only
    STATUS            mandatory
    DESCRIPTION
        "Upload Stats file.



        (From mibCtl ElementType 2836 UploadStatsFile)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCStatsFiles 6 }


_625k-MCSysInterfaces                    OBJECT IDENTIFIER
    -- DESCRIPTION       "System Interfaces"
    ::= { _625k-MCSystem 3 }



_625k-MCInterfaceNetwork                 OBJECT IDENTIFIER
    -- DESCRIPTION       "Network Interfaces"
    ::= { _625k-MCSysInterfaces 1 }



_625k-MCTypeOfNetworkProtocol            OBJECT-TYPE
    SYNTAX            INTEGER -- NetworkProtocolType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Type of Network Protocol used with the Network.
```

```
        Type of Network Protocol is Ethernet or ATM.

        (From mibCtl ElementType 3002 TypeOfNetworkProtocol)
        Description for mibCtl Type 214 NetworkProtocolType :
            Network Protocol type.
        Description for mibCtl NetworkProtocolType 0 Unknown :
            Network protocol type is unknown.
        Description for mibCtl NetworkProtocolType 1 Ethernet :
            Ethernet interface.
        Description for mibCtl NetworkProtocolType 2 ATM :
            ATM interface.
        "
    ::= { _625k-MCInterfaceNetwork 1 }




_625k-MCMgmtNetConfigTable              OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCMgmtNetConfigTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "Mgmt Network Configuration"
    ::= { _625k-MCInterfaceNetwork 2 }



_625k-MCMgmtNetConfigTableEntry         OBJECT-TYPE
    SYNTAX           _625k-MCMgmtNetConfigTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCMgmtNetConfigTableIndex }
    ::= { _625k-MCMgmtNetConfigTable 1 }

_625k-MCMgmtNetConfigTableEntry ::= SEQUENCE {
   _625k-MCMgmtNetConfigTableIndex        INTEGER, -- MoNerdAddressType
   _625k-MCEthernetIPAddress              OCTET STRING (SIZE(0..15)), --
IPAddressTextType
   _625k-MCEthernetIPLocalBits            OCTET STRING (SIZE(0..15)), --
IPAddressTextType
   _625k-MCEthernetHostName               OCTET STRING (SIZE(0..20)) --
TextType X 20
    }



_625k-MCMgmtNetConfigTableIndex         OBJECT-TYPE
    SYNTAX            INTEGER -- MoNerdAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 204 MoNerdAddressType :
            Base station network component address.

            A network address is a subset of Base Station component addresses,
            restricted to network components only.
            Network components interface with a telephony switch or similar.
            [Limits: 0 1 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCMgmtNetConfigTableEntry 1 }
```

```
_625k-MCEthernetIPAddress               OBJECT-TYPE
    SYNTAX           OCTET STRING (SIZE(0..15)) -- IPAddressTextType
    ACCESS           read-write
    STATUS           mandatory
    DESCRIPTION
        "Internet Protocol (IP) address for ethernet port of Module.

        This is the actual IP address in use for the ethernet port
        of a given Module.
        If IP is not being used on the ethernet port, or there is
        no ethernet port, then an empty string is provided for this element.

        (From mibCtl ElementType 2811 EthernetIPAddress)
        Description for mibCtl Type 420 IPAddressTextType :
            Internet Protocol Address (Text).

            This text must currently be in the dotted abc.def.ghi.jkl format.
            In the future, hostnames might be allowed.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCMgmtNetConfigTableEntry 2 }



_625k-MCEthernetIPLocalBits             OBJECT-TYPE
    SYNTAX           OCTET STRING (SIZE(0..15)) -- IPAddressTextType
    ACCESS           read-write
    STATUS           mandatory
    DESCRIPTION
        "Ethernet IP (Internet Protocol) local routing bit count.

        This indicates how many of the low-order bits of
        the IP address of the ethernet connection are used
        within the local network.
        The remaining (high-order) bits are the same for all
        hosts on the local network.
        This is used as the first part of the routing algorithm.
        IP addresses that do not share the upper bits of the ethernet
        IP address and which are not otherwise resolved will be sent
        through the gateway, if defined.

        For example, 255.255.255.0

        (From mibCtl ElementType 2812 EthernetIPLocalBits)
        Description for mibCtl Type 420 IPAddressTextType :
            Internet Protocol Address (Text).

            This text must currently be in the dotted abc.def.ghi.jkl format.
            In the future, hostnames might be allowed.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCMgmtNetConfigTableEntry 3 }



_625k-MCEthernetHostName                OBJECT-TYPE
    SYNTAX           OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS           read-write
    STATUS           mandatory
```

```
    DESCRIPTION
        "Ethernet IP host name for module.

        (From mibCtl ElementType 2813 EthernetHostName)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCMgmtNetConfigTableEntry 4 }




_625k-MCUserNetConfigTable              OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCUserNetConfigTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "User Network Configuration"
    ::= { _625k-MCInterfaceNetwork 3 }



_625k-MCUserNetConfigTableEntry         OBJECT-TYPE
    SYNTAX           _625k-MCUserNetConfigTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCUserNetConfigTableIndex }
    ::= { _625k-MCUserNetConfigTable 1 }

_625k-MCUserNetConfigTableEntry ::= SEQUENCE {
   _625k-MCUserNetConfigTableIndex        INTEGER, -- MoNerdAddressType
   _625k-MCUserEthernetIPAddress          OCTET STRING (SIZE(0..15)), --
IPAddressTextType
   _625k-MCUserEthernetIPLocalBits        OCTET STRING (SIZE(0..15)), --
IPAddressTextType
   _625k-MCUserEthernetHostName           OCTET STRING (SIZE(0..20)) --
TextType X 20
    }



_625k-MCUserNetConfigTableIndex         OBJECT-TYPE
    SYNTAX            INTEGER -- MoNerdAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 204 MoNerdAddressType :
            Base station network component address.

            A network address is a subset of Base Station component addresses,
            restricted to network components only.
            Network components interface with a telephony switch or similar.
            [Limits: 0 1 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCUserNetConfigTableEntry 1 }



_625k-MCUserEthernetIPAddress           OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..15)) -- IPAddressTextType
```

```
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Internet Protocol (IP) address for user ethernet port of Module.

        This is the actual IP address in use for the ethernet port
        of a given Module.
        If IP is not being used on the ethernet port, or there is
        no ethernet port, then an empty string is provided for this element.

        (From mibCtl ElementType 2817 UserEthernetIPAddress)
        Description for mibCtl Type 420 IPAddressTextType :
            Internet Protocol Address (Text).

            This text must currently be in the dotted abc.def.ghi.jkl format.
            In the future, hostnames might be allowed.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCUserNetConfigTableEntry 2 }



_625k-MCUserEthernetIPLocalBits         OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..15)) -- IPAddressTextType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Ethernet IP subnet mask for user network.

        Ethernet IP subnet mask for user network

        (From mibCtl ElementType 2818 UserEthernetIPLocalBits)
        Description for mibCtl Type 420 IPAddressTextType :
            Internet Protocol Address (Text).

            This text must currently be in the dotted abc.def.ghi.jkl format.
            In the future, hostnames might be allowed.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCUserNetConfigTableEntry 3 }



_625k-MCUserEthernetHostName            OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "User ethernet IP host name for module.

        (From mibCtl ElementType 2819 UserEthernetHostName)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCUserNetConfigTableEntry 4 }
```

```
_625k-MCUserNetStatusTable              OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCUserNetStatusTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "Network Status"
    ::= { _625K-MCInterfaceNetwork 4 }



_625k-MCUserNetStatusTableEntry         OBJECT-TYPE
    SYNTAX           _625k-MCUserNetStatusTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCUserNetStatusTableIndex }
    ::= { _625k-MCUserNetStatusTable 1 }

_625k-MCUserNetStatusTableEntry ::= SEQUENCE {
   _625k-MCUserNetStatusTableIndex        INTEGER, -- MoNerdAddressType
   _625k-MCNetworkInOctets                Counter, -- Counter32Type
   _625k-MCNetworkOutOctets               Counter -- Counter32Type
    }



_625k-MCUserNetStatusTableIndex         OBJECT-TYPE
    SYNTAX            INTEGER -- MoNerdAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 204 MoNerdAddressType :
            Base station network component address.

            A network address is a subset of Base Station component addresses,
            restricted to network components only.
            Network components interface with a telephony switch or similar.
            [Limits: 0 1 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCUserNetStatusTableEntry 1 }



_625k-MCNetworkInOctets                 OBJECT-TYPE
    SYNTAX            Counter -- Counter32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "In octets user data of network.



        (From mibCtl ElementType 1000 NetworkInOctets)
        Description for mibCtl Type 19 Counter32Type :
            32 bits of Counter data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCUserNetStatusTableEntry 2 }



_625k-MCNetworkOutOctets                OBJECT-TYPE
```

```
    SYNTAX            Counter -- Counter32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Out octets user data of network.



        (From mibCtl ElementType 1001 NetworkOutOctets)
        Description for mibCtl Type 19 Counter32Type :
            32 bits of Counter data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCUserNetStatusTableEntry 3 }




_625k-MCL2TPConfigTable                 OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCL2TPConfigTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "L2TP Configuration Table"
    ::= { _625k-MCInterfaceNetwork 5 }



_625k-MCL2TPConfigTableEntry            OBJECT-TYPE
    SYNTAX           _625k-MCL2TPConfigTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCL2TPConfigTableIndex }
    ::= { _625k-MCL2TPConfigTable 1 }

_625k-MCL2TPConfigTableEntry ::= SEQUENCE {
   _625k-MCL2TPConfigTableIndex           INTEGER, -- MoNerdAddressType
   _625k-MCL2TPPeerName                   OCTET STRING (SIZE(0..20)), --
TextType X 20
   _625k-MCL2TPPeerIPAddress              OCTET STRING (SIZE(0..15)), --
IPAddressTextType
   _625k-MCL2TPAVPHostName                OCTET STRING (SIZE(0..20)), --
TextType X 20
   _625k-MCL2TPAVPChallAndRes             OCTET STRING (SIZE(0..20)) --
TextType X 20
    }



_625k-MCL2TPConfigTableIndex            OBJECT-TYPE
    SYNTAX            INTEGER -- MoNerdAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 204 MoNerdAddressType :
            Base station network component address.

            A network address is a subset of Base Station component addresses,
            restricted to network components only.
            Network components interface with a telephony switch or similar.
            [Limits: 0 1 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCL2TPConfigTableEntry 1 }
```

```
_625k-MCL2TPPeerName                    OBJECT-TYPE
    SYNTAX           OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS           read-write
    STATUS           mandatory
    DESCRIPTION
        "L2TP peer name.

        Tunnel switch host name

        (From mibCtl ElementType 2000 L2TPPeerName)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCL2TPConfigTableEntry 2 }



_625k-MCL2TPPeerIPAddress               OBJECT-TYPE
    SYNTAX           OCTET STRING (SIZE(0..15)) -- IPAddressTextType
    ACCESS           read-write
    STATUS           mandatory
    DESCRIPTION
        "L2TP peer IP Address.

        Tunnel switch IP Address

        (From mibCtl ElementType 2001 L2TPPeerIPAddress)
        Description for mibCtl Type 420 IPAddressTextType :
            Internet Protocol Address (Text).

            This text must currently be in the dotted abc.def.ghi.jkl format.
            In the future, hostnames might be allowed.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCL2TPConfigTableEntry 3 }



_625k-MCL2TPAVPHostName                 OBJECT-TYPE
    SYNTAX           OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS           read-write
    STATUS           mandatory
    DESCRIPTION
        "BS host name using L2TP.



        (From mibCtl ElementType 2008 L2TPAVPHostName)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCL2TPConfigTableEntry 4 }
```

```
_625k-MCL2TPAVPChallAndRes             OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "AVP challenge and response name.



        (From mibCtl ElementType 2012 L2TPAVPChallAndRes)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCL2TPConfigTableEntry 5 }




_625k-MCL2TPStatusTable                OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCL2TPStatusTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "L2TP Status Table"
    ::= { _625k-MCInterfaceNetwork 6 }



_625k-MCL2TPStatusTableEntry           OBJECT-TYPE
    SYNTAX           _625k-MCL2TPStatusTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCL2TPStatusTableIndex }
    ::= { _625k-MCL2TPStatusTable 1 }

_625k-MCL2TPStatusTableEntry ::= SEQUENCE {
   _625k-MCL2TPStatusTableIndex           INTEGER, -- MoNerdAddressType
   _625k-MCL2TPActiveSession              INTEGER, -- Unsigned32Type
   _625k-MCL2TPActiveTunnel               INTEGER -- Unsigned32Type
    }



_625k-MCL2TPStatusTableIndex           OBJECT-TYPE
    SYNTAX            INTEGER -- MoNerdAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 204 MoNerdAddressType :
            Base station network component address.

            A network address is a subset of Base Station component addresses,
            restricted to network components only.
            Network components interface with a telephony switch or similar.
            [Limits: 0 1 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCL2TPStatusTableEntry 1 }
```

```
_625k-MCL2TPActiveSession                 OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "L2TP active session.



        (From mibCtl ElementType 2013 L2TPActiveSession)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCL2TPStatusTableEntry 2 }



_625k-MCL2TPActiveTunnel                  OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "L2TP active tunnel.



        (From mibCtl ElementType 2014 L2TPActiveTunnel)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCL2TPStatusTableEntry 3 }




_625k-MCATMConfigTable                    OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCATMConfigTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "ATM Configuration Table"
    ::= { _625k-MCInterfaceNetwork 7 }



_625k-MCATMConfigTableEntry               OBJECT-TYPE
    SYNTAX           _625k-MCATMConfigTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCATMConfigTableIndex }
    ::= { _625k-MCATMConfigTable 1 }

_625k-MCATMConfigTableEntry ::= SEQUENCE {
   _625k-MCATMConfigTableIndex            INTEGER, -- MoNerdAddressType
   _625k-MCAtmAddress                     OCTET STRING (SIZE(0..40)), --
TextType X 40
   _625k-MCAtmVCTypes                     INTEGER, -- AtmVCType
   _625k-MCAtmFrameTypes                  INTEGER, -- AtmFrameType
   _625k-MCAtmUNIVersion                  INTEGER, -- AtmUNIVersionType
   _625k-MCAtmLineStatus                  INTEGER, -- LineStatusType
   _625k-MCAtmParameterFailReason         INTEGER, -- Unsigned32Type
   _625k-MCAtmOpenChannelFailReason       INTEGER, -- Unsigned32Type
```

```
     _625k-MCAtmChannelNumber               INTEGER, -- Unsigned32Type
     _625k-MCAtmAlarmCauseRegister          INTEGER, -- Unsigned32Type
     _625k-MCAtmPHYIntrCauseRegister        INTEGER -- Unsigned32Type
      }



  _625k-MCATMConfigTableIndex            OBJECT-TYPE
      SYNTAX            INTEGER -- MoNerdAddressType
      ACCESS            read-only
      STATUS            mandatory
      DESCRIPTION       "
          Description for mibCtl Type 204 MoNerdAddressType :
              Base station network component address.

              A network address is a subset of Base Station component addresses,
              restricted to network components only.
              Network components interface with a telephony switch or similar.
              [Limits: 0 1 ]
          Type derived from mibCtl Type 14 Unsigned32Type :
              32 bit unsigned integer.
          Type derived from mibCtl Type 11 Word32Type :
              32 bits of raw opaque data.
          Derived from basic 32 bit word type.
      "
      ::= { _625k-MCATMConfigTableEntry 1 }



  _625k-MCAtmAddress                     OBJECT-TYPE
      SYNTAX            OCTET STRING (SIZE(0..40)) -- TextType X 40
      ACCESS            read-write
      STATUS            mandatory
      DESCRIPTION
          "Atm Address.



          (From mibCtl ElementType 1950 AtmAddress)
          Description for mibCtl Type 15 TextType :
              ASCII or compatible text.
          Type derived from mibCtl Type 12 OctetType :
              8 bits of raw opaque data.
          Derived from basic 8 bit word type.
          "
      ::= { _625k-MCATMConfigTableEntry 2 }



  _625k-MCAtmVCTypes                     OBJECT-TYPE
      SYNTAX            INTEGER -- AtmVCType
      ACCESS            read-write
      STATUS            mandatory
      DESCRIPTION
          "Atm VC Type.



          (From mibCtl ElementType 1951 AtmVCTypes)
          Description for mibCtl Type 300 AtmVCType :
              ATM VC Type.
          Description for mibCtl AtmVCType 0 Unknown :
              ATM VC Type is Unknown.
          Description for mibCtl AtmVCType 1 PVC :
              ATM VC Type is PVC.
          Description for mibCtl AtmVCType 2 SVC :
              ATM VC Type is SVC.
          Description for mibCtl AtmVCType 3 PVCSVC :
```

```
            ATM VC Type is PVC & SVC.
        "
    ::= { _625k-MCATMConfigTableEntry 3 }



_625k-MCAtmFrameTypes                   OBJECT-TYPE
    SYNTAX            INTEGER -- AtmFrameType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Atm Frame Type.



        (From mibCtl ElementType 1952 AtmFrameTypes)
        Description for mibCtl Type 301 AtmFrameType :
            ATM Frame Type.
        Description for mibCtl AtmFrameType 0 Unknown :
            ATM Frame Type is Unknown.
        Description for mibCtl AtmFrameType 1 OC48 :
            ATM Frame Type is OC48.
        Description for mibCtl AtmFrameType 2 OC36 :
            ATM Frame Type is OC36.
        Description for mibCtl AtmFrameType 3 OC24 :
            ATM Frame Type is OC24.
        Description for mibCtl AtmFrameType 4 OC18 :
            ATM Frame Type is OC18.
        Description for mibCtl AtmFrameType 5 OC12 :
            ATM Frame Type is OC12.
        Description for mibCtl AtmFrameType 6 OC9 :
            ATM Frame Type is OC9.
        Description for mibCtl AtmFrameType 7 OC3 :
            ATM Frame Type is OC3.
        Description for mibCtl AtmFrameType 8 OC1 :
            ATM Frame Type is OC1.
        Description for mibCtl AtmFrameType 9 STM16 :
            ATM Frame Type is STM16.
        Description for mibCtl AtmFrameType 10 STM4 :
            ATM Frame Type is STM4.
        Description for mibCtl AtmFrameType 11 STM1 :
            ATM Frame Type is STM1.
        Description for mibCtl AtmFrameType 12 DS3 :
            ATM Frame Type is DS3.
        Description for mibCtl AtmFrameType 13 DS2 :
            ATM Frame Type is DS2.
        Description for mibCtl AtmFrameType 14 DS1 :
            ATM Frame Type is DS1.
        Description for mibCtl AtmFrameType 15 DS0 :
            ATM Frame Type is DS0.
        Description for mibCtl AtmFrameType 16 E3 :
            ATM Frame Type is E3.
        Description for mibCtl AtmFrameType 17 E2 :
            ATM Frame Type is E2.
        Description for mibCtl AtmFrameType 18 E1 :
            ATM Frame Type is E1.
        Description for mibCtl AtmFrameType 19 E0 :
            ATM Frame Type is E0.
        "
    ::= { _625k-MCATMConfigTableEntry 4 }



_625k-MCAtmUNIVersion                   OBJECT-TYPE
    SYNTAX            INTEGER -- AtmUNIVersionType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
```

```
        "Atm UNI version.



        (From mibCtl ElementType 1953 AtmUNIVersion)
        Description for mibCtl Type 302 AtmUNIVersionType :
            ATM UNI Version Type.
        Description for mibCtl AtmUNIVersionType 0 Unknown :
            ATM UNI Version Type is Unknown.
        Description for mibCtl AtmUNIVersionType 1 V30 :
            ATM UNI Version Type is 3.0.
        Description for mibCtl AtmUNIVersionType 2 V31 :
            ATM UNI Version Type is 3.1.
        Description for mibCtl AtmUNIVersionType 3 V40 :
            ATM UNI Version Type is 4.0.
        "
    ::= { _625k-MCATMConfigTableEntry 5 }



_625k-MCAtmLineStatus                   OBJECT-TYPE
    SYNTAX            INTEGER -- LineStatusType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Atm line status.

        ATM line status

        (From mibCtl ElementType 1956 AtmLineStatus)
        Description for mibCtl Type 72 LineStatusType :
            Line status type.
        Description for mibCtl LineStatusType 0 LinkUp :
            Line status is link up.
        Description for mibCtl LineStatusType 1 LinkDown :
            Line status is link down.
        "
    ::= { _625k-MCATMConfigTableEntry 6 }



_625k-MCAtmParameterFailReason          OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Atm parameter fail reason.

        ATM parameter fail reason

        (From mibCtl ElementType 1957 AtmParameterFailReason)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCATMConfigTableEntry 7 }



_625k-MCAtmOpenChannelFailReason        OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Atm open channel fail reason.

```

```
        ATM open channel fail reason

        (From mibCtl ElementType 1958 AtmOpenChannelFailReason)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCATMConfigTableEntry 8 }



_625k-MCAtmChannelNumber                OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Atm open channel fail reason.

        ATM open channel fail reason

        (From mibCtl ElementType 1959 AtmChannelNumber)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCATMConfigTableEntry 9 }



_625k-MCAtmAlarmCauseRegister           OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Atm alarm cause register.

        ATM alarm cause register

        (From mibCtl ElementType 1960 AtmAlarmCauseRegister)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCATMConfigTableEntry 10 }



_625k-MCAtmPHYIntrCauseRegister         OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Atm PHY Interrupt cause register.

        ATM PHY interrupt cause register

        (From mibCtl ElementType 1961 AtmPHYIntrCauseRegister)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
```

```
        "
    ::= { _625k-MCATMConfigTableEntry 11 }




_625k-MCA10ConfigTable                  OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCA10ConfigTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "A10 Configuration Table"
    ::= { _625k-MCInterfaceNetwork 9 }



_625k-MCA10ConfigTableEntry             OBJECT-TYPE
    SYNTAX           _625k-MCA10ConfigTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCA10ConfigTableIndex }
    ::= { _625k-MCA10ConfigTable 1 }

_625k-MCA10ConfigTableEntry ::= SEQUENCE {
   _625k-MCA10ConfigTableIndex            INTEGER, -- MoNerdAddressType
   _625k-MC3GPP2PriPDSNIPAddress          OCTET STRING (SIZE(0..15)), --
IPAddressTextType
   _625k-MC3GPP2PriPDSNSharedSecret       OCTET STRING (SIZE(0..64)), --
TextType X 64
   _625k-MC3GPP2PriPDSNSPI                INTEGER, -- Unsigned32Type
   _625k-MC3GPP2SecPDSNIPAddress          OCTET STRING (SIZE(0..15)), --
IPAddressTextType
   _625k-MC3GPP2SecPDSNSharedSecret       OCTET STRING (SIZE(0..64)), --
TextType X 64
   _625k-MC3GPP2SecPDSNSPI                INTEGER -- Unsigned32Type
    }



_625k-MCA10ConfigTableIndex             OBJECT-TYPE
    SYNTAX            INTEGER -- MoNerdAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 204 MoNerdAddressType :
            Base station network component address.

            A network address is a subset of Base Station component addresses,
            restricted to network components only.
            Network components interface with a telephony switch or similar.
            [Limits: 0 1 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCA10ConfigTableEntry 1 }



_625k-MC3GPP2PriPDSNIPAddress           OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..15)) -- IPAddressTextType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "3GPP2 primary PDSN IP address.
```

```
        3GPP2 Primary PDSN IP Address

        (From mibCtl ElementType 2100 3GPP2PriPDSNIPAddress)
        Description for mibCtl Type 420 IPAddressTextType :
            Internet Protocol Address (Text).

            This text must currently be in the dotted abc.def.ghi.jkl format.
            In the future, hostnames might be allowed.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCA10ConfigTableEntry 2 }



_625k-MC3GPP2PriPDSNSharedSecret        OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..64)) -- TextType X 64
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "3GPP2 primary PDSN shared secret.

        3GPP2 primary PDSN shared secret

        (From mibCtl ElementType 2101 3GPP2PriPDSNSharedSecret)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCA10ConfigTableEntry 3 }



_625k-MC3GPP2PriPDSNSPI                 OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "3GPP2 primary PDSN SPI.

        3GPP2 primary PDSN SPI

        (From mibCtl ElementType 2102 3GPP2PriPDSNSPI)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCA10ConfigTableEntry 4 }



_625k-MC3GPP2SecPDSNIPAddress           OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..15)) -- IPAddressTextType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "3GPP2 secondary PDSN IP address.

        3GPP2 secondary PDSN IP address

        (From mibCtl ElementType 2103 3GPP2SecPDSNIPAddress)
```

```
        Description for mibCtl Type 420 IPAddressTextType :
            Internet Protocol Address (Text).

            This text must currently be in the dotted abc.def.ghi.jkl format.
            In the future, hostnames might be allowed.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCA10ConfigTableEntry 5 }



_625k-MC3GPP2SecPDSNSharedSecret        OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..64)) -- TextType X 64
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "3GPP2 secondary PDSN shared secret.

        3GPP2 secondary PDSN shared secret

        (From mibCtl ElementType 2104 3GPP2SecPDSNSharedSecret)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCA10ConfigTableEntry 6 }



_625k-MC3GPP2SecPDSNSPI                 OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "3GPP2 secondary PDSN SPI.

        3GPP2 secondary PDSN SPI

        (From mibCtl ElementType 2105 3GPP2SecPDSNSPI)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCA10ConfigTableEntry 7 }




_625k-MCA10StatusTable                  OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCA10StatusTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "A10 Status Table"
    ::= { _625k-MCInterfaceNetwork 10 }



_625k-MCA10StatusTableEntry             OBJECT-TYPE
    SYNTAX           _625k-MCA10StatusTableEntry
    ACCESS            not-accessible
```

```
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCA10StatusTableIndex }
    ::= { _625k-MCA10StatusTable 1 }

_625k-MCA10StatusTableEntry ::= SEQUENCE {
   _625k-MCA10StatusTableIndex            INTEGER, -- MoNerdAddressType
   _625k-MC3GPP2PDSNIPAddress             OCTET STRING (SIZE(0..15)) --
IPAddressTextType
    }



_625k-MCA10StatusTableIndex             OBJECT-TYPE
    SYNTAX            INTEGER -- MoNerdAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 204 MoNerdAddressType :
            Base station network component address.

            A network address is a subset of Base Station component addresses,
            restricted to network components only.
            Network components interface with a telephony switch or similar.
            [Limits: 0 1 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCA10StatusTableEntry 1 }



_625k-MC3GPP2PDSNIPAddress              OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..15)) -- IPAddressTextType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "3GPP2 PDSN IP Address.

        Current main using PDSN IP address

        (From mibCtl ElementType 2113 3GPP2PDSNIPAddress)
        Description for mibCtl Type 420 IPAddressTextType :
            Internet Protocol Address (Text).

            This text must currently be in the dotted abc.def.ghi.jkl format.
            In the future, hostnames might be allowed.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCA10StatusTableEntry 2 }



_625k-MCInterfaceRF                     OBJECT IDENTIFIER
    -- DESCRIPTION       "Radio Frequency Interfaces"
    ::= { _625k-MCSysInterfaces 2 }



_625k-MCCarrierTable                    OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCCarrierTableEntry
```

```
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "Carrier Frequency Table"
    ::= { _625k-MCInterfaceRF 1 }



_625k-MCCarrierTableEntry              OBJECT-TYPE
    SYNTAX           _625k-MCCarrierTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCCarrierTableIndex }
    ::= { _625k-MCCarrierTable 1 }

_625k-MCCarrierTableEntry ::= SEQUENCE {
   _625k-MCCarrierTableIndex              INTEGER, -- BaseStationCarrierType
   _625k-MCCarrierUsage                   INTEGER -- CarrierUsageType
    }



_625k-MCCarrierTableIndex              OBJECT-TYPE
    SYNTAX            INTEGER -- BaseStationCarrierType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 219 BaseStationCarrierType :
            Base station carrier number.

            Base station carriers are a contiguous set of carriers
            that are used by the Base Station;
            they are numbered from 0 to a current maximum of 32-1.
            [Limits: 0 15 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCCarrierTableEntry 1 }



_625k-MCCarrierUsage                   OBJECT-TYPE
    SYNTAX            INTEGER -- CarrierUsageType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Current assigned usage per base station carrier.



        (From mibCtl ElementType 54 CarrierUsage)
        Description for mibCtl Type 220 CarrierUsageType :
            The assigned use of a radio carrier.

            A radio carrier is a frequency band.
            The assigned use of a carrier can be Reserved, Control
            or Traffic.
            Base station transmits control information on one of the time
slots
            of given Control carrier.
            Base station does not transmit anything on Reserved carriers.
        Description for mibCtl CarrierUsageType 0 NotUse :
            Not Use for this carrier.
        Description for mibCtl CarrierUsageType 1 TCH :
            All timeslots in this carrier are for traffic only.
```

```
        Description for mibCtl CarrierUsageType 2 TCHBCH :
            One timeslot in this carrier is for BCH, others for TCH.
        "
    ::= { _625k-MCCarrierTableEntry 2 }



_625k-MCBSCC                           OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Base Station Color Code.

        (From mibCtl ElementType 61 BSCC)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCInterfaceRF 2 }



_625k-MCBSLowestCarrier                OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "The lowest carrier of the base station operating band.

        This is an extended carrier number
        that identifies the lowest carrier of the bandwidth
        to which the base station is tuned.
        This value cannot be changed while the Base Station state
        is Operating.

        (From mibCtl ElementType 52 BSLowestCarrier)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCInterfaceRF 3 }



_625k-MCBCHModuleAddress               OBJECT-TYPE
    SYNTAX            INTEGER -- ModuleAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Which module is handling the broadcast channel.

        (From mibCtl ElementType 57 BCHModuleAddress)
        Description for mibCtl Type 202 ModuleAddressType :
            Base station bus slot address.

            Most components of the Base Station for which data can
            be obtained are identified by a ModuleAddressType address
            and possibly a subsidiary address.
            [Limits: 0 7 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
```

```
          Derived from basic 32 bit word type.
          "
      ::= { _625k-MCInterfaceRF 4 }



  _625k-MCBCHCarrierNumber                OBJECT-TYPE
      SYNTAX             OCTET STRING (SIZE(0..20)) -- TextType X 20
      ACCESS             read-write
      STATUS             mandatory
      DESCRIPTION
          "Number Of BCH Carrier.

          (From mibCtl ElementType 58 BCHCarrierNumber)
          Description for mibCtl Type 15 TextType :
              ASCII or compatible text.
          Type derived from mibCtl Type 12 OctetType :
              8 bits of raw opaque data.
          Derived from basic 8 bit word type.
          "
      ::= { _625k-MCInterfaceRF 5 }



  _625k-MCRACHCarrierMask                 OBJECT-TYPE
      SYNTAX             INTEGER -- Unsigned32Type
      ACCESS             read-write
      STATUS             mandatory
      DESCRIPTION
          "RACH carrier mask.



          (From mibCtl ElementType 73 RACHCarrierMask)
          Description for mibCtl Type 14 Unsigned32Type :
              32 bit unsigned integer.
          Type derived from mibCtl Type 11 Word32Type :
              32 bits of raw opaque data.
          Derived from basic 32 bit word type.
          "
      ::= { _625k-MCInterfaceRF 6 }



  _625k-MCRACHSlotMask                    OBJECT-TYPE
      SYNTAX             INTEGER -- Unsigned32Type
      ACCESS             read-write
      STATUS             mandatory
      DESCRIPTION
          "RACH slot mask.



          (From mibCtl ElementType 72 RACHSlotMask)
          Description for mibCtl Type 14 Unsigned32Type :
              32 bit unsigned integer.
          Type derived from mibCtl Type 11 Word32Type :
              32 bits of raw opaque data.
          Derived from basic 32 bit word type.
          "
      ::= { _625k-MCInterfaceRF 7 }



  _625k-MCCalibrationInterval             OBJECT-TYPE
      SYNTAX             INTEGER -- Unsigned32Type
      ACCESS             read-write
      STATUS             mandatory
```

```
    DESCRIPTION
        "Calibration interval time.



        (From mibCtl ElementType 75 CalibrationInterval)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCInterfaceRF 8 }



_625k-MCSpatialParameter                OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Spatial parameter.



        (From mibCtl ElementType 78 SpatialParameter)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCInterfaceRF 9 }



_625k-MCCostCalcParameter               OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Cost calculation parameter.



        (From mibCtl ElementType 79 CostCalcParameter)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCInterfaceRF 10 }



_625k-MCBSRegistrationCapacity          OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Capacity of UT registration on BS.



        (From mibCtl ElementType 76 BSRegistrationCapacity)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
```

```
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCInterfaceRF 11 }



_625k-MCBSRegistrationTimer             OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Timer of keeping UT registration on BS.



        (From mibCtl ElementType 77 BSRegistrationTimer)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCInterfaceRF 12 }



_625k-MCPCHFrequencyHopping             OBJECT-TYPE
    SYNTAX            INTEGER -- BooleanType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Propriety of frequency hopping (PCH).



        (From mibCtl ElementType 70 PCHFrequencyHopping)
        Description for mibCtl Type 16 BooleanType :
            Truth value, 0=FALSE, 1=TRUE.

            This is a subset of TriStateType; no UNDEFINED value is provided.
            [Limits: 0 1 ]
        Description for mibCtl BooleanType 0 FALSE :
            False.
        Description for mibCtl BooleanType 1 TRUE :
            True.
        "
    ::= { _625k-MCInterfaceRF 13 }



_625k-MCTCHFrequencyHopping             OBJECT-TYPE
    SYNTAX            INTEGER -- BooleanType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Propriety of frequency hopping (TCH).



        (From mibCtl ElementType 71 TCHFrequencyHopping)
        Description for mibCtl Type 16 BooleanType :
            Truth value, 0=FALSE, 1=TRUE.

            This is a subset of TriStateType; no UNDEFINED value is provided.
            [Limits: 0 1 ]
        Description for mibCtl BooleanType 0 FALSE :
```

```
            False.
        Description for mibCtl BooleanType 1 TRUE :
            True.
        "
    ::= { _625k-MCInterfaceRF 14 }



_625k-MCRFStatusTable                   OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCRFStatusTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "RF Status Table"
    ::= { _625k-MCInterfaceRF 15 }



_625k-MCRFStatusTableEntry              OBJECT-TYPE
    SYNTAX           _625k-MCRFStatusTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCRFStatusTableIndex }
    ::= { _625k-MCRFStatusTable 1 }

_625k-MCRFStatusTableEntry ::= SEQUENCE {
   _625k-MCRFStatusTableIndex             INTEGER, -- MoNerdAddressType
   _625k-MCBSAirBitRateUpLink             INTEGER, -- Unsigned32Type
   _625k-MCBSAirBitRateDownLink           INTEGER, -- Unsigned32Type
   _625k-MCBSActiveStream                 INTEGER, -- Unsigned32Type
   _625k-MCBSActiveRegistration           INTEGER -- Unsigned32Type
    }



_625k-MCRFStatusTableIndex              OBJECT-TYPE
    SYNTAX            INTEGER -- MoNerdAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 204 MoNerdAddressType :
            Base station network component address.

            A network address is a subset of Base Station component addresses,
            restricted to network components only.
            Network components interface with a telephony switch or similar.
            [Limits: 0 1 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCRFStatusTableEntry 1 }



_625k-MCBSAirBitRateUpLink              OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Radio bit rate of up link per Modem control board.



        (From mibCtl ElementType 4022 BSAirBitRateUpLink)
```

```
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCRFStatusTableEntry 2 }



_625k-MCBSAirBitRateDownLink            OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Radio bit rate of down link per Modem control board.



        (From mibCtl ElementType 4023 BSAirBitRateDownLink)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCRFStatusTableEntry 3 }



_625k-MCBSActiveStream                  OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Number of streams currently connected in a base station.

        Number of active streams.

        (From mibCtl ElementType 4020 BSActiveStream)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCRFStatusTableEntry 4 }



_625k-MCBSActiveRegistration            OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Number of registrations currently existed in a base station.

        The call capacity is determined by the available resources in a
        base station.

        (From mibCtl ElementType 4021 BSActiveRegistration)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCRFStatusTableEntry 5 }
```

```
_625k-MCSysScalars                        OBJECT IDENTIFIER
    -- DESCRIPTION       "System Scalars"
    ::= { _625k-MCSystem 4 }



_625k-MCBaseStationID                     OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..18)) -- TextType X 18
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Base Station Identification Code.

        This text string must represent in hexadecimal a 42 bit number
        to be used as the Base Station Identification Code (BSID).
        The BSID is used by the base station to identify itself to
        subscriber units.
        The BSID of a base station must at a minimum
        differ from that of any other base station
        where both would be within radio reception distance
        of any subscriber unit.

        This cannot be changed while the Base Station state is Operating.

        (From mibCtl ElementType 60 BaseStationID)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCSysScalars 1 }



_625k-MCBaseStationTypeID                 OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Type ID of base station.



        (From mibCtl ElementType 66 BaseStationTypeID)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCSysScalars 2 }



_625k-MCBaseStationGroupID                OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Group ID of base station.



        (From mibCtl ElementType 67 BaseStationGroupID)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
```

```
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCSysScalars 3 }



_625k-MCBaseStationSubGroupID           OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Sub group ID of base station.



        (From mibCtl ElementType 68 BaseStationSubGroupID)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCSysScalars 4 }



_625k-MCDesiredStateOfBaseStation       OBJECT-TYPE
    SYNTAX            INTEGER -- ComponentStateType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Desired state of base station as a whole.

        This indicates the Base Station state desired by the operator.
        These desired states are currently supported:

        Operating -  for normal operation.

        Ready -  to avoid taking any new calls.
        Existing calls will not be terminated except normally or by
        command from the operator.
        While existing calls remain, the base station state will
        remain as Operating.

        This information is permanently stored on the base station.

        (From mibCtl ElementType 42 DesiredStateOfBaseStation)
        Description for mibCtl Type 71 ComponentStateType :
            Component operational state.

            A component begins in the Unknown state.
            If not detected, it enters and remains in the NotPresent state.
            If detected, it enters the Uninitialized state, from where it
            may go to the Testing and Initializing states and then to the
            Standby or Operating state depending upon permissions.
            Due to loss of permissions or resources, it may revert from
            the Operating state to the Standby state.
            Due to failure or loss of permission, it may revert to the
            Uninitialized state, perhaps by way of the ShuttingDown state
            depending on the device.
            From the Uninitialized state it may return to more advanced
            states depending upon permissions.
            In case of a waiting period before (again) initializing,
            the component is considered to be Initializing.

            Permissions include administrative permissions (from the
```

```
            operator); excessive failure restrictions; etc.
        Description for mibCtl ComponentStateType 0 Unknown :
            Component state not known.
        Description for mibCtl ComponentStateType 1 NotPresent :
            Component is not present.
        Description for mibCtl ComponentStateType 2 PowerOff :
            Component is present but powered off.
        Description for mibCtl ComponentStateType 3 Uninitialized :
            Component is present but not in use.

            The power on/off state of the component is not specified in
            this case.
        Description for mibCtl ComponentStateType 4 Testing :
            Component is being tested.
        Description for mibCtl ComponentStateType 5 Initializing :
            Component is being initialized.
        Description for mibCtl ComponentStateType 6 Ready :
            Component is ready but not operating.
        Description for mibCtl ComponentStateType 7 Operating :
            Component is operating for normal use without restriction.

            The component is either in actual use or may be used at any time,
            without restriction.
        Description for mibCtl ComponentStateType 8 Abandoned :
            Component state is not the desired state due to excessive errors.

            The component state is not that desired, and the Base Station
            software has abandoned attempts to place the component in
            the desired state.
            The actual state of the component is undefined.
            The Base Station software will resume attempting to place the
            component in the desired state if the appropriate Reinitialize
            action element is written with the correct value.
            Also, the software may resume attempts under other conditions,
            not all of which may be documented.
        Description for mibCtl ComponentStateType 9 InitialSetUp :
            Component is initial set up..

            Initial set up state.
        Description for mibCtl ComponentStateType 10 Degrading :
            Component is degrading..

            Degrading state.
        Description for mibCtl ComponentStateType 11 Restriction :
            Component is restriction..

            Restriction state.
        "
    ::= { _625k-MCSysScalars 5 }



_625k-MCTypeOfReboot                    OBJECT-TYPE
    SYNTAX            INTEGER -- RebootType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Type of reboot for base station.



        (From mibCtl ElementType 63 TypeOfReboot)
        Description for mibCtl Type 250 RebootType :
            Reboot Type.
        Description for mibCtl RebootType 0 Force :
            Force  mode.
        Description for mibCtl RebootType 1 Graceful :
            Graceful  mode.
```

```
        "
    ::= { _625k-MCSysScalars 6 }



_625k-MCBaseStationRebootTime           OBJECT-TYPE
    SYNTAX            Gauge -- AbsoluteTimeType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Time of base station reboot.

        This is the base station reboot time (GPS time).

        (From mibCtl ElementType 62 BaseStationRebootTime)
        Description for mibCtl Type 801 AbsoluteTimeType :
            Absolute time in GPS seconds.

            GPS (Global Positioning System) time in seconds since Jan. 6,
1980.
            Note that this differs from UTC (in addition to a possible
            offset due to starting time) due to leap seconds; see
            the GpsLeapSecond element.
        Type derived from mibCtl Type 18 Gauge32Type :
            32 bits of Gauge data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCSysScalars 7 }



_625k-MCTypeOfBSDiagnosis               OBJECT-TYPE
    SYNTAX            INTEGER -- DiagnosisType
    ACCESS            read-write
    STATUS            mandatory
    DESCRIPTION
        "Type of diagnosis for base station.



        (From mibCtl ElementType 64 TypeOfBSDiagnosis)
        Description for mibCtl Type 251 DiagnosisType :
            Diagnosis Type.
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCSysScalars 8 }



_625k-MCBSDiagnosisStatus               OBJECT-TYPE
    SYNTAX            INTEGER -- DiagnosisStatusType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Diagnosis status for base station.



        (From mibCtl ElementType 370 BSDiagnosisStatus)
        Description for mibCtl Type 252 DiagnosisStatusType :
            Diagnosis status Type.
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
```

```
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCSysScalars 9 }



_625k-MCBSDiagnosisFailReason           OBJECT-TYPE
    SYNTAX            INTEGER -- DiagFailReasonType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Diagnosis fail reason for base station.



        (From mibCtl ElementType 371 BSDiagnosisFailReason)
        Description for mibCtl Type 253 DiagFailReasonType :
            Diagnosis fail reason Type.
        Description for mibCtl DiagFailReasonType 1 PDSNPing :
            Diagnosis fail reason is PDSN Ping.
        Description for mibCtl DiagFailReasonType 2 Calibration :
            Diagnosis fail reason is Calibration.
        Description for mibCtl DiagFailReasonType 3 AntPath :
            Diagnosis fail reason is TRx Antenna Path.
        Description for mibCtl DiagFailReasonType 4 LOAlive :
            Diagnosis fail reason is Local Oscilator DSP Alive.
        Description for mibCtl DiagFailReasonType 5 GCLoopBack :
            Diagnosis fail reason is GCLoopBack.
        Description for mibCtl DiagFailReasonType 6 SlaveNM :
            Diagnosis fail reason is Slave Modem control board.
        Description for mibCtl DiagFailReasonType 7 GPSAnt :
            Diagnosis fail reason is GPS Antenna.
        Description for mibCtl DiagFailReasonType 8 SlotDSP :
            Diagnosis fail reason is Modem control board DSP Alive.
        Description for mibCtl DiagFailReasonType 9 ATMAlive :
            Diagnosis fail reason is ATM Alive.
        Description for mibCtl DiagFailReasonType 96 UndefineName :
            Diagnosis fail reason is Undefine Diag Name.
        Description for mibCtl DiagFailReasonType 97 TimeOut :
            Diagnosis fail reason is Time Out.
        Description for mibCtl DiagFailReasonType 98 InvalidStateExec :
            Diagnosis fail reason is Invalid State Execute.
        Description for mibCtl DiagFailReasonType 99 ExecFail :
            Diagnosis fail reason is Execute Fail.
        "
    ::= { _625k-MCSysScalars 10 }



_625k-MCDiskDbUpdateSequence            OBJECT-TYPE
    SYNTAX            Gauge -- Gauge32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Base station Flach update sequence number.

        This number is incremented on disk every time any other
        database element is actually changed on flash.
        It is not incremented on redundant sets.
        This number may also be set to a desired value.

        (From mibCtl ElementType 6 DiskDbUpdateSequence)
        Description for mibCtl Type 18 Gauge32Type :
            32 bits of Gauge data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCSysScalars 11 }
```

```
_625k-MCStateOfBaseStation             OBJECT-TYPE
    SYNTAX            INTEGER -- ComponentStateType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "State of base station as a whole.

        This will not have values of Unknown or PowerOff since the
        base station would be unable to report such values.

        When sufficiently initialized, the state will be Operating if
        accepting new calls (according to the desired state of the
        base station) or continuing ongoing calls;
        or Ready if the desired state is Ready and their are no
        ongoing calls.

        (From mibCtl ElementType 41 StateOfBaseStation)
        Description for mibCtl Type 71 ComponentStateType :
            Component operational state.

            A component begins in the Unknown state.
            If not detected, it enters and remains in the NotPresent state.
            If detected, it enters the Uninitialized state, from where it
            may go to the Testing and Initializing states and then to the
            Standby or Operating state depending upon permissions.
            Due to loss of permissions or resources, it may revert from
            the Operating state to the Standby state.
            Due to failure or loss of permission, it may revert to the
            Uninitialized state, perhaps by way of the ShuttingDown state
            depending on the device.
            From the Uninitialized state it may return to more advanced
            states depending upon permissions.
            In case of a waiting period before (again) initializing,
            the component is considered to be Initializing.

            Permissions include administrative permissions (from the
            operator); excessive failure restrictions; etc.
        Description for mibCtl ComponentStateType 0 Unknown :
            Component state not known.
        Description for mibCtl ComponentStateType 1 NotPresent :
            Component is not present.
        Description for mibCtl ComponentStateType 2 PowerOff :
            Component is present but powered off.
        Description for mibCtl ComponentStateType 3 Uninitialized :
            Component is present but not in use.

            The power on/off state of the component is not specified in
            this case.
        Description for mibCtl ComponentStateType 4 Testing :
            Component is being tested.
        Description for mibCtl ComponentStateType 5 Initializing :
            Component is being initialized.
        Description for mibCtl ComponentStateType 6 Ready :
            Component is ready but not operating.
        Description for mibCtl ComponentStateType 7 Operating :
            Component is operating for normal use without restriction.

            The component is either in actual use or may be used at any time,
            without restriction.
        Description for mibCtl ComponentStateType 8 Abandoned :
            Component state is not the desired state due to excessive errors.

            The component state is not that desired, and the Base Station
            software has abandoned attempts to place the component in
            the desired state.
```

```
            The actual state of the component is undefined.
            The Base Station software will resume attempting to place the
            component in the desired state if the appropriate Reinitialize
            action element is written with the correct value.
            Also, the software may resume attempts under other conditions,
            not all of which may be documented.
        Description for mibCtl ComponentStateType 9 InitialSetUp :
            Component is initial set up..

            Initial set up state.
        Description for mibCtl ComponentStateType 10 Degrading :
            Component is degrading..

            Degrading state.
        Description for mibCtl ComponentStateType 11 Restriction :
            Component is restriction..

            Restriction state.
        "
    ::= { _625k-MCSysScalars 12 }



_625k-MCBSTotalIndication               OBJECT-TYPE
    SYNTAX            INTEGER -- IndicationType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Status of BS total indicator.

        In the current implementation,
        this indication is set to the value On by BS.

        (From mibCtl ElementType 525 BSTotalIndication)
        Description for mibCtl Type 230 IndicationType :
            Hardware indication status (LEDs).


        Description for mibCtl IndicationType 0 Off :
            Off.
        Description for mibCtl IndicationType 1 Amber :
            Amber.
        Description for mibCtl IndicationType 2 Red :
            Red.
        Description for mibCtl IndicationType 3 Green :
            Green.
        Description for mibCtl IndicationType 4 NotPresent :
            Not present.
        "
    ::= { _625k-MCSysScalars 13 }



_625k-MCMasterAddress                   OBJECT-TYPE
    SYNTAX            INTEGER -- ModuleAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Bus slot address of master Modem control board.

        This indicates which Modem control board is master
        of the base station.

        (From mibCtl ElementType 32 MasterAddress)
        Description for mibCtl Type 202 ModuleAddressType :
            Base station bus slot address.

            Most components of the Base Station for which data can
```

```
            be obtained are identified by a ModuleAddressType address
            and possibly a subsidiary address.
            [Limits: 0 7 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCSysScalars 14 }



_625k-MCBSManufactureID                 OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Base Station manufacture identification number.

        The manufacture identification assigned by Vendor

        (From mibCtl ElementType 201 BSManufactureID)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCSysScalars 15 }



_625k-MCBSSerialNumber                  OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Base Station serial number.

        This is the character serial number of the base station.
        This serial number will be unique among all base stations
        of this type regardless of manufacturer.

        (From mibCtl ElementType 203 BSSerialNumber)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCSysScalars 16 }



_625k-MCDiagnosisBaseStation            OBJECT-TYPE
    SYNTAX            INTEGER -- BooleanType
    ACCESS            read-write -- REALLY: write-only
    STATUS            mandatory
    DESCRIPTION
        "Diagnosis base station.

        This is a write-only element; only a value of TRUE is valid.

        (From mibCtl ElementType 47 DiagnosisBaseStation)
        Description for mibCtl Type 16 BooleanType :
            Truth value, 0=FALSE, 1=TRUE.
```

```
            This is a subset of TriStateType; no UNDEFINED value is provided.
            [Limits: 0 1 ]
        Description for mibCtl BooleanType 0 FALSE :
            False.
        Description for mibCtl BooleanType 1 TRUE :
            True.
        "
    ::= { _625k-MCSysScalars 17 }



_625k-MCRebootBaseStation               OBJECT-TYPE
    SYNTAX            INTEGER -- BooleanType
    ACCESS            read-write -- REALLY: write-only
    STATUS            mandatory
    DESCRIPTION
        "Reboot base station.

        This is a write-only element; only a value of TRUE is valid.
        All existing calls will be terminated abruptly.
        All components of the base station will be reinitialized
        according to the permanent contents of the Base Station database.
        The base station may be incommunicado for a period of time.

        The reinitialization may be delayed by a few seconds to allow
        for a clean shutdown.

        (From mibCtl ElementType 44 RebootBaseStation)
        Description for mibCtl Type 16 BooleanType :
            Truth value, 0=FALSE, 1=TRUE.

            This is a subset of TriStateType; no UNDEFINED value is provided.
            [Limits: 0 1 ]
        Description for mibCtl BooleanType 0 FALSE :
            False.
        Description for mibCtl BooleanType 1 TRUE :
            True.
        "
    ::= { _625k-MCSysScalars 18 }



_625k-MCBSModelNumber                   OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Base Station model number.

        Base Station model number

        (From mibCtl ElementType 204 BSModelNumber)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCSysScalars 19 }



_625k-MCBSManufactureDate               OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Base Station manufacture date.
```

```
        Base Station manufacture date

        (From mibCtl ElementType 205 BSManufactureDate)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCSysScalars 20 }



_625k-MCBSHardwareRevision              OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Base Station hardware revision.

        Base Station hardware revision

        (From mibCtl ElementType 206 BSHardwareRevision)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCSysScalars 21 }



_625k-MCMiscComponents                  OBJECT IDENTIFIER
    -- DESCRIPTION       "Miscellaneous Component"
    ::= { IEEE802dot20-625k-MC-MIB 2 }



_625k-MCAntenna                         OBJECT IDENTIFIER
    -- DESCRIPTION       "Antenna"
    ::= { _625k-MCMiscComponents 1 }



_625k-MCAntennaTable                    OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCAntennaTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "Antenna Table"
    ::= { _625k-MCAntenna 1 }



_625k-MCAntennaTableEntry               OBJECT-TYPE
    SYNTAX           _625k-MCAntennaTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCAntennaTableIndex }
    ::= { _625k-MCAntennaTable 1 }

_625k-MCAntennaTableEntry ::= SEQUENCE {
   _625k-MCAntennaTableIndex              INTEGER, -- AntennaAddressType
   _625k-MCStateOfAntenna                 INTEGER -- ComponentStateType
    }
```

```
_625k-MCAntennaTableIndex                OBJECT-TYPE
    SYNTAX            INTEGER -- AntennaAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 210 AntennaAddressType :
            Component antenna address.
            [Limits: 0 11 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCAntennaTableEntry 1 }



_625k-MCStateOfAntenna                  OBJECT-TYPE
    SYNTAX            INTEGER -- ComponentStateType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "State of Antenna as a whole.


        (From mibCtl ElementType 211 StateOfAntenna)
        Description for mibCtl Type 71 ComponentStateType :
            Component operational state.

            A component begins in the Unknown state.
            If not detected, it enters and remains in the NotPresent state.
            If detected, it enters the Uninitialized state, from where it
            may go to the Testing and Initializing states and then to the
            Standby or Operating state depending upon permissions.
            Due to loss of permissions or resources, it may revert from
            the Operating state to the Standby state.
            Due to failure or loss of permission, it may revert to the
            Uninitialized state, perhaps by way of the ShuttingDown state
            depending on the device.
            From the Uninitialized state it may return to more advanced
            states depending upon permissions.
            In case of a waiting period before (again) initializing,
            the component is considered to be Initializing.

            Permissions include administrative permissions (from the
            operator); excessive failure restrictions; etc.
        Description for mibCtl ComponentStateType 0 Unknown :
            Component state not known.
        Description for mibCtl ComponentStateType 1 NotPresent :
            Component is not present.
        Description for mibCtl ComponentStateType 2 PowerOff :
            Component is present but powered off.
        Description for mibCtl ComponentStateType 3 Uninitialized :
            Component is present but not in use.

            The power on/off state of the component is not specified in
            this case.
        Description for mibCtl ComponentStateType 4 Testing :
            Component is being tested.
        Description for mibCtl ComponentStateType 5 Initializing :
            Component is being initialized.
        Description for mibCtl ComponentStateType 6 Ready :
            Component is ready but not operating.
        Description for mibCtl ComponentStateType 7 Operating :
            Component is operating for normal use without restriction.
```

```
            The component is either in actual use or may be used at any time,
            without restriction.
        Description for mibCtl ComponentStateType 8 Abandoned :
            Component state is not the desired state due to excessive errors.

            The component state is not that desired, and the Base Station
            software has abandoned attempts to place the component in
            the desired state.
            The actual state of the component is undefined.
            The Base Station software will resume attempting to place the
            component in the desired state if the appropriate Reinitialize
            action element is written with the correct value.
            Also, the software may resume attempts under other conditions,
            not all of which may be documented.
        Description for mibCtl ComponentStateType 9 InitialSetUp :
            Component is initial set up..

            Initial set up state.
        Description for mibCtl ComponentStateType 10 Degrading :
            Component is degrading..

            Degrading state.
        Description for mibCtl ComponentStateType 11 Restriction :
            Component is restriction..

            Restriction state.
        "
    ::= { _625k-MCAntennaTableEntry 2 }



_625k-MCBSTemperatures                  OBJECT IDENTIFIER
    -- DESCRIPTION       "BS Temperature"
    ::= { _625k-MCMiscComponents 3 }



_625k-MCBSTemperature                   OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..4)) -- DegreesCelsiusType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "The temperature of Base station (degrees Celsius).

        The latest recorded temperature of a given BS.

        (From mibCtl ElementType 536 BSTemperature)
        Description for mibCtl Type 807 DegreesCelsiusType :
            Temperature in degrees Celsius.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCBSTemperatures 1 }



_625k-MCCableInfo                       OBJECT IDENTIFIER
    -- DESCRIPTION       "Cable Info"
    ::= { _625k-MCMiscComponents 4 }



_625k-MCCableLossValueForLoCal          OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
```

```
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Value of cable loss for Local Oscilator.

        Value of cable loss.

        (From mibCtl ElementType 801 CableLossValueForLoCal)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCCableInfo 1 }



_625k-MCAntCableTable                  OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCAntCableTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "Antenna Cable"
    ::= { _625k-MCCableInfo 2 }



_625k-MCAntCableTableEntry             OBJECT-TYPE
    SYNTAX           _625k-MCAntCableTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCAntCableTableIndex }
    ::= { _625k-MCAntCableTable 1 }

_625k-MCAntCableTableEntry ::= SEQUENCE {
   _625k-MCAntCableTableIndex             INTEGER, -- AntennaAddressType
   _625k-MCCableLossValueForAntenna       OCTET STRING (SIZE(0..20)) --
TextType X 20
    }



_625k-MCAntCableTableIndex             OBJECT-TYPE
    SYNTAX            INTEGER -- AntennaAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 210 AntennaAddressType :
            Component antenna address.
            [Limits: 0 11 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCAntCableTableEntry 1 }



_625k-MCCableLossValueForAntenna       OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Value of cable loss for Antenna.
```

```
        Value of cable loss.

        (From mibCtl ElementType 802 CableLossValueForAntenna)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCAntCableTableEntry 2 }




_625k-MCGPSCableTable                  OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCGPSCableTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "GPS Cable"
    ::= { _625k-MCCableInfo 10 }



_625k-MCGPSCableTableEntry             OBJECT-TYPE
    SYNTAX           _625k-MCGPSCableTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCGPSCableTableIndex }
    ::= { _625k-MCGPSCableTable 1 }

_625k-MCGPSCableTableEntry ::= SEQUENCE {
   _625k-MCGPSCableTableIndex             INTEGER, -- GpsAddressType
   _625k-MCCableLengthForGps              OCTET STRING (SIZE(0..20)) --
TextType X 20
    }



_625k-MCGPSCableTableIndex             OBJECT-TYPE
    SYNTAX            INTEGER -- GpsAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 209 GpsAddressType :
            Base station GPS component address.


            [Limits: 0 1 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCGPSCableTableEntry 1 }



_625k-MCCableLengthForGps              OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Cable length for Gps.

        This cable is used for calibration.
```

```
        (From mibCtl ElementType 803 CableLengthForGps)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCGPSCableTableEntry 2 }



_625k-MCGPS                              OBJECT IDENTIFIER
    -- DESCRIPTION       "GPS"
    ::= { _625k-MCMiscComponents 6 }



_625k-MCGPSTable                         OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCGPSTableEntry
    ACCESS             not-accessible
    STATUS             mandatory
    DESCRIPTION        "GPS Table"
    ::= { _625k-MCGPS 1 }



_625k-MCGPSTableEntry                    OBJECT-TYPE
    SYNTAX            _625k-MCGPSTableEntry
    ACCESS             not-accessible
    STATUS             mandatory
    DESCRIPTION        ""
    INDEX  { _625k-MCGPSTableIndex }
    ::= { _625k-MCGPSTable 1 }

_625k-MCGPSTableEntry ::= SEQUENCE {
   _625k-MCGPSTableIndex                   INTEGER, -- GpsAddressType
   _625k-MCStateOfGps                      INTEGER, -- ComponentStateType
   _625k-MCGpsNumberOfSatelliteSeen        INTEGER, -- Unsigned32Type
   _625k-MCGpsIndication                   INTEGER, -- IndicationType
   _625k-MCGpsSerialNumber                 OCTET STRING (SIZE(0..20)) --
TextType X 20
    }



_625k-MCGPSTableIndex                    OBJECT-TYPE
    SYNTAX             INTEGER -- GpsAddressType
    ACCESS             read-only
    STATUS             mandatory
    DESCRIPTION        "
        Description for mibCtl Type 209 GpsAddressType :
            Base station GPS component address.


            [Limits: 0 1 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCGPSTableEntry 1 }



_625k-MCStateOfGps                       OBJECT-TYPE
    SYNTAX             INTEGER -- ComponentStateType
```

```
ACCESS            read-only
STATUS            mandatory
DESCRIPTION
    "GPS state.

    The state of the GPS (Global Positioning System)
    on the active local oscillator unit

    (From mibCtl ElementType 420 StateOfGps)
    Description for mibCtl Type 71 ComponentStateType :
        Component operational state.

        A component begins in the Unknown state.
        If not detected, it enters and remains in the NotPresent state.
        If detected, it enters the Uninitialized state, from where it
        may go to the Testing and Initializing states and then to the
        Standby or Operating state depending upon permissions.
        Due to loss of permissions or resources, it may revert from
        the Operating state to the Standby state.
        Due to failure or loss of permission, it may revert to the
        Uninitialized state, perhaps by way of the ShuttingDown state
        depending on the device.
        From the Uninitialized state it may return to more advanced
        states depending upon permissions.
        In case of a waiting period before (again) initializing,
        the component is considered to be Initializing.

        Permissions include administrative permissions (from the
        operator); excessive failure restrictions; etc.
    Description for mibCtl ComponentStateType 0 Unknown :
        Component state not known.
    Description for mibCtl ComponentStateType 1 NotPresent :
        Component is not present.
    Description for mibCtl ComponentStateType 2 PowerOff :
        Component is present but powered off.
    Description for mibCtl ComponentStateType 3 Uninitialized :
        Component is present but not in use.

        The power on/off state of the component is not specified in
        this case.
    Description for mibCtl ComponentStateType 4 Testing :
        Component is being tested.
    Description for mibCtl ComponentStateType 5 Initializing :
        Component is being initialized.
    Description for mibCtl ComponentStateType 6 Ready :
        Component is ready but not operating.
    Description for mibCtl ComponentStateType 7 Operating :
        Component is operating for normal use without restriction.

        The component is either in actual use or may be used at any time,
        without restriction.
    Description for mibCtl ComponentStateType 8 Abandoned :
        Component state is not the desired state due to excessive errors.

        The component state is not that desired, and the Base Station
        software has abandoned attempts to place the component in
        the desired state.
        The actual state of the component is undefined.
        The Base Station software will resume attempting to place the
        component in the desired state if the appropriate Reinitialize
        action element is written with the correct value.
        Also, the software may resume attempts under other conditions,
        not all of which may be documented.
    Description for mibCtl ComponentStateType 9 InitialSetUp :
        Component is initial set up..

        Initial set up state.
    Description for mibCtl ComponentStateType 10 Degrading :
```

```
            Component is degrading..

            Degrading state.
        Description for mibCtl ComponentStateType 11 Restriction :
            Component is restriction..

            Restriction state.
        "
    ::= { _625k-MCGPSTableEntry 2 }



_625k-MCGpsNumberOfSatelliteSeen        OBJECT-TYPE
    SYNTAX            INTEGER -- Unsigned32Type
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Number of satellites seen by GPS.

        The number of satellites seen by the GPS (Global Positioning System)
        on the active local oscillator unit

        (From mibCtl ElementType 421 GpsNumberOfSatelliteSeen)
        Description for mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
        "
    ::= { _625k-MCGPSTableEntry 3 }



_625k-MCGpsIndication                   OBJECT-TYPE
    SYNTAX            INTEGER -- IndicationType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Status of GPS indicator.



        (From mibCtl ElementType 530 GpsIndication)
        Description for mibCtl Type 230 IndicationType :
            Hardware indication status (LEDs).


        Description for mibCtl IndicationType 0 Off :
            Off.
        Description for mibCtl IndicationType 1 Amber :
            Amber.
        Description for mibCtl IndicationType 2 Red :
            Red.
        Description for mibCtl IndicationType 3 Green :
            Green.
        Description for mibCtl IndicationType 4 NotPresent :
            Not present.
        "
    ::= { _625k-MCGPSTableEntry 4 }



_625k-MCGpsSerialNumber                 OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "GPS serial number text.
```

```
        Factory set uniquely for each component.

        (From mibCtl ElementType 570 GpsSerialNumber)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCGPSTableEntry 5 }



_625k-MCPowerAmplifier                  OBJECT IDENTIFIER
    -- DESCRIPTION       "Power Amplifier"
    ::= { _625k-MCMiscComponents 8 }




_625k-MCPAUnitTable                     OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCPAUnitTableEntry
    ACCESS             not-accessible
    STATUS             mandatory
    DESCRIPTION        "PA Table"
    ::= { _625k-MCPowerAmplifier 1 }



_625k-MCPAUnitTableEntry                OBJECT-TYPE
    SYNTAX            _625k-MCPAUnitTableEntry
    ACCESS             not-accessible
    STATUS             mandatory
    DESCRIPTION        ""
    INDEX  { _625k-MCPAUnitTableIndex }
    ::= { _625k-MCPAUnitTable 1 }

_625k-MCPAUnitTableEntry ::= SEQUENCE {
   _625k-MCPAUnitTableIndex               INTEGER, -- PAUnitAddressType
   _625k-MCRebootPAUnit                   INTEGER, -- BooleanType
   _625k-MCStateOfPAUnit                  INTEGER, -- ComponentStateType
   _625k-MCPAUnitIndication               INTEGER, -- IndicationType
   _625k-MCPAUnitSerialNumber             OCTET STRING (SIZE(0..20)), --
TextType X 20
   _625k-MCPAUnitModelNumber              OCTET STRING (SIZE(0..20)), --
TextType X 20
   _625k-MCPAUnitManufactureDate          OCTET STRING (SIZE(0..20)), --
TextType X 20
   _625k-MCPAUnitHardwareRevision         OCTET STRING (SIZE(0..20)), --
TextType X 20
   _625k-MCPAUnitManufactureID            OCTET STRING (SIZE(0..20)), --
TextType X 20
   _625k-MCPAUnitTemperature              OCTET STRING (SIZE(0..4)) --
DegreesCelsiusType
    }



_625k-MCPAUnitTableIndex                OBJECT-TYPE
    SYNTAX             INTEGER -- PAUnitAddressType
    ACCESS             read-only
    STATUS             mandatory
    DESCRIPTION        "
        Description for mibCtl Type 207 PAUnitAddressType :
            Base station power amplifier component unit address.

            A power amplifier unit address is a subset of Base Station
```

```
            component addresses,
            restricted to power amplifier components only.
            Power amplifiers boost radio frequency
            signal levels.
            [Limits: 0 3 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCPAUnitTableEntry 1 }



_625k-MCRebootPAUnit                    OBJECT-TYPE
    SYNTAX            INTEGER -- BooleanType
    ACCESS            read-write -- REALLY: write-only
    STATUS            mandatory
    DESCRIPTION
        "Action to reboot a PA unit.

        This is a write-only element; only a value of TRUE is valid.

        (From mibCtl ElementType 506 RebootPAUnit)
        Description for mibCtl Type 16 BooleanType :
            Truth value, 0=FALSE, 1=TRUE.

            This is a subset of TriStateType; no UNDEFINED value is provided.
            [Limits: 0 1 ]
        Description for mibCtl BooleanType 0 FALSE :
            False.
        Description for mibCtl BooleanType 1 TRUE :
            True.
        "
    ::= { _625k-MCPAUnitTableEntry 2 }



_625k-MCStateOfPAUnit                   OBJECT-TYPE
    SYNTAX            INTEGER -- ComponentStateType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "State of PA as a whole.


        (From mibCtl ElementType 212 StateOfPAUnit)
        Description for mibCtl Type 71 ComponentStateType :
            Component operational state.

            A component begins in the Unknown state.
            If not detected, it enters and remains in the NotPresent state.
            If detected, it enters the Uninitialized state, from where it
            may go to the Testing and Initializing states and then to the
            Standby or Operating state depending upon permissions.
            Due to loss of permissions or resources, it may revert from
            the Operating state to the Standby state.
            Due to failure or loss of permission, it may revert to the
            Uninitialized state, perhaps by way of the ShuttingDown state
            depending on the device.
            From the Uninitialized state it may return to more advanced
            states depending upon permissions.
            In case of a waiting period before (again) initializing,
            the component is considered to be Initializing.

            Permissions include administrative permissions (from the
            operator); excessive failure restrictions; etc.
```

```
        Description for mibCtl ComponentStateType 0 Unknown :
            Component state not known.
        Description for mibCtl ComponentStateType 1 NotPresent :
            Component is not present.
        Description for mibCtl ComponentStateType 2 PowerOff :
            Component is present but powered off.
        Description for mibCtl ComponentStateType 3 Uninitialized :
            Component is present but not in use.

            The power on/off state of the component is not specified in
            this case.
        Description for mibCtl ComponentStateType 4 Testing :
            Component is being tested.
        Description for mibCtl ComponentStateType 5 Initializing :
            Component is being initialized.
        Description for mibCtl ComponentStateType 6 Ready :
            Component is ready but not operating.
        Description for mibCtl ComponentStateType 7 Operating :
            Component is operating for normal use without restriction.

            The component is either in actual use or may be used at any time,
            without restriction.
        Description for mibCtl ComponentStateType 8 Abandoned :
            Component state is not the desired state due to excessive errors.

            The component state is not that desired, and the Base Station
            software has abandoned attempts to place the component in
            the desired state.
            The actual state of the component is undefined.
            The Base Station software will resume attempting to place the
            component in the desired state if the appropriate Reinitialize
            action element is written with the correct value.
            Also, the software may resume attempts under other conditions,
            not all of which may be documented.
        Description for mibCtl ComponentStateType 9 InitialSetUp :
            Component is initial set up..

            Initial set up state.
        Description for mibCtl ComponentStateType 10 Degrading :
            Component is degrading..

            Degrading state.
        Description for mibCtl ComponentStateType 11 Restriction :
            Component is restriction..

            Restriction state.
        "
    ::= { _625k-MCPAUnitTableEntry 3 }



_625k-MCPAUnitIndication                OBJECT-TYPE
    SYNTAX            INTEGER -- IndicationType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "Status of PA Unit indicator.



        (From mibCtl ElementType 526 PAUnitIndication)
        Description for mibCtl Type 230 IndicationType :
            Hardware indication status (LEDs).


        Description for mibCtl IndicationType 0 Off :
            Off.
        Description for mibCtl IndicationType 1 Amber :
```

```
            Amber.
        Description for mibCtl IndicationType 2 Red :
            Red.
        Description for mibCtl IndicationType 3 Green :
            Green.
        Description for mibCtl IndicationType 4 NotPresent :
            Not present.
        "
    ::= { _625k-MCPAUnitTableEntry 4 }



_625k-MCPAUnitSerialNumber              OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "PA unit serial number text.

        Factory set uniquely for each component.

        (From mibCtl ElementType 560 PAUnitSerialNumber)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCPAUnitTableEntry 5 }



_625k-MCPAUnitModelNumber               OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "PA unit model number.

        Factory set with description of component type, including
        the major revision level.

        (From mibCtl ElementType 561 PAUnitModelNumber)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCPAUnitTableEntry 6 }



_625k-MCPAUnitManufactureDate           OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "PA unit manufacture date.

        Factory set to month and date of manufacture date of the module.

        (From mibCtl ElementType 562 PAUnitManufactureDate)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
```

```
        "
    ::= { _625k-MCPAUnitTableEntry 7 }



_625k-MCPAUnitHardwareRevision          OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "PA unit hardware revision name.

        Set at the factory to indicate the minor hardware revision
        level of the module.

        (From mibCtl ElementType 563 PAUnitHardwareRevision)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCPAUnitTableEntry 8 }



_625k-MCPAUnitManufactureID             OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..20)) -- TextType X 20
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "PA unit manufacture ID.



        (From mibCtl ElementType 564 PAUnitManufactureID)
        Description for mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCPAUnitTableEntry 9 }



_625k-MCPAUnitTemperature               OBJECT-TYPE
    SYNTAX            OCTET STRING (SIZE(0..4)) -- DegreesCelsiusType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION
        "The temperature of PAUnit (degrees Celsius).

        The latest recorded temperature of a given PAUnit.

        (From mibCtl ElementType 539 PAUnitTemperature)
        Description for mibCtl Type 807 DegreesCelsiusType :
            Temperature in degrees Celsius.
        Type derived from mibCtl Type 15 TextType :
            ASCII or compatible text.
        Type derived from mibCtl Type 12 OctetType :
            8 bits of raw opaque data.
        Derived from basic 8 bit word type.
        "
    ::= { _625k-MCPAUnitTableEntry 10 }
```

```
_625k-MCPAModuleTable                    OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCPAModuleTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       "PA Module Table"
    ::= { _625k-MCPowerAmplifier 2 }



_625k-MCPAModuleTableEntry               OBJECT-TYPE
    SYNTAX            _625k-MCPAModuleTableEntry
    ACCESS            not-accessible
    STATUS            mandatory
    DESCRIPTION       ""
    INDEX  { _625k-MCPAModuleTableIndex1,_625k-MCPAModuleTableIndex2 }
    ::= { _625k-MCPAModuleTable 1 }

_625k-MCPAModuleTableEntry ::= SEQUENCE {
   _625k-MCPAModuleTableIndex1            INTEGER, -- PAUnitAddressType
   _625k-MCPAModuleTableIndex2            INTEGER -- PAModuleAddressType
    }



_625k-MCPAModuleTableIndex1              OBJECT-TYPE
    SYNTAX            INTEGER -- PAUnitAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 207 PAUnitAddressType :
            Base station power amplifier component unit address.

            A power amplifier unit address is a subset of Base Station
            component addresses,
            restricted to power amplifier components only.
            Power amplifiers boost radio frequency
            signal levels.
            [Limits: 0 3 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCPAModuleTableEntry 1 }



_625k-MCPAModuleTableIndex2              OBJECT-TYPE
    SYNTAX            INTEGER -- PAModuleAddressType
    ACCESS            read-only
    STATUS            mandatory
    DESCRIPTION       "
        Description for mibCtl Type 208 PAModuleAddressType :
            Base station power amplifier component module address.

            A power amplifier module address is a subset of Base Station
            component addresses,
            [Limits: 0 2 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCPAModuleTableEntry 2 }
```

```
_625k-MCPowerSupply                    OBJECT IDENTIFIER
    -- DESCRIPTION       "Power supply"
    ::= { _625k-MCMiscComponents 9 }



_625k-MCPowerSupplyTable               OBJECT-TYPE
    SYNTAX SEQUENCE OF _625k-MCPowerSupplyTableEntry
    ACCESS           not-accessible
    STATUS           mandatory
    DESCRIPTION      "Power supply Table"
    ::= { _625k-MCPowerSupply 1 }


_625k-MCPowerSupplyTableEntry          OBJECT-TYPE
    SYNTAX          _625k-MCPowerSupplyTableEntry
    ACCESS           not-accessible
    STATUS           mandatory
    DESCRIPTION      ""
    INDEX  { _625k-MCPowerSupplyTableIndex }
    ::= { _625k-MCPowerSupplyTable 1 }

_625k-MCPowerSupplyTableEntry ::= SEQUENCE {
   _625k-MCPowerSupplyTableIndex          INTEGER, -- PowerAddressType
   _625k-MCStateOfPowerSupply             INTEGER, -- ComponentStateType
   _625k-MCPowerSupplyIndication          INTEGER -- IndicationType
    }


_625k-MCPowerSupplyTableIndex          OBJECT-TYPE
    SYNTAX           INTEGER -- PowerAddressType
    ACCESS           read-only
    STATUS           mandatory
    DESCRIPTION      "
        Description for mibCtl Type 211 PowerAddressType :
            Component power supply address.
            [Limits: 0 2 ]
        Type derived from mibCtl Type 14 Unsigned32Type :
            32 bit unsigned integer.
        Type derived from mibCtl Type 11 Word32Type :
            32 bits of raw opaque data.
        Derived from basic 32 bit word type.
    "
    ::= { _625k-MCPowerSupplyTableEntry 1 }


_625k-MCStateOfPowerSupply             OBJECT-TYPE
    SYNTAX           INTEGER -- ComponentStateType
    ACCESS           read-only
    STATUS           mandatory
    DESCRIPTION
        "State of Power supply as a whole.


        (From mibCtl ElementType 213 StateOfPowerSupply)
        Description for mibCtl Type 71 ComponentStateType :
            Component operational state.

            A component begins in the Unknown state.
            If not detected, it enters and remains in the NotPresent state.
            If detected, it enters the Uninitialized state, from where it
            may go to the Testing and Initializing states and then to the
```

```
                Standby or Operating state depending upon permissions.
                Due to loss of permissions or resources, it may revert from
                the Operating state to the Standby state.
                Due to failure or loss of permission, it may revert to the
                Uninitialized state, perhaps by way of the ShuttingDown state
                depending on the device.
                From the Uninitialized state it may return to more advanced
                states depending upon permissions.
                In case of a waiting period before (again) initializing,
                the component is considered to be Initializing.

                Permissions include administrative permissions (from the
                operator); excessive failure restrictions; etc.
            Description for mibCtl ComponentStateType 0 Unknown :
                Component state not known.
            Description for mibCtl ComponentStateType 1 NotPresent :
                Component is not present.
            Description for mibCtl ComponentStateType 2 PowerOff :
                Component is present but powered off.
            Description for mibCtl ComponentStateType 3 Uninitialized :
                Component is present but not in use.

                The power on/off state of the component is not specified in
                this case.
            Description for mibCtl ComponentStateType 4 Testing :
                Component is being tested.
            Description for mibCtl ComponentStateType 5 Initializing :
                Component is being initialized.
            Description for mibCtl ComponentStateType 6 Ready :
                Component is ready but not operating.
            Description for mibCtl ComponentStateType 7 Operating :
                Component is operating for normal use without restriction.

                The component is either in actual use or may be used at any time,
                without restriction.
            Description for mibCtl ComponentStateType 8 Abandoned :
                Component state is not the desired state due to excessive errors.

                The component state is not that desired, and the Base Station
                software has abandoned attempts to place the component in
                the desired state.
                The actual state of the component is undefined.
                The Base Station software will resume attempting to place the
                component in the desired state if the appropriate Reinitialize
                action element is written with the correct value.
                Also, the software may resume attempts under other conditions,
                not all of which may be documented.
            Description for mibCtl ComponentStateType 9 InitialSetUp :
                Component is initial set up..

                Initial set up state.
            Description for mibCtl ComponentStateType 10 Degrading :
                Component is degrading..

                Degrading state.
            Description for mibCtl ComponentStateType 11 Restriction :
                Component is restriction..

                Restriction state.
            "
        ::= { _625k-MCPowerSupplyTableEntry 2 }



    _625k-MCPowerSupplyIndication           OBJECT-TYPE
        SYNTAX            INTEGER -- IndicationType
        ACCESS            read-only
        STATUS            mandatory
```

```
        DESCRIPTION
            "Status of Power Supply indicator.



            (From mibCtl ElementType 527 PowerSupplyIndication)
            Description for mibCtl Type 230 IndicationType :
                Hardware indication status (LEDs).


            Description for mibCtl IndicationType 0 Off :
                Off.
            Description for mibCtl IndicationType 1 Amber :
                Amber.
            Description for mibCtl IndicationType 2 Red :
                Red.
            Description for mibCtl IndicationType 3 Green :
                Green.
            Description for mibCtl IndicationType 4 NotPresent :
                Not present.
            "
        ::= { _625k-MCPowerSupplyTableEntry 3 }



    END
```

625K-MC Appendix – A (Informative)

As specified in the Appendix Section-A of baseline specification HC-SDMA [25].