Project	**IEEE 802.16 Broadband Wireless Access Working Group <http://ieee802.org/16>**
Title	**ARQ Proposal for TG3/TG4 MAC**
Date Submitted	**2000-09-13**

Source(s)		
	Itzik Kitroser Runcom Technologies LTD. 2 Ahoma St. 75655, Rishon Lezion Israel	itzikk@runcom.co.il
	Yigal Leiba Runcom Technologies LTD. 2 Ahoma St. 75655, Rishon Lezion Israel	Voice: 972-3-9528440 Fax: 972-3-9528805 yigall@runcom.co.il
	Subbu Ponnuswamy Malibu Networks 1035 Suncast Lane El Dorado Hills, CA 95762	Voice: 916-941-8815 Fax: 916-941-8850 mailto:subbu@malibunetworks.com
	Juha Salokannel Nokia Visiokatu 1 Tampere, FINLAND, FIN-33720	Voice: +358 3 272 5494 Fax: mailto: juha.salokannel@nokia.com
	Vladimir Yanover Alvarion Ltd 21 A Habarzel St. Ramat-Hahayal Tel - Aviv, Israel, 69710	Voice: +972-36457834 Fax: +972-36456290 mailto:vladimiry@breezecom.co.il
	Huanchun Ye BeamReach Networks 755 North Mathilda Avenue Sunnyvale, CA 94086	Voice: 408-869-8748 Fax: 408-869-8701 mailto:hye@beamreachnetworks.com

Re:	TG3/4 MAC Ad Hoc output with Session 15 802.16ab MAC resolutions

Abstract	This ARQ proposal incorporates many of the elements from TG3/4 ARQ AdHoc documents, related comments and contributions and Session 15 TG3/4 MAC resolutions. It is intended for consideration by the TG3/4 MAC group for incorporation into 802.16ab MAC standard.

Purpose	Incorporate the text of this document under ARQ section of TG3/4 MAC document (80216ab-01_01r1)
Notice	This document has been prepared to assist IEEE 802.16. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein.
Release	The contributor grants a free, irrevocable license to the IEEE to incorporate text contained in this contribution, and any modifications thereof, in the creation of an IEEE Standards publication; to copyright in the IEEE's name any IEEE Standards publication even though it may include portions of this contribution; and at the IEEE's sole discretion to permit others to reproduce in whole or in part the resulting IEEE Standards publication. The contributor also acknowledges and accepts that this contribution may be made public by IEEE 802.16.
Patent Policy and Procedures	The contributor is familiar with the IEEE 802.16 Patent Policy and Procedures (Version 1.0) <http://ieee802.org/16/ipr/patents/policy.html>, including the statement "IEEE standards may include the known use of patent(s), including patent applications, if there is technical justification in the opinion of the standards-developing committee and provided the IEEE receives assurance from the patent holder that it will license applicants under reasonable terms and conditions for the purpose of implementing the standard." Early disclosure to the Working Group of patent information that might be relevant to the standard is essential to reduce the possibility for delays in the development process and increase the likelihood that the draft publication will be approved for publication. Please notify the Chair <mailto:r.b.marks@ieee.org> as early as possible, in written or electronic form, of any patents (granted or under application) that may cover technology that is under consideration by or has been approved by IEEE 802.16. The Chair will disclose this notification via the IEEE 802.16 web site <http://ieee802.org/16/ipr/patents/notices>.

1 Introduction

The ARQ mechanism is part of the MAC layer and can be enabled on a per-connection basis. The per-connection ARQ and associated parameters should be specified and negotiated during connection creation or change. A connection cannot have a mixed ARQ and non-ARQ traffic. Similar to other properties of the MAC protocol the scope of a specific instance of ARQ is limited to one unidirectional connection.

The ARQ feedback information can be sent as a standalone MAC management message on the appropriate basic management connection, or as piggybacked sub-headers on an existing connection. ARQ feedback cannot be fragmented. The implementation of ARQ is optional.

The term MPDU (MAC Protocol Data Unit), used throughout this document, refers to a MAC PDU with a single MAC header, zero or more optional main or packing sub-headers and a payload, where the presence of sub-headers is indicated by the TYPE field in the generic MAC header. An MPDU may carry one or more whole or fragmented MAC SDUs.

2 Block Numbering Scheme

An ARQ block is a uniquely identifiable entity on which the ARQ algorithm operates. Each ARQ block is identified by an *ARQ block number*, which is assigned to it by the MAC. ARQ block numbers are assigned in increasing order, modulo **ARQ_MAX_BSN,** where **ARQ_MAX_BSN** is 2048 (2^{11}). When the MAC decides to transmit a certain MAC SDU for the first time, it assigns it block numbers starting from the current block index, and according to the **ARQ_BLOCK_SIZE** parameter, that will determine how many ARQ blocks are contained in the MAC SDU. The last block of an MAC SDU may be of size less than the **ARQ_BLOCK_SIZE**. Note that the ARQ numbering is merely a numbering scheme that identifies both the MAC SDU transmission order, and the order of the ARQ blocks comprising each MAC SDU. Note the ARQ block numbering implies nothing on the order and size of MPDU transmission. The **ARQ_BLOCK_SIZE** shall be between 4 to 2048 bytes, with a resolution of one byte. The block size is negotiated between the peers during the connection creation/change. A sequence of blocks following a MAC header or sub-header must have contiguous block numbers.

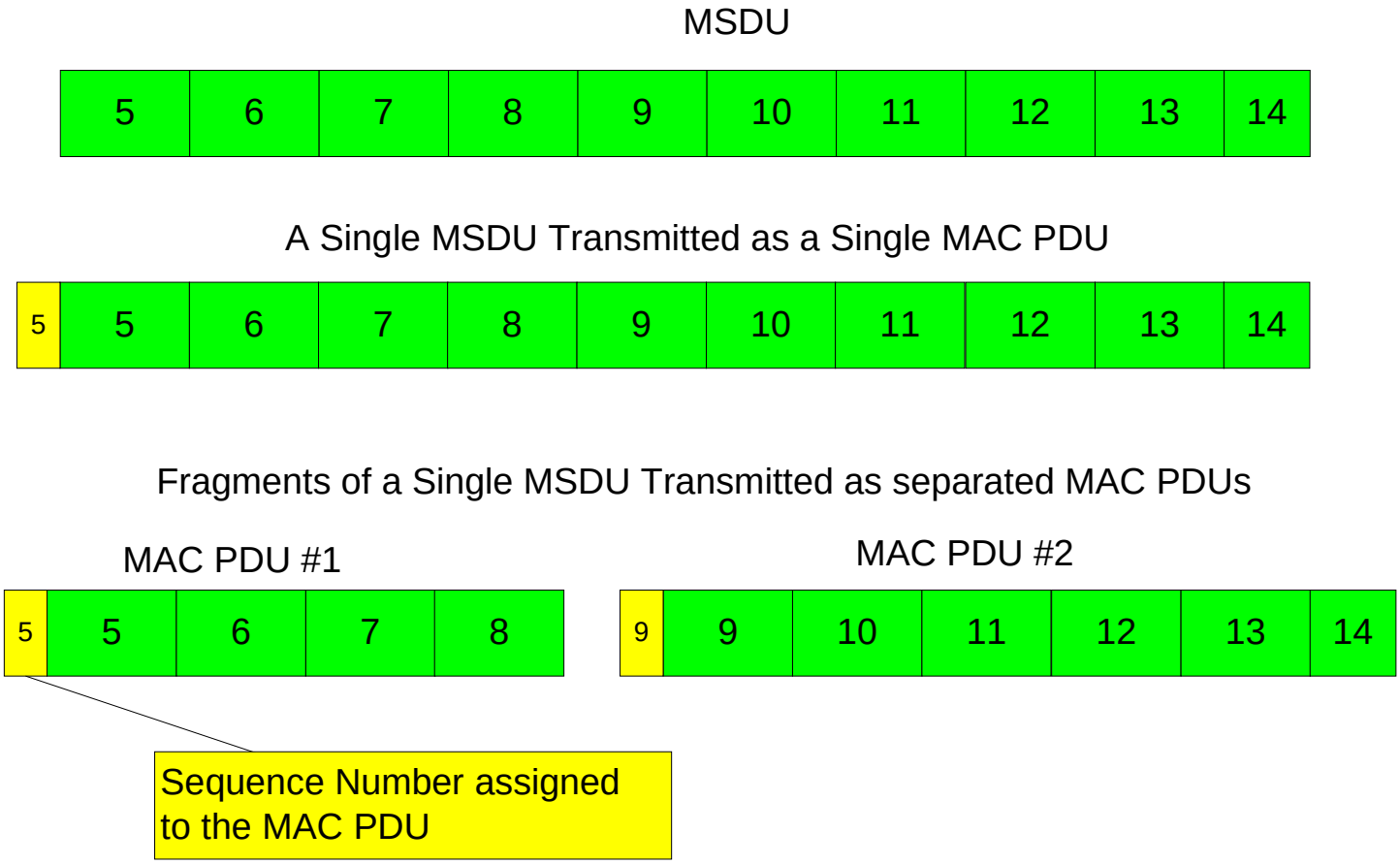


Figure 1: Basic block numbering

3 ARQ Acknowledgement, Sub-header and Type fields

3.1 Acknowledgment Message Format

Table 1 shows the basic ARQ acknowledgment information element used by the receiver to signal positive or negative acknowledgments. This information element may be a transported as part of the ARQ feedback sub-header or as a payload in a standalone MAC PDU.

Table 1: Acknowledgement information element

Syntax	Size	Notes
ARQ_feedback_IE () {		
CID	16 bits	The ID of the connection being referenced
LAST	1 bit	0 = More ARQ feedback IE in the list 1 = Last ARQ feedback IE in the list
ACK Type	2 bits	00 = Selective ACK entry 01 = Cumulative ACK entry
BSN	11 bits	Block sequential number for the acknowledged ARQ block
Number of 16 bits ACK Maps	2 bits	00 = 16 bits 01 = 32 bits 10 = 48 bits 11 = 64 bits
If (ACK Type == 00) {		
ACK MAP	16 bits	Each bit set to one means the corresponding ARQ block has been received without errors. The first bit corresponds to the ARQ block whose number is the BSN above.
}		
}		

3.2 Updated Generic MAC header type field encoding

The presence of the sub-headers is indicated by the value of the TYPE field in the generic MAC header. The table below lists the encoding of the TYPE field. The ARQ feedback sub-header enables piggybacking of ARQ feedback information elements. When ARQ-feedback is present, it is always the last sub-header preceding the fragmentation and packing sub-headers that always precede an MAC SDU or fragment. Figure 1 illustrated the piggybacked ARQ feedback as a variable size sub-header.

Figure 2: Example structure of the MAC PDU with ARQ feedback as a piggybacked sub-header

Table 2: Downlink type encoding

Type	Description
0x00	No sub-headers present
0x01	Reserved
0x02	ARQ sub-header with packing or TG1 packing sub-header present
0x03	Reserved
0x04	ARQ sub-header without packing or TG1 Fragmentation sub-header present
0x05	Reserved
0x06	Reserved
0x07	Reserved
0x08	ARQ-feedback sub-header present
0x09	Reserved
0x0A	ARQ-feedback and packing sub-headers present
0x0B	Reserved
0x0C	ARQ-feedback and fragmentation sub-headers present
0x0D – 0x3F	Reserved

Table 3: Uplink type encoding

Type	Description
0x00	No sub-headers present
0x01	Grant management sub-header present
0x02	ARQ sub-header with packing or TG1 packing sub-header present
0x03	Grant management and packing sub-headers present
0x04	ARQ sub-header without packing or TG1 Fragmentation sub-header present
0x05	Grant management and fragmentation sub-headers present
0x06	Reserved
0x07	Reserved
0x08	ARQ-feedback sub-header present
0x09	ARQ-feedback and grant management sub-headers present
0x0A	ARQ-feedback and packing sub-headers present
0x0B	ARQ-feedback and packing and grant management sub-headers present
0x0C	ARQ-feedback and fragmentation sub-headers present
0x0D	ARQ-feedback and fragmentation and grant management sub-headers present
0x0E – 0x3F	Reserved

3.3 ARQ sub-header without packing

In this case each MPDU will contain a whole MAC SDU or fragment of an MAC SDU. Knowledge of the BSN of the first ARQ block, the length of the MAC SDU or fragment (conveyed in the MAC header) and the **ARQ_BLOCK_SIZE** parameter enable the calculation of the range of ARQ blocks contained in the MAC PDU. The ARQ with fragmentation sub-header position and its contents is shown below.

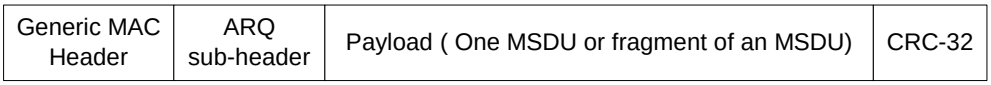


Figure 3: Structure of the MAC PDU with ARQ sub-header without packing

Table 4: Format of ARQ sub-header with no packing

Syntax	Size	Notes
Fragmentation_and_ARQ_Sub_Header_Format() {		
FC	2 bits	Fragmentation Control Indicates the fragmentation state of the payload: 00 = no fragmentation 01 = last fragment 10 = first fragment 11 = continuing (middle) fragment
Reserved	2 bits	
ACK Request ("A-bit")	1 bit	Receiver must send an acknowledgement when this bit is set
BSN	11 bits	Block Sequence number for the first ARQ block in the MAC SDU fragment
}		

3.4 ARQ sub-header with packing

In this case each MPDU may contain multiple MAC SDUs or fragments thereof. Each of the packed MAC SDU or MAC SDU fragments requires its own ARQ sub-header, as some of them may be transmissions while other are re-transmissions. Knowledge of the BSN of the first ARQ block, the length of the each MAC SDU fragment (conveyed in the packing sub-header) and the **ARQ_BLOCK_SIZE** parameter enable the calculation of the range of ARQ blocks contained in each of the packed message. The ARQ with packing sub-headers position and the contents of an ARQ with packing sub-header is shown below. Note that the A-bit is not present in this sub-header.

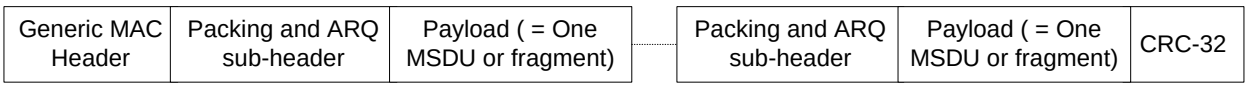


Figure 4: Structure of the MAC PDU with ARQ sub-header with packing

Table 5: Format of ARQ sub-header with packing

Syntax	Size	Notes
Packing_and_ARQ_Sub_Header_Format() {		
FC	2 bits	Fragmentation Control Indicates the fragmentation state of the payload: 00 = no fragmentation 01 = last fragment 10 = first fragment 11 = continuing (middle) fragment
Length	11 bits	The length in bytes of the MAC SDU or MAC SDU fragment, including the three-byte Packing_and_ARQ sub-header
BSN	11 bits	Block sequential number for the first ARQ block in the MAC SDU or MAC SDU fragment
}		

3.5 ARQ Feedback Sub-header

The ARQ feedback sub-header concatenates one or more ARQ feedback information elements. Therefore the ARQ feedback information element can be “piggybacked” on any connection in the form of a sub-header. Several information elements may be present within the same sub-header. The LAST flag (Table 1) of the information element indicates whether this is the last information element or not, within this sub-header.

4 ARQ Parameters

The parameters defined in this section are negotiated between the transmitter and the receiver at the connection set-up.

4.1 ARQ_BLOCK_SIZE

ARQ_BLOCK_SIZE is the basic retransmission unit and it should be negotiated during connection setup. The concept of blocks is described in Section 2.

4.2 ARQ_WINDOW_SIZE

ARQ_WINDOW_SIZE must be less than or equal to half of the **ARQ_MAX_BSN**.

4.3 ARQ_BLOCK_LIFETIME

ARQ_BLOCK_LIFETIME is the maximum time interval beyond which a transmitter shall discard unacknowledged ARQ blocks.

4.4 ARQ_RETRY_TIMEOUT

ARQ_RETRY_TIMEOUT is the minimum time interval a transmitter shall wait before retransmitting an unacknowledged ARQ block.

4.5 ARQ_SYNC_LOSS_TIMEOUT

ARQ_SYNC_LOSS_TIMEOUT is the minimum time interval after which the ARQ synchronization shall be considered lost.

5 ARQ Procedures

5.1 ARQ State Machine Variables

5.1.1 Transmitter Variables

- *ARQ_TX_WINDOW_START*: All BSN up to (*ARQ_TX_WINDOW_START* - 1) have been acknowledged.
- *ARQ_TX_NEXT_BSN*: BSN of the next block to send. This value shall be between *ARQ_TX_WINDOW_START* and (*ARQ_TX_WINDOW_START* + **ARQ_WINDOW_SIZE).**

5.1.2 Receiver Variables

- *ARQ_RX_WINDOW_START*: All BSN up to (*ARQ_RX_WINDOW_START* - 1) have been correctly received.
- *ARQ_RX_HIGHEST_BSN*: BSN of the highest block received, plus one. This value shall be between *ARQ_RX_WINDOW_START* and (*ARQ_RX_WINDOW_START* + **ARQ_WINDOW_SIZE).**

5.2 ARQ Connection Setup and Negotiation

Connections are set and defined either statically through the configuration file, or dynamically through the DSA/DSC class of messages. CRC-32 shall be used for error detection of MAC PDUs for all ARQ connections. All the ARQ parameters (Section 4) shall be set when an ARQ connection is set up. The transmitter and receiver variables (defined in Section 5.1) shall be reset on connection setup. This section describes the TLV fields for the ARQ algorithm that are required for both static and dynamic connection creation methods.

5.2.1 ARQ_SUPPORT

This field indicates whether or not ARQ is available for the connection that is being setup. A value of 0 indicates the non-availability of ARQ support and a value 1 indicates otherwise. The DSA-REQ/DSC-REQ shall contain the request to use ARQ or not. The DSA-RSP/DSC-RSP message shall contain the acceptance or rejection of the request. ARQ shall be enabled for this connection only if both sides support it

Table 6: ARQ_SUPPORT TLV

Type	Length	Value	Scope
[24/25].?	1	0 = ARQ Not Supported 1= ARQ Supported	DSx-REQ DSx-RSP Configuration file

5.2.2 ARQ_BLOCK_SIZE

This parameter is negotiated upon connection setup. The DSA-REQ/DSC-REQ shall contain the suggested value of this parameter. The DSA-RSP/DSC-RSP message shall contain the confirmation value or an alternate value for this parameter. The smaller of the two shall be used as the **ARQ_BLOCK_SIZE**.

Table 7: ARQ_BLOCK_SIZE TLV

Type	Length	Value	Scope
[24/25].?	2	4 - 2048	DSx-REQ DSx-RSP Configuration file

5.2.3 ARQ_WINDOW_SIZE

This parameter is negotiated upon connection setup. The DSA-REQ/DSC-REQ message shall contain the suggested value for this parameter. The DSA-RSP/DSC-RSP message shall contain the confirmation value or an alternate value for this parameter. The smaller of the two shall be used as the **ARQ_WINDOW_SIZE**.

Table 8: ARQ_WINDOW_SIZE TLV

Type	Length	Value	Scope
[24/25].?	2	1 – (ARQ_MAX_BSN/2)	DSx-REQ DSx-RSP Configuration file

5.2.4 ARQ_RETRY_TIMEOUT

The ARQ retry timeout should account for the transmitter and receiver processing delays and any other delays relevant to the system.

- TRANSMITTER_DELAY: This is the total transmitter delay, including sending (e.g., MAC PDUs) and receiving (e.g., ARQ feedback) delays and other implementation dependent processing delays. If the transmitter is the BS, it may include other delays such as scheduling and propagation delay.
- RECEIVER_DELAY: This is the total receiver delay, including receiving (e.g., MAC PDUs) and sending (e.g., ARQ feedback) delays and other implementation dependent processing delays. If the receiver is the BS, it may include other delays such as scheduling and propagation delay.

The DSA-REQ/DSC-REQ and DSA-RSP/DSC-RSP messages shall contain the values for these parameters, where the receiver and transmitter each declare their capabilities. When the DSA/DSC handshake is completed, each party shall calculate **ARQ_RETRY_TIMEOUT** to be the sum of TRANSMITTER_DELAY and RECEIVER_DELAY. The table below lists the relevant TLVs.

Table 9: ACK_RETRY_TIMEOUT related TLVs

Name	Type	Length	Value	Scope
TRANSMITTER_DELAY	[24/25].?.?	2	0 – 65535 (In microseconds)	DSx-REQ DSx-RSP Configuration file
RECEIVER_DELAY	[24/25].?.?	2	0 – 65535 (In microseconds)	DSx-REQ DSx-RSP Configuration file

5.2.5 ARQ_BLOCK_LIFETIME

The BS shall set this parameter. The DSA-REQ/DSC-REQ or DSA-RSP/DSC-RSP messages shall contain the value of this parameter as set by the BS.

Table 10: ARQ_BLOCK_LIFETIME TLV

Type	Length	Value	Scope
[24/25].?	2	0 – 65535 (In microseconds)	DSx-REQ DSx-RSP Configuration file

5.2.6 ARQ_SYNC_LOSS_TIMEOUT

The BS shall set this parameter. The DSA-REQ/DSC-REQ or DSA-RSP/DSC-RSP messages shall contain the value of this parameter as set by the BS.

Table 11: ARQ_SYNC_LOSS_TIMEOUT TLV

Type	Length	Value	Scope
[24/25].?	2	0 – 65535 (In microseconds)	DSx-REQ DSx-RSP Configuration file

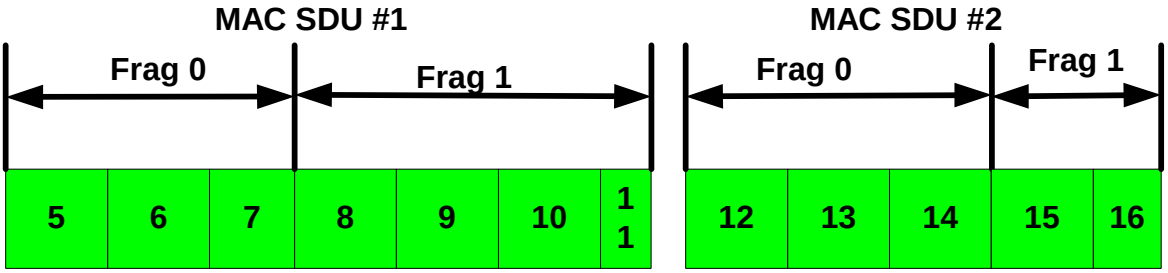
5.3 ARQ Operation

5.3.1 ARQ Block Usage

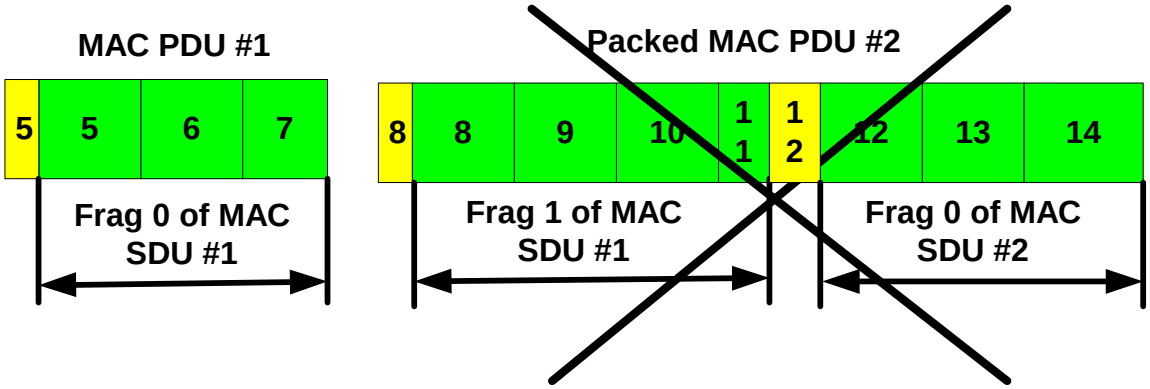
The section describes the use of blocks for ARQ. A MAC SDU is logically divided into blocks of given **ARQ_BLOCK_SIZE**. The last block of a MAC SDU may be smaller than **ARQ_BLOCK_SIZE**. Once defined, the block never changes.

A set of blocks selected for transmission or retransmission is encapsulated into a MAC PDU. A MAC PDU may contain blocks that are transmitted for the first time as well as retransmitted blocks. If fragmentation is enabled for this connection, the fragmentation shall occur only on ARQ block boundaries. Note that a sequence of blocks following a MAC header or sub-header must have contiguous block numbers.

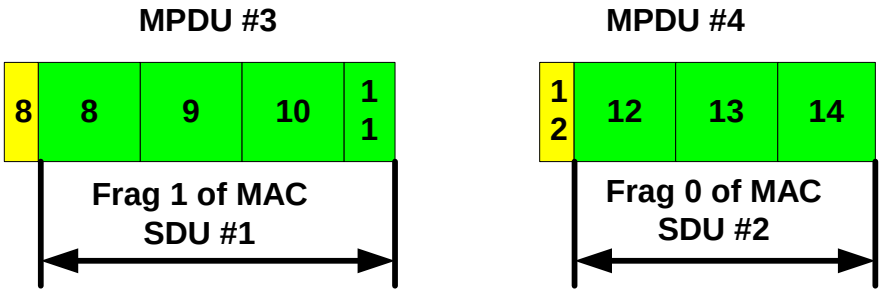
Each ARQ sub-header contains a BSN, which is the sequence number of the first ARQ block in the sequence of blocks following this sub-header. It is a matter of transmitter's policy whether a set of blocks once transmitted as a single MAC PDU should be retransmitted also as a single MAC PDU or not**.** The Figure below illustrates the use of blocks for ARQ transmissions and retransmissions.



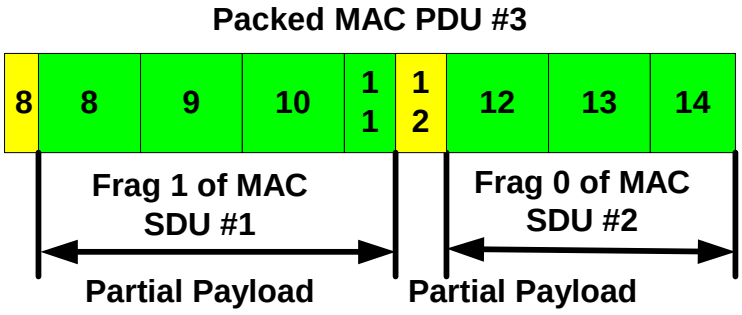
Original Transmission



Retransmission with rearrangement of MAC PDU



Retransmission without rearrangement of MAC PDUs



5.3.2 Transmitter state machine

An ARQ block may be in one of the following four states, *not-sent*, *outstanding*, *discarded* and *waiting-for-retransmission*. Any ARQ block begins as *not-sent*. After it is sent it becomes *outstanding* for a period of time termed **ACK_RETRY_TIMEOUT**. While a block is in *outstanding* state, it either is acknowledged and is *discarded*, or transitions to *waiting-for-retransmission* after **ACK_RETRY_TIMEOUT**. An ARQ block can become *waiting-for-retransmission* before the **ACK_RETRY_TIMEOUT** period expires if it is negatively acknowledged. An ARQ block may also change from *waiting-for-retransmission* to *discarded* when an ACK message for it is received or after a timeout **ARQ_BLOCK_LIFETIME**. The specification of timeout values is outside the scope of the ARQ specification.

The transmitter policy is that if any *waiting-for-retransmission* ARQ blocks exist, they should be given precedence over *not-sent* packets for the same connection. ARQ blocks that are *outstanding* or *discarded* should never be transmitted. When blocks are retransmitted, the block with the lowest BSN shall be retransmitted first.

The ARQ block state sequence is shown below (Figure 5).

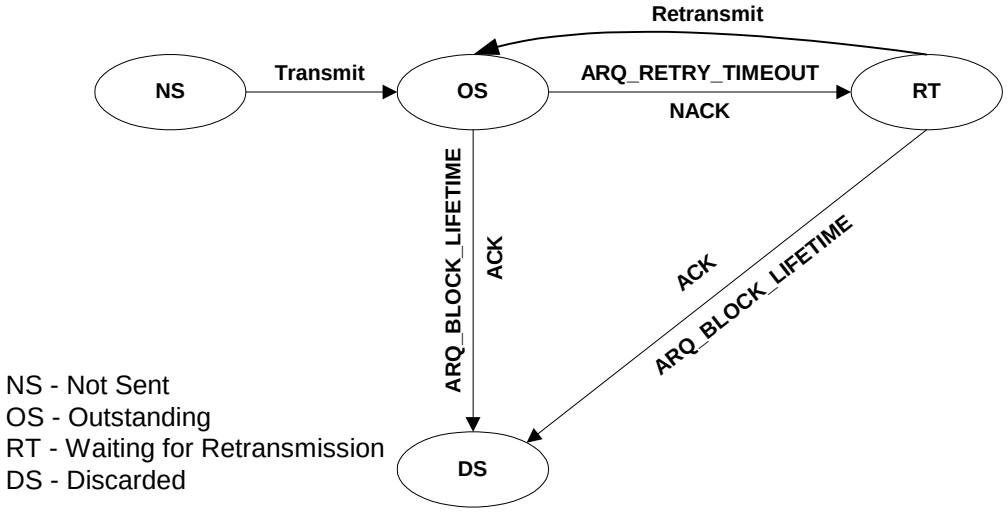


Figure 5: ARQ block states

The transmitter checks (*ARQ_TX_WINDOW_START* + **ARQ_WINDOW_SIZE** - *ARQ_TX_NEXT_BSN)* to see how many ARQ blocks can be transmitted, and creates a full or partial MAC PDU that does not exceed this value. The state variable *ARQ_TX_NEXT_BSN* is copied into the BSN field before transmission, and *ARQ_TX_NEXT_BSN* is incremented after transmission by the number of blocks in the full or partial MAC PDU. If the **ARQ_WINDOW_SIZE** limit is reached due to the transmission of MAC PDU, then the A-bit must be set to 1. Otherwise, the choice of setting A-bit or not is implementation dependent.

When an acknowledgement is received, the transmitter shall check the validity of BSN. A valid BSN is one in the range [*ARQ_TX_WINDOW_START, ARQ_TX_NEXT_BSN*]. If BSN is not valid, the transmitter shall ignore the acknowledgement.

If BSN is valid and the acknowledgement is a cumulative, then the transmitter shall consider all blocks in the range [*ARQ_TX_WINDOW_START*, BSN) as acknowledged, set *ARQ_TX_WINDOW_START* to BSN. All timers associated with acknowledged blocks shall be cancelled.

On receiving a valid selective acknowledgement message, the transmitter shall consider all blocks with the corresponding bit set to 1 in the bitmap as acknowledged and those with the corresponding bit set to 0 as

negatively acknowledged. Note that an MAC PDU may consist of ARQ blocks transmitted for the first time as well as retransmitted ARQ blocks.

5.3.3 Receiver State Machine

When an MPDU is received, its integrity is determined based on the CRC-32 checksum. If an MPDU passes the checksum, it is unpacked and de-fragmented, if necessary. The receiver maintains a sliding-window defined by *ARQ_RX_WINDOW_START* state variable and the **ARQ_WINDOW_SIZE** parameter. When an ARQ block with a number that falls in the range defined by sliding window is received, the receiver shall accept it. ARQ block numbers outside the sliding window shall be rejected as out of order. The receiver should discard duplicate ARQ blocks (i.e. ARQ blocks that where already received correctly) within the window.

The sliding window is maintained such that the *ARQ_RX_WINDOW_START* variable always points to the lowest numbered ARQ block that has not been received or has been received with errors. When an ARQ block with a number corresponding to the *ARQ_RX_WINDOW_START* is received, the window is advanced (i.e. *ARQ_RX_WINDOW_START* is incremented modulo **ARQ_MAX_BSN**) such that the *ARQ_RX_WINDOW_START* variable points to the next lowest numbered ARQ block that has not been received or has been received with errors. The timer associated with **ARQ_SYNC_LOSS_TIMEOUT** shall be reset.

If ARQ blocks are being received, yet the *ARQ_RX_WINDOW_START* variable has been pointing to a specific ARQ block for more than **ARQ_SYNC_LOSS_TIMEOUT**, the ARQ synchronization shall be considered lost. In such a case, the window shall slide until an ARQ block that has been correctly received is found. The *ARQ_RX_WINDOW_START* shall point to the next lowest numbered ARQ block that has not been received or has been received with errors.

For each ARQ block accepted fully and without errors (including duplicates), an acknowledgment message may be sent to the transmitter. Acknowledgments may be either for specific ARQ blocks (i.e. contain information on the acknowledged ARQ block numbers), or cumulative (i.e. contain the highest ARQ block number below which all ARQ blocks have been received correctly). Acknowledgments shall be sent in the order of the ARQ block numbers they acknowledge. The frequency of acknowledgement generation is not specified here and is implementation dependent. The receiver shall respond with an acknowledgement, whenever an ARQ sub-header is received with the A-bit set.

An MAC SDU is handed to the upper layers when all the ARQ blocks of the MAC SDU have been correctly received within the timeout values defined.

5.4 Standalone ARQ Feedback

The ARQ feedback message may take the format of a stand-alone MAC message as shown in Table 12. It can be used to signal a cumulative ACK or several selective ACKs similar to the piggybacked sub-header mechanism. This feedback shall be sent as a MAC management message on the appropriate basic management connection (i.e. it cannot be fragmented)

Syntax	Size	Notes
ACK_Message_Format() {		
Management Message Type = ?	8 bits	
for (*i* =1; *I* <*n* ; *i* ++) {		Repeat as many times as required
ARQ_feedback_IE ()	16 bits	The connection ID being referenced
}		
}		

Table 12: Standalone ACK Feedback