

Project	<b>IEEE 802.16 Broadband Wireless Access Working Group</b> < <a href="http://ieee802.org/16">http://ieee802.org/16</a> >	
Title	<b>MAC and PHY MIB for WirelessMAN and WirelessHUMAN BS and SS</b>	
Date Submitted	<b>2004-03-12</b>	
Source(s)	Joey Chou Intel Corporation	<a href="mailto:joey.chou@intel.com">[mailto:joey.chou@intel.com]</a>
	Russ Reynolds Alex Raji Proxim Corporation	<a href="mailto:RReynolds@proxim.com">[mailto:RReynolds@proxim.com]</a> <a href="mailto:adr@telecomminnovations.com">[mailto:adr@telecomminnovations.com]</a>
	Vladimir Yanover Alvarion	<a href="mailto:Vladimir.Yanover@alvarion.com">[mailto: Vladimir.Yanover@alvarion.com]</a>
	Shlomi Eini Airspan	<a href="mailto:seini@Airspan.com">[mailto:seini@Airspan.com]</a>
	Radu Selea Redline Communications	<a href="mailto:radu@redlinecommunications.com">[mailto:radu@redlinecommunications.com]</a>
Re:		
Abstract	IEEE 802.16 working group defines WirelessMAN and WirelessHUMAN air interface specifications for the development of standard based Base Station (BS) and Subscriber Station (SS) to provide broadband wireless services to Metropolitan Area Networks (MANs). This contribution defines the 802.16 MIB for MAC and PHY layers to achieve management interoperability and provide the remote management capability that are urgently needed for massive WirelessMAN and WirelessHUMAN deployment by carriers.	
Purpose	Adoption	
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 material 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	<p>The contributor is familiar with the IEEE 802.16 Patent Policy and Procedures (Version 1.0) &lt;<a href="http://ieee802.org/16/ipr/patents/policy.html">http://ieee802.org/16/ipr/patents/policy.html</a>&gt;, 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."</p> <p>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 &lt;<a href="mailto:r.b.marks@ieee.org">mailto:r.b.marks@ieee.org</a>&gt; as early as possible, in written or electronic form, of any patents</p>	

---

(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>>.

---

## *Table of Content*

<b>1. Introduction .....</b>	<b>5</b>
<b>1.1 Scope.....</b>	<b>5</b>
<b>1.2 References.....</b>	<b>5</b>
<b>2. BWA Network Management Reference Model.....</b>	<b>6</b>
<b>3. Relationship with Interface MIB .....</b>	<b>7</b>
<b>3.1 MIB-2 Integration .....</b>	<b>7</b>
<b>3.2 Usage of MIB-II Tables .....</b>	<b>7</b>
<b>4. Service Flow Creation.....</b>	<b>9</b>
<b>5. 802.16 MIB Structure .....</b>	<b>10</b>
<b>5.1 wmanIfBsObjects .....</b>	<b>11</b>
<b>5.1.1 wmanIfBsSystem.....</b>	<b>11</b>
<b>5.1.1.1 wmanIfBsInventoryInfoTable.....</b>	<b>11</b>
<b>5.1.1.2 wmanIfBsRegisteredSsTable.....</b>	<b>11</b>
<b>5.1.2 wmanIfBsPacketCs .....</b>	<b>11</b>
<b>5.1.2.1 wmanIfBsPreProvisionedSfTable.....</b>	<b>11</b>
<b>5.1.2.2 wmanIfBsServiceClassTable .....</b>	<b>12</b>
<b>5.1.3 wmanIfBsCps.....</b>	<b>12</b>
<b>5.1.3.1 wmanIfBsConfigurationTable .....</b>	<b>12</b>
<b>5.1.3.2 wmanIfBsStatisticsCountersTable .....</b>	<b>13</b>
<b>5.1.4 wmanIfBsPkm .....</b>	<b>13</b>
<b>5.1.4.1 wmanIfBsPkmBaselineTable .....</b>	<b>13</b>
<b>5.1.4.2 wmanIfBsAuthTable.....</b>	<b>13</b>
<b>5.1.4.3 wmanIfBsTekTable.....</b>	<b>13</b>
<b>5.1.5 wmanIfBsTraps .....</b>	<b>13</b>
<b>5.2 wmanIfSsObjects.....</b>	<b>13</b>
<b>5.2.1 wmanSsSystem .....</b>	<b>13</b>
<b>5.2.1.1 wmanSsInventoryInfoTable .....</b>	<b>13</b>
<b>5.2.2 wmanIfSsCps .....</b>	<b>13</b>
<b>5.2.2.1 wmanIfSsConfigurationTable.....</b>	<b>13</b>
<b>5.2.2.2 wmanIfSsStatisticsCountersTable.....</b>	<b>14</b>
<b>5.2.3 wmanIfSsPkm.....</b>	<b>14</b>
<b>5.2.3.1 wmanIfSsPkmAuthTable.....</b>	<b>14</b>
<b>5.2.3.2 wmanIfSsPkmTekTable.....</b>	<b>14</b>
<b>5.2.3.3 wmanIfSsPkmCertificatesTable .....</b>	<b>14</b>

5.2.4	wmanIfSsTraps.....	14
5.3	wmanIfCommonObjects.....	14
5.3.1	wmanIfCmnPacketCs.....	14
5.3.1.1	wmanIfCmnClassifierRuleTable.....	14
5.3.2	wmanIfCmnCps.....	14
5.3.2.1	wmanIfCmnSystemParameters.....	14
5.3.2.2	wmanIfCmnSsCapabilitiesTable (index = ifIndex).....	14
5.3.2.3	wmanIfCmnServiceFlowTable.....	15
5.3.3	wmanIfCmnPrivacy.....	15
5.3.3.1	wmanIfCmnCryptoSuiteTable.....	15
5.3.4	wmanIfCmnOfdmPhy.....	15
5.3.4.1	wmanIfOfdmUplinkChannelTable.....	15
5.3.4.2	wmanIfOfdmDownlinkChannelTable.....	15
5.3.4.3	wmanIfOfdmUcdBurstProfileTable.....	16
5.3.4.4	wmanIfOfdmDcdBurstProfileTable.....	16
6.	<i>ASN.1 Definition of 802.16 MIB</i> .....	17

## 1. Introduction

IEEE 802.16 working group defines WirelessMAN and WirelessHUMAN air interface specifications for the development of standard based Base Station (BS) and Subscriber Station (SS) to provide broadband wireless services to Metropolitan Area Networks (MANs). This contribution defines the 802.16 MIB for MAC and PHY layers to achieve management interoperability and provide the remote management capability that are urgently needed for massive WirelessMAN and WirelessHUMAN deployment by carriers.

### 1.1 Scope

The scope of this contribution is to define the 802.16 MAC and PHY MIB for SS and BS, based on IEEE 802.16REVd/D3 specification [3]. The definition of managed objects in this MIB is based on SNMPv2 Structure of Management Information (SMI) [4] and Textual Conventions [5]. Therefore, 802.16 MIB is compliant to SNMPV2, but is backward compatible to SNMPv1 through appropriate translation.

Since 802.16 MIB has to be accessed through MIB tree, its relationship with Interface MIB—RFC2863 [7] will be described. Additional MIBs may be necessary to manage other interfaces in SS or BS, such as Ethernet, T1/E1, and ATM, but they are outside the scope of this contribution.

### 1.2 References

- [1] IEEE 802.16-2001, "IEEE Standard for Local and Metropolitan area networks – Part 16: Air Interface for Fixed Wireless Access Systems".
- [2] IEEE 802.16a-2003, "IEEE Standard for Local and Metropolitan area networks – Part 16: Air Interface for Fixed Wireless Access Systems – Amendment 2: Medium Access Control Modifications and Additional Physical Layer Specifications for 2-11 GHz.
- [3] IEEE 802.16REVd/D3-2004, "Draft IEEE Standard for Local and Metropolitan area networks – Part 16: Air Interface for Fixed Broadband Wireless Access Systems.
- [4] RFC1902, "Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2)", January 1996
- [5] RFC1903, "Textual Convention for Version 2 of the Simple Network Management Protocol (SNMPv2)", January 1996
- [6] RFC 1213, " Management Information Base for Network Management of TCP/IP-based internets: MIB-II", IETF, March 1991
- [7] RFC2863, "The Interfaces Group MIB", June, 2000
- [8] RFC2515, "Definitions of Managed Objects for ATM Management", February, 1999

## 2. BWA Network Management Reference Model

Figure 1 shows the management reference model of Broadband Wireless Access (BWA) networks. It consists of a network Management System (NMS), performing the network manager role, and managed nodes, which provide access to managed objects via MIB or virtual information store. SSs and BSs are managed nodes that act in the SNMP agent role. Furthermore, managed SSs, which have a secondary management CID, may be managed indirectly through the BS to which they are registered. In this case, the BS acts in an SNMP Proxy role on behalf of managed SSs. SS can be managed by NMS directly as well.

The management information between SS and BS will be carried over Second Management CID for managed SS. If it is an unmanaged SS where 2<sup>nd</sup> management CID does not exist, the SNMP messages shall go through another interface in the customer premise. However, the definition of 802.16 MIB should be independent of managed / unmanaged SS, or proxy / direct mode.

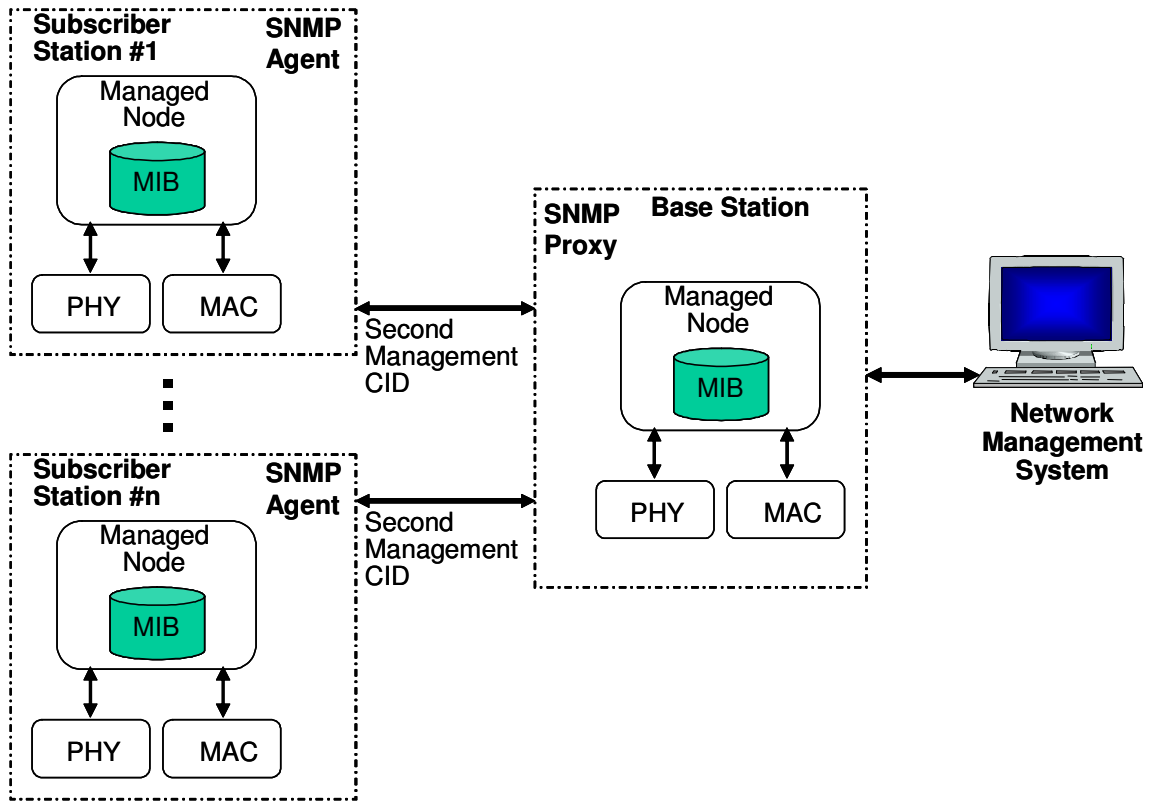


Figure 1 - BWA Network Management Reference Model

### 3. Relationship with Interface MIB

This section describes the integration with MIB-II [6] under Interface Group MIB defined in RFC2863, as 802.16 MIB will need to be integrated in the MIB tree. It describes where 802.16 MIB is located in the MIB-II subtree, and how it can be accessed by NMS.

#### 3.1 MIB-2 Integration

The IANA has assigned the following `ifType` to point to multipoint broadband wireless access.

```

IANAifType ::= TEXTUAL-COVENTION
SYNTAX INTEGER {
    propBWAp2Mp (184) -- prop broadband wireless access
                    -- point to multipoint
}

```

Therefore, upon 802.16 MIB being approved by the IETF, this MIB can be accessed through

```

iso.org.dod.internet.mgmt.mib-2.transmission.ifType
(1.3.6.1.2.1.10.184)

```

Wireless MAN interface table is located under transmission subtree, as follows.

```

wmanMIB ::= {transmission 184} -- wMAN interface table

```

Before the approval of the IETF; however, 802.16 MIB is temporary located under enterprise via

```

iso.org.dod.internet.private.enterprise.wmanMIB
(1.3.6.1.4.1.n)

```

Or

```

iso.org.dod.internet.private.enterprise.vendorID.wmanMIB
(1.3.6.1.4.1.xxx.n)

```

If `propBWAp2Mp` has been designed for other BWA interface, it will be required to ask IANA to assign another `IANAifType` for 802.16. In addition, two `IANAifType` should be assigned by IANA to represent 802.16 downstream and upstream interfaces. The following shows the structure of `IANAifType` that should be assigned by IANA.

```

IANAifType ::= TEXTUAL-COVENTION
SYNTAX INTEGER {
    wmanMAC (?) -- wireless MAN MAC Layer
    wmanDownstream (?) -- wirelesss MAN downstream interface
    wmanUpstream (?) -- wireless MAN upstream interface
}

```

#### 3.2 Usage of MIB-II Tables

"Interfaces" group of MIB-II, in RFC1573, has been designed to manage various sub-layers (e.g. MAC and PHY) beneath the internetwork-layer for numerous media-specific interfaces. `ifTable` in MIB-II is used to access the `wmanIfMib`. Table 1 describes

1 some key attributes in the *ifTable* that will be reused in the *wmanIfMib*, and provides an  
 2 example on how *ifIndex* will be assigned for each sub-layer.

3  
 4 Three *ifEntry*.will be created for each BS or radio channel installed in the system. One *ifIndex*  
 5 is for BS MAC sub-layer, while two *ifIndex* will be created for BS upstream and downstream  
 6 PHY sub-layers, respectively. When the SS has entered the network, three *ifEntry*.will be  
 7 created as well , including one *ifIndex* for BS MAC sub-layer, and two *ifIndex* for  
 8 upstream and downstream PHY sub-layers, respectively.

<i>ifTable</i>	<i>ifIndex</i>	<i>ifType (IANA)</i>	<i>ifSpeed</i>	<i>ifPhysAddress</i>	<i>ifAdminStatus</i>	<i>ifOperStatus</i>
<b>BS MAC</b>	An <i>ifEntry</i> per BS (1)	wmanMAC (?)	Null	MAC address	Administration Status	Operational Status
<b>BS Down-Stream</b>	An <i>ifEntry</i> per BS Downstream Channel (2)	wmanDownstream (?)	Speed of downstream channel in Mbps/s	Null	Administration Status	Operational Status
<b>BS Upstream</b>	An <i>ifEntry</i> per BS Upstream Channel (3)	wmanUpstream (?)	Speed of upstream channel in Mbps/s	Null	Administration Status	Operational Status
<b>SS MAC</b>	An <i>ifEntry</i> per SS (4)	wmanMAC (?)	Null	MAC address	Administration Status	Operational Status
<b>SS Down-stream</b>	An <i>ifEntry</i> per SS Downstream Channel (5)	wmanDownstream (?)	Speed of downstream channel in Mbps/s	Null	Administration Status	Operational Status
<b>SS upstream</b>	An <i>ifEntry</i> per SS Upstream Channel (6)	wmanUpstream (?)	Speed of upstream channel in Mbps/s	Null	Administration Status	Operational Status

30 **Table 1 – Usage of ifTable objects**

31  
 32 *ifStackTable* must be used to define the association of MAC, downstream, and  
 33 upstream sub-layers in SS and BS. Table 2 shows the example of *ifStackTable* for SS  
 34 and BS sub-layers as illustrated in Table 1.

<i>ifStackTable</i>	<i>ifIndex</i>									
<i>ifStackHigherLayer</i>	0	1	1	2	3	0	4	4	5	6
<i>ifStackLowerLayer</i>	1	2	3	0	0	4	5	6	0	0

42 **Table 2 – Usage of ifStackTable**



## 1 **4. Service Flow Creation**

2 Section 6.4.13.7 of IEEE 802.16REVd/D3 standard [3] specifies that there are two types of  
3 service flow creation, which are pre-provisioning and dynamic service flow creation, similar  
4 to PVC (Permanent Virtual Circuit) and SVC (Switched Virtual Circuit) for ATM. In most  
5 cases, BWA will provide subscribed services; therefore, pre-provisioning shall be required.  
6 802.16 MIB defines pre-provisioned service flow table to support pre-provisioning service  
7 flow creation. In the mobile environment, where a mobile SS requests Internet access in  
8 the hotspot, it may be necessary to use dynamic service flow creation.

9 Section 6.4.13.6 of IEEE 802.16REVd/D3 standard [3] specifies that there are three types  
10 of service flows, which are Provisioned, Admitted, and Active service flows. This section  
11 describes the usage scenario of the 802.16 MIB related to the transition of service flow  
12 types.

- 13
- 14     ▪ **Provisioned Service Flow:** When the NMS pre-provisions a service flow, an entry  
15 in `wmanIfBsPreProvisionedSfTable` in the BS MIB is created. It is assigned a  
16 Provisioned Service Flow state, but is not active yet. An attribute—  
17 `wmanIfBsSfState`—in `wmanIfBsPreProvisionedSfTable` determines how the  
18 service flow shall be transitioned to the next state, when a trigger is generated.  
19 When `wmanIfBsSfState` is in one of the following state:
  - 20         • **Admitted or Active state:** The pre-provisioned service flow will be  
21 transitioned to the Admitted or Active state, as soon as the SS passes the  
22 network entry procedure, and connection admission control. An entry will  
23 be created in the SS and BS service flow tables.
  - 24         • **Provisioned state:** After SS enters the network; the pre-provisioned  
25 service flow will remain in the Provisioned state until NMS set it different  
26 state. An entry will be created in the SS and BS service flow tables.
- 27     ▪ **Admitted Service Flow:** Network resource will be reserved when the service flow  
28 is in the Admitted service flow state. The service flow can be transitioned to the  
29 active service flow state by application triggers, such as off-hook for UGS service  
30 flows.
- 31     ▪ **Active Service Flow:** Active Service Flow state indicates the data packets are sent  
32 upstream and downstream by UL-MAP and DL-MAP. The service flow can be  
33 transitioned directly from Provision service flow state to the Active service flow  
34 state.

35  
36

## 5. 802.16 MIB Structure

Figure 2 shows the MIB structure of wmanIfMib for 802.16 [3]. The MIB structure is organized based on the the reference model as defined in IEEE 802.16REVd/D3 standard [3].

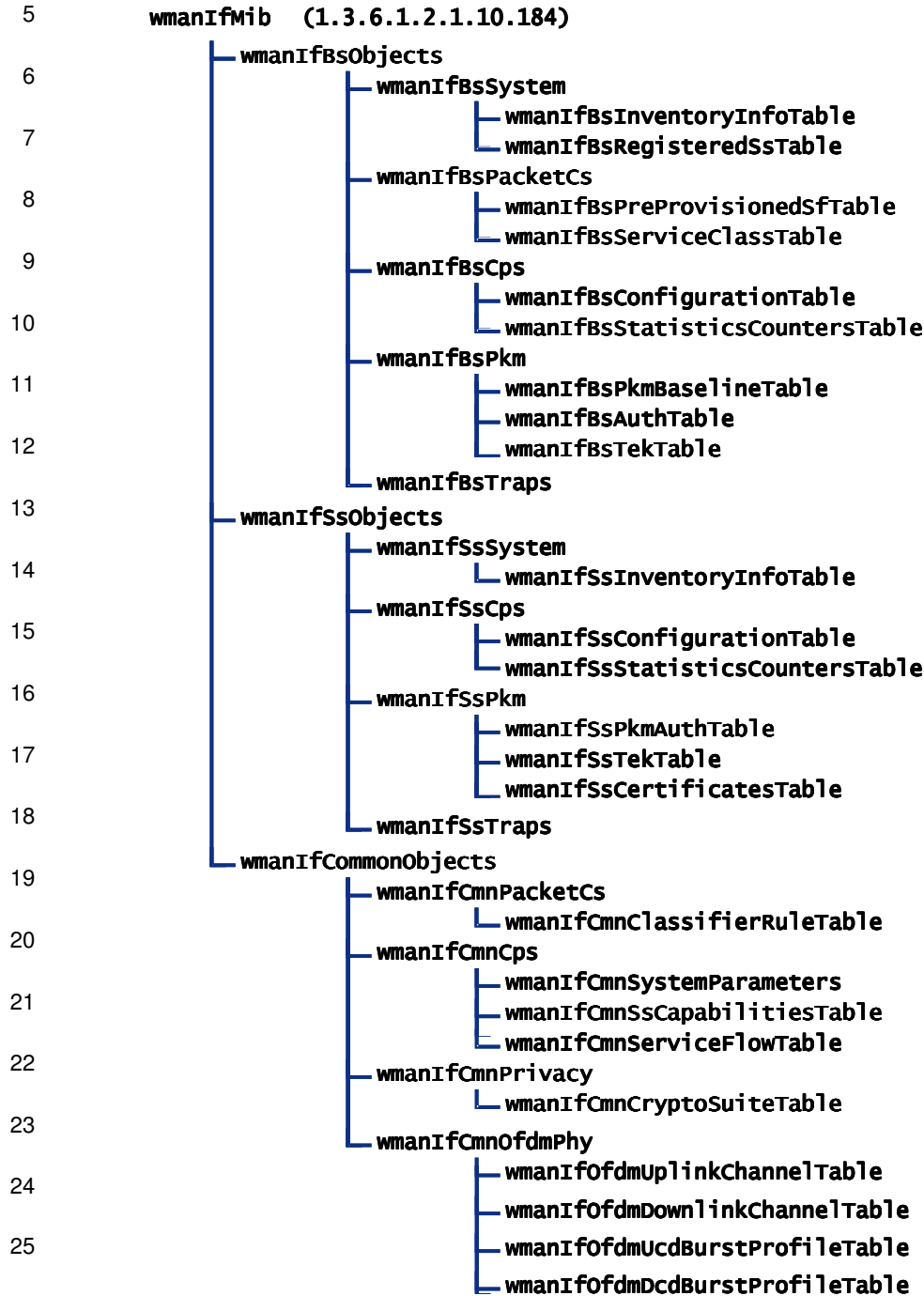


Figure 2 – WirelessMan MIB Structure

- 1 wmanIfMib is composed of three groups:
- 2     ▪ wmanIfBsObjects: This group contains managed objects to be implemented
  - 3         in the SNMP agent in BS.
  - 4     ▪ wmanIfSsObjects: This group contains managed objects to be implemented
  - 5         in the SNMP agent in SS.
  - 6     ▪ wmanIfCommonObjects: This group contains common managed objects to be
  - 7         implemented in the SNMP agent in BS and SS.

## 8 **5.1 wmanIfBsObjects**

### 9 5.1.1 **wmanIfBsSystem**

10 wmanIfBsSystem group contains system level BS managed objects.

#### 11 5.1.1.1 wmanIfBsInventoryInfoTable

12 This table is indexed by BS ifIndex and contains inventory information about the Base  
13 Station such as manufacturer, hardware model, serial number, and software or  
14 firmware revision.

#### 15 5.1.1.2 wmanIfBsRegisteredSsTable

16 This table is indexed by SS ifIndex and contains SS information obtained from REG-  
17 REQ message as defined in section 6.4.3.2.7 in [2]. Each entry in the table may  
18 contain the following objects.

- 19     ▪ Basic CID
- 20     ▪ Primary management CID
- 21     ▪ Secondary Management CID
- 22     ▪ HMAC tuple
- 23     ▪ Uplink CID support
- 24     ▪ SS management support
- 25     ▪ SS capability
- 26     ▪ IP version
- 27     ▪ CS sublayer capabilities

### 28 5.1.2 **wmanIfBsPacketCs**

29 wmanIfBsPacketCs group contains BS managed objects relating to the Packet CS  
30 management entity layer in figure 1 of [3].

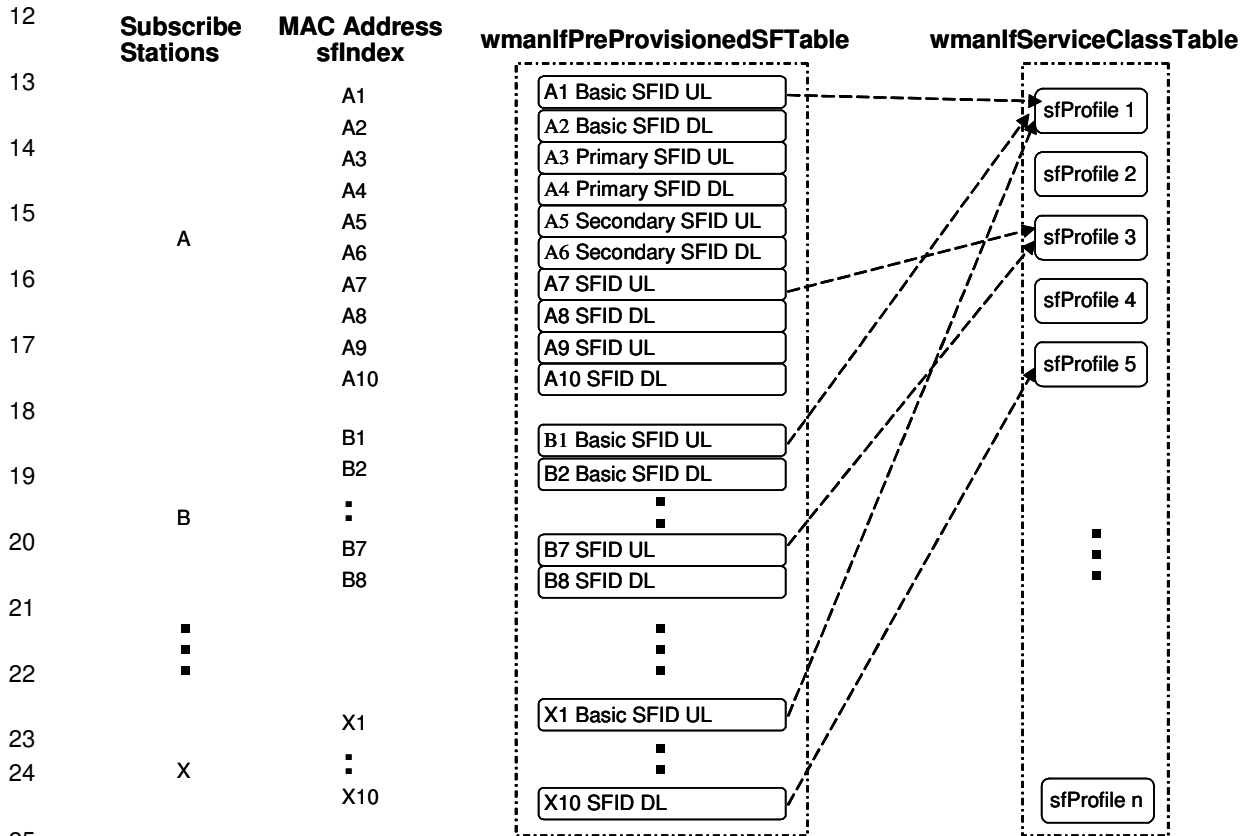
#### 31 5.1.2.1 wmanIfBsPreProvisionedSfTable

32 This table is doubly indexed (SS MAC address, SF ID) and contains pre-provisioned  
33 service flow profiles, Per SS. These connection parameters shall be provisioned for the  
34 SS using DSA messages, as specified in [3], 6.4.13. Admittance and activation of  
35 provisioned service flow may be postponed.

1 5.1.2.2 wmanIfBsServiceClassTable

2 This table is provisioned and is indexed by Service Class ID. Each entry of the table  
 3 contains corresponding service flow characteristic attributes (e.g. QoS parameter set)  
 4 as defined in section 6.4.13.4 in [3].

5 To facilitate the NMS task of provisioning service flow attributes for hundreds or even  
 6 thousands of subscriber stations supported by each BS, the concept of Pre-provisioned  
 7 Service Classes are devised. Figure 3 shows an example of QoS profiles that are  
 8 created to define the service flow attributes that can be shared by multiple service  
 9 flows. For example, Basic CID UL for SSs A1, B1, and X1 uses profile 1. Service flow  
 10 attribute profiles can be added or deleted dynamically to meet different QoS demands  
 11 from subscribers.



26 **Figure 3 – Service Classes – Service Flows Mapping**

27 5.1.3 wmanIfBsCps

28 wmanIfBsCpsParameters group contains BS managed objects relating to the MAC  
 29 CPS management entity layer in figure 1 of [3].

30 5.1.3.1 wmanIfBsConfigurationTable

31 This table contains objects for BS system parameters and constants as defined in  
 32 section 10.1, Table 264 of [3].

- 1           ▪ Base station ID
- 2 5.1.3.2           wmanIfBsStatisticsCountersTable
- 3           This table is indexed by BS ifIndex and contains statistics about the packet count and  
4           various error conditions related to the operation of the BS.
- 5 5.1.4           **wmanIfBsPkm**
- 6           wmanIfBsPkm group contains BS managed objects relating to the MAC CPS privacy  
7           management entity section in figure 1 of [3].
- 8 5.1.4.1           wmanIfBsPkmBaselineTable
- 9           This table is indexed by BS ifIndex and contains base station PKM operational  
10          parameters described in section 10.2 and table 270 of [3].
- 11 5.1.4.2           wmanIfBsAuthTable
- 12          This table is double indexed by ifIndex and SsMacAddress and contains runtime  
13          subscriber station authentication and authorization parameters for each base station.
- 14 5.1.4.3           wmanIfBsTekTable
- 15          This table is double indexed by ifIndex and SAId and contains runtime Security association  
16          parameters for each base station.
- 17 5.1.5           **wmanIfBsTraps**
- 18          wmanIfBsTraps group contains BS traps to report fault events and exceptions, such as  
19          link-up and link-down.
- 20 **5.2 wmanIfSsObjects**
- 21 5.2.1           **wmanSsSystem**
- 22          wmanIfSsSystem group contains subscriber station system level objects.
- 23 5.2.1.1           wmanSsInventoryInfoTable
- 24          This table is indexed by SS MAC address and contains inventory information about the  
25          subscriber station such as manufacturer, hardware model, serial number, and software  
26          or firmware revision.
- 27 5.2.2           **wmanIfSsCps**
- 28          wmanIfSsCpsParameters group contains subscriber station manageable objects  
29          relating to the MAC CPS management entity layer in figure 1 of [3].
- 30 5.2.2.1           wmanIfSsConfigurationTable
- 31          This table is indexed by SS MAC address and contains objects for SS system  
32          parameters and constants as defined in section 10.1, Table 264 of [3].

- 1 5.2.2.2 **wmanlfSsStatisticsCountersTable**  
2 This object contains the performance monitoring data for SS.
- 3 5.2.3 **wmanlfSsPkm**  
4 wmanlfSsPkmParameters group contains subscriber station manageable objects  
5 relating to the MAC CPS privacy management entity section in figure 1 of [3].
- 6 5.2.3.1 **wmanlfSsPkmAuthTable**  
7 This table is indexed by SS MAC address and contains subscriber station  
8 authentication and authorization parameters including those described in section 10.2  
9 and table 270 of [3].
- 10 5.2.3.2 **wmanlfSsPkmTekTable**  
11 This table is doubly indexed by SS MAC address and SAId and contains subscriber  
12 station runtime parameters for each active security association.
- 13 5.2.3.3 **wmanlfSsPkmCertificatesTable**  
14 This table is indexed by SS MAC address and contains subscriber station and SS  
15 manufacturer certificates.
- 16 5.2.4 **wmanlfSsTraps**  
17 wmanlfBsTraps group contains SS traps to report fault events and exceptions, such as  
18 link-up and link-down.
- 19 **5.3 wmanlfCommonObjects**
- 20 5.3.1 **wmanlfCmnPacketCs**
- 21 5.3.1.1 **wmanlfCmnClassifierRuleTable**  
22 wmanlfClassifierRuleTable is indexed by service flow ID and contains runtime classifier  
23 rules screening criteria for each service flow as described in section 11.13.22.3.4 of [3].
- 24 5.3.2 **wmanlfCmnCps**
- 25 5.3.2.1 **wmanlfCmnSystemParameters**  
26 wmanlfCommonSystemParameters contain objects for common BS and SS system  
27 parameters and constants as defined in section 10.1, Table 264 of [3].
- 28 5.3.2.2 **wmanlfCmnSsCapabilitiesTable (index = ifIndex)**  
29 This table is indexed by SS MAC address and negotiated SS capability parameters  
30 and constants as defined in sections 11.4.2.2 and 11.4.2.6 in [3]. In the BS, this table  
31 contains SS capability parameters for all SSs, while in the BS, this table should have  
32 an entry indicating the capability of such SS. In the BS, NMS has read / write access to  
33 this table. The SS capability parameters are provided to SS via SBC-RSP message  
34 during the network entry; hence NMS has only read access to SS.

### 1 5.3.2.3 **wmanIfCmnServiceFlowTable**

2 This table is doubly indexed by ifIndex and service flow ID. In the BS, it represents the  
3 totality of all provisioned, admitted, and active service flow for both DL and UL  
4 directions. In the SS, this table should contain the service flows, both DL and UL, being  
5 allocated to a specific SS.

6 A Service Flow is represented by parameters, such as

- 7     ▪ Service Flow common parameters, like SFID and CID
- 8     ▪ Classifiers associated with Service Flow, see [3] , 5.2.2, 5.2.5 – 5.2.7
- 9     ▪ Service Flow QoS parameters like QoS parameters of specific Service Flow,  
10     like Max Sustained Traffic Rate, QoS status (admitted etc.)
- 11     ▪ Service Flow Header Suppression parameters like associated classifier and  
12     PHS rule, see [3] , 5.2.4

### 13 5.3.3 **wmanIfCmnPrivacy**

#### 14 5.3.3.1 **wmanIfCmnCryptoSuiteTable**

15 This table is doubly indexed by ifIndex and wmanIfCryptoSuiteIndex and contains  
16 supported crypto suites for the particular SS and other crypto parameters such as key  
17 lifetimes. See sections 11.2.14 and 11.2.15 of [3].

### 18 5.3.4 **wmanIfCmnOfdmPhy**

19 wmanIfOfdmPhy is a group containing objects specific to OFDM PHY.

#### 20 5.3.4.1 **wmanIfOfdmUplinkChannelTable**

21 This table contains the uplink channels that the BS is able to receive. In the SS, this  
22 table should have an entry indicating the uplink channel that the SS can transmit. Each  
23 entry contains the parameters needed to describe uplink channel descriptor as defined  
24 in section 11, Table 268 of [3], and include the following objects.

- 25     ▪ Uplink center frequency (KHz)
- 26     ▪ Subchannelization REQ Region-Full Parameters
- 27     ▪ Bandwidth request opportunity size
- 28     ▪ Ranging request opportunity size

#### 29 5.3.4.2 **wmanIfOfdmDownlinkChannelTable**

30 This table contains the downlink channels that the BS is able to transmit. In the SS, this  
31 table should have an entry indicating the downlink channel that the SS can receive.  
32 Each entry contains the parameters needed to describe downlink channel descriptor as  
33 defined in section 11, Table 273 of [3], and including the following.

- 34     ▪ channel number (for license exempt operation only)
- 35     ▪ Frequency (downlink center frequency (kHz))
- 36     ▪ BS EIRP

- 1           ▪ TTG
- 2           ▪ RTG
- 3           ▪ MAC Version

4 5.3.4.3           wmanIfOfdmUcdBurstProfileTable

5           Each entry in this table contains the parameters needed for the UCD burst profile as  
6           defined in section 11, Table 271 of [3].

7 5.3.4.4           wmanIfOfdmDcdBurstProfileTable

8           wmanIfDcdBurstProfileTable – Each entry in this table contains the parameters  
9           needed for the UCD burst profile as defined in section 11, Table 276 of [3].



## 1 **6. ASN.1 Definition of 802.16 MIB**

```

2  WMAN-IF-MIB DEFINITIONS ::= BEGIN
3
4  IMPORTS
5      MODULE-IDENTITY,
6      OBJECT-TYPE,
7      Unsigned32,
8      Integer32,
9      Counter32,
10     Counter64,
11     TimeTicks,
12     IPAddress,
13     transmission
14         FROM SNMPv2-SMI
15     SnmpAdminString
16         FROM SNMP-FRAMEWORK-MIB
17     TEXTUAL-CONVENTION,
18     MacAddress,
19     RowStatus,
20     TruthValue,
21     DateAndTime,
22     DisplayString,
23     TimeInterval,
24     TimeStamp
25         FROM SNMPv2-TC
26     InetAddressType, InetAddress
27         FROM INET-ADDRESS-MIB
28     OBJECT-GROUP,
29
30     MODULE-COMPLIANCE
31         FROM SNMPv2-CONF
32     ifIndex, InterfaceIndexOrZero
33         FROM IF-MIB;
34
35 wmanIfMib MODULE-IDENTITY
36     LAST-UPDATED      "0403110000Z" -- March 11, 2004
37     ORGANIZATION      "IETF IPCDN Working Group"
38     CONTACT-INFO
39         "              Joey Chou
40         Postal: Intel Corporation
41         5000 W. Chandler Blvd, Chandler, AZ 85227, USA
42         E-mail: joey.chou@intel.com
43
44         Russ Reynolds
45         Postal: Proxim Corporation
46         935 Stewart Drive, Sunnyvale, CA 94085, USA
47         E-mail: RReynolds@proxim.com
48
49         Shlomi Eini
50         Postal: Airspan Networks
51         Airport city 70100, Israel

```

```

1           E-mail: seini@airspan.com"
2
3     DESCRIPTION
4       "This MIB Module defines managed objects for 802.16 based
5       Subscriber Station and Base Station."
6       ::= { transmission 184 }
7
8     -- Textual Conventions
9
10    wmanIfMibObjects OBJECT IDENTIFIER ::= { wmanIfMib 1 }
11    wmanIfBsObjects OBJECT IDENTIFIER ::= { wmanIfMibObjects 1 }
12    wmanIfSsObjects OBJECT IDENTIFIER ::= { wmanIfMibObjects 2 }
13    wmanIfCommonObjects OBJECT IDENTIFIER ::= { wmanIfMibObjects 3 }
14
15    --
16    -- BS object group - containing tables and objects to be implemented in
17    -- the Base station
18    --
19    -- wmanIfBsSystem contain the Base Station system objects
20    wmanIfBsSystem OBJECT IDENTIFIER ::= { wmanIfBsObjects 1 }
21
22    wmanIfBsRegisteredSsTable OBJECT-TYPE
23      SYNTAX      SEQUENCE OF wmanIfBsRegisteredSsEntry
24      MAX-ACCESS  not-accessible
25      STATUS      current
26      DESCRIPTION
27        "This table contains entries of Ss that have been
28        registered through REG-REQ message"
29      REFERENCE
30        "Section 6.4.3.2.7 in IEEE 802.16REVd/D3-2004"
31      ::= { wmanIfBsSystem 1 }
32
33    wmanIfBsRegisteredSsEntry OBJECT-TYPE
34      SYNTAX      wmanIfBsRegisteredSsEntry
35      MAX-ACCESS  not-accessible
36      STATUS      current
37      DESCRIPTION
38        "This table provides one row for each SS that has been
39        registered in the BS, and is indexed by SS ifIndex. An
40        entry in this table exists for each ifEntry of SS with an
41        ifType of wmanMac."
42      INDEX { ifIndex }
43      ::= { wmanIfBsRegisteredSsTable 1 }
44
45    wmanIfBsRegisteredSsEntry ::= SEQUENCE {
46      wmanIfBsSsMacAddress      MacAddress,
47      wmanIfBsSsBaseStationIndex  INTEGER,
48      wmanIfBsSsBasicCid        INTEGER,
49      wmanIfBsSsPrimaryCid      INTEGER,
50      wmanIfBsSsSecondaryCid    INTEGER,
51      wmanIfBsHmacTuple         OCTET STRING,
52      wmanIfBsUlcidSupport      INTEGER,
53      wmanIfBsSsManagementSupport  INTEGER,
54      wmanIfBsSsArqSupport      INTEGER,

```

```

1      wmanIfBsSsDsxFwControl      INTEGER,
2      wmanIfBsSsMacCrcSupport     INTEGER,
3      wmanIfBsSsMcaFlowControl    INTEGER,
4      wmanIfBsSsMcpGroupCidSupport INTEGER,
5      wmanIfBsSsPkmFlowControl    INTEGER,
6      wmanIfBsIpVersion           INTEGER,
7      wmanIfBsSsMacCsSupportBitMap BITS,
8      wmanIfBsSsMaxNumOfClassifier INTEGER,
9      wmanIfBsSsPhsSupport        INTEGER
10     }
11
12     wmanIfBsSsMacAddress OBJECT-TYPE
13         SYNTAX      MacAddress
14         MAX-ACCESS  read-only
15         STATUS      current
16         DESCRIPTION
17             "The MAC address of SS is received from the RNG-REQ
18             message. This MAC address can be used as the
19             index to find out the BS and its associated SSS."
20         REFERENCE
21             "Section 6.4.2.3.5 in IEEE 802.16REvD/D3-2004"
22         ::= { wmanIfBsRegisteredSsEntry 1 }
23
24
25     wmanIfBsSsBaseStationIndex OBJECT-TYPE
26         SYNTAX      INTEGER
27         MAX-ACCESS  read-only
28         STATUS      current
29         DESCRIPTION
30             "wmanIfBsSsBaseStationIndex identifies with which BS the
31             SS is associated."
32         REFERENCE
33             "Section 6.4.2.3.5 in IEEE 802.16REvD/D3-2004"
34         ::= { wmanIfBsRegisteredSsEntry 2 }
35
36     wmanIfBsSsBasicCid OBJECT-TYPE
37         SYNTAX      INTEGER
38         MAX-ACCESS  read-only
39         STATUS      current
40         DESCRIPTION
41             "The value of this object indicates the SS's basic CID
42             that was sent in the RNG-RSP message."
43         REFERENCE
44             "Section 6.4.9.5 in IEEE 802.16REvD/D3-2004"
45         ::= { wmanIfBsRegisteredSsEntry 3 }
46
47     wmanIfBsSsPrimaryCid OBJECT-TYPE
48         SYNTAX      INTEGER
49         MAX-ACCESS  read-only
50         STATUS      current
51         DESCRIPTION
52             "The value of this object indicates the basic CID of the
53             SS received from the RNG-RSP message."
54         REFERENCE

```

```

1         "Section 6.4.9.5 in IEEE 802.16REvD/D3-2004"
2         ::= { wmanIfBsRegisteredSsEntry 4 }
3
4     wmanIfBsSsSecondaryCid OBJECT-TYPE
5         SYNTAX      INTEGER
6         MAX-ACCESS  read-only
7         STATUS      current
8         DESCRIPTION
9             "The value of this object indicates the secondary
10            management CID present in the REG-REQ message."
11         REFERENCE
12            "Section 6.4.2.3.8 in IEEE 802.16REvD/D3-2004"
13         ::= { wmanIfBsRegisteredSsEntry 5 }
14
15     wmanIfBsHmacTuple OBJECT-TYPE
16         SYNTAX      OCTET STRING
17         MAX-ACCESS  read-only
18         STATUS      current
19         DESCRIPTION
20            "This parameter contains the HMAC Key Sequence Number
21            concatenated with an HMAC-Digest used for message
22            authentication. The HMAC Key Sequence Number is stored
23            in the four least significant bits of the first byte of
24            the HMAC Tuple, and the most significant four bits
25            are reserved."
26         REFERENCE
27            "Section 11.1.2 in IEEE 802.16REvD/D3-2004"
28         ::= { wmanIfBsRegisteredSsEntry 6 }
29
30     wmanIfBsUlCidSupport OBJECT-TYPE
31         SYNTAX      INTEGER
32         MAX-ACCESS  read-only
33         STATUS      current
34         DESCRIPTION
35            "This object shows the number of Uplink CIDs the SS can
36            support."
37         REFERENCE
38            "Section 11.7.4 in IEEE 802.16REvD/D3-2004"
39         ::= { wmanIfBsRegisteredSsEntry 7 }
40
41     wmanIfBsSsManagementSupport OBJECT-TYPE
42         SYNTAX      INTEGER {unmanagedSs(0),
43                        managedSs(1)}
44         MAX-ACCESS  read-only
45         STATUS      current
46         DESCRIPTION
47            "This object indicates whether or not the SS is managed."
48         REFERENCE
49            "Section 11.7.1.1 in IEEE 802.16REvD/D3-2004"
50         ::= { wmanIfBsRegisteredSsEntry 8 }
51
52     wmanIfBsSsArqSupport OBJECT-TYPE
53         SYNTAX      INTEGER {arqOn(0),
54                        arqOff(1)}

```

```
1      MAX-ACCESS read-only
2      STATUS current
3      DESCRIPTION
4          "This object indicates whether the SS support ARQ."
5      REFERENCE
6          "Section 11.7.6.1 in IEEE 802.16REvd/D3-2004"
7      ::= { wmanIfBsRegisteredSsEntry 9 }
8
9      wmanIfBsSsDsxFwControl OBJECT-TYPE
10     SYNTAX INTEGER (0..255)
11     MAX-ACCESS read-only
12     STATUS current
13     DESCRIPTION
14         "This object specifies the maximum number of concurrent
15         DSA, DSC, or DSD transactions that may be outstanding."
16     REFERENCE
17         "Section 11.7.6.2 in IEEE 802.16REvd/D3-2004"
18     ::= { wmanIfBsRegisteredSsEntry 10 }
19
20     wmanIfBsSsMacCrcSupport OBJECT-TYPE
21     SYNTAX INTEGER {noMacCrcSupport(0),
22                 macCrcSupport(1)}
23     MAX-ACCESS read-only
24     STATUS current
25     DESCRIPTION
26         "This object indicates whether or not the SS supports MAC
27         level CRC."
28     REFERENCE
29         "Section 11.7.6.3 in IEEE 802.16REvd/D3-2004"
30     ::= { wmanIfBsRegisteredSsEntry 11 }
31
32     wmanIfBsSsMcaFlowControl OBJECT-TYPE
33     SYNTAX INTEGER (0..255)
34     MAX-ACCESS read-only
35     STATUS current
36     DESCRIPTION
37         "This object specifies the maximum number of concurrent
38         MCA transactions that may be outstanding."
39     REFERENCE
40         "Section 11.7.6.4 in IEEE 802.16REvd/D3-2004"
41     ::= { wmanIfBsRegisteredSsEntry 12 }
42
43     wmanIfBsSsMcpGroupCidSupport OBJECT-TYPE
44     SYNTAX INTEGER (0..255)
45     MAX-ACCESS read-only
46     STATUS current
47     DESCRIPTION
48         "This object indicates the maximum number of
49         simultaneous Multicast Polling Groups the SS is
50         capable of belonging to."
51     REFERENCE
52         "Section 11.7.6.5 in IEEE 802.16REvd/D3-2004"
53     ::= { wmanIfBsRegisteredSsEntry 13 }
54
```

```

1  wmanIfBsSsPkmFlowControl OBJECT-TYPE
2      SYNTAX      INTEGER (0..255)
3      MAX-ACCESS  read-only
4      STATUS      current
5      DESCRIPTION
6          "This object specifies the maximum number of concurrent PKM
7          transactions that may be outstanding."
8      REFERENCE
9          "Section 11.7.6.6 in IEEE 802.16REvD/D3-2004"
10     ::= { wmanIfBsRegisteredSsEntry 14 }
11
12  wmanIfBsIpVersion OBJECT-TYPE
13      SYNTAX      INTEGER {ipv4(1),
14                      ipv6(2)}
15      MAX-ACCESS  read-only
16      STATUS      current
17      DESCRIPTION
18          "This object indicates the version of IP used on the
19          Secondary Management Connection."
20      REFERENCE
21          "Section 11.7.2.1 in IEEE 802.16REvD/D3-2004"
22     ::= { wmanIfBsRegisteredSsEntry 15 }
23
24  wmanIfBsSSMacCsSupportBitMap OBJECT-TYPE
25      SYNTAX      BITS {atm(0),
26                      packetIpv4(1),
27                      packetIpv6(2),
28                      packet802-3(3),
29                      packet802-1Q(4),
30                      packetIpv4Over802-3(5),
31                      packetIpv6Over802-3(6),
32                      packetIpv4Over802-1Q(7),
33                      packetIpv6Over802-1Q(8)}
34      MAX-ACCESS  read-only
35      STATUS      current
36      DESCRIPTION
37          "This object indicates the set of MAC convergence
38          sublayer support. When a bit is set, it indicates
39          the corresponding CS feature is supported."
40      REFERENCE
41          "Section 11.7.5.1 in IEEE 802.16REvD/D3-2004"
42     ::= { wmanIfBsRegisteredSsEntry 16 }
43
44  wmanIfBsSSMaxNumOfClassifier OBJECT-TYPE
45      SYNTAX      INTEGER
46      MAX-ACCESS  read-only
47      STATUS      current
48      DESCRIPTION
49          "This object indicates the maximum number of admitted
50          Classifiers that the SS is allowed to have."
51      REFERENCE
52          "Section 11.7.5.2 in IEEE 802.16REvD/D3-2004"
53     ::= { wmanIfBsRegisteredSsEntry 17 }
54

```

```

1
2 wmanIfBsSSPhsSupport OBJECT-TYPE
3     SYNTAX      INTEGER {noPhsSupport(0),
4                   atmPhsSupport(1),
5                   packetPhsSupport(2)}
6     MAX-ACCESS  read-only
7     STATUS      current
8     DESCRIPTION
9         "This object indicates indicates the level of PHS support."
10    REFERENCE
11        "Section 11.7.5.3 in IEEE 802.16REvd/D3-2004"
12    ::= { wmanIfBsRegisteredSsEntry 18 }
13
14 --
15 -- wmanIfBsPacketCs contain the Base Station Packet Convergence Sublayer
16 -- objects
17 wmanIfBsPacketCs OBJECT IDENTIFIER ::= { wmanIfBsObjects 2 }
18
19 wmanUlschedulingType ::= TEXTUAL-CONVENTION
20     STATUS      current
21     DESCRIPTION
22         "The scheduling service provided by a SC for an
23         upstream service flow. If the parameter is omitted
24         from an upstream QOS Parameter Set, this object takes
25         the value of bestEffort (2). This parameter must be
26         reported as undefined (1) for downstream QOS Parameter
27         Sets."
28     SYNTAX      INTEGER {undefined(1),
29                   bestEffort(2),
30                   nonRealTimePollingService(3),
31                   realTimePollingService(4),
32                   unsolicitedGrantService(6)}
33
34 wmanIfBsPreProvisionedSfTable OBJECT-TYPE
35     SYNTAX      SEQUENCE OF wmanIfBsPreProvisionedSfEntry
36     MAX-ACCESS  not-accessible
37     STATUS      current
38     DESCRIPTION
39         "This table is doubly indexed (SS MAC address, SF ID) and
40         contains pre-provisioned service flow profiles, Per SS.
41         These connection parameters shall be provisioned for the SS
42         using DSA messages. NMS shall pre-provisioning the service
43         class table - wmanIfBsServiceClassTable by using
44         wmanIfBsServiceClassIndex, and packet classifier rule table
45         - wmanIfBsClassifierRuleTable by using wmanIfBSSfId"
46     REFERENCE
47         "Section 6.4.13 in IEEE 802.16REvd/D3-2004"
48     ::= { wmanIfBsPacketCs 1 }
49
50 wmanIfBsPreProvisionedSfEntry OBJECT-TYPE
51     SYNTAX      wmanIfBsPreProvisionedSfEntry
52     MAX-ACCESS  not-accessible
53     STATUS      current
54     DESCRIPTION

```

```

1           "This table provides one row for each service flow been
2           pre-provisioned by NMS."
3     INDEX { wmanIfBsSsProvMacAddress, wmanIfBsSfId}
4     ::= { wmanIfBsPreProvisionedSfTable 1 }
5
6     wmanIfBsPreProvisionedSfEntry ::= SEQUENCE {
7         wmanIfBsSfId                INTEGER,
8         wmanIfBsSfDirection          INTEGER,
9         wmanIfBsServiceClassIndex    INTEGER,
10        wmanIfBsSsProvMacAddress      MacAddress,
11        wmanIfBsSfState               INTEGER,
12        wmanIfBsSfCreateTime          TimeStamp,
13        wmanIfBsPreProvisionedSfRowStatus RowStatus
14    }
15
16    wmanIfBsSfId OBJECT-TYPE
17        SYNTAX      INTEGER
18        MAX-ACCESS  read-only
19        STATUS      current
20        DESCRIPTION
21            "A 32 bit quantity that uniquely identifies a service flow
22             to both the subscriber station and base station (BS)."
23        ::= { wmanIfBsPreProvisionedSfEntry 1 }
24
25    wmanIfBsSfDirection OBJECT-TYPE
26        SYNTAX      INTEGER {downstream(1),
27                        upstream(2)}
28        MAX-ACCESS  read-write
29        STATUS      current
30        DESCRIPTION
31            "An attribute indicating the service flow is downstream or
32             upstream."
33        ::= { wmanIfBsPreProvisionedSfEntry 2 }
34
35    wmanIfBsServiceClassIndex OBJECT-TYPE
36        SYNTAX      INTEGER
37        MAX-ACCESS  read-write
38        STATUS      current
39        DESCRIPTION
40            "The index in wmanIfBsServiceClassTable describing the
41             service class or QoS parameters for such service flow.
42             If no associated entry in wmanIfBsServiceClassTable
43             exists, this object returns a value of zero."
44        ::= { wmanIfBsPreProvisionedSfEntry 3 }
45
46    wmanIfBsSsProvMacAddress OBJECT-TYPE
47        SYNTAX      MacAddress
48        MAX-ACCESS  read-write
49        STATUS      current
50        DESCRIPTION
51            "The MAC address of the SS, where the service flow resides.
52             It can be used as the index to associate service flows
53             with the SS."
54        ::= { wmanIfBsPreProvisionedSfEntry 4 }

```



```

1
2 wmanIfBsSfState OBJECT-TYPE
3     SYNTAX      INTEGER {provisionedState(1),
4                   admittedState(2),
5                   activeState(3)}
6     MAX-ACCESS  read-write
7     STATUS      current
8     DESCRIPTION
9         "wmanIfBsSfState determines how the service flow will be
10        transitioned to the Admitted or Active state.
11        Admitted or Active state: The pre-provisioned service flow
12        will be transitioned to the Admitted or Active state, as
13        soon as the SS passes the network entry procedure, and
14        connection admission control. An entry will be created
15        in the SS and BS service flow tables.
16        Provisioned state: After SS enters the network; the
17        pre-provisioned service flow will remain in the Provisioned
18        state until NMS set it different state. An entry will be
19        created in the SS and BS service flow tables"
20     REFERENCE
21         "Section 6.4.13.6, in IEEE 802.16REvD/D3-2004"
22     ::= { wmanIfBsPreProvisionedSfEntry 5 }
23
24 wmanIfBsSfCreateTime OBJECT-TYPE
25     SYNTAX      TimeStamp
26     MAX-ACCESS  read-write
27     STATUS      current
28     DESCRIPTION
29         "Indicates the data and time when the service flow is
30        created."
31     ::= { wmanIfBsPreProvisionedSfEntry 6 }
32
33 wmanIfBsPreProvisionedSfRowStatus OBJECT-TYPE
34     SYNTAX      RowStatus
35     MAX-ACCESS  read-create
36     STATUS      current
37     DESCRIPTION
38         "This object is used to create a new row or modify or
39        delete an existing row in this table.
40
41        If the implementator of this MIB has chosen not
42        to implement 'dynamic assignment' of profiles, this
43        object is not useful and should return noSuchName
44        upon SNMP request."
45     ::= { wmanIfBsPreProvisionedSfEntry 7 }
46
47 wmanIfBsServiceClassTable OBJECT-TYPE
48     SYNTAX      SEQUENCE OF wmanIfBsServiceClassEntry
49     MAX-ACCESS  not-accessible
50     STATUS      current
51     DESCRIPTION
52         "This table is provisioned and is indexed by Service Class
53        ID. Each entry of the table contains corresponding service
54        flow characteristic attributes (e.g. QoS parameter set)"

```

```

1      REFERENCE
2      "Section 6.4.13.4 in IEEE 802.16REvd/D3-2004"
3      ::= { wmanIfBsPacketCs 2 }
4
5      wmanIfBsServiceClassEntry OBJECT-TYPE
6          SYNTAX      WmanIfBsServiceClassEntry
7          MAX-ACCESS  not-accessible
8          STATUS      current
9          DESCRIPTION
10             "This table provides one row for each service class"
11             INDEX { wmanIfBsQoSProfileIndex }
12             ::= { wmanIfBsServiceClassTable 1 }
13
14      wmanIfBsServiceClassEntry ::= SEQUENCE {
15          wmanIfBsQoSProfileIndex      INTEGER,
16          wmanIfBsQoSServiceClassName  DisplayString,
17          wmanIfBsQoSTrafficPriority    INTEGER,
18          wmanIfBsQoSMaxSustainedRate  INTEGER,
19          wmanIfBsQoSMaxTrafficBurst    INTEGER,
20          wmanIfBsQoSMinReservedRate   INTEGER,
21          wmanIfBsQoSToToleratedJitter INTEGER,
22          wmanIfBsQoSMaxLatency         INTEGER,
23          wmanIfBsQoSScSchedulingType   wmanUlschedulingType,
24          wmanIfBsQoSScArqEnable        TruthValue,
25          wmanIfBsQoSScArqWindowSize    INTEGER,
26          wmanIfBsQoSScArqFragmentLifetime  INTEGER,
27          wmanIfBsQoSScArqSyncLossTimeout  INTEGER,
28          wmanIfBsQoSScArqDeliverInOrder  TruthValue,
29          wmanIfBsQoSScArqRxPurgeTimeout  INTEGER,
30          wmanIfBsQoSScFragmentLen      INTEGER,
31          wmanIfBsQoSScMinRsvdTolerableRate  INTEGER,
32          wmanIfBsQoSServiceClassRowStatus  RowStatus
33      }
34
35      wmanIfBsQoSProfileIndex OBJECT-TYPE
36          SYNTAX      INTEGER
37          MAX-ACCESS  not-accessible
38          STATUS      current
39          DESCRIPTION
40             "The index value which uniquely identifies an entry
41             in the wmanIfBsServiceClassTable"
42             ::= { wmanIfBsServiceClassEntry 1 }
43
44      wmanIfBsQoSServiceClassName OBJECT-TYPE
45          SYNTAX      DisplayString
46          MAX-ACCESS  read-create
47          STATUS      current
48          DESCRIPTION "Refers to the Service Class Name"
49          REFERENCE   "802.16 Chapter 11.4.8"
50          ::= { wmanIfBsServiceClassEntry 2 }
51
52      wmanIfBsQoSTrafficPriority OBJECT-TYPE
53          SYNTAX      INTEGER
54          MAX-ACCESS  read-create

```

```

1      STATUS      current
2      DESCRIPTION
3          "The value of this parameter specifies the priority
4          assigned to a service flow. For uplink service flows,
5          the BS should use this parameter when determining
6          precedence in request service and grant generation,
7          and the SS shall preferentially select contention
8          Request opportunities for Priority Request CIDs
9          based on this priority"
10     REFERENCE
11         "Section 11.13.7 in IEEE 802.16REVd/D3-2004"
12     ::= { wmanIfBsServiceClassEntry 3 }
13
14     wmanIfBsQoSMaxSustainedRate OBJECT-TYPE
15         SYNTAX      INTEGER
16         MAX-ACCESS  read-create
17         STATUS      current
18         DESCRIPTION
19             "This parameter defines the peak information rate
20             of the service. The rate is expressed in bits per
21             second and pertains to the SDUs at the input to
22             the system."
23         REFERENCE
24             "Section 11.13.8 in IEEE 802.16REVd/D3-2004"
25     ::= { wmanIfBsServiceClassEntry 4 }
26
27     wmanIfBsQoSMaxTrafficBurst OBJECT-TYPE
28         SYNTAX      INTEGER
29         MAX-ACCESS  read-create
30         STATUS      current
31         DESCRIPTION
32             "This parameter defines the maximum burst size that
33             must be accommodated for the service."
34         REFERENCE
35             "Section 11.13.9 in IEEE 802.16REVd/D3-2004"
36     ::= { wmanIfBsServiceClassEntry 5 }
37
38     wmanIfBsQoSMinReservedRate OBJECT-TYPE
39         SYNTAX      INTEGER
40         MAX-ACCESS  read-create
41         STATUS      current
42         DESCRIPTION
43             "This parameter specifies the minimum rate reserved
44             for this service flow."
45         REFERENCE
46             "Section 11.13.10 in IEEE 802.16REVd/D3-2004"
47     ::= { wmanIfBsServiceClassEntry 6 }
48
49     wmanIfBsQoSToLeratedJitter OBJECT-TYPE
50         SYNTAX      INTEGER
51         MAX-ACCESS  read-create
52         STATUS      current
53         DESCRIPTION
54             "This parameter defines the Maximum delay

```

```

1         variation (jitter) for the connection."
2     REFERENCE
3         "Section 11.13.15 in IEEE 802.16REvd/D3-2004"
4     ::= { wmanIfBsServiceClassEntry 7 }
5
6     wmanIfBsQoSMaxLatency OBJECT-TYPE
7         SYNTAX      INTEGER
8         MAX-ACCESS  read-create
9         STATUS      current
10        DESCRIPTION
11            "The value of this parameter specifies the maximum
12            latency between the reception of a packet by the BS
13            or SS on its network interface and the forwarding
14            of the packet to its RF Interface."
15        REFERENCE
16            "Section 11.13.16 in IEEE 802.16REvd/D3-2004"
17        ::= { wmanIfBsServiceClassEntry 8 }
18
19        wmanIfBsQoSScSchedulingType OBJECT-TYPE
20            SYNTAX      wmanUlSchedulingType
21            MAX-ACCESS  read-create
22            STATUS      current
23            DESCRIPTION
24                "Specifies the upstream scheduling service used for
25                upstream service flow. If the referenced parameter
26                is not present in the corresponding 802.16 QoS
27                Parameter Set of an upstream service flow, the
28                default value of this object is bestEffort(2)."
29            REFERENCE      "802.16 Chapter 11.4.8"
30            DEFVAL        {2}
31            ::= { wmanIfBsServiceClassEntry 9}
32
33        wmanIfBsQoSScArqEnable OBJECT-TYPE
34            SYNTAX      TruthValue
35            MAX-ACCESS  read-create
36            STATUS      current
37            DESCRIPTION
38                "True(1) ARQ enabling is requested for the connection."
39            ::= { wmanIfBsServiceClassEntry 10}
40
41        wmanIfBsQoSScArqWindowSize OBJECT-TYPE
42            SYNTAX      INTEGER (1 .. 255)
43            MAX-ACCESS  read-create
44            STATUS      current
45            DESCRIPTION
46                "Indicates the maximum number of unacknowledged
47                fragments at any time."
48            ::= { wmanIfBsServiceClassEntry 11}
49
50        wmanIfBsQoSScArqFragmentLifetime OBJECT-TYPE
51            SYNTAX      INTEGER (0 .. 65535)
52            UNITS        "10 us"
53            MAX-ACCESS  read-create
54            STATUS      current

```

```

1      DESCRIPTION
2          "The maximum time interval an ARQ fragment will be
3          managed by the transmitter ARQ machine, once
4          initial transmission of the fragment has occurred.
5          If transmission or retransmission of the fragment
6          is not acknowledged by the receiver before the
7          time limit is reached, the fragmnet is discarded.
8          A value of 0 means Infinite."
9      DEFVAL      {0}
10     ::= { wmanIfBsServiceClassEntry 12 }
11
12     wmanIfBsQosScArqSyncLosTimeout OBJECT-TYPE
13         SYNTAX      INTEGER (0 .. 65535 )
14         UNITS       "10 us"
15         MAX-ACCESS  read-create
16         STATUS      current
17         DESCRIPTION
18             "The maximum interval before declaring a loss
19             of synchronization of the sender and receiver
20             state machines. A value of 0 means Infinite."
21         DEFVAL      {0}
22         ::= { wmanIfBsServiceClassEntry 13}
23
24     wmanIfBsQosScArqDeliverInOrder OBJECT-TYPE
25         SYNTAX      TruthValue
26         MAX-ACCESS  read-create
27         STATUS      current
28         DESCRIPTION
29             "Indicates whether or not data is to be delivered
30             by the receiving MAC to its client application
31             in the order in which data was handed off to the
32             originating MAC."
33         ::= { wmanIfBsServiceClassEntry 14 }
34
35     wmanIfBsQosScArqRXPurgeTimeout OBJECT-TYPE
36         SYNTAX      INTEGER (0 .. 65535)
37         UNITS       "10 us"
38         MAX-ACCESS  read-create
39         STATUS      current
40         DESCRIPTION
41             "Indicates the time interval the ARQ window is advanced
42             after a fragment is received. A value of 0 means
43             Infinite."
44         DEFVAL      {0}
45         ::= { wmanIfBsServiceClassEntry 15}
46
47     wmanIfBsQosScFragmentLen OBJECT-TYPE
48         SYNTAX      INTEGER (32 .. 2040)
49         MAX-ACCESS  read-create
50         STATUS      current
51         DESCRIPTION
52             "The maximum size fragment a transmitter shall form
53             or a receiver shall expect to receive."
54         ::= { wmanIfBsServiceClassEntry 16 }

```

```

1
2  wmanIfBsQoSsCMinRsvdTolerableRate OBJECT-TYPE
3      SYNTAX      INTEGER
4      MAX-ACCESS  read-create
5      STATUS      current
6      DESCRIPTION
7          "Minimum Tolerable Traffic Rate = R (bits/sec) with
8          time base T(sec) means the following. Let S denote
9          additional demand accumulated at the MAC SAP of the
10         transmitter during an arbitrary time interval of the
11         length T. Then the amount of data forwarded at the
12         receiver to CS (in bits) during this interval should
13         be not less than min {S, R * T}."
14     REFERENCE  "Section 11.13.11 in IEEE 802.16REvd/D3-2004"
15     ::= { wmanIfBsServiceClassEntry 17 }
16
17  wmanIfBsQoSServiceClassRowStatus OBJECT-TYPE
18      SYNTAX      RowStatus
19      MAX-ACCESS  read-create
20      STATUS      current
21      DESCRIPTION
22          "This object is used to create a new row or modify or
23          delete an existing row in this table.
24
25          If the implementator of this MIB has chosen not
26          to implement 'dynamic assignment' of profiles, this
27          object is not useful and should return noSuchName
28          upon SNMP request."
29     ::= { wmanIfBsServiceClassEntry 18 }
30
31  wmanIfBsClassifierRuleTable OBJECT-TYPE
32      SYNTAX      SEQUENCE OF wmanIfBsClassifierRuleEntry
33      MAX-ACCESS  not-accessible
34      STATUS      current
35      DESCRIPTION
36          "This table contains packet classifier rules associated
37          with service flows."
38     ::= { wmanIfBsPacketCs 3 }
39
40  wmanIfBsClassifierRuleEntry OBJECT-TYPE
41      SYNTAX      wmanIfBsClassifierRuleEntry
42      MAX-ACCESS  not-accessible
43      STATUS      current
44      DESCRIPTION
45          "This table provides one row for each packet classifier
46          rule, and is indexed by wmanIfBsSfId and
47          wmanIfBsClassifierRuleIndex. wmanIfBsSfId
48          identifies the service flow, while
49          wmanIfBsClassifierRuleIndex identifies the packet
50          classifier rule."
51     INDEX { wmanIfBsSfId, wmanIfBsClassifierRuleIndex }
52     ::= { wmanIfBsClassifierRuleTable 1 }
53
54  wmanIfBsClassifierRuleEntry ::= SEQUENCE {

```

```

1      wmanIfBsClassifierRuleIndex          Unsigned32,
2      wmanIfBsClassifierRuleServiceFlowId Unsigned32,
3      wmanIfBsClassifierRulePriority        INTEGER,
4      wmanIfBsClassifierRuleIpTosLow       OCTET STRING,
5      wmanIfBsClassifierRuleIpTosHigh      OCTET STRING,
6      wmanIfBsClassifierRuleIpTosMask      OCTET STRING,
7      wmanIfBsClassifierRuleIpProtocol     Integer32,
8      wmanIfBsClassifierRuleInetAddressType InetAddressType,
9      wmanIfBsClassifierRuleInetAddress     InetAddress,
10     wmanIfBsClassifierRuleInetAddressMask InetAddress,
11     wmanIfBsClassifierRuleInetAddress     InetAddress,
12     wmanIfBsClassifierRuleInetAddressMask InetAddress,
13     wmanIfBsClassifierRuleSourcePortStart Integer32,
14     wmanIfBsClassifierRuleSourcePortEnd   Integer32,
15     wmanIfBsClassifierRuleDestPortStart   Integer32,
16     wmanIfBsClassifierRuleDestPortEnd     Integer32,
17     wmanIfBsClassifierRuleDestMacAddr     MacAddress,
18     wmanIfBsClassifierRuleDestMacMask    MacAddress,
19     wmanIfBsClassifierRuleSourceMacAddr   MacAddress,
20     wmanIfBsClassifierRuleSourceMacMask   MacAddress,
21     wmanIfBsClassifierRuleEnetProtocolType INTEGER,
22     wmanIfBsClassifierRuleEnetProtocol   Integer32,
23     wmanIfBsClassifierRuleUserPriLow      Integer32,
24     wmanIfBsClassifierRuleUserPriHigh    Integer32,
25     wmanIfBsClassifierRuleVlanId         Integer32,
26     wmanIfBsClassifierRuleState          INTEGER,
27     wmanIfBsClassifierRulePkts           Counter64,
28     wmanIfBsClassifierRuleRowStatus      RowStatus
29     }
30
31     wmanIfBsClassifierRuleIndex OBJECT-TYPE
32         SYNTAX      Unsigned32 (1..4294967295)
33         MAX-ACCESS  not-accessible
34         STATUS      current
35         DESCRIPTION
36             "An index is assigned to a classifier in BS classifiers
37             table"
38         REFERENCE   ""
39         ::= { wmanIfBsClassifierRuleEntry 1 }
40
41
42     wmanIfBsClassifierRuleServiceFlowId OBJECT-TYPE
43         SYNTAX      Unsigned32 (1..4294967295)
44         MAX-ACCESS  read-write
45         STATUS      current
46         DESCRIPTION
47             "An index assigned to a service flow by SC (SFID)."

```

```

1      DESCRIPTION
2          "The value specifies the order of evaluation of the
3          classifiers. The higher the value the higher the
4          priority. The value of 0 is used as default in
5          provisioned service flows classifiers. The default
6          value of 64 is used for dynamic service flow classifiers.
7          If the referenced parameter is not present in a classifier,
8          this object reports the default value as defined above"
9      ::= { wmanIfBsClassifierRuleEntry 3 }
10
11  wmanIfBsClassifierRuleIpTosLow OBJECT-TYPE
12      SYNTAX      OCTET STRING (SIZE(1))
13      MAX-ACCESS  read-write
14      STATUS      current
15      DESCRIPTION
16          "The low value of a range of TOS byte values. If the
17          referenced parameter is not present in a classifier, this
18          object reports the value of 0."
19      REFERENCE   "802.16 Chapter 11.4.9"
20      ::= { wmanIfBsClassifierRuleEntry 4 }
21
22  wmanIfBsClassifierRuleIpTosHigh OBJECT-TYPE
23      SYNTAX      OCTET STRING (SIZE(1))
24      MAX-ACCESS  read-write
25      STATUS      current
26      DESCRIPTION
27          "The 8-bit high value of a range of TOS byte values.
28          If the referenced parameter is not present in a classifier,
29          this object reports the value of 0."
30      REFERENCE   "802.16 Chapter 11.4.9"
31      ::= { wmanIfBsClassifierRuleEntry 5 }
32
33  wmanIfBsClassifierRuleIpTosMask OBJECT-TYPE
34      SYNTAX      OCTET STRING (SIZE(1))
35      MAX-ACCESS  read-write
36      STATUS      current
37      DESCRIPTION
38          "The mask value is bitwise ANDed with TOS byte in an IP
39          packet and this value is used check range checking of
40          TosLow and TosHigh. If the referenced parameter is not
41          present in a classifier, this object reports the value
42          of 0."
43      REFERENCE   "802.16 Chapter 11.4.9"
44      ::= { wmanIfBsClassifierRuleEntry 6 }
45
46  wmanIfBsClassifierRuleIpProtocol OBJECT-TYPE
47      SYNTAX      Integer32 (0..255)
48      MAX-ACCESS  read-write
49      STATUS      current
50      DESCRIPTION
51          "This object indicates the value of the IP Protocol field
52          required for IP packets to match this rule. If the
53          referenced parameter is not present in a classifier, this
54          object reports the value of 0."

```



```

1      REFERENCE      "802.16 Chapter 11.4.9"
2      ::= { wmanIfBsClassifierRuleEntry 7 }
3
4      wmanIfBsClassifierRuleInetAddressType OBJECT-TYPE
5          SYNTAX      InetAddressType
6          MAX-ACCESS  read-write
7          STATUS      current
8          DESCRIPTION
9              "The type of the internet address for
10             wmanIfBsClassifierRuleInetAddressSourceAddr,
11             wmanIfBsClassifierRuleInetAddressSourceMask,
12             wmanIfBsClassifierRuleInetAddressDestAddr, and
13             wmanIfBsClassifierRuleInetAddressDestMask.
14             If the referenced parameter is not present in a classifier,
15             this object reports the value of ipv4(1)."

```

```

1      DESCRIPTION
2          "This object specifies the value of the IP Destination
3          Address required for packets to match this rule. An IP
4          packet matches the rule when the packet IP destination
5          address bitwise ANDed with the
6          wmanIfBsClassifierRuleInetDestMask value equals the
7          wmanIfBsClassifierRuleInetDestAddr value.
8          If the referenced parameter is not present in a
9          classifier, this object reports the value of 0.0.0.0."
10     REFERENCE      "802.16 Chapter 11.4.9"
11     ::= { wmanIfBsClassifierRuleEntry 11 }
12
13     wmanIfBsClassifierRuleInetDestMask OBJECT-TYPE
14         SYNTAX      InetAddress
15         MAX-ACCESS  read-write
16         STATUS      current
17         DESCRIPTION
18             "This object specifies which bits of a packet's IP
19             Destination Address that are compared to match this rule.
20             An IP packet matches the rule when the packet destination
21             address bitwise ANDed with the
22             wmanIfBsClassifierRuleInetDestMask value equals the
23             wmanIfBsClassifierRuleInetDestAddr value.
24             If the referenced parameter is not present in a classifier
25             , this object reports the value of 0.0.0.0."
26         REFERENCE  "802.16 Chapter 11.4.9"
27         ::= { wmanIfBsClassifierRuleEntry 12}
28
29     wmanIfBsClassifierRuleSourcePortStart OBJECT-TYPE
30         SYNTAX      Integer32 (0..65535)
31         MAX-ACCESS  read-write
32         STATUS      current
33         DESCRIPTION
34             "This object specifies the low end inclusive range of
35             TCP/UDP source port numbers to which a packet is compared
36             . This object is irrelevant for non-TCP/UDP IP packets.
37             If the referenced parameter is not present in a
38             classifier, this object reports the value of 0."
39         REFERENCE  "802.16 Chapter 11.4.9"
40         ::= { wmanIfBsClassifierRuleEntry 13 }
41
42     wmanIfBsClassifierRuleSourcePortEnd OBJECT-TYPE
43         SYNTAX      Integer32 (0..65535)
44         MAX-ACCESS  read-write
45         STATUS      current
46         DESCRIPTION
47             "This object specifies the high end inclusive range of
48             TCP/UDP source port numbers to which a packet is compared.
49             This object is irrelevant for non-TCP/UDP IP packets.
50             If the referenced parameter is not present in a classifier,
51             this object reports the value of 65535."
52         REFERENCE  "802.16 Chapter 11.4.9"
53         ::= { wmanIfBsClassifierRuleEntry 14 }
54

```

```

1  wmanIfBsClassifierRuleDestPortStart OBJECT-TYPE
2      SYNTAX      Integer32 (0..65535)
3      MAX-ACCESS  read-write
4      STATUS      current
5      DESCRIPTION
6          "This object specifies the low end inclusive range of
7          TCP/UDP destination port numbers to which a packet is
8          compared. If the referenced parameter is not present
9          in a classifier, this object reports the value of 0."
10     REFERENCE   "802.16 Chapter 11.4.9"
11     ::= { wmanIfBsClassifierRuleEntry 15 }
12
13  wmanIfBsClassifierRuleDestPortEnd OBJECT-TYPE
14     SYNTAX      Integer32 (0..65535)
15     MAX-ACCESS  read-write
16     STATUS      current
17     DESCRIPTION
18         "This object specifies the high end inclusive range of
19         TCP/UDP destination port numbers to which a packet is
20         compared. If the referenced parameter is not present
21         in a classifier, this object reports the value of
22         65535."
23     REFERENCE   "802.16 Chapter 11.4.9"
24     ::= { wmanIfBsClassifierRuleEntry 16 }
25
26  wmanIfBsClassifierRuleDestMacAddr OBJECT-TYPE
27     SYNTAX      MacAddress
28     MAX-ACCESS  read-write
29     STATUS      current
30     DESCRIPTION
31         "An Ethernet packet matches an entry when its destination
32         MAC address bitwise ANDed with
33         wmanIfBsClassifierRuleDestMacMask equals the value of
34         wmanIfBsClassifierRuleDestMacAddr. If the referenced
35         parameter is not present in a classifier, this object
36         reports the value of '000000000000'H."
37     REFERENCE   "802.16 Chapter 11.4.9"
38     ::= { wmanIfBsClassifierRuleEntry 17 }
39
40  wmanIfBsClassifierRuleDestMacMask OBJECT-TYPE
41     SYNTAX      MacAddress
42     MAX-ACCESS  read-write
43     STATUS      current
44     DESCRIPTION
45         "An Ethernet packet matches an entry when its destination
46         MAC address bitwise ANDed with
47         wmanIfBsClassifierRuleDestMacMask equals the value of
48         wmanIfBsClassifierRuleDestMacAddr. If the referenced
49         parameter is not present in a classifier, this object
50         reports the value of '000000000000'H."
51     REFERENCE   "802.16 Chapter 11.4.9"
52     ::= { wmanIfBsClassifierRuleEntry 18 }
53
54  wmanIfBsClassifierRuleSourceMacAddr OBJECT-TYPE

```

```

1      SYNTAX      MacAddress
2      MAX-ACCESS  read-write
3      STATUS      current
4      DESCRIPTION
5          "An Ethernet packet matches this entry when its source
6          MAC address bitwise ANDed with
7          wmanIfBsClassifierRuleSourceMacMask equals the value
8          of wmanIfBsClassifierRuleSourceMacAddr. If the
9          referenced parameter is not present in a classifier,
10         this object reports the value of 'FFFFFFFFFFFF'H."
11     REFERENCE    "802.16 Chapter 11.4.9"
12     ::= { wmanIfBsClassifierRuleEntry 19 }
13
14     wmanIfBsClassifierRuleSourceMacMask OBJECT-TYPE
15         SYNTAX      MacAddress
16         MAX-ACCESS  read-write
17         STATUS      current
18         DESCRIPTION
19             "An Ethernet packet matches an entry when its destination
20             MAC address bitwise ANDed with
21             wmanIfBsClassifierRuleSourceMacMask equals the value of
22             wmanIfBsClassifierRuleSourceMacAddr. If the referenced
23             parameter is not present in a classifier, this object
24             reports the value of '000000000000'H."
25         REFERENCE    "802.16 Chapter 11.4.9"
26         ::= { wmanIfBsClassifierRuleEntry 20 }
27
28     wmanIfBsClassifierRuleEnetProtocolType OBJECT-TYPE
29         SYNTAX      INTEGER {none(0),
30                     ethertype(1),
31                     dsap(2)}
32         MAX-ACCESS  read-write
33         STATUS      current
34         DESCRIPTION
35             "This object indicates the format of the layer 3 protocol
36             id in the Ethernet packet. A value of none(0) means that
37             the rule does not use the layer 3 protocol type as a
38             matching criteria. A value of ethertype(1) means that the
39             rule applies only to frames which contains an EtherType
40             value. Ethertype values are contained in packets using
41             the Dec-Intel-Xerox (DIX) encapsulation or the RFC1042
42             Sub-Network Access Protocol (SNAP) encapsulation formats.
43             A value of dsap(2) means that the rule applies only to
44             frames using the IEEE802.3 encapsulation format with a
45             Destination Service Access Point (DSAP) other than 0xAA
46             (which is reserved for SNAP). If the Ethernet frame
47             contains an 802.1P/Q Tag header (i.e. EtherType 0x8100),
48             this object applies to the embedded EtherType field within
49             the 802.1P/Q header. If the referenced parameter is not
50             present in a classifier, this object reports the value of
51             0."
52         REFERENCE    "802.16 Chapter 11.4.9"
53         ::= { wmanIfBsClassifierRuleEntry 21 }
54

```

```

1  wmanIfBsClassifierRuleEnetProtocol OBJECT-TYPE
2      SYNTAX      Integer32 (0..65535)
3      MAX-ACCESS  read-write
4      STATUS      current
5      DESCRIPTION
6          "If wmanIfBsClassifierRuleEnetProtocolType is none(0),
7           this object is ignored when considering whether a packet
8           matches the current rule.
9           If wmanIfBsClassifierRuleEnetProtocolType is ethertype(1),
10          this object gives the 16-bit value of the EtherType that
11          the packet must match in order to match the rule.
12          If wmanIfBsClassifierRuleEnetProtocolType is dsap(2), the
13          lower 8 bits of this object's value must match the DSAP
14          byte of the packet in order to match the rule.
15          If wmanIfBsClassifierRuleEnetProtocolType is mac(3), the
16          lower 8 bits of this object value represent a lower bound
17          (inclusive) of MAC management message type codes matched,
18          and the upper 8 bits of this object value represent the
19          upper bound (inclusive) of matched MAC message type codes.
20          Certain message type codes are excluded from matching, as
21          specified in the reference.
22          If the Ethernet frame contains an 802.1P/Q Tag header
23          (i.e. EtherType 0x8100), this object applies to the
24          embedded EtherType field within the 802.1P/Q header.
25          If the referenced parameter is not present in the
26          classifier, the value of this object is reported as 0."
27      REFERENCE   "802.16 Chapter 11.4.9"
28      ::= { wmanIfBsClassifierRuleEntry 22 }
29
30  wmanIfBsClassifierRuleUserPriLow OBJECT-TYPE
31      SYNTAX      Integer32 (0..7)
32      MAX-ACCESS  read-write
33      STATUS      current
34      DESCRIPTION
35          "This object applies only to Ethernet frames using the
36          802.1P/Q tag header (indicated with EtherType 0x8100).
37          Such frames include a 16-bit Tag that contains a 3 bit
38          Priority field and a 12 bit VLAN number.
39          Tagged Ethernet packets must have a 3-bit Priority field
40          within the range of wmanIfBsClassifierRulePriLow and
41          wmanIfBsClassifierRulePriHigh in order to match this
42          rule.
43          If the referenced parameter is not present in the
44          classifier, the value of this object is reported as 0."
45      REFERENCE   "802.16 Chapter 11.4.9"
46      ::= { wmanIfBsClassifierRuleEntry 23 }
47
48  wmanIfBsClassifierRuleUserPriHigh OBJECT-TYPE
49      SYNTAX      Integer32 (0..7)
50      MAX-ACCESS  read-write
51      STATUS      current
52      DESCRIPTION
53          "This object applies only to Ethernet frames using the
54          802.1P/Q tag header (indicated with EtherType 0x8100).

```

```

1      Such frames include a 16-bit Tag that contains a 3 bit
2      Priority field and a 12 bit VLAN number.
3      Tagged Ethernet packets must have a 3-bit Priority
4      field within the range of wmanIfBsClassifierRulePriLow
5      and wmanIfBsClassifierRulePriHigh in order to match
6      this rule.
7      If the referenced parameter is not present in the
8      classifier, the value of this object is reported as 7."
9      REFERENCE    "802.16 Chapter 11.4.9"
10     ::= { wmanIfBsClassifierRuleEntry 24 }
11
12     wmanIfBsClassifierRuleVlanId OBJECT-TYPE
13         SYNTAX      Integer32 (0..4095)
14         MAX-ACCESS  read-write
15         STATUS      current
16         DESCRIPTION
17             "This object applies only to Ethernet frames using the
18             802.1P/Q tag header.
19             If this object's value is nonzero, tagged packets must
20             have a VLAN Identifier that matches the value in order
21             to match the rule.
22             Only the least significant 12 bits of this object's
23             value are valid.
24             If the referenced parameter is not present in the
25             classifier, the value of this object is reported as 0."
26         REFERENCE    "802.16 Chapter 11.4.9"
27         ::= { wmanIfBsClassifierRuleEntry 25 }
28
29     wmanIfBsClassifierRuleState OBJECT-TYPE
30         SYNTAX      INTEGER {active(1),
31                       inactive(2)}
32         MAX-ACCESS  read-write
33         STATUS      current
34         DESCRIPTION
35             "This object indicates whether or not the classifier is
36             enabled to classify packets to a Service Flow.
37             If the referenced parameter is not present in the
38             classifier, the value of this object is reported
39             as active(1)."
40         REFERENCE    "802.16 Chapter 11.4.9"
41         ::= { wmanIfBsClassifierRuleEntry 26 }
42
43     wmanIfBsClassifierRulePkts OBJECT-TYPE
44         SYNTAX      Counter64
45         MAX-ACCESS  read-write
46         STATUS      current
47         DESCRIPTION
48             "This object counts the number of packets that have
49             been classified using this entry."
50         ::= { wmanIfBsClassifierRuleEntry 27 }
51
52     wmanIfBsClassifierRuleRowStatus OBJECT-TYPE
53         SYNTAX      RowStatus
54         MAX-ACCESS  read-create

```

```

1      STATUS      current
2      DESCRIPTION
3          "This object is used to create a new row or modify or
4          delete an existing row in this table.
5
6          If the implementator of this MIB has chosen not
7          to implement 'dynamic assignment' of profiles, this
8          object is not useful and should return noSuchName
9          upon SNMP request."
10     ::= { wmanIfBsClassifierRuleEntry 28 }
11
12     --
13     -- wmanIfBsCps contain the Base Station Common Part Sublayer objects
14     wmanIfBsCps OBJECT IDENTIFIER ::= { wmanIfBsObjects 3 }
15
16     --
17     -- Base station PKM group
18     -- wmanIfBsPkmObjects contain the Base Station Privacy Sublayer objects
19     wmanIfBsPkmObjects OBJECT IDENTIFIER ::= { wmanIfBsObjects 4 }
20
21     --
22     -- Table wmanIfBsPkmBaseTable
23     --
24     wmanIfBsPkmBaseTable OBJECT-TYPE
25         SYNTAX      SEQUENCE OF      WmanIfBsPkmBaseEntry
26         MAX-ACCESS  not-accessible
27         STATUS      current
28         DESCRIPTION
29             "This table describes the basic PKM attributes of each Base
30             Station wireless interface."
31         ::= { wmanIfBsPkmObjects 1 }
32
33     wmanIfBsPkmBaseEntry OBJECT-TYPE
34         SYNTAX      WmanIfBsPkmBaseEntry
35         MAX-ACCESS  not-accessible
36         STATUS      current
37         DESCRIPTION
38             "Each entry contains objects describing attributes of one
39             BS wireless interface."
40         INDEX      { ifIndex }
41         ::= { wmanIfBsPkmBaseTable 1 }
42
43     wmanIfBsPkmBaseEntry ::= SEQUENCE {
44         wmanIfBsPkmDefaultAuthLifetime      Integer32,
45         wmanIfBsPkmDefaultTEKLifetime       Integer32,
46         wmanIfBsPkmDefaultSelfSignedManufCertTrust  INTEGER,
47         wmanIfBsPkmCheckCertValidityPeriods  TruthValue,
48         wmanIfBsPkmAuthentInfos              Counter64,
49         wmanIfBsPkmAuthRequests              Counter64,
50         wmanIfBsPkmAuthReplies               Counter64,
51         wmanIfBsPkmAuthRejects               Counter64,
52         wmanIfBsPkmAuthInvalids              Counter64
53     }
54

```

```
1
2
3 wmanIfBsPkmDefaultAuthLifetime OBJECT-TYPE
4     SYNTAX      Integer32 (86400..6048000)
5     UNITS       "seconds"
6     MAX-ACCESS  read-write
7     STATUS      current
8     DESCRIPTION
9         "The value of this object is the default lifetime, in
10        seconds, the BS assigns to a new authorization key."
11     REFERENCE
12        "IEEE 802.16 standard; Table 270"
13     DEFVAL     { 604800 }
14     ::= { wmanIfBsPkmBaseEntry 1 }
15
16
17 wmanIfBsPkmDefaultTEKLifetime OBJECT-TYPE
18     SYNTAX      Integer32 (1800..604800)
19     UNITS       "seconds"
20     MAX-ACCESS  read-write
21     STATUS      current
22     DESCRIPTION
23        "The value of this object is the default lifetime, in
24        seconds, the BS assigns to a new Traffic Encryption
25        Key(TEK)."
```



```
1         current time of day."
2 ::= { wmanIfBsPkmBaseEntry 4 }
3
4
5 wmanIfBsPkmAuthentInfos OBJECT-TYPE
6     SYNTAX      Counter64
7     MAX-ACCESS  read-only
8     STATUS      current
9     DESCRIPTION
10        "The value of this object is the count of times the BS has
11        received an Authentication Information message from any
12        SS."
13 ::= { wmanIfBsPkmBaseEntry 5 }
14
15
16 wmanIfBsPkmAuthRequests OBJECT-TYPE
17     SYNTAX      Counter64
18     MAX-ACCESS  read-only
19     STATUS      current
20     DESCRIPTION
21        "The value of this object is the count of times the BS has
22        received an Authorization Request message from any SS"
23 ::= { wmanIfBsPkmBaseEntry 6 }
24
25
26 wmanIfBsPkmAuthReplies OBJECT-TYPE
27     SYNTAX      Counter64
28     MAX-ACCESS  read-only
29     STATUS      current
30     DESCRIPTION
31        "The value of this object is the count of times the BS has
32        transmitted an Authorization Reply message to any SS."
33 ::= { wmanIfBsPkmBaseEntry 7 }
34
35
36 wmanIfBsPkmAuthRejects OBJECT-TYPE
37     SYNTAX      Counter64
38     MAX-ACCESS  read-only
39     STATUS      current
40     DESCRIPTION
41        "The value of this object is the count of times the BS has
42        transmitted an Authorization Reject message to any SS."
43 ::= { wmanIfBsPkmBaseEntry 8 }
44
45
46 wmanIfBsPkmAuthInvalids OBJECT-TYPE
47     SYNTAX      Counter64
48     MAX-ACCESS  read-only
49     STATUS      current
50     DESCRIPTION
51        "The value of this object is the count of times the BS has
52        transmitted an Authorization Invalid message to any SS."
53 ::= { wmanIfBsPkmBaseEntry 9 }
54
```

```

1
2  --
3  -- Table wmanIfBsPkmAuthTable
4  --
5
6  wmanIfBsPkmAuthTable OBJECT-TYPE
7      SYNTAX      SEQUENCE OF  wmanIfBsPkmAuthEntry
8      MAX-ACCESS  not-accessible
9      STATUS      current
10     DESCRIPTION
11         "This table describes the attributes of each SS
12         authorization association. The BS maintains one
13         authorization association with each Baseline
14         Privacy-enabled SS on each BS wireless interface."
15     ::= { wmanIfBsPkmObjects 2 }
16
17
18  wmanIfBsPkmAuthEntry OBJECT-TYPE
19      SYNTAX      wmanIfBsPkmAuthEntry
20      MAX-ACCESS  not-accessible
21      STATUS      current
22      DESCRIPTION
23         "Each entry contains objects describing attributes of one
24         authorization association. The BS MUST create one entry per
25         SS per wireless interface, based on the receipt of an
26         Authorization Request message, and MUST not delete the
27         entry before the SS authorization permanently expires."
28      INDEX      { ifIndex, wmanIfBsPkmAuthSsMacAddress }
29     ::= { wmanIfBsPkmAuthTable 1 }
30
31
32     wmanIfBsPkmAuthEntry ::= SEQUENCE {
33         wmanIfBsPkmAuthSsMacAddress           MacAddress,
34         wmanIfBsPkmAuthSsPublicKey           OCTET STRING,
35         wmanIfBsPkmAuthSsKeySequenceNumber   Integer32,
36         wmanIfBsPkmAuthSsExpiresOld         DateAndTime,
37         wmanIfBsPkmAuthSsExpiresNew        DateAndTime,
38         wmanIfBsPkmAuthSsLifetime           Integer32,
39         wmanIfBsPkmAuthSsGraceTime          Integer32,
40         wmanIfBsPkmAuthSsReset              INTEGER,
41         wmanIfBsPkmAuthSsInfos              Counter64,
42         wmanIfBsPkmAuthSsRequests           Counter64,
43         wmanIfBsPkmAuthSsReplies            Counter64,
44         wmanIfBsPkmAuthSsRejects            Counter64,
45         wmanIfBsPkmAuthSsInvalids           Counter64,
46         wmanIfBsPkmAuthRejectErrorCode      INTEGER,
47         wmanIfBsPkmAuthRejectErrorString    SnmpAdminString,
48         wmanIfBsPkmAuthInvalidErrorCode     INTEGER,
49         wmanIfBsPkmAuthInvalidErrorString   SnmpAdminString,
50         wmanIfBsPkmAuthPrimarySAId          Integer32,
51         wmanIfBsPkmAuthBpkmSsCertValid     INTEGER,
52         wmanIfBsPkmAuthBpkmSsCert          OCTET STRING
53     }
54

```

```

1
2   wmanIfBsPkmAuthSsMacAddress OBJECT-TYPE
3       SYNTAX      MacAddress
4       MAX-ACCESS  not-accessible
5       STATUS      current
6       DESCRIPTION
7           "The value of this object is the physical address of the SS
8             to which the authorization association applies."
9       ::= { wmanIfBsPkmAuthEntry 1 }
10
11
12  wmanIfBsPkmAuthSsPublicKey OBJECT-TYPE
13      SYNTAX      OCTET STRING (SIZE (140))
14      MAX-ACCESS  read-only
15      STATUS      current
16      DESCRIPTION
17          "The value of this object is a DER-encoded RSAPublicKey
18            ASN.1 type string, as defined in the RSA Encryption
19            Standard (PKCS #1) [10], corresponding to the public key of
20            the SS. The 74, 106, 140, 204, and 270 byte key encoding
21            lengths correspond to 512 bit, 768 bit, 1024 bit, 1536 bit,
22            and 2048 public moduli respectively. This is a zero-length
23            string if the BS does not retain the public key."
24      ::= { wmanIfBsPkmAuthEntry 2 }
25
26
27  wmanIfBsPkmAuthSsKeySequenceNumber OBJECT-TYPE
28      SYNTAX      Integer32 (0..15)
29      MAX-ACCESS  read-only
30      STATUS      current
31      DESCRIPTION
32          "The value of this object is the most recent authorization
33            key sequence number for this SS."
34      ::= { wmanIfBsPkmAuthEntry 3 }
35
36
37  wmanIfBsPkmAuthSsExpiresOld OBJECT-TYPE
38      SYNTAX      DateAndTime
39      MAX-ACCESS  read-only
40      STATUS      current
41      DESCRIPTION
42          "The value of this object is the actual clock time for
43            expiration of the immediate predecessor of the most recent
44            authorization key for this FSM. If this FSM has only one
45            authorization key, then the value is the time of activation
46            of this FSM."
47      ::= { wmanIfBsPkmAuthEntry 4 }
48
49
50  wmanIfBsPkmAuthSsExpiresNew OBJECT-TYPE
51      SYNTAX      DateAndTime
52      MAX-ACCESS  read-only
53      STATUS      current
54      DESCRIPTION

```

```

1           "The value of this object is the actual clock time for
2           expiration of the most recent authorization key for this
3           FSM"
4 ::= { wmanIfBsPkmAuthEntry 5 }
5
6
7 wmanIfBsPkmAuthSsLifetime OBJECT-TYPE
8     SYNTAX      Integer32 (86400..6048000)
9     UNITS       "seconds"
10    MAX-ACCESS  read-write
11    STATUS      current
12    DESCRIPTION
13        "The vaue of this object is the lifetime, in seconds, the
14        BS assigns to an authorization key for this SS."
15    REFERENCE
16        "IEEE 802.16 standard; Table 270"
17    DEFVAL     { 604800 }
18 ::= { wmanIfBsPkmAuthEntry 6 }
19
20
21 wmanIfBsPkmAuthSsGraceTime OBJECT-TYPE
22     SYNTAX      Integer32 (300..3024000)
23     UNITS       "seconds"
24     MAX-ACCESS  read-only
25     STATUS      current
26     DESCRIPTION
27         "The value of this object is the grace time for the
28         authorization key in seconds. The SS is expected to start
29         trying to get a new authorization key beginning
30         AuthGraceTime seconds before the authorization key actually
31         expires."
32     REFERENCE
33         "IEEE 802.16 standard; Table 270"
34     DEFVAL     { 600 }
35 ::= { wmanIfBsPkmAuthEntry 7 }
36
37
38 wmanIfBsPkmAuthSsReset OBJECT-TYPE
39     SYNTAX      INTEGER { noResetRequested(1),
40                    invalidateAuth(2),
41                    sendAuthInvalid(3),
42                    invalidateTeks(4) }
43     MAX-ACCESS  read-write
44     STATUS      current
45     DESCRIPTION
46         "Setting this object to invalidateAuth(2) causes the BS to
47         invalidate the current SS authorization key(s), but not to
48         transmit an Authorization Invalid message nor to invalidate
49         unicast TEKS. Setting this object to sendAuthInvalid(3)
50         causes the BS to invalidate the current SS authorization
51         key(s), and to transmit an Authorization Invalid message to
52         the SS, but not to invalidate unicast TEKS. Setting this
53         object to invalidateTeks(4) causes the BS to invalidate the
54         current SS authorization key(s), to transmit an

```

```

1           Authorization Invalid message to the SS, and to
2           invalidate all unicast TEKs associated with this SS
3           authorization. Reading this object returns the
4           most-recently-set value of this object, or returns
5           noResetRequested(1) if the object has not been set since
6           the last BS reboot."
7       ::= { wmanIfBsPkmAuthEntry 8 }
8
9
10      wmanIfBsPkmAuthSsInfos OBJECT-TYPE
11          SYNTAX      Counter64
12          MAX-ACCESS  read-only
13          STATUS      current
14          DESCRIPTION
15              "The value of this object is the count of times the BS has
16              received an Authentication Information message from this
17              SS."
18          ::= { wmanIfBsPkmAuthEntry 9 }
19
20
21      wmanIfBsPkmAuthSsRequests OBJECT-TYPE
22          SYNTAX      Counter64
23          MAX-ACCESS  read-only
24          STATUS      current
25          DESCRIPTION
26              "The value of this object is the count of times the BS has
27              received an Authorization Request message from this SS."
28          ::= { wmanIfBsPkmAuthEntry 10 }
29
30
31      wmanIfBsPkmAuthSsReplies OBJECT-TYPE
32          SYNTAX      Counter64
33          MAX-ACCESS  read-only
34          STATUS      current
35          DESCRIPTION
36              "The value of this object is the count of times the BS has
37              transmitted an Authorization Reply message to this SS."
38          ::= { wmanIfBsPkmAuthEntry 11 }
39
40
41      wmanIfBsPkmAuthSsRejects OBJECT-TYPE
42          SYNTAX      Counter64
43          MAX-ACCESS  read-only
44          STATUS      current
45          DESCRIPTION
46              "The value of this object is the count of times the BS has
47              transmitted an Authorization Reject message to this SS."
48          ::= { wmanIfBsPkmAuthEntry 12 }
49
50
51      wmanIfBsPkmAuthSsInvalids OBJECT-TYPE
52          SYNTAX      Counter64
53          MAX-ACCESS  read-only
54          STATUS      current

```

```

1      DESCRIPTION
2          "The value of this object is the count of times the BS has
3          transmitted an Authorization Invalid message to this SS."
4      ::= { wmanIfBSPkmAuthEntry 13 }
5
6
7      wmanIfBSPkmAuthRejectErrorCode OBJECT-TYPE
8          SYNTAX      INTEGER {none(1),
9                      unknown(2),
10                     unauthorizedSs(3),
11                     unauthorizedSaid(4),
12                     permanentAuthorizationFailure(8),
13                     timeOfDayNotAcquired(11)}
14          MAX-ACCESS  read-only
15          STATUS      current
16          DESCRIPTION
17              "The value of this object is the enumerated description of
18              the Error-Code in most recent Authorization Reject message
19              transmitted to the SS. This has value unknown(2) if the
20              last Error-Code value was 0, and none(1) if no
21              Authorization Reject message has been transmitted to the
22              SS."
23      ::= { wmanIfBSPkmAuthEntry 14 }
24
25
26      wmanIfBSPkmAuthRejectErrorString OBJECT-TYPE
27          SYNTAX      SnmpAdminString (SIZE (0..128))
28          MAX-ACCESS  read-only
29          STATUS      current
30          DESCRIPTION
31              "The value of this object is the Display-String in most
32              recent Authorization Reject message transmitted to the SS.
33              This is a zero length string if no Authorization Reject
34              message has been transmitted to the SS."
35      ::= { wmanIfBSPkmAuthEntry 15 }
36
37
38      wmanIfBSPkmAuthInvalidErrorCode OBJECT-TYPE
39          SYNTAX      INTEGER {none(1),
40                      unknown(2),
41                      unauthorizedSs(3),
42                      unsolicited(5),
43                      invalidKeySequence(6),
44                      keyRequestAuthenticationFailure(7)}
45          MAX-ACCESS  read-only
46          STATUS      current
47          DESCRIPTION
48              "The value of this object is the enumerated description of
49              the Error-Code in most recent Authorization Invalid message
50              transmitted to the SS. This has value unknown(2) if the
51              last Error-Code value was 0, and none(1) if no
52              Authorization Invalid message has been transmitted to the
53              SS."
54      ::= { wmanIfBSPkmAuthEntry 16 }

```

```

1
2
3   wmanIfBSPkmAuthInvalidErrorString OBJECT-TYPE
4       SYNTAX      SnmpAdminString (SIZE (0..128))
5       MAX-ACCESS  read-only
6       STATUS      current
7       DESCRIPTION
8           "The value of this object is the Display-String in most
9           recent Authorization Invalid message transmitted to the SS.
10          This is a zero length string if no Authorization Invalid
11          message has been transmitted to the SS."
12      ::= { wmanIfBSPkmAuthEntry 17 }
13
14
15   wmanIfBSPkmAuthPrimarySAId OBJECT-TYPE
16       SYNTAX      Integer32 (0..16383)
17       MAX-ACCESS  read-only
18       STATUS      current
19       DESCRIPTION
20           "The value of this object is the Primary Security
21           Association identifier."
22      ::= { wmanIfBSPkmAuthEntry 18 }
23
24
25   wmanIfBSPkmAuthBpkmSsCertValid OBJECT-TYPE
26       SYNTAX      INTEGER {unknown (0),
27                   validSsChained (1),
28                   validSsTrusted (2),
29                   invalidSsUntrusted (3),
30                   invalidCAUntrusted (4),
31                   invalidSsOther (5),
32                   invalidCAOther (6) }
33       MAX-ACCESS  read-only
34       STATUS      current
35       DESCRIPTION
36           "Contains the reason why a SS's certificate is deemed valid
37           or invalid. Return unknown if the SS is running PKM mode.
38           ValidSsChained means the certificate is valid because it
39           chains to a valid certificate. ValidSsTrusted means the
40           certificate is valid because it has been provisioned to be
41           trusted. InvalidSsUntrusted means the certificate is
42           invalid because it has been provisioned to be untrusted.
43           InvalidCAUntrusted means the certificate is invalid
44           because it chains to an untrusted certificate.
45           InvalidSsOther and InvalidCAOther refer to errors in
46           parsing, validity periods, etc, which are attributable to
47           the SS certificate or its chain respectively."
48      ::= { wmanIfBSPkmAuthEntry 19 }
49
50
51   wmanIfBSPkmAuthBpkmSsCert OBJECT-TYPE
52       SYNTAX      OCTET STRING
53       MAX-ACCESS  read-only
54       STATUS      current

```

```

1      DESCRIPTION
2          "The X509 SS Certificate sent as part of a PKM
3          Authorization Request."
4      ::= { wmanIfBsPkmAuthEntry 20 }
5
6
7      --
8      -- Table wmanIfBsPkmTEKTable
9      --
10
11     wmanIfBsPkmTEKTable OBJECT-TYPE
12         SYNTAX      SEQUENCE OF WmanIfBsPkmTEKEntry
13         MAX-ACCESS  not-accessible
14         STATUS      current
15         DESCRIPTION
16             "This table describes the attributes of each Traffic
17             Encryption Key (TEK) association. The BS maintains one TEK
18             association per SAID on each BS wireless interface."
19     ::= { wmanIfBsPkmObjects 3 }
20
21
22     wmanIfBsPkmTEKEntry OBJECT-TYPE
23         SYNTAX      WmanIfBsPkmTEKEntry
24         MAX-ACCESS  not-accessible
25         STATUS      current
26         DESCRIPTION
27             "Each entry contains objects describing attributes of one
28             TEK association on a particular BS wireless interface. The
29             BS MUST create one entry per SAID per wireless interface,
30             based on the receipt of a Key Request message, and MUST not
31             delete the entry before the SS authorization for the SAID
32             permanently expires."
33         INDEX      { ifIndex, wmanIfBsPkmTEKSAID }
34     ::= { wmanIfBsPkmTEKTable 1 }
35
36
37     wmanIfBsPkmTEKEntry ::= SEQUENCE {
38         wmanIfBsPkmTEKSAID                Integer32,
39         wmanIfBsPkmTEKSAType              INTEGER,
40         wmanIfBsPkmTEKDataEncryptAlg      INTEGER,
41         wmanIfBsPkmTEKDataAuthentAlg      INTEGER,
42         wmanIfBsPkmTEKEncryptAlg          INTEGER,
43         wmanIfBsPkmTEKLifetime             Integer32,
44         wmanIfBsPkmTEKGraceTime           Integer32,
45         wmanIfBsPkmTEKKeySequenceNumber   Integer32,
46         wmanIfBsPkmTEKExpiresOld          DateAndTime,
47         wmanIfBsPkmTEKExpiresNew         DateAndTime,
48         wmanIfBsPkmTEKReset               TruthValue,
49         wmanIfBsPkmKeyRequests            Counter64,
50         wmanIfBsPkmKeyReplies             Counter64,
51         wmanIfBsPkmKeyRejects            Counter64,
52         wmanIfBsPkmTEKInvalids           Counter64,
53         wmanIfBsPkmKeyRejectErrorCode     INTEGER,
54         wmanIfBsPkmKeyRejectErrorString   SnmpAdminString,

```



```

1          wmanIfBsPkmTEKInvalidErrorCode          INTEGER,
2          wmanIfBsPkmTEKInvalidErrorString      SnmpAdminString
3      }
4
5
6      wmanIfBsPkmTEKSAId OBJECT-TYPE
7          SYNTAX          Integer32
8          MAX-ACCESS      not-accessible
9          STATUS          current
10         DESCRIPTION
11             "The value of this object is the WiMAX Security Association
12             ID (SAID)."
```

```

13         ::= { wmanIfBsPkmTEKEntry 1 }
14
15
16     wmanIfBsPkmTEKSAType OBJECT-TYPE
17         SYNTAX          INTEGER {none(0),
18                             primary(1),
19                             static(2),
20                             dynamic(3) }
21         MAX-ACCESS      read-only
22         STATUS          current
23         DESCRIPTION
24             "The value of this object is the type of security
25             association. Dynamic does not apply to SSS running in PKM
26             mode."
27         ::= { wmanIfBsPkmTEKEntry 2 }
28
29
30     wmanIfBsPkmTEKDataEncryptAlg OBJECT-TYPE
31         SYNTAX          INTEGER { none(0),
32                             des56CbcMode(1) }
33         MAX-ACCESS      read-only
34         STATUS          current
35         DESCRIPTION
36             "The value of this object is the data encryption algorithm
37             being utilized."
38         REFERENCE
39             "IEEE 802.16 standard; Table 301"
40         ::= { wmanIfBsPkmTEKEntry 3 }
41
42
43     wmanIfBsPkmTEKDataAuthentAlg OBJECT-TYPE
44         SYNTAX          INTEGER { none(0) }
45         MAX-ACCESS      read-only
46         STATUS          current
47         DESCRIPTION
48             "The value of this object is the data authentication
49             algorithm being utilized."
50         REFERENCE
51             "IEEE 802.16 standard; Table 302"
52         ::= { wmanIfBsPkmTEKEntry 4 }
53
54
```

```

1      wmanIfBsPkmTEKEncryptAlg OBJECT-TYPE
2          SYNTAX      INTEGER { tripleDES(0),
3                          rsa1024(1) }
4          MAX-ACCESS  read-only
5          STATUS      current
6          DESCRIPTION
7              "The value of this object is the TEK key encryption
8              algorithm being utilized."
9          REFERENCE
10             "IEEE 802.16 standard; Table 303"
11             ::= { wmanIfBsPkmTEKEntry 5 }
12
13
14     wmanIfBsPkmTEKLifetime OBJECT-TYPE
15         SYNTAX      Integer32 (1800..604800)
16         UNITS       "seconds"
17         MAX-ACCESS  read-write
18         STATUS      current
19         DESCRIPTION
20             "The value of this object is the lifetime, in seconds, the
21             BS assigns to keys for this TEK association."
22         REFERENCE
23             "IEEE 802.16 standard; Table 270"
24         DEFVAL      { 43200 }
25         ::= { wmanIfBsPkmTEKEntry 6 }
26
27
28     wmanIfBsPkmTEKGraceTime OBJECT-TYPE
29         SYNTAX      Integer32 (300..302399)
30         UNITS       "seconds"
31         MAX-ACCESS  read-only
32         STATUS      current
33         DESCRIPTION
34             "The value of this object is the grace time for the TEK in
35             seconds. The SS is expected to start trying to acquire a
36             new TEK beginning TEK GraceTime seconds before the TEK
37             actually expires."
38         REFERENCE
39             "IEEE 802.16 standard; Table 270"
40         DEFVAL      { 3600 }
41         ::= { wmanIfBsPkmTEKEntry 7 }
42
43
44     wmanIfBsPkmTEKKeySequenceNumber OBJECT-TYPE
45         SYNTAX      Integer32 (0..15)
46         MAX-ACCESS  read-only
47         STATUS      current
48         DESCRIPTION
49             "The value of this object is the most recent TEK key
50             sequence number for this SAID."
51         ::= { wmanIfBsPkmTEKEntry 8 }
52
53
54     wmanIfBsPkmTEKExpiresOld OBJECT-TYPE

```

```

1      SYNTAX      DateAndTime
2      MAX-ACCESS  read-only
3      STATUS      current
4      DESCRIPTION
5          "The value of this object is the actual clock time for
6          expiration of the immediate predecessor of the most recent
7          TEK for this FSM. If this FSM has only one TEK, then the
8          value is the time of activation of this FSM."
9      ::= { wmanIfBsPkmTEKEntry 9 }
10
11
12     wmanIfBsPkmTEKExpiresNew OBJECT-TYPE
13         SYNTAX      DateAndTime
14         MAX-ACCESS  read-only
15         STATUS      current
16         DESCRIPTION
17             "The value of this object is the actual clock time for
18             expiration of the most recent TEK for this FSM."
19         ::= { wmanIfBsPkmTEKEntry 10 }
20
21
22     wmanIfBsPkmTEKReset OBJECT-TYPE
23         SYNTAX      TruthValue
24         MAX-ACCESS  read-write
25         STATUS      current
26         DESCRIPTION
27             "Setting this object to TRUE causes the BS to invalidate
28             the current active TEK(s) (plural due to key transition
29             periods), and to generate a new TEK for the associated
30             SAID; the BS MAY also generate an unsolicited TEK Invalid
31             message, to optimize the TEK synchronization between the BS
32             and the SS. Reading this object always returns FALSE."
33         ::= { wmanIfBsPkmTEKEntry 11 }
34
35
36     wmanIfBsPkmKeyRequests OBJECT-TYPE
37         SYNTAX      Counter64
38         MAX-ACCESS  read-only
39         STATUS      current
40         DESCRIPTION
41             "The value of this object is the count of times the BS has
42             received a Key Request message."
43         ::= { wmanIfBsPkmTEKEntry 12 }
44
45
46     wmanIfBsPkmKeyReplies OBJECT-TYPE
47         SYNTAX      Counter64
48         MAX-ACCESS  read-only
49         STATUS      current
50         DESCRIPTION
51             "The value of this object is the count of times the BS has
52             transmitted a Key Reply message."
53         ::= { wmanIfBsPkmTEKEntry 13 }
54

```

```

1
2   wmanIfBsPkmKeyRejects OBJECT-TYPE
3       SYNTAX          Counter64
4       MAX-ACCESS      read-only
5       STATUS          current
6       DESCRIPTION
7           "The value of this object is the count of times the BS has
8             transmitted a Key Reject message."
9       ::= { wmanIfBsPkmTEKEntry 14 }
10
11
12   wmanIfBsPkmTEKInvalids OBJECT-TYPE
13       SYNTAX          Counter64
14       MAX-ACCESS      read-only
15       STATUS          current
16       DESCRIPTION
17           "The value of this object is the count of times the BS has
18             transmitted a TEK Invalid message."
19       ::= { wmanIfBsPkmTEKEntry 15 }
20
21
22   wmanIfBsPkmKeyRejectErrorCode OBJECT-TYPE
23       SYNTAX          INTEGER {none(1),
24                               unknown(2),
25                               unauthorizedSaid(4)}
26       MAX-ACCESS      read-only
27       STATUS          current
28       DESCRIPTION
29           "The value of this object is the enumerated; description of
30             the Error-Code in the most recent Key Reject message sent
31             in response to a Key Request for this SAID. This has value
32             unknown(2) if the last Error-Code value was 0, and none(1)
33             if no Key Reject message has been received since reboot."
34       ::= { wmanIfBsPkmTEKEntry 16 }
35
36
37   wmanIfBsPkmKeyRejectErrorString OBJECT-TYPE
38       SYNTAX          SnmpAdminString (SIZE (0..128))
39       MAX-ACCESS      read-only
40       STATUS          current
41       DESCRIPTION
42           "The value of this object is the Display-String in the most
43             recent Key Reject message sent in response to a Key Request
44             for this SAID. This is a zero length string if no Key
45             Reject message has been received since reboot."
46       ::= { wmanIfBsPkmTEKEntry 17 }
47
48
49   wmanIfBsPkmTEKInvalidErrorCode OBJECT-TYPE
50       SYNTAX          INTEGER {none(1),
51                               unknown(2),
52                               invalidKeySequence(6)}
53       MAX-ACCESS      read-only
54       STATUS          current

```

```

1      DESCRIPTION
2          "The value of this object is the enumerated description of
3          the Error-Code in the most recent TEK Invalid message sent
4          in association with this SAID. This has value unknown(2)
5          if the last Error-Code value was 0, and none(1) if no TEK
6          Invalid message has been received since reboot."
7      ::= { wmanIfBsPkmTEKEntry 18 }
8
9
10     wmanIfBsPkmTEKInvalidErrorString OBJECT-TYPE
11         SYNTAX      SnmpAdminString (SIZE (0..128))
12         MAX-ACCESS  read-only
13         STATUS      current
14         DESCRIPTION
15             "The value of this object is the Display-String in the most
16             recent TEK Invalid message sent in association with this
17             SAID. This is a zero length string if no TEK Invalid
18             message has been received since reboot."
19         ::= { wmanIfBsPkmTEKEntry 19 }
20
21     --
22     -- SS object group - containing tables and objects to be implemented in
23     -- the Subscriber station
24     --
25     -- wmanIfSsSystem contain the Subscriber Station System objects
26     wmanIfSsSystem OBJECT IDENTIFIER ::= { wmanIfSsObjects 1 }
27
28     wmanIfSsConfigFileEncodingTable OBJECT-TYPE
29         SYNTAX      SEQUENCE OF wmanIfSsConfigFileEncodingEntry
30         MAX-ACCESS  not-accessible
31         STATUS      current
32         DESCRIPTION
33             "This table contains configuration file encoding
34             information of the SS."
35         REFERENCE
36             "Section 11.2 in IEEE 802.16REvD/D3-2004"
37         ::= { wmanIfSsSystem 1 }
38
39     wmanIfSsConfigFileEncodingEntry OBJECT-TYPE
40         SYNTAX      wmanIfSsConfigFileEncodingEntry
41         MAX-ACCESS  not-accessible
42         STATUS      current
43         DESCRIPTION
44             "This table provides one row for each SS, and is indexed
45             by BS ifIndex. An entry in this table exists for each
46             ifEntry of SS with an ifType of wmanMac"
47         INDEX { ifIndex }
48         ::= { wmanIfSsConfigFileEncodingTable 1 }
49
50     wmanIfSsConfigFileEncodingEntry ::= SEQUENCE {
51         wmanIfSsMicConfigSetting      OCTET STRING,
52         wmanIfSsVendorId               OCTET STRING,
53         wmanIfSsHwId                   OCTET STRING,
54         wmanIfSsSwVersion               OCTET STRING,

```

```

1      wmanIfSsUpgradeFileName          OCTET STRING,
2      wmanIfSsSwUpgradeTftpServer      InetAddress,
3      wmanIfSsTftpServerTimeStamp      DateAndTime
4      }
5
6      wmanIfSsMicConfigSetting OBJECT-TYPE
7          SYNTAX          OCTET STRING (SIZE(16))
8          MAX-ACCESS      read-only
9          STATUS          current
10         DESCRIPTION
11             "The value field contains the SS MIC code. This is used
12             to detect unauthorized modification or corruption of
13             the configuration file."
14         ::= { wmanIfSsConfigFileEncodingEntry 1 }
15
16     wmanIfSsVendorId OBJECT-TYPE
17         SYNTAX          OCTET STRING (SIZE(3))
18         MAX-ACCESS      read-only
19         STATUS          current
20         DESCRIPTION
21             "This value identifies the managed SS vendor to which the
22             software upgrade is to be applied."
23         ::= { wmanIfSsConfigFileEncodingEntry 2 }
24
25     wmanIfSsHwId OBJECT-TYPE
26         SYNTAX          OCTET STRING
27         MAX-ACCESS      read-only
28         STATUS          current
29         DESCRIPTION
30             "This value identifies the hardware version to which the
31             software upgrade is to be applied."
32         ::= { wmanIfSsConfigFileEncodingEntry 3 }
33
34     wmanIfSsSwVersion OBJECT-TYPE
35         SYNTAX          OCTET STRING
36         MAX-ACCESS      read-only
37         STATUS          current
38         DESCRIPTION
39             "The value field contains the SS MIC code. This is used
40             to detect unauthorized modification or corruption of
41             the configuration file."
42         ::= { wmanIfSsConfigFileEncodingEntry 4 }
43
44     wmanIfSsUpgradeFileName OBJECT-TYPE
45         SYNTAX          OCTET STRING
46         MAX-ACCESS      read-only
47         STATUS          current
48         DESCRIPTION
49             "The filename is a fully qualified directory path
50             name which is in a format appropriate to the server."
51         ::= { wmanIfSsConfigFileEncodingEntry 5 }
52
53     wmanIfSsSwUpgradeTftpServer OBJECT-TYPE
54         SYNTAX          InetAddress

```

```

1      MAX-ACCESS  read-only
2      STATUS      current
3      DESCRIPTION
4          "This object is the IP address of the TFTP server on
5          which the software upgrade file for the SS resides."
6      ::= { wmanIfSsConfigFileEncodingEntry 6 }
7
8  wmanIfSsTftpServerTimeStamp OBJECT-TYPE
9      SYNTAX      DateAndTime
10     MAX-ACCESS  read-only
11     STATUS      current
12     DESCRIPTION
13         "This is the sending time of the configuration file in
14     seconds.
15         The definition of time is as in IETF RFC 868."
16     ::= { wmanIfSsConfigFileEncodingEntry 7 }
17
18     --
19     -- wmanIfSsCps contain the Base Station Common Part Sublayer objects
20 wmanIfSsCps OBJECT IDENTIFIER ::= { wmanIfSsObjects 2 }
21
22
23
24     -- Subscriber station PKM group
25     -- wmanIfSsPkmObjects contain the Subscriber Station Privacy Sublayer
26     -- objects
27 wmanIfSsPkmObjects OBJECT IDENTIFIER ::= { wmanIfSsObjects 3 }
28
29
30     --
31     -- Table wmanIfSsPkmBaseTable
32     --
33
34 wmanIfSsPkmBaseTable OBJECT-TYPE
35     SYNTAX      SEQUENCE OF wmanIfSsPkmBaseEntry
36     MAX-ACCESS  not-accessible
37     STATUS      current
38     DESCRIPTION
39         "This table describes the basic and authorization related
40     PKM attributes of each SS wireless interface."
41     ::= { wmanIfSsPkmObjects 1 }
42
43
44 wmanIfSsPkmBaseEntry OBJECT-TYPE
45     SYNTAX      wmanIfSsPkmBaseEntry
46     MAX-ACCESS  not-accessible
47     STATUS      current
48     DESCRIPTION
49         "Each entry contains objects describing attributes of one
50     SS wireless interface."
51     INDEX      { ifIndex }
52     ::= { wmanIfSsPkmBaseTable 1 }
53
54

```

```

1      wmanIfSsPkmBaseEntry ::= SEQUENCE {
2          wmanIfSsPkmPrivacyEnable          TruthValue,
3          wmanIfSsPkmPublicKey              OCTET STRING,
4          wmanIfSsPkmAuthState              INTEGER,
5          wmanIfSsPkmAuthKeySequenceNumber Integer32,
6          wmanIfSsPkmAuthExpiresOld         DateAndTime,
7          wmanIfSsPkmAuthExpiresNew        DateAndTime,
8          wmanIfSsPkmAuthReset              TruthValue,
9          wmanIfSsPkmAuthGraceTime          Integer32,
10         wmanIfSsPkmTEKGraceTime           Integer32,
11         wmanIfSsPkmAuthWaitTimeout        Integer32,
12         wmanIfSsPkmReauthWaitTimeout      Integer32,
13         wmanIfSsPkmOpWaitTimeout          Integer32,
14         wmanIfSsPkmRekeyWaitTimeout       Integer32,
15         wmanIfSsPkmAuthRejectWaitTimeout  Integer32,
16         wmanIfSsPkmAuthentInfos           Counter64,
17         wmanIfSsPkmAuthRequests           Counter64,
18         wmanIfSsPkmAuthReplies            Counter64,
19         wmanIfSsPkmAuthRejects            Counter64,
20         wmanIfSsPkmAuthInvalids           Counter64,
21         wmanIfSsPkmAuthRejectErrorCode    INTEGER,
22         wmanIfSsPkmAuthRejectErrorString  SnmpAdminString,
23         wmanIfSsPkmAuthInvalidErrorCode   INTEGER,
24         wmanIfSsPkmAuthInvalidErrorString SnmpAdminString
25     }
26
27
28     wmanIfSsPkmPrivacyEnable OBJECT-TYPE
29         SYNTAX      TruthValue
30         MAX-ACCESS  read-only
31         STATUS      current
32         DESCRIPTION
33             "This object identifies whether this SS is provisioned to
34             run Baseline Privacy Plus."
35         ::= { wmanIfSsPkmBaseEntry 1 }
36
37
38     wmanIfSsPkmPublicKey OBJECT-TYPE
39         SYNTAX      OCTET STRING (SIZE (140))
40         MAX-ACCESS  read-only
41         STATUS      current
42         DESCRIPTION
43             "The value of this object is a DER-encoded RSAPublicKey
44             ASN.1 type string, as defined in the RSA Encryption
45             Standard (PKCS#1) [10], corresponding to the public key of
46             the SS. The 74, 106, 140, 204, and 270 byte key encoding
47             lengths correspond to 512 bit, 768 bit, 1024 bit, 1536 bit,
48             and 2048 public moduli respectively."
49         ::= { wmanIfSsPkmBaseEntry 2 }
50
51
52     wmanIfSsPkmAuthState OBJECT-TYPE
53         SYNTAX      INTEGER {start(1),
54                     authwait(2),

```



```

1           authorized(3),
2           reauthWait(4),
3           authRejectWait(5),
4           silent(6)}
5     MAX-ACCESS    read-only
6     STATUS        current
7     DESCRIPTION
8         "The value of this object is the state of the SS
9         authorization FSM. The start state indicates that FSM is
10        in its initial state."
11    ::= { wmanIfSsPkmBaseEntry 3 }
12
13
14    wmanIfSsPkmAuthKeySequenceNumber OBJECT-TYPE
15        SYNTAX      Integer32 (0..15)
16        MAX-ACCESS  read-only
17        STATUS      current
18        DESCRIPTION
19            "The value of this object is the most recent authorization
20            key sequence number for this FSM."
21        ::= { wmanIfSsPkmBaseEntry 4 }
22
23
24    wmanIfSsPkmAuthExpiresOld OBJECT-TYPE
25        SYNTAX      DateAndTime
26        MAX-ACCESS  read-only
27        STATUS      current
28        DESCRIPTION
29            "The value of this object is the actual clock time for
30            expiration of the immediate predecessor of the most recent
31            authorization key for this FSM. If this FSM has only one
32            authorization key, then the value is the time of activation
33            of this FSM."
34        ::= { wmanIfSsPkmBaseEntry 5 }
35
36
37    wmanIfSsPkmAuthExpiresNew OBJECT-TYPE
38        SYNTAX      DateAndTime
39        MAX-ACCESS  read-only
40        STATUS      current
41        DESCRIPTION
42            "The value of this object is the actual clock time for
43            expiration of the most recent authorization key for this
44            FSM."
45        ::= { wmanIfSsPkmBaseEntry 6 }
46
47
48    wmanIfSsPkmAuthReset OBJECT-TYPE
49        SYNTAX      TruthValue
50        MAX-ACCESS  read-write
51        STATUS      current
52        DESCRIPTION
53            "Setting this object to TRUE generates a Reauthorize event
54            in the authorization FSM. Reading this object always

```

```

1         returns FALSE."
2     ::= { wmanIfSsPkmBaseEntry 7 }
3
4
5     wmanIfSsPkmAuthGraceTime OBJECT-TYPE
6         SYNTAX      Integer32 (300..3024000)
7         UNITS       "seconds"
8         MAX-ACCESS  read-only
9         STATUS      current
10        DESCRIPTION
11            "The value of this object is the grace time for an
12             authorization key. A SS is expected to start trying to get
13             a new authorization key beginning AuthGraceTime seconds
14             before the authorization key actually expires."
15        REFERENCE
16            "IEEE 802.16 standard; Table 270"
17        DEFVAL      { 600 }
18        ::= { wmanIfSsPkmBaseEntry 8 }
19
20
21     wmanIfSsPkmTEKGraceTime OBJECT-TYPE
22         SYNTAX      Integer32 (300..3024000)
23         UNITS       "seconds"
24         MAX-ACCESS  read-only
25         STATUS      current
26        DESCRIPTION
27            "The value of this object is the grace time for the TEK in
28             seconds. The SS is expected to start trying to acquire a
29             new TEK beginning TEK GraceTime seconds before the
30             expiration of the most recent TEK."
31        REFERENCE
32            "IEEE 802.16 standard; Table 270"
33        DEFVAL      { 3600 }
34        ::= { wmanIfSsPkmBaseEntry 9 }
35
36
37     wmanIfSsPkmAuthWaitTimeout OBJECT-TYPE
38         SYNTAX      Integer32 (2..30)
39         UNITS       "seconds"
40         MAX-ACCESS  read-only
41         STATUS      current
42        DESCRIPTION
43            "The value of this object is the Authorize wait Timeout."
44        REFERENCE
45            "IEEE 802.16 standard; Table 270"
46        DEFVAL      { 10 }
47        ::= { wmanIfSsPkmBaseEntry 10 }
48
49
50     wmanIfSsPkmReauthWaitTimeout OBJECT-TYPE
51         SYNTAX      Integer32 (2..30)
52         UNITS       "seconds"
53         MAX-ACCESS  read-only
54         STATUS      current

```

```

1      DESCRIPTION
2          "The value of this object is the Reauthorize Wait Timeout
3          in seconds."
4      REFERENCE
5          "IEEE 802.16 standard; Table 270"
6      DEFVAL      { 10 }
7      ::= { wmanIfSsPkmBaseEntry 11 }
8
9
10     wmanIfSsPkmOpwaitTimeout OBJECT-TYPE
11         SYNTAX      Integer32 (1..10)
12         UNITS       "seconds"
13         MAX-ACCESS  read-only
14         STATUS      current
15         DESCRIPTION
16             "The value of this object is the Operational Wait Timeout
17             in seconds."
18         REFERENCE
19             "IEEE 802.16 standard; Table 270"
20         DEFVAL      { 1 }
21         ::= { wmanIfSsPkmBaseEntry 12 }
22
23
24     wmanIfSsPkmRekeywaitTimeout OBJECT-TYPE
25         SYNTAX      Integer32 (1..10)
26         UNITS       "seconds"
27         MAX-ACCESS  read-only
28         STATUS      current
29         DESCRIPTION
30             "The value of this object is the Rekey Wait Timeout in
31             seconds."
32         REFERENCE
33             "IEEE 802.16 standard; Table 270"
34         DEFVAL      { 1 }
35         ::= { wmanIfSsPkmBaseEntry 13 }
36
37
38     wmanIfSsPkmAuthRejectwaitTimeout OBJECT-TYPE
39         SYNTAX      Integer32 (10..600)
40         UNITS       "seconds"
41         MAX-ACCESS  read-only
42         STATUS      current
43         DESCRIPTION
44             "The value of this object is the Authorization Reject wait
45             Timeout in seconds."
46         REFERENCE
47             "IEEE 802.16 standard; Table 270"
48         DEFVAL      { 60 }
49         ::= { wmanIfSsPkmBaseEntry 14 }
50
51
52     wmanIfSsPkmAuthentInfos OBJECT-TYPE
53         SYNTAX      Counter64
54         MAX-ACCESS  read-only

```

```

1      STATUS      current
2      DESCRIPTION
3          "The value of this object is the count of times the SS has
4          transmitted an Authentication Information message."
5      ::= { wmanIfSsPkmBaseEntry 15 }
6
7
8      wmanIfSsPkmAuthRequests OBJECT-TYPE
9          SYNTAX      Counter64
10         MAX-ACCESS  read-only
11         STATUS      current
12         DESCRIPTION
13             "The value of this object is the count of times the SS has
14             transmitted an Authorization Request message."
15         ::= { wmanIfSsPkmBaseEntry 16 }
16
17
18         wmanIfSsPkmAuthReplies OBJECT-TYPE
19             SYNTAX      Counter64
20             MAX-ACCESS  read-only
21             STATUS      current
22             DESCRIPTION
23                 "The value of this object is the count of times the SS has
24                 received an Authorization Reply message."
25             ::= { wmanIfSsPkmBaseEntry 17 }
26
27
28         wmanIfSsPkmAuthRejects OBJECT-TYPE
29             SYNTAX      Counter64
30             MAX-ACCESS  read-only
31             STATUS      current
32             DESCRIPTION
33                 "The value of this object is the count of times the SS has
34                 received an Authorization Reject message."
35             ::= { wmanIfSsPkmBaseEntry 18 }
36
37
38         wmanIfSsPkmAuthInvalids OBJECT-TYPE
39             SYNTAX      Counter64
40             MAX-ACCESS  read-only
41             STATUS      current
42             DESCRIPTION
43                 "The value of this object is the count of times the SS has
44                 received an Authorization Invalid message."
45             ::= { wmanIfSsPkmBaseEntry 19 }
46
47
48         wmanIfSsPkmAuthRejectErrorCode OBJECT-TYPE
49             SYNTAX      INTEGER {none(1),
50                             unknown(2),
51                             unauthorizedSs(3),
52                             unauthorizedSaid(4),
53                             permanentAuthorizationFailure(8),
54                             timeOfDayNotAcquired(11)}

```

```

1      MAX-ACCESS  read-only
2      STATUS      current
3      DESCRIPTION
4          "The value of this object is the enumerated description of
5          the Error-Code in most recent Authorization Reject message
6          received by the SS. This has value unknown(2)if the last
7          Error-Code value was 0, and none(1) if no Authorization
8          Reject message has been received since reboot."
9      ::= { wmanIfSsPkmBaseEntry 20 }
10
11
12     wmanIfSsPkmAuthRejectErrorString OBJECT-TYPE
13         SYNTAX      SnmpAdminString (SIZE (0..128))
14         MAX-ACCESS  read-only
15         STATUS      current
16         DESCRIPTION
17             "The value of this object is the Display-String in most
18             recent Authorization Reject message received by the SS.
19             This is a zero length string if no Authorization Reject
20             message has been received since reboot."
21         ::= { wmanIfSsPkmBaseEntry 21 }
22
23
24     wmanIfSsPkmAuthInvalidErrorCode OBJECT-TYPE
25         SYNTAX      INTEGER {none(1),
26                     unknown(2),
27                     unauthorizedSs(3),
28                     unsolicited(5),
29                     invalidKeySequence(6),
30                     keyRequestAuthenticationFailure(7)}
31         MAX-ACCESS  read-only
32         STATUS      current
33         DESCRIPTION
34             "The value of this object is the enumerated description of
35             the Error-Code in most recent Authorization Invald message
36             received by the SS. This has value unknown(2) if the last
37             Error-Code value was 0, and none(1) if no Authorization
38             Invalid message has been received since reboot."
39         ::= { wmanIfSsPkmBaseEntry 22 }
40
41
42     wmanIfSsPkmAuthInvalidErrorString OBJECT-TYPE
43         SYNTAX      SnmpAdminString (SIZE (0..128))
44         MAX-ACCESS  read-only
45         STATUS      current
46         DESCRIPTION
47             "The value of this object is the Display-String in most
48             recent Authorization Invalid message received by the SS.
49             This is a zero length string if no Authorization Invalid
50             message has been received since reboot."
51         ::= { wmanIfSsPkmBaseEntry 23 }
52
53
54     --

```

```

1      -- Table wmanIfSsPkmTEKTable
2      --
3
4      wmanIfSsPkmTEKTable OBJECT-TYPE
5          SYNTAX      SEQUENCE OF  WmanIfSsPkmTEKEntry
6          MAX-ACCESS  not-accessible
7          STATUS      current
8          DESCRIPTION
9              "This table describes the attributes of each SS Traffic
10             Encryption Key(TEK) association. The SS maintains (no more
11             than) one TEK association per SAID per SS wireless
12             interface."
13      ::= { wmanIfSsPkmObjects 2 }
14
15
16      wmanIfSsPkmTEKEntry OBJECT-TYPE
17          SYNTAX      WmanIfSsPkmTEKEntry
18          MAX-ACCESS  not-accessible
19          STATUS      current
20          DESCRIPTION
21              "Each entry contains objects describing the TEK association
22             attributes of one SAID. The SS MUST create one entry per
23             SAID, regardless of whether the SAID was obtained from a
24             Registration Response message, from an Authorization Reply
25             message, or from any dynamic SAID establishment
26             mechanisms."
27          INDEX      { ifIndex, wmanIfSsPkmTEKSAId }
28      ::= { wmanIfSsPkmTEKTable 1 }
29
30
31      wmanIfSsPkmTEKEntry ::= SEQUENCE {
32          wmanIfSsPkmTEKSAId          Integer32,
33          wmanIfSsPkmTEKSAType        INTEGER,
34          wmanIfSsPkmTEKDataEncryptAlg  INTEGER,
35          wmanIfSsPkmTEKDataAuthentAlg  INTEGER,
36          wmanIfSsPkmTEKEncryptAlg     INTEGER,
37          wmanIfSsPkmTEKState          INTEGER,
38          wmanIfSsPkmTEKKeySequenceNumber Integer32,
39          wmanIfSsPkmTEKExpiresOld      DateAndTime,
40          wmanIfSsPkmTEKExpiresNew     DateAndTime,
41          wmanIfSsPkmTEKKeyRequests     Counter64,
42          wmanIfSsPkmTEKKeyReplies     Counter64,
43          wmanIfSsPkmTEKKeyRejects     Counter64,
44          wmanIfSsPkmTEKInvalids       Counter64,
45          wmanIfSsPkmTEKAuthPends      Counter64,
46          wmanIfSsPkmTEKKeyRejectErrorCode INTEGER,
47          wmanIfSsPkmTEKKeyRejectErrorString SnmpAdminString,
48          wmanIfSsPkmTEKInvalidErrorCode INTEGER,
49          wmanIfSsPkmTEKInvalidErrorString SnmpAdminString
50      }
51
52
53      wmanIfSsPkmTEKSAId OBJECT-TYPE
54          SYNTAX      Integer32 (1..16383)

```

```

1      MAX-ACCESS  not-accessible
2      STATUS      current
3      DESCRIPTION
4          "The value of this object is the WiMAX Security Association
5              ID (SAID)."
```

::= { wmanIfSsPkmTEKEntry 1 }

```

7
8
9      wmanIfSsPkmTEKSAType OBJECT-TYPE
10     SYNTAX      INTEGER {none(0),
11                                     primary(1),
12                                     static(2),
13                                     dynamic(3)}
14     MAX-ACCESS  read-only
15     STATUS      current
16     DESCRIPTION
17         "The value of this object is the type of security
18             association."
```

::= { wmanIfSsPkmTEKEntry 2 }

```

20
21
22     wmanIfSsPkmTEKDataEncryptAlg OBJECT-TYPE
23     SYNTAX      INTEGER { none(0),
24                                     des56CbcMode(1) }
25     MAX-ACCESS  read-only
26     STATUS      current
27     DESCRIPTION
28         "The value of this object is the data encryption algorithm
29             being utilized."
```

REFERENCE

"IEEE 802.16 standard; Table 301"

::= { wmanIfSsPkmTEKEntry 3 }

```

33
34
35     wmanIfSsPkmTEKDataAuthentAlg OBJECT-TYPE
36     SYNTAX      INTEGER { none(0) }
37     MAX-ACCESS  read-only
38     STATUS      current
39     DESCRIPTION
40         "The value of this object is the data authentication
41             algorithm being utilized."
```

REFERENCE

"IEEE 802.16 standard; Table 302"

::= { wmanIfSsPkmTEKEntry 4 }

```

45
46
47     wmanIfSsPkmTEKEncryptAlg OBJECT-TYPE
48     SYNTAX      INTEGER { tripleDES(0),
49                                     rsa1024(1) }
50     MAX-ACCESS  read-only
51     STATUS      current
52     DESCRIPTION
53         "The value of this object is the TEK key encryption
54             algorithm for this cryptographic suite capability."
```

```

1      REFERENCE
2          "IEEE 802.16 standard; Table 303"
3      ::= { wmanIfSsPkmTEKEntry 5 }
4
5
6      wmanIfSsPkmTEKState OBJECT-TYPE
7          SYNTAX      INTEGER { start(1),
8                      opwait(2),
9                      opReauthwait(3),
10                     operational(4),
11                     rekeywait(5),
12                     rekeyReauthwait(6) }
13
14          MAX-ACCESS  read-only
15          STATUS      current
16          DESCRIPTION
17              "The value of this object is the state of the indicated TEK
18              FSM. The start(1) state indicates that FSM is in its
19              initial state."
20
21          ::= { wmanIfSsPkmTEKEntry 6 }
22
23      wmanIfSsPkmTEKKeySequenceNumber OBJECT-TYPE
24          SYNTAX      Integer32 (0..15)
25          MAX-ACCESS  read-only
26          STATUS      current
27          DESCRIPTION
28              "The value of this object is the most recent TEK key
29              sequence number for this TEK FSM."
30
31          ::= { wmanIfSsPkmTEKEntry 7 }
32
33      wmanIfSsPkmTEKExpiresOld OBJECT-TYPE
34          SYNTAX      DateAndTime
35          MAX-ACCESS  read-only
36          STATUS      current
37          DESCRIPTION
38              "The value of this object is the actual clock time for
39              expiration of the immediate predecessor of the most recent
40              TEK for this FSM. If this FSM has only one TEK, then the
41              value is the time of activation of this FSM."
42
43          ::= { wmanIfSsPkmTEKEntry 8 }
44
45      wmanIfSsPkmTEKExpiresNew OBJECT-TYPE
46          SYNTAX      DateAndTime
47          MAX-ACCESS  read-only
48          STATUS      current
49          DESCRIPTION
50              "The value of this object is the actual clock time for
51              expiration of the most recent TEK for this FSM."
52
53          ::= { wmanIfSsPkmTEKEntry 9 }
54
55      wmanIfSsPkmTEKKeyRequests OBJECT-TYPE

```



```
1          SYNTAX      Counter64
2          MAX-ACCESS  read-only
3          STATUS      current
4          DESCRIPTION
5              "The value of this object is the count of times the SS has
6              transmitted a Key Request message."
7          ::= { wmanIfSsPkmTEKEntry 10 }
8
9
10         wmanIfSsPkmTEKKeyReplies OBJECT-TYPE
11             SYNTAX      Counter64
12             MAX-ACCESS  read-only
13             STATUS      current
14             DESCRIPTION
15                 "The value of this object is the count of times the SS has
16                 received a Key Reply message, including a message whose
17                 authentication failed."
18             ::= { wmanIfSsPkmTEKEntry 11 }
19
20
21         wmanIfSsPkmTEKKeyRejects OBJECT-TYPE
22             SYNTAX      Counter64
23             MAX-ACCESS  read-only
24             STATUS      current
25             DESCRIPTION
26                 "The value of this object is the count of times the SS has
27                 received a Key Reject message, including a message whose
28                 authentication failed."
29             ::= { wmanIfSsPkmTEKEntry 12 }
30
31
32         wmanIfSsPkmTEKInvalids OBJECT-TYPE
33             SYNTAX      Counter64
34             MAX-ACCESS  read-only
35             STATUS      current
36             DESCRIPTION
37                 "The value of this object is the count of times the SS has
38                 received a TEK Invalid message, including a message whose
39                 authentication failed."
40             ::= { wmanIfSsPkmTEKEntry 13 }
41
42
43         wmanIfSsPkmTEKAuthPends OBJECT-TYPE
44             SYNTAX      Counter64
45             MAX-ACCESS  read-only
46             STATUS      current
47             DESCRIPTION
48                 "The value of this object is the count of times an
49                 Authorization Pending (Auth Pend) event occurred in this
50                 FSM."
51             ::= { wmanIfSsPkmTEKEntry 14 }
52
53
54         wmanIfSsPkmTEKKeyRejectErrorCode OBJECT-TYPE
```

```

1          SYNTAX      INTEGER {none(1),
2                                unknown(2),
3                                unauthorizedSaid(4)}
4          MAX-ACCESS  read-only
5          STATUS      current
6          DESCRIPTION
7              "The value of this object is the enumerated description of
8              the Error-Code in most recent Key Reject message received
9              by the SS. This has value unknown(2) if the last Error-Code
10             value was 0, and none(1) if no Key Reject message has been
11             received since reboot."
12             ::= { wmanIfSsPkmTEKEntry 15 }
13
14
15         wmanIfSsPkmTEKKeyRejectErrorString OBJECT-TYPE
16             SYNTAX      SnmpAdminString (SIZE (0..128))
17             MAX-ACCESS  read-only
18             STATUS      current
19             DESCRIPTION
20                 "The value of this object is the Display-String in most
21                 recent Key Reject message received by the SS. This is a
22                 zero length string if no Key Reject message has been
23                 received since reboot."
24                 ::= { wmanIfSsPkmTEKEntry 16 }
25
26
27         wmanIfSsPkmTEKInvalidErrorCode OBJECT-TYPE
28             SYNTAX      INTEGER {none(1),
29                                unknown(2),
30                                invalidKeySequence(6)}
31             MAX-ACCESS  read-only
32             STATUS      current
33             DESCRIPTION
34                 "The value of this object is the enumerated description of
35                 the Error-Code in most recent TEK Invalid message received
36                 by the SS. This has value unknown(2) if the last
37                 Error-Code value was 0, and none(1) if no TEK Invalid
38                 message has been received since reboot."
39                 ::= { wmanIfSsPkmTEKEntry 17 }
40
41
42         wmanIfSsPkmTEKInvalidErrorString OBJECT-TYPE
43             SYNTAX      SnmpAdminString (SIZE (0..128))
44             MAX-ACCESS  read-only
45             STATUS      current
46             DESCRIPTION
47                 "The value of this object is the Display-String in most
48                 recent TEK Invalid message received by the SS. This is a
49                 zero length string if no TEK Invalid message has been
50                 received since reboot."
51                 ::= { wmanIfSsPkmTEKEntry 18 }
52
53         --
54

```

```

1      -- Table wmanIfsDeviceCertTable
2      --
3
4      wmanIfsDeviceCertTable OBJECT-TYPE
5          SYNTAX      SEQUENCE OF WmanIfsDeviceCertEntry
6          MAX-ACCESS  not-accessible
7          STATUS      current
8          DESCRIPTION
9              "This table describes the PKM device certificates for each
10             SS wireless interface."
11      ::= { wmanIfsPkmObjects 3 }
12
13
14     wmanIfsDeviceCertEntry OBJECT-TYPE
15         SYNTAX      WmanIfsDeviceCertEntry
16         MAX-ACCESS  not-accessible
17         STATUS      current
18         DESCRIPTION
19             "Each entry contains the device certificate of one SS."
20         INDEX      { ifIndex }
21     ::= { wmanIfsDeviceCertTable 1 }
22
23
24     wmanIfsDeviceCertEntry ::= SEQUENCE {
25         wmanIfsDeviceCert      OCTET STRING,
26         wmanIfsDeviceManufCert  OCTET STRING
27     }
28
29
30     wmanIfsDeviceCert OBJECT-TYPE
31         SYNTAX      OCTET STRING
32         MAX-ACCESS  read-only
33         STATUS      current
34         DESCRIPTION
35             "The X509 DER-encoded subscriber station certificate."
36     ::= { wmanIfsDeviceCertEntry 1 }
37
38
39     wmanIfsDeviceManufCert OBJECT-TYPE
40         SYNTAX      OCTET STRING
41         MAX-ACCESS  read-only
42         STATUS      current
43         DESCRIPTION
44             "The X509 DER-encoded manufacturer certificate which is
45             signed by the CA root authority certificate."
46     ::= { wmanIfsDeviceCertEntry 2 }
47
48
49
50
51     --
52     -- Common object group - containing common tables and objects to be
53     -- implemented in both Base Station and Subscriber Station
54     --

```

```

1  -- wmanIfCmnPacketCs contain the Packet Convergence Sublayer objects
2  -- that are common to both Base Station and Subscriber Station
3  wmanIfCmnPacketCs OBJECT IDENTIFIER ::= { wmanIfCommonObjects 1 }
4
5
6  --
7  -- wmanIfCmnCps contain the Common Part Sublayer objects that are common
8  -- to both Base Station and Subscriber Station
9  wmanIfCmnCps OBJECT IDENTIFIER ::= { wmanIfCommonObjects 2 }
10
11 wmanIfCmnCpsServiceFlowTable OBJECT-TYPE
12     SYNTAX      SEQUENCE OF wmanIfCmnCpsServiceFlowEntry
13     MAX-ACCESS  not-accessible
14     STATUS      current
15     DESCRIPTION
16         "This table contains Service Flows that are created in both
17         BS and SS."
18     ::= { wmanIfCmnCps 1 }
19
20 wmanIfCmnCpsServiceFlowEntry OBJECT-TYPE
21     SYNTAX      wmanIfCmnCpsServiceFlowEntry
22     MAX-ACCESS  not-accessible
23     STATUS      current
24     DESCRIPTION
25         "This table provides one row for each service flow, and is
26         indexed by ifIndex and wmanIfCmnCpsServiceFlowId. The
27         ifIndex is provided by a ifEntry for BS or SS with ifType
28         wmanMac. ifIndex is used to identify the SS or BS, while
29         wmanIfCmnCpsServiceFlowId is used to identify the service
30         flow."
31     INDEX       { ifIndex, wmanIfCmnCpsServiceFlowId }
32     ::= { wmanIfCmnCpsServiceFlowTable 1 }
33
34 wmanIfCmnCpsServiceFlowEntry ::= SEQUENCE {
35     wmanIfCpsSfId          Integer32,
36     wmanIfCpsSfCid        INTEGER,
37     wmanIfCpsSfDirection  INTEGER,
38     wmanIfCpsServiceClassIndex  INTEGER,
39     wmanIfCpsSfState      INTEGER,
40     wmanIfCpsServiceClassName  DisplayString,
41     wmanIfCpsTrafficPriority  INTEGER,
42     wmanIfCpsMaxSustainedRate  INTEGER,
43     wmanIfCpsMaxTrafficBurst  INTEGER,
44     wmanIfCpsMinReservedRate  INTEGER,
45     wmanIfCpsToleratedJitter  INTEGER,
46     wmanIfCpsMaxLatency      INTEGER,
47     wmanIfCpsScSchedulingType  wmanUlschedulingType,
48     wmanIfCpsScArqEnable     TruthValue,
49     wmanIfCpsScArqWindowSize  INTEGER,
50     wmanIfCpsScArqFragmentLifetime  INTEGER,
51     wmanIfCpsScArqSyncLossTimeout  INTEGER,
52     wmanIfCpsScArqDeliverInOrder  TruthValue,
53     wmanIfCpsScArqRXPurgeTimeout  INTEGER,
54     wmanIfCpsScFragmentLen    INTEGER,

```

```

1      wmanIfCpsSCMinRsvdTolerableRate  INTEGER
2      }
3
4  wmanIfCpsSfId OBJECT-TYPE
5      SYNTAX      Integer32
6      MAX-ACCESS  read-only
7      STATUS      current
8      DESCRIPTION
9          "A 32 bit quantity that uniquely identifies a service flow
10         to both the subscriber station and base station (BS)."
```

::= { wmanIfCmnCpsServiceFlowEntry 1 }

```

12
13 wmanIfCpsSfCid OBJECT-TYPE
14     SYNTAX      INTEGER
15     MAX-ACCESS  read-only
16     STATUS      current
17     DESCRIPTION
18         "A 16 bit channel identifier to identify the connection
19         being created by DSA."
20     ::= { wmanIfCmnCpsServiceFlowEntry 2 }
21
22 wmanIfCpsSfDirection OBJECT-TYPE
23     SYNTAX      INTEGER {downstream(1),
24                     upstream(2)}
25     MAX-ACCESS  read-only
26     STATUS      current
27     DESCRIPTION
28         "An attribute indicating the service flow is downstream or
29         upstream."
30     ::= { wmanIfCmnCpsServiceFlowEntry 3 }
31
32 wmanIfCpsServiceClassIndex OBJECT-TYPE
33     SYNTAX      INTEGER
34     MAX-ACCESS  read-only
35     STATUS      current
36     DESCRIPTION
37         "The index in wmanIfCpsServiceFlowTable describing the
38         service flow and associated QoS parameters.
39         If no associated entry in wmanIfCpsServiceFlowTable
40         exists, this object returns a value of zero."
41     ::= { wmanIfCmnCpsServiceFlowEntry 4 }
42
43 wmanICpsSfState OBJECT-TYPE
44     SYNTAX      INTEGER {provisionedState(1),
45                     admittedState(2),
46                     activeState(3)}
47     MAX-ACCESS  read-only
48     STATUS      current
49     DESCRIPTION
50         "wmanIfCpsSfState indicates the service flow state:
51         Provisioned, AdmittedState(2), and Active service flow
52         state."
53     REFERENCE
54         "Section 6.4.13.6, in IEEE 802.16REVd/D3-2004"
```

```

1      ::= { wmanIfCmnCpsServiceFlowEntry 5 }
2
3  wmanIfCpsServiceClassName OBJECT-TYPE
4      SYNTAX      DisplayString
5      MAX-ACCESS  read-only
6      STATUS      current
7      DESCRIPTION
8          "Refers to the Service Class Name"
9      REFERENCE   "802.16 Chapter 11.4.8"
10     ::= { wmanIfCmnCpsServiceFlowEntry 6 }
11
12  wmanIfCpsTrafficPriority OBJECT-TYPE
13     SYNTAX      INTEGER
14     MAX-ACCESS  read-only
15     STATUS      current
16     DESCRIPTION
17         "The value of this parameter specifies the priority
18         assigned to a service flow. For uplink service flows,
19         the BS should use this parameter when determining
20         precedence in request service and grant generation,
21         and the SS shall preferentially select contention
22         Request opportunities for Priority Request CIDs
23         based on this priority"
24     REFERENCE   "Section 11.13.7 in IEEE 802.16REVd/D3-2004"
25     ::= { wmanIfCmnCpsServiceFlowEntry 7 }
26
27
28  wmanIfCpsMaxSustainedRate OBJECT-TYPE
29     SYNTAX      INTEGER
30     MAX-ACCESS  read-only
31     STATUS      current
32     DESCRIPTION
33         "This parameter defines the peak information rate
34         of the service. The rate is expressed in bits per
35         second and pertains to the SDUs at the input to
36         the system."
37     REFERENCE   "Section 11.13.8 in IEEE 802.16REVd/D3-2004"
38     ::= { wmanIfCmnCpsServiceFlowEntry 8 }
39
40
41  wmanIfCpsMaxTrafficBurst OBJECT-TYPE
42     SYNTAX      INTEGER
43     MAX-ACCESS  read-only
44     STATUS      current
45     DESCRIPTION
46         "This parameter defines the maximum burst size that
47         must be accommodated for the service."
48     REFERENCE   "Section 11.13.9 in IEEE 802.16REVd/D3-2004"
49     ::= { wmanIfCmnCpsServiceFlowEntry 9 }
50
51
52  wmanIfCpsMinReservedRate OBJECT-TYPE
53     SYNTAX      INTEGER
54     MAX-ACCESS  read-only

```

```

1      STATUS      current
2      DESCRIPTION
3          "This parameter specifies the minimum rate reserved
4          for this service flow."
5      REFERENCE
6          "Section 11.13.10 in IEEE 802.16REvD/D3-2004"
7      ::= { wmanIfCmnCpsServiceFlowEntry 10 }
8
9      wmanIfCpsToleratedJitter OBJECT-TYPE
10     SYNTAX      INTEGER
11     MAX-ACCESS  read-only
12     STATUS      current
13     DESCRIPTION
14         "This parameter defines the Maximum delay
15         variation (jitter) for the connection."
16     REFERENCE
17         "Section 11.13.15 in IEEE 802.16REvD/D3-2004"
18     ::= { wmanIfCmnCpsServiceFlowEntry 11 }
19
20     wmanIfCpsMaxLatency OBJECT-TYPE
21     SYNTAX      INTEGER
22     MAX-ACCESS  read-only
23     STATUS      current
24     DESCRIPTION
25         "The value of this parameter specifies the maximum
26         latency between the reception of a packet by the BS
27         or SS on its network interface and the forwarding
28         of the packet to its RF Interface."
29     REFERENCE
30         "Section 11.13.16 in IEEE 802.16REvD/D3-2004"
31     ::= { wmanIfCmnCpsServiceFlowEntry 12 }
32
33     wmanIfCpsScsSchedulingType OBJECT-TYPE
34     SYNTAX      WmanUlSchedulingType
35     MAX-ACCESS  read-only
36     STATUS      current
37     DESCRIPTION
38         "Specifies the upstream scheduling service used for
39         upstream service flow. If the referenced parameter
40         is not present in the corresponding 802.16 QoS
41         Parameter Set of an upstream service flow, the
42         default value of this object is bestEffort(2)."
43     REFERENCE      "802.16 Chapter 11.4.8"
44     ::= { wmanIfCmnCpsServiceFlowEntry 13 }
45
46     wmanIfCpsScArqEnable OBJECT-TYPE
47     SYNTAX      TruthValue
48     MAX-ACCESS  read-only
49     STATUS      current
50     DESCRIPTION
51         "True(1) ARQ enabling is requested for the connection."
52     ::= { wmanIfCmnCpsServiceFlowEntry 14 }
53
54     wmanIfCpsScArqWindowSize      OBJECT-TYPE

```

```

1          SYNTAX      INTEGER (1 .. 255)
2          MAX-ACCESS  read-only
3          STATUS      current
4          DESCRIPTION
5              "Indicates the maximum number of unacknowledged
6              fragments at any time."
7          ::= { wmanIfCmnCpsServiceFlowEntry 15 }
8
9  wmanIfCpsScArqFragmentLifetime OBJECT-TYPE
10         SYNTAX      INTEGER (0 .. 65535)
11         UNITS        "10 us"
12         MAX-ACCESS  read-only
13         STATUS      current
14         DESCRIPTION
15             "The maximum time interval an ARQ fragment will be
16             managed by the transmitter ARQ machine, once
17             initial transmission of the fragment has occurred.
18             If transmission or retransmission of the fragment
19             is not acknowledged by the receiver before the
20             time limit is reached, the fragment is discarded.
21             A value of 0 means Infinite."
22         ::= { wmanIfCmnCpsServiceFlowEntry 16 }
23
24  wmanIfCpsScArqSyncLossTimeout OBJECT-TYPE
25         SYNTAX      INTEGER (0 .. 65535 )
26         UNITS        "10 us"
27         MAX-ACCESS  read-only
28         STATUS      current
29         DESCRIPTION
30             "The maximum interval before declaring a loss
31             of synchronization of the sender and receiver
32             state machines. A value of 0 means Infinite."
33         ::= { wmanIfCmnCpsServiceFlowEntry 17}
34
35  wmanIfCpsScArqDeliverInOrder OBJECT-TYPE
36         SYNTAX      TruthValue
37         MAX-ACCESS  read-only
38         STATUS      current
39         DESCRIPTION
40             "Indicates whether or not data is to be delivered
41             by the receiving MAC to its client application
42             in the order in which data was handed off to the
43             originating MAC."
44         ::= { wmanIfCmnCpsServiceFlowEntry 18 }
45
46  wmanIfCpsScArqRxPurgeTimeout OBJECT-TYPE
47         SYNTAX      INTEGER (0 .. 65535)
48         UNITS        "10 us"
49         MAX-ACCESS  read-only
50         STATUS      current
51         DESCRIPTION
52             "Indicates the time interval the ARQ window is advanced
53             after a fragment is received. A value of 0 means
54             Infinite."

```



```

1      ::= { wmanIfCmnCpsServiceFlowEntry 19}
2
3  wmanIfCpsScFragmentLen OBJECT-TYPE
4      SYNTAX      INTEGER (32 .. 2040)
5      MAX-ACCESS  read-only
6      STATUS      current
7      DESCRIPTION
8          "The maximum size fragment a transmitter shall form
9          or a receiver shall expect to receive."
10     ::= { wmanIfCmnCpsServiceFlowEntry 20 }
11
12  wmanIfCpsSCMinRsvdTolerableRate OBJECT-TYPE
13      SYNTAX      INTEGER
14      MAX-ACCESS  read-only
15      STATUS      current
16      DESCRIPTION
17          "Minimum Tolerable Traffic Rate = R (bits/sec) with
18          time base T(sec) means the following. Let S denote
19          additional demand accumulated at the MAC SAP of the
20          transmitter during an arbitrary time interval of the
21          length T. Then the amount of data forwarded at the
22          receiver to CS (in bits) during this interval should
23          be not less than min {S, R * T}."
24      REFERENCE  "Section 11.13.11 in IEEE 802.16REvd/D3-2004"
25     ::= { wmanIfCmnCpsServiceFlowEntry 17 }
26
27  wmanIfCmnClassifierRuleTable OBJECT-TYPE
28      SYNTAX      SEQUENCE OF wmanIfCmnClassifierRuleEntry
29      MAX-ACCESS  not-accessible
30      STATUS      current
31      DESCRIPTION
32          "This table contains packet classifier rules associated
33          with service flows."
34     ::= { wmanIfCmnPacketCs 3 }
35
36  wmanIfCmnClassifierRuleEntry OBJECT-TYPE
37      SYNTAX      wmanIfCmnClassifierRuleEntry
38      MAX-ACCESS  not-accessible
39      STATUS      current
40      DESCRIPTION
41          "This table provides one row for each packet classifier
42          rule, and is indexed by ifIndex, wmanIfCmnSfId and
43          wmanIfCmnClassifierRuleIndex. ifIndex identifies the SS.
44          wmanIfCmnSfId identifies the service flow, and
45          wmanIfCmnClassifierRuleIndexAn identifies the packet
46          classifier rule."
47      INDEX { ifIndex, wmanIfCpsSfId, wmanIfCmnClassifierRuleIndex }
48     ::= { wmanIfCmnClassifierRuleTable 1 }
49
50  wmanIfCmnClassifierRuleEntry ::= SEQUENCE {
51      wmanIfCmnClassifierRuleIndex      Unsigned32,
52      wmanIfCmnClassifierRuleServiceFlowId Unsigned32,
53      wmanIfCmnClassifierRulePriority    INTEGER,
54      wmanIfCmnClassifierRuleIpTosLow   OCTET STRING,

```

```

1      wmanIfCmnClassifierRuleIpTosHigh      OCTET STRING,
2      wmanIfCmnClassifierRuleIpTosMask     OCTET STRING,
3      wmanIfCmnClassifierRuleIpProtocol    Integer32,
4      wmanIfCmnClassifierRuleInetAddressType InetAddressType,
5      wmanIfCmnClassifierRuleInetAddress   InetAddress,
6      wmanIfCmnClassifierRuleInetAddressMask InetAddress,
7      wmanIfCmnClassifierRuleInetAddress   InetAddress,
8      wmanIfCmnClassifierRuleInetAddressMask InetAddress,
9      wmanIfCmnClassifierRuleSourcePortStart Integer32,
10     wmanIfCmnClassifierRuleSourcePortEnd  Integer32,
11     wmanIfCmnClassifierRuleDestPortStart  Integer32,
12     wmanIfCmnClassifierRuleDestPortEnd    Integer32,
13     wmanIfCmnClassifierRuleDestMacAddr    MacAddress,
14     wmanIfCmnClassifierRuleDestMacMask    MacAddress,
15     wmanIfCmnClassifierRuleSourceMacAddr   MacAddress,
16     wmanIfCmnClassifierRuleSourceMacMask   MacAddress,
17     wmanIfCmnClassifierRuleEnetProtocolType INTEGER,
18     wmanIfCmnClassifierRuleEnetProtocol   Integer32,
19     wmanIfCmnClassifierRuleUserPriLow     Integer32,
20     wmanIfCmnClassifierRuleUserPriHigh    Integer32,
21     wmanIfCmnClassifierRuleVlanId         Integer32,
22     wmanIfCmnClassifierRuleState          INTEGER,
23     wmanIfCmnClassifierRulePkts          Counter64,
24     wmanIfCmnClassifierRuleRowStatus      RowStatus
25     }
26
27     wmanIfCmnClassifierRuleIndex OBJECT-TYPE
28         SYNTAX      Unsigned32 (1..4294967295)
29         MAX-ACCESS  not-accessible
30         STATUS      current
31         DESCRIPTION
32             "An index is assigned to each classifier in the classifiers
33             table"
34         REFERENCE   ""
35         ::= { wmanIfCmnClassifierRuleEntry 1 }
36
37     wmanIfCmnClassifierRuleServiceFlowId OBJECT-TYPE
38         SYNTAX      Unsigned32 (1..4294967295)
39         MAX-ACCESS  read-write
40         STATUS      current
41         DESCRIPTION
42             "An index assigned to a service flow by SC (SFID)."

```

```

1         value of 64 is used for dynamic service flow classifiers.
2         If the referenced parameter is not present in a classifier,
3         this object reports the default value as defined above"
4 ::= { wmanIfCmnClassifierRuleEntry 3 }
5
6 wmanIfCmnClassifierRuleIpTosLow OBJECT-TYPE
7     SYNTAX      OCTET STRING (SIZE(1))
8     MAX-ACCESS  read-write
9     STATUS      current
10    DESCRIPTION
11        "The low value of a range of TOS byte values. If the
12        referenced parameter is not present in a classifier, this
13        object reports the value of 0."
14    REFERENCE   "802.16 Chapter 11.4.9"
15    ::= { wmanIfCmnClassifierRuleEntry 4 }
16
17 wmanIfCmnClassifierRuleIpTosHigh OBJECT-TYPE
18     SYNTAX      OCTET STRING (SIZE(1))
19     MAX-ACCESS  read-write
20     STATUS      current
21     DESCRIPTION
22        "The 8-bit high value of a range of TOS byte values.
23        If the referenced parameter is not present in a classifier,
24        this object reports the value of 0."
25    REFERENCE   "802.16 Chapter 11.4.9"
26    ::= { wmanIfCmnClassifierRuleEntry 5 }
27
28 wmanIfCmnClassifierRuleIpTosMask OBJECT-TYPE
29     SYNTAX      OCTET STRING (SIZE(1))
30     MAX-ACCESS  read-write
31     STATUS      current
32     DESCRIPTION
33        "The mask value is bitwise ANDed with TOS byte in an IP
34        packet and this value is used check range checking of
35        TosLow and TosHigh. If the referenced parameter is not
36        present in a classifier, this object reports the value
37        of 0."
38    REFERENCE   "802.16 Chapter 11.4.9"
39    ::= { wmanIfCmnClassifierRuleEntry 6 }
40
41 wmanIfCmnClassifierRuleIpProtocol OBJECT-TYPE
42     SYNTAX      Integer32 (0..255)
43     MAX-ACCESS  read-write
44     STATUS      current
45     DESCRIPTION
46        "This object indicates the value of the IP Protocol field
47        required for IP packets to match this rule. If the
48        referenced parameter is not present in a classifier, this
49        object reports the value of 0."
50    REFERENCE   "802.16 Chapter 11.4.9"
51    ::= { wmanIfCmnClassifierRuleEntry 7 }
52
53 wmanIfCmnClassifierRuleInetAddressType OBJECT-TYPE
54     SYNTAX      InetAddressType

```

```

1      MAX-ACCESS  read-write
2      STATUS      current
3      DESCRIPTION
4          "The type of the internet address for
5          wmanIfCmnClassifierRuleInetSourceAddr,
6          wmanIfCmnClassifierRuleInetSourceMask,
7          wmanIfCmnClassifierRuleInetDestAddr, and
8          wmanIfCmnClassifierRuleInetDestMask.
9          If the referenced parameter is not present in a classifier,
10         this object reports the value of ipv4(1)."
```

REFERENCE ""

```

11     ::= { wmanIfCmnClassifierRuleEntry 8 }
12
13
14 wmanIfCmnClassifierRuleInetSourceAddr OBJECT-TYPE
15     SYNTAX      InetAddress
16     MAX-ACCESS  read-write
17     STATUS      current
18     DESCRIPTION
19         "This object specifies the value of the IP Source Address
20         required for packets to match this rule. An IP packet
21         matches the rule when the packet ip source address bitwise
22         ANDed with the wmanIfCmnClassifierRuleInetSourceMask value
23         equals the wmanIfCmnClassifierRuleInetSourceAddr value.
24         If the referenced parameter is not present n a classifier,
25         this object reports the value of 0.0.0.0."
```

REFERENCE "802.16 Chapter 11.4.9"

```

26     ::= { wmanIfCmnClassifierRuleEntry 9 }
27
28
29 wmanIfCmnClassifierRuleInetSourceMask OBJECT-TYPE
30     SYNTAX      InetAddress
31     MAX-ACCESS  read-write
32     STATUS      current
33     DESCRIPTION
34         "This object specifies which bits of a packet's IP Source
35         Address that are compared to match this rule. An IP packet
36         matches the rule when the packet source address bitwise
37         ANDed with the
38         wmanIfCmnClassifierRuleInetSourceMask value equals the
39         wmanIfCmnClassifierRuleInetSourceAddr value.
40         If the referenced parameter is not present in a classifier,
41         this object reports the value of 0.0.0.0."
```

REFERENCE "802.16 Chapter 11.4.9"

```

42     ::= { wmanIfCmnClassifierRuleEntry 10 }
43
44
45 wmanIfCmnClassifierRuleInetDestAddr OBJECT-TYPE
46     SYNTAX      InetAddress
47     MAX-ACCESS  read-write
48     STATUS      current
49     DESCRIPTION
50         "This object specifies the value of the IP Destination
51         Address required for packets to match this rule. An IP
52         packet matches the rule when the packet IP destination
53         address bitwise ANDed with the
54         wmanIfCmnClassifierRuleInetDestMask value equals the
```

```

1         wmanIfCmnClassifierRuleInetDestAddr value.
2         If the referenced parameter is not present in a
3         classifier, this object reports the value of 0.0.0.0."
4     REFERENCE      "802.16 Chapter 11.4.9"
5     ::= { wmanIfCmnClassifierRuleEntry 11 }
6
7     wmanIfCmnClassifierRuleInetDestMask OBJECT-TYPE
8         SYNTAX      InetAddress
9         MAX-ACCESS  read-write
10        STATUS      current
11        DESCRIPTION
12            "This object specifies which bits of a packet's IP
13            Destination Address that are compared to match this rule.
14            An IP packet matches the rule when the packet destination
15            address bitwise ANDed with the
16            wmanIfCmnClassifierRuleInetDestMask value equals the
17            wmanIfCmnClassifierRuleInetDestAddr value.
18            If the referenced parameter is not present in a classifier
19            , this object reports the value of 0.0.0.0."
20        REFERENCE   "802.16 Chapter 11.4.9"
21        ::= { wmanIfCmnClassifierRuleEntry 12}
22
23        wmanIfCmnClassifierRuleSourcePortStart OBJECT-TYPE
24            SYNTAX      Integer32 (0..65535)
25            MAX-ACCESS  read-write
26            STATUS      current
27            DESCRIPTION
28                "This object specifies the low end inclusive range of
29                TCP/UDP source port numbers to which a packet is compared
30                . This object is irrelevant for non-TCP/UDP IP packets.
31                If the referenced parameter is not present in a
32                classifier, this object reports the value of 0."
33            REFERENCE   "802.16 Chapter 11.4.9"
34            ::= { wmanIfCmnClassifierRuleEntry 13 }
35
36        wmanIfCmnClassifierRuleSourcePortEnd OBJECT-TYPE
37            SYNTAX      Integer32 (0..65535)
38            MAX-ACCESS  read-write
39            STATUS      current
40            DESCRIPTION
41                "This object specifies the high end inclusive range of
42                TCP/UDP source port numbers to which a packet is compared.
43                This object is irrelevant for non-TCP/UDP IP packets.
44                If the referenced parameter is not present in a classifier,
45                this object reports the value of 65535."
46            REFERENCE   "802.16 Chapter 11.4.9"
47            ::= { wmanIfCmnClassifierRuleEntry 14 }
48
49        wmanIfCmnClassifierRuleDestPortStart OBJECT-TYPE
50            SYNTAX      Integer32 (0..65535)
51            MAX-ACCESS  read-write
52            STATUS      current
53            DESCRIPTION
54                "This object specifies the low end inclusive range of

```

```

1         TCP/UDP destination port numbers to which a packet is
2         compared. If the referenced parameter is not present
3         in a classifier, this object reports the value of 0."
4     REFERENCE      "802.16 Chapter 11.4.9"
5     ::= { wmanIfCmnClassifierRuleEntry 15 }
6
7     wmanIfCmnClassifierRuleDestPortEnd OBJECT-TYPE
8     SYNTAX         Integer32 (0..65535)
9     MAX-ACCESS     read-write
10    STATUS         current
11    DESCRIPTION
12        "This object specifies the high end inclusive range of
13        TCP/UDP destination port numbers to which a packet is
14        compared. If the referenced parameter is not present
15        in a classifier, this object reports the value of
16        65535."
17    REFERENCE      "802.16 Chapter 11.4.9"
18    ::= { wmanIfCmnClassifierRuleEntry 16 }
19
20    wmanIfCmnClassifierRuleDestMacAddr OBJECT-TYPE
21    SYNTAX         MacAddress
22    MAX-ACCESS     read-write
23    STATUS         current
24    DESCRIPTION
25        "An Ethernet packet matches an entry when its destination
26        MAC address bitwise ANDed with
27        wmanIfCmnClassifierRuleDestMacMask equals the value of
28        wmanIfCmnClassifierRuleDestMacAddr. If the referenced
29        parameter is not present in a classifier, this object
30        reports the value of '000000000000'H."
31    REFERENCE      "802.16 Chapter 11.4.9"
32    ::= { wmanIfCmnClassifierRuleEntry 17 }
33
34    wmanIfCmnClassifierRuleDestMacMask OBJECT-TYPE
35    SYNTAX         MacAddress
36    MAX-ACCESS     read-write
37    STATUS         current
38    DESCRIPTION
39        "An Ethernet packet matches an entry when its destination
40        MAC address bitwise ANDed with
41        wmanIfCmnClassifierRuleDestMacMask equals the value of
42        wmanIfCmnClassifierRuleDestMacAddr. If the referenced
43        parameter is not present in a classifier, this object
44        reports the value of '000000000000'H."
45    REFERENCE      "802.16 Chapter 11.4.9"
46    ::= { wmanIfCmnClassifierRuleEntry 18 }
47
48    wmanIfCmnClassifierRuleSourceMacAddr OBJECT-TYPE
49    SYNTAX         MacAddress
50    MAX-ACCESS     read-write
51    STATUS         current
52    DESCRIPTION
53        "An Ethernet packet matches this entry when its source
54        MAC address bitwise ANDed with

```

```

1           wmanIfCmnClassifierRuleSourceMacMask equals the value
2           of wmanIfCmnClassifierRuleSourceMacAddr. If the
3           referenced parameter is not present in a classifier,
4           this object reports the value of 'FFFFFFFFFFFF'H."
5           REFERENCE      "802.16 Chapter 11.4.9"
6           ::= { wmanIfCmnClassifierRuleEntry 19 }
7
8   wmanIfCmnClassifierRuleSourceMacMask OBJECT-TYPE
9       SYNTAX      MacAddress
10      MAX-ACCESS  read-write
11      STATUS      current
12      DESCRIPTION
13          "An Ethernet packet matches an entry when its destination
14          MAC address bitwise ANDed with
15          wmanIfCmnClassifierRuleSourceMacMask equals the value of
16          wmanIfCmnClassifierRuleSourceMacAddr. If the referenced
17          parameter is not present in a classifier, this object
18          reports the value of '000000000000'H."
19      REFERENCE      "802.16 Chapter 11.4.9"
20      ::= { wmanIfCmnClassifierRuleEntry 20 }
21
22  wmanIfCmnClassifierRuleEnetProtocolType OBJECT-TYPE
23      SYNTAX      INTEGER {none(0),
24                  ethertype(1),
25                  dsap(2)}
26      MAX-ACCESS  read-write
27      STATUS      current
28      DESCRIPTION
29          "This object indicates the format of the layer 3 protocol
30          id in the Ethernet packet. A value of none(0) means that
31          the rule does not use the layer 3 protocol type as a
32          matching criteria. A value of ethertype(1) means that the
33          rule applies only to frames which contains an EtherType
34          value. Ethertype values are contained in packets using
35          the Dec-Intel-Xerox (DIX) encapsulation or the RFC1042
36          Sub-Network Access Protocol (SNAP) encapsulation formats.
37          A value of dsap(2) means that the rule applies only to
38          frames using the IEEE802.3 encapsulation format with a
39          Destination Service Access Point (DSAP) other than 0xAA
40          (which is reserved for SNAP). If the Ethernet frame
41          contains an 802.1P/Q Tag header (i.e. EtherType 0x8100),
42          this object applies to the embedded EtherType field within
43          the 802.1P/Q header. If the referenced parameter is not
44          present in a classifier, this object reports the value of
45          0."
46      REFERENCE      "802.16 Chapter 11.4.9"
47      ::= { wmanIfCmnClassifierRuleEntry 21 }
48
49  wmanIfCmnClassifierRuleEnetProtocol OBJECT-TYPE
50      SYNTAX      Integer32 (0..65535)
51      MAX-ACCESS  read-write
52      STATUS      current
53      DESCRIPTION
54          "If wmanIfCmnClassifierRuleEnetProtocolType is none(0),

```

```

1      this object is ignored when considering whether a packet
2      matches the current rule.
3      If wmanIfCmnClassifierRuleEnetProtocolType is ethertype(1),
4      this object gives the 16-bit value of the EtherType that
5      the packet must match in order to match the rule.
6      If wmanIfCmnClassifierRuleEnetProtocolType is dsap(2), the
7      lower 8 bits of this object's value must match the DSAP
8      byte of the packet in order to match the rule.
9      If wmanIfCmnClassifierRuleEnetProtocolType is mac(3), the
10     lower 8 bits of this object value represent a lower bound
11     (inclusive) of MAC management message type codes matched,
12     and the upper 8 bits of this object value represent the
13     upper bound (inclusive) of matched MAC message type codes.
14     Certain message type codes are excluded from matching, as
15     specified in the reference.
16     If the Ethernet frame contains an 802.1P/Q Tag header
17     (i.e. EtherType 0x8100), this object applies to the
18     embedded EtherType field within the 802.1P/Q header.
19     If the referenced parameter is not present in the
20     classifier, the value of this object is reported as 0."
21     REFERENCE    "802.16 Chapter 11.4.9"
22     ::= { wmanIfCmnClassifierRuleEntry 22 }
23
24 wmanIfCmnClassifierRuleUserPriLow OBJECT-TYPE
25     SYNTAX      Integer32 (0..7)
26     MAX-ACCESS  read-write
27     STATUS      current
28     DESCRIPTION
29         "This object applies only to Ethernet frames using the
30         802.1P/Q tag header (indicated with EtherType 0x8100).
31         Such frames include a 16-bit Tag that contains a 3 bit
32         Priority field and a 12 bit VLAN number.
33         Tagged Ethernet packets must have a 3-bit Priority field
34         within the range of wmanIfCmnClassifierRulePriLow and
35         wmanIfCmnClassifierRulePriHigh in order to match this
36         rule.
37         If the referenced parameter is not present in the
38         classifier, the value of this object is reported as 0."
39     REFERENCE    "802.16 Chapter 11.4.9"
40     ::= { wmanIfCmnClassifierRuleEntry 23 }
41
42 wmanIfCmnClassifierRuleUserPriHigh OBJECT-TYPE
43     SYNTAX      Integer32 (0..7)
44     MAX-ACCESS  read-write
45     STATUS      current
46     DESCRIPTION
47         "This object applies only to Ethernet frames using the
48         802.1P/Q tag header (indicated with EtherType 0x8100).
49         Such frames include a 16-bit Tag that contains a 3 bit
50         Priority field and a 12 bit VLAN number.
51         Tagged Ethernet packets must have a 3-bit Priority
52         field within the range of wmanIfCmnClassifierRulePriLow
53         and wmanIfCmnClassifierRulePriHigh in order to match
54         this rule.

```



```

1           If the referenced parameter is not present in the
2           classifier, the value of this object is reported as 7."
3     REFERENCE   "802.16 Chapter 11.4.9"
4     ::= { wmanIfCmnClassifierRuleEntry 24 }
5
6     wmanIfCmnClassifierRuleVlanId OBJECT-TYPE
7         SYNTAX      Integer32 (0..4095)
8         MAX-ACCESS  read-write
9         STATUS      current
10        DESCRIPTION
11            "This object applies only to Ethernet frames using the
12            802.1P/Q tag header.
13            If this object's value is nonzero, tagged packets must
14            have a VLAN Identifier that matches the value in order
15            to match the rule.
16            Only the least significant 12 bits of this object's
17            value are valid.
18            If the referenced parameter is not present in the
19            classifier, the value of this object is reported as 0."
20        REFERENCE   "802.16 Chapter 11.4.9"
21        ::= { wmanIfCmnClassifierRuleEntry 25 }
22
23        wmanIfCmnClassifierRuleState OBJECT-TYPE
24            SYNTAX      INTEGER {active(1),
25                        inactive(2)}
26            MAX-ACCESS  read-write
27            STATUS      current
28            DESCRIPTION
29                "This object indicates whether or not the classifier is
30                enabled to classify packets to a Service Flow.
31                If the referenced parameter is not present in the
32                classifier, the value of this object is reported
33                as active(1)."
34            REFERENCE   "802.16 Chapter 11.4.9"
35            ::= { wmanIfCmnClassifierRuleEntry 26 }
36
37        wmanIfCmnClassifierRulePkts OBJECT-TYPE
38            SYNTAX      Counter64
39            MAX-ACCESS  read-write
40            STATUS      current
41            DESCRIPTION
42                "This object counts the number of packets that have
43                been classified using this entry."
44            ::= { wmanIfCmnClassifierRuleEntry 27 }
45
46        wmanIfCmnClassifierRuleRowStatus OBJECT-TYPE
47            SYNTAX      RowStatus
48            MAX-ACCESS  read-create
49            STATUS      current
50            DESCRIPTION
51                "This object is used to create a new row or modify or
52                delete an existing row in this table.
53
54                If the implementator of this MIB has chosen not

```

```

1         to implement 'dynamic assignment' of profiles, this
2         object is not useful and should return noSuchName
3         upon SNMP request."
4         ::= { wmanIfCmnClassifierRuleEntry 28 }
5
6     -- Common PKM group
7     -- wmanIfCmnPkmObjects contain the Privacy Sublayer objects that are
8     -- common to both Base Station and Subscriber Station
9     wmanIfCmnPkmObjects OBJECT IDENTIFIER ::= { wmanIfCommonObjects 3 }
10
11     --
12     -- Table wmanIfCmnCryptoSuiteTable
13     --
14
15     wmanIfCmnCryptoSuiteTable OBJECT-TYPE
16         SYNTAX      SEQUENCE OF WmanIfCmnCryptoSuiteEntry
17         MAX-ACCESS  not-accessible
18         STATUS      current
19         DESCRIPTION
20             "This table describes the PKM cryptographic suite
21             capabilities for each SS or BS wireless interface."
22         ::= { wmanIfCmnPkmObjects 4 }
23
24
25     wmanIfCmnCryptoSuiteEntry OBJECT-TYPE
26         SYNTAX      WmanIfCmnCryptoSuiteEntry
27         MAX-ACCESS  not-accessible
28         STATUS      current
29         DESCRIPTION
30             "Each entry contains the cryptographic suite pair that SS
31             or BS supports."
32         INDEX       { ifIndex, wmanIfSsCryptoSuiteIndex }
33         ::= { wmanIfCmnCryptoSuiteTable 1 }
34
35
36     WmanIfCmnCryptoSuiteEntry ::= SEQUENCE {
37         wmanIfSsCryptoSuiteIndex          Integer32,
38         wmanIfCmnCryptoSuiteDataEncryptAlg  INTEGER,
39         wmanIfCmnCryptoSuiteDataAuthentAlg  INTEGER,
40         wmanIfCmnCryptoSuiteTEKEncryptAlg   INTEGER
41     }
42
43
44     wmanIfSsCryptoSuiteIndex OBJECT-TYPE
45         SYNTAX      Integer32
46         MAX-ACCESS  not-accessible
47         STATUS      current
48         DESCRIPTION
49             "The index for a cryptographic suite row."
50         ::= { wmanIfCmnCryptoSuiteEntry 1 }
51
52
53     wmanIfCmnCryptoSuiteDataEncryptAlg OBJECT-TYPE
54         SYNTAX      INTEGER { none(0),

```

```

1           des56CbcMode(1) }
2     MAX-ACCESS read-only
3     STATUS current
4     DESCRIPTION
5         "The value of this object is the data encryption algorithm
6         for this cryptographic suite capability."
7     REFERENCE
8         "IEEE 802.16 standard; Table 301"
9     ::= { wmanIfCmnCryptoSuiteEntry 2 }
10
11
12     wmanIfCmnCryptoSuiteDataAuthentAlg OBJECT-TYPE
13         SYNTAX INTEGER { none(0) }
14         MAX-ACCESS read-only
15         STATUS current
16         DESCRIPTION
17             "The value of this object is the data authentication
18             algorithm for this cryptographic suite capability."
19         REFERENCE
20             "IEEE 802.16 standard; Table 302"
21         ::= { wmanIfCmnCryptoSuiteEntry 3 }
22
23
24     wmanIfCmnCryptoSuiteTEKEncryptAlg OBJECT-TYPE
25         SYNTAX INTEGER { tripleDES(0),
26             rsa1024(1) }
27         MAX-ACCESS read-only
28         STATUS current
29         DESCRIPTION
30             "The value of this object is the TEK key encryption
31             algorithm for this cryptographic suite capability."
32         REFERENCE
33             "IEEE 802.16 standard; Table 303"
34         ::= { wmanIfCmnCryptoSuiteEntry 4 }
35
36     --
37     -- wmanIfCmnOfdmPhy contain the OFDM PHY objects that are common to both
38     -- Base Station and Subscriber Station. When the objects are implemented
39     -- in the BS, they should have the read-write access. When the objects
40     -- are implemented the SS, they should have the read-only access.
41     --
42     wmanIfCmnOfdmPhy OBJECT IDENTIFIER ::= { wmanIfCommonObjects 4 }
43
44     wmanIfOfdmUplinkChannelTable OBJECT-TYPE
45         SYNTAX SEQUENCE OF wmanIfOfdmUplinkChannelEntry
46         MAX-ACCESS not-accessible
47         STATUS current
48         DESCRIPTION
49             "This table contains UCD channel attributes, defining the
50             transmission characteristics of uplink channels"
51         REFERENCE
52             "Section 11.3.1, table 276 and 279, in IEEE
53             802.16REVd/D3-2004"
54         ::= { wmanIfCmnOfdmPhy 1 }

```

```

1
2  wmanIfOfdmUplinkChannelEntry OBJECT-TYPE
3      SYNTAX      wmanIfOfdmUplinkChannelEntry
4      MAX-ACCESS  not-accessible
5      STATUS      current
6      DESCRIPTION
7          "This table provides one row for each uplink channel of
8          multi-sector BS, and is indexed by BS ifIndex. An entry
9          in this table exists for each ifEntry of BS with an
10         iftype of wmanUpstream.
11         The objects in each entry will be implemented as
12         read-create in BS and read-only in SS."
13     INDEX { ifIndex }
14     ::= { wmanIfOfdmUplinkChannelTable 1 }
15
16  wmanIfOfdmUplinkChannelEntry ::= SEQUENCE {
17      wmanIfOfdmCtBasedResvTimeout      INTEGER,
18      wmanIfOfdmBwReqOppSize            INTEGER,
19      wmanIfOfdmRangReqOppSize          INTEGER,
20      wmanIfOfdmUplinkCenterFreq        INTEGER,
21      wmanIfOfdmUlAntennaNmr            INTEGER,
22      wmanIfOfdmSubChReqRegionFull      INTEGER,
23      wmanIfOfdmSubChFocusCtCode        INTEGER,
24      wmanIfOfdmChannelWidth            INTEGER,
25      wmanIfOfdmUplinkChannelRowStatus  RowStatus
26  }
27
28  wmanIfOfdmCtBasedResvTimeout OBJECT-TYPE
29      SYNTAX      INTEGER (1..255)
30      MAX-ACCESS  read-write
31      STATUS      current
32      DESCRIPTION
33          "The number of UL-MAPS to receive before contention-based
34          reservation is attempted again for the same connection."
35      REFERENCE
36          "Section 11.3.1, table 276, in IEEE 802.16REVd/D3-2004"
37      ::= { wmanIfOfdmUplinkChannelEntry 1 }
38
39  wmanIfOfdmBwReqOppSize OBJECT-TYPE
40      SYNTAX      INTEGER (1..65535)
41      MAX-ACCESS  read-write
42      STATUS      current
43      DESCRIPTION
44          " Size (in units of PS) of PHY payload that SS may use to
45          format and transmit a bandwidth request message in a
46          contention request opportunity. The value includes all
47          PHY overhead as well as allowance for the MAC data the
48          message may hold."
49      REFERENCE
50          "Section 11.3.1, table 276, in IEEE 802.16REVd/D3-2004"
51      ::= { wmanIfOfdmUplinkChannelEntry 2 }
52
53  wmanIfOfdmRangReqOppSize OBJECT-TYPE
54      SYNTAX      INTEGER (1..65535)

```

```

1      UNITS      "PS"
2      MAX-ACCESS read-write
3      STATUS     current
4      DESCRIPTION
5          " Size (in units of PS) of PHY payload that SS may use to
6          format and transmit a RNG-REQ message in a contention
7          request opportunity. The value includes all PHY overhead
8          as well as allowance for the MAC data the message may
9          hold and the maximum SS/BS roundtrip propagation delay."
10     REFERENCE
11         "Section 11.3.1, table 276, in IEEE 802.16REVd/D3-2004"
12     ::= { wmanIfOfdmUplinkChannelEntry 3 }
13
14     wmanIfOfdmUplinkCenterFreq OBJECT-TYPE
15         SYNTAX      INTEGER
16         UNITS      "KHZ"
17         MAX-ACCESS read-write
18         STATUS     current
19         DESCRIPTION
20             " Uplink center frequency (KHz)"
21         REFERENCE
22             "Section 11.3.1, table 276, in IEEE 802.16REVd/D3-2004"
23         ::= { wmanIfOfdmUplinkChannelEntry 4 }
24
25     wmanIfOfdmUplAntennaNmr OBJECT-TYPE
26         SYNTAX      INTEGER
27         MAX-ACCESS read-write
28         STATUS     current
29         DESCRIPTION
30             " Antenna number or pointing reference"
31         REFERENCE
32             "Section 11.3.1, table 276, in IEEE 802.16REVd/D3-2004"
33         ::= { wmanIfOfdmUplinkChannelEntry 5 }
34
35     wmanIfOfdmSubChReqRegionFull OBJECT-TYPE
36         SYNTAX      INTEGER {oneSubchannel(0),
37                             twoSubchannels(1),
38                             fourSubchannels(2),
39                             eightSubchannels(3),
40                             sixteenSubchannels(4)}
41         MAX-ACCESS read-write
42         STATUS     current
43         DESCRIPTION
44             "Bits 0..2 Number of subchannels used by each transmit
45             opportunity when REQ Region-Full is allocated in
46             subchannelization region, per the following enumeration:
47             0: 1 Subchannel.
48             1: 2 Subchannels.
49             2: 4 Subchannels.
50             3: 8 Subchannels.
51             4: 16 Subchannels.
52             5-7: Shall not be used.
53             Bits 3..7: Number of OFDM symbols used by each transmit
54             opportunity when REQ Region-Full is allocated in

```

```

1         subchannelization region.
2     REFERENCE
3         Section 11.3.1, table 279, in IEEE 802.16REVd/D3-2004"
4     ::= { wmanIfOfdmUplinkChannelEntry 6 }
5
6     wmanIfOfdmSubChFocusCtCode OBJECT-TYPE
7         SYNTAX      INTEGER
8         MAX-ACCESS  read-write
9         STATUS      current
10        DESCRIPTION
11            "Number of contention codes (CSE) that shall only be used to
12            request a subchannelized allocation. Default value 0.
13        Allowed
14            values 0-48."
15        REFERENCE
16            "Section 11.3.1, table 279, in IEEE 802.16REVd/D3-2004"
17        ::= { wmanIfOfdmUplinkChannelEntry 7 }
18
19        wmanIfOfdmChannelwidth OBJECT-TYPE
20            SYNTAX      INTEGER
21            UNITS      "10KHz"
22            MAX-ACCESS  read-write
23            STATUS      current
24            DESCRIPTION
25                "Channel width, increments of 10 KHz Shall not be used
26                in license-exempt bands."
27            ::= { wmanIfOfdmUplinkChannelEntry 8 }
28
29        wmanIfOfdmUplinkChannelRowStatus OBJECT-TYPE
30            SYNTAX      RowStatus
31            MAX-ACCESS  read-create
32            STATUS      current
33            DESCRIPTION
34                "This object is used to create a new row or modify or
35                delete an existing row in this table.
36
37                If the implementator of this MIB has choosen not
38                to implement 'dynamic assignment' of profiles, this
39                object is not useful and should return noSuchName
40                upon SNMP request."
41            ::= { wmanIfOfdmUplinkChannelEntry 9 }
42
43        wmanIfOfdmDownlinkChannelTable OBJECT-TYPE
44            SYNTAX      SEQUENCE OF wmanIfOfdmDownlinkChannelEntry
45            MAX-ACCESS  not-accessible
46            STATUS      current
47            DESCRIPTION
48                "This table contains DCD channel attributes, defining the
49                transmission characteristics of downlink channels"
50            REFERENCE
51                "Section 11.4.1, Table 286, in IEEE 802.16REVd/D3-2004"
52            ::= { wmanIfCmnOfdmPhy 2 }
53
54        wmanIfOfdmDownlinkChannelEntry OBJECT-TYPE

```

```

1      SYNTAX      wmanIfOfdmDownlinkChannelEntry
2      MAX-ACCESS  not-accessible
3      STATUS      current
4      DESCRIPTION
5          "This table provides one row for each downlink channel of
6          multi-sector BS, and is indexed by BS ifIndex. An entry
7          in this table exists for each ifEntry of BS with an
8          iftype of wmanDownstream.
9          The objects in each entry will be implemented as
10         read-create in BS and read-only in SS."
11     INDEX { ifIndex }
12     ::= { wmanIfOfdmDownlinkChannelTable 1 }
13
14     wmanIfOfdmDownlinkChannelEntry ::= SEQUENCE {
15         wmanIfOfdmBSEIRP                INTEGER,
16         wmanIfOfdmChannelNumber         INTEGER,
17         wmanIfOfdmTTG                   INTEGER,
18         wmanIfOfdmRTG                   INTEGER,
19         wmanIfOfdmInitRngMaxRSS         INTEGER,
20         wmanIfOfdmChSwitchFrameNmr     INTEGER,
21         wmanIfOfdmDlAntennaNmr         INTEGER,
22         wmanIfOfdmDownlinkCenterFreq   INTEGER,
23         wmanIfOfdmMacVersion            INTEGER,
24         wmanIfOfdmDownlinkChannelRowStatus RowStatus
25     }
26
27     wmanIfOfdmBSEIRP OBJECT-TYPE
28         SYNTAX      INTEGER
29         UNITS       "dbm"
30         MAX-ACCESS  read-write
31         STATUS      current
32         DESCRIPTION
33             " Signed in units of 1 dBm."
34         REFERENCE
35             "Section 11.4.1, table 286, in IEEE 802.16REVd/D3-2004"
36         ::= { wmanIfOfdmDownlinkChannelEntry 1 }
37
38     wmanIfOfdmChannelNumber OBJECT-TYPE
39         SYNTAX      INTEGER
40         MAX-ACCESS  read-write
41         STATUS      current
42         DESCRIPTION
43             " Downlink channel number as defined in 8.5.
44             Used for license-exempt operation only."
45         REFERENCE
46             "Section 11.4.1, table 286, in IEEE 802.16REVd/D3-2004"
47         ::= { wmanIfOfdmDownlinkChannelEntry 2 }
48
49     wmanIfOfdmTTG OBJECT-TYPE
50         SYNTAX      INTEGER
51         MAX-ACCESS  read-write
52         STATUS      current
53         DESCRIPTION
54             " Transmit / Receive Transition Gap."

```

```

1      REFERENCE
2      "Section 11.4.1, table 286, in IEEE 802.16REVd/D3-2004"
3      ::= { wmanIfOfdmDownlinkChannelEntry 3 }
4
5      wmanIfOfdmRTG OBJECT-TYPE
6          SYNTAX      INTEGER
7          MAX-ACCESS  read-write
8          STATUS      current
9          DESCRIPTION
10             " Receive / Transmit Transition Gap."
11          REFERENCE
12             "Section 11.4.1, table 286, in IEEE 802.16REVd/D3-2004"
13             ::= { wmanIfOfdmDownlinkChannelEntry 4 }
14
15      wmanIfOfdmInitRngMaxRSS OBJECT-TYPE
16          SYNTAX      INTEGER
17          UNITS        "dbm"
18          MAX-ACCESS  read-write
19          STATUS      current
20          DESCRIPTION
21             " Initial Ranging Max. Received Signal Strength at BS
22             Signed in units of 1 dBm."
23          REFERENCE
24             "Section 11.4.1, table 286, in IEEE 802.16REVd/D3-2004"
25             ::= { wmanIfOfdmDownlinkChannelEntry 5 }
26
27      wmanIfOfdmChSwitchFrameNmr OBJECT-TYPE
28          SYNTAX      INTEGER
29          MAX-ACCESS  read-write
30          STATUS      current
31          DESCRIPTION
32             " Channel switch frame number as defined in 6.4.14.7,
33             Used for license-exempt operation only."
34          REFERENCE
35             "Section 11.4.1, table 286, in IEEE 802.16REVd/D3-2004"
36             ::= { wmanIfOfdmDownlinkChannelEntry 6 }
37
38      wmanIfOfdmDlAntennaNmr OBJECT-TYPE
39          SYNTAX      INTEGER
40          MAX-ACCESS  read-write
41          STATUS      current
42          DESCRIPTION
43             " Antenna number or pointing reference."
44          REFERENCE
45             "Section 11.4.1, table 286, in IEEE 802.16REVd/D3-2004"
46             ::= { wmanIfOfdmDownlinkChannelEntry 7 }
47
48      wmanIfOfdmDownlinkCenterFreq OBJECT-TYPE
49          SYNTAX      INTEGER
50          UNITS        "KHZ"
51          MAX-ACCESS  read-write
52          STATUS      current
53          DESCRIPTION
54             " Downlink center frequency (kHz)."
```



```

1      REFERENCE
2      "Section 11.4.1, table 286, in IEEE 802.16REVd/D3-2004"
3      ::= { wmanIfOfdmDownlinkChannelEntry 8 }
4
5      wmanIfOfdmMacVersion OBJECT-TYPE
6          SYNTAX      INTEGER
7          MAX-ACCESS  read-write
8          STATUS      current
9          DESCRIPTION
10             " This parameter specifies the version of 802.16 to which
11             the message originator conforms."
12          REFERENCE
13             "Section 11.4.1, table 286, in IEEE 802.16REVd/D3-2004"
14             ::= { wmanIfOfdmDownlinkChannelEntry 9 }
15
16      wmanIfOfdmDownlinkChannelRowStatus OBJECT-TYPE
17          SYNTAX      RowStatus
18          MAX-ACCESS  read-create
19          STATUS      current
20          DESCRIPTION
21             "This object is used to create a new row or modify or
22             delete an existing row in this table.
23
24             If the implementator of this MIB has chosen not
25             to implement 'dynamic assignment' of profiles, this
26             object is not useful and should return noSuchName
27             upon SNMP request."
28             ::= { wmanIfOfdmDownlinkChannelEntry 10 }
29
30      wmanIfOfdmUcdBurstProfileTable OBJECT-TYPE
31          SYNTAX      SEQUENCE OF wmanIfOfdmUcdBurstProfileEntry
32          MAX-ACCESS  not-accessible
33          STATUS      current
34          DESCRIPTION
35             "This table contains UCD burst profiles for each uplink
36             channel"
37          REFERENCE
38             "Section 11.3.1.1, table 281 and 284, in IEEE
39             802.16REVd/D3-2004"
40             ::= { wmanIfCmnOfdmPhy 3 }
41
42      wmanIfOfdmUcdBurstProfileEntry OBJECT-TYPE
43          SYNTAX      wmanIfOfdmUcdBurstProfileEntry
44          MAX-ACCESS  not-accessible
45          STATUS      current
46          DESCRIPTION
47             "This table provides one row for each UCD burst profile.
48             This table is double indexed. The primary index is an
49             ifIndex with an ifType of wmanUpstream. The secondary
50             index
51             is wmanIfOfdmOfdmUcdBurstProfIndex.
52             The objects in each entry will be implemented as
53             read-create in BS and read-only in SS."
54          INDEX { ifIndex, wmanIfOfdmOfdmUcdBurstProfIndex }

```

```

1      ::= { wmanIfOfdmUcdBurstProfileTable 1 }
2
3  wmanIfOfdmUcdBurstProfileEntry ::= SEQUENCE {
4      wmanIfOfdmOfdmUcdBurstProfIndex    INTEGER,
5      wmanIfOfdmUplinkFrequency          INTEGER,
6      wmanIfOfdmUcdAntennaNmr           INTEGER,
7      wmanIfOfdmUcdFecCodeType           INTEGER,
8      wmanIfOfdmFocusCtPowerBoost        INTEGER,
9      wmanIfOfdmUcdBurstProfileRowStatus RowStatus
10     }
11
12  wmanIfOfdmOfdmUcdBurstProfIndex OBJECT-TYPE
13      SYNTAX      INTEGER
14      MAX-ACCESS  not-accessible
15      STATUS      current
16      DESCRIPTION
17          "ifIndex and wmanIfOfdmOfdmUcdBurstProfIndex uniquely
18           identify an entry in the wmanIfOfdmUcdBurstProfileTable."
19      ::= { wmanIfOfdmUcdBurstProfileEntry 1 }
20
21  wmanIfOfdmUplinkFrequency OBJECT-TYPE
22      SYNTAX      INTEGER
23      UNITS       "KHZ"
24      MAX-ACCESS  read-write
25      STATUS      current
26      DESCRIPTION
27          "Uplink Frequency (kHz)."

```

```

1          qpskCtc1-2(12),
2          qpskCtc2-3(13),
3          qpskCtc3-4(14),
4          sixteenQamCtc3-4(16),
5          sixteenQamCtc2-3(17),
6          sixtyFourQamCtc3-4(18)}
7  MAX-ACCESS  read-write
8  STATUS      current
9  DESCRIPTION
10     " 0= QPSK (RS+CC/CC) 1/2
11       1= QPSK (RS+CC/CC) 3/4
12       2= 16-QAM (RS+CC/CC) 1/2
13       3= 16-QAM (RS+CC/CC) 3/4
14       4= 64-QAM (RS+CC/CC) 2/3
15       5= 64-QAM (RS+CC/CC) 3/4
16       6= QPSK (BTC) 1/2
17       7= QPSK (BTC) 3/4
18       8= 16-QAM (BTC) 3/5
19       9= 16-QAM (BTC) 4/5
20       10 = 64-QAM (BTC) 2/3
21       11 = 64-QAM (BTC) 5/6
22       12 = QPSK (CTC) 1/2
23       13 = QPSK (CTC) 2/3
24       14 = QPSK (CTC) 3/4
25       15 = 16-QAM (CTC) 1/2
26       16 = 16-QAM (CTC) 3/4
27       17 = 64-QAM (CTC) 2/3
28       18 = 64-QAM (CTC) 3/4
29       19-255 Reserved."
30  REFERENCE
31     "Section 11.3.1.1, table 284, in IEEE 802.16REVd/D3-2004"
32     ::= { wmanIfOfdmUcdBurstProfileEntry 4 }
33
34  wmanIfOfdmFocusCtPowerBoost OBJECT-TYPE
35      SYNTAX      INTEGER
36      MAX-ACCESS  read-write
37      STATUS      current
38      DESCRIPTION
39          "The power boost in dB of focused contention carriers, as
40          described in 8.3.6.3.3."
41      REFERENCE
42          "Section 11.3.1.1, table 284, in IEEE 802.16REVd/D3-2004"
43          ::= { wmanIfOfdmUcdBurstProfileEntry 5 }
44
45  wmanIfOfdmUcdBurstProfileRowStatus OBJECT-TYPE
46      SYNTAX      RowStatus
47      MAX-ACCESS  read-create
48      STATUS      current
49      DESCRIPTION
50          "This object is used to create a new row or modify or
51          delete an existing row in this table.
52
53          If the implementator of this MIB has chosen not
54          to implement 'dynamic assignment' of profiles, this

```

```

1         object is not useful and should return noSuchName
2         upon SNMP request."
3         ::= { wmanIfOfdmUcdBurstProfileEntry 6 }
4
5 wmanIfOfdmDcdBurstProfileTable OBJECT-TYPE
6     SYNTAX      SEQUENCE OF wmanIfOfdmDcdBurstProfileEntry
7     MAX-ACCESS  not-accessible
8     STATUS      current
9     DESCRIPTION
10        "This table provides one row for each DCD burst profile.
11        This table is double indexed. The primary index is an
12        ifIndex with an ifType of wmanDownstream. The secondary
13 index
14        is wmanIfOfdmOfdmDcdBurstProfIndex"
15        ::= { wmanIfCmnOfdmPhy 4 }
16
17
18 wmanIfOfdmDcdBurstProfileEntry OBJECT-TYPE
19     SYNTAX      wmanIfOfdmDcdBurstProfileEntry
20     MAX-ACCESS  not-accessible
21     STATUS      current
22     DESCRIPTION
23        "This table provides one row for each DCD burst profile.
24        This table is double indexed. The primary index is an
25        ifIndex with an ifType of wmanDownstream. The secondary
26 index
27        is wmanIfOfdmOfdmDcdBurstProfIndex.
28        The objects in each entry will be implemented as
29        read-create in BS and read-only in SS."
30     INDEX { ifIndex, wmanIfOfdmOfdmDcdBurstProfIndex }
31     ::= { wmanIfOfdmDcdBurstProfileTable 1 }
32
33 wmanIfOfdmDcdBurstProfileEntry ::= SEQUENCE {
34     wmanIfOfdmOfdmDcdBurstProfIndex    INTEGER,
35     wmanIfOfdmDownlinkFrequency       INTEGER,
36     wmanIfOfdmDcdAntennaNmr           INTEGER,
37     wmanIfOfdmDcdFecCodeType          INTEGER,
38     wmanIfOfdmDiucMandatoryExitThresh INTEGER,
39     wmanIfOfdmDiucMinEntryThresh      INTEGER,
40     wmanIfOfdmDcdBurstProfileRowStatus RowStatus
41     }
42
43 wmanIfOfdmOfdmDcdBurstProfIndex OBJECT-TYPE
44     SYNTAX      INTEGER
45     MAX-ACCESS  not-accessible
46     STATUS      current
47     DESCRIPTION
48        "ifIndex and wmanIfOfdmOfdmDcdBurstProfIndex uniquely
49        identify an entry in the wmanIfOfdmOfdmDcdBurstProfIndex."
50     ::= { wmanIfOfdmDcdBurstProfileEntry 1 }
51
52 wmanIfOfdmDownlinkFrequency OBJECT-TYPE
53     SYNTAX      INTEGER
54     UNITS       "KHZ"

```

```

1      MAX-ACCESS  read-write
2      STATUS      current
3      DESCRIPTION
4          "Downlink Frequency (kHz)."

```

```

1           14 = QPSK (CTC) ¾
2           15 = 16-QAM (CTC) ½
3           16 = 16-QAM (CTC) ¾
4           17 = 64-QAM (CTC) 2/3
5           18 = 64-QAM (CTC) ¾
6           19-255 Reserved."
7     REFERENCE
8       "Section 11.4.1, table 290, in IEEE 802.16REVd/D3-2004"
9     ::= { wmanIfOfdmDcdBurstProfileEntry 4 }
10
11    wmanIfOfdmDiucMandatoryExitThresh OBJECT-TYPE
12      SYNTAX      INTEGER
13      MAX-ACCESS  read-write
14      STATUS      current
15      DESCRIPTION
16        "DIUC mandatory exit threshold: 0-63.75 dB CINR at or below
17        where this DIUC can no longer be used and where this change
18        to a more robust DIUC is required, in 0.25 dB units."
19      REFERENCE
20        "Section 11.4.1, table 290, in IEEE 802.16REVd/D3-2004"
21      ::= { wmanIfOfdmDcdBurstProfileEntry 5 }
22
23    wmanIfOfdmDiucMinEntryThresh OBJECT-TYPE
24      SYNTAX      INTEGER
25      MAX-ACCESS  read-write
26      STATUS      current
27      DESCRIPTION
28        "DIUC minimum entry threshold: 0-63.75 dB The minimum CINR
29        required to start using this DIUC when changing from a more
30        robust DIUC is required, in 0.25 dB units."
31      REFERENCE
32        "Section 11.4.1, table 290, in IEEE 802.16REVd/D3-2004"
33      ::= { wmanIfOfdmDcdBurstProfileEntry 6 }
34
35
36    wmanIfOfdmDcdBurstProfileRowStatus OBJECT-TYPE
37      SYNTAX      RowStatus
38      MAX-ACCESS  read-create
39      STATUS      current
40      DESCRIPTION
41        "This object is used to create a new row or modify or
42        delete an existing row in this table.
43
44        If the implementator of this MIB has chosen not
45        to implement 'dynamic assignment' of profiles, this
46        object is not useful and should return noSuchName
47        upon SNMP request."
48      ::= { wmanIfOfdmDcdBurstProfileEntry 7 }
49
50    END
51
52

```

