# Updates to Section 8 on MAC Management as the Result of Letter Ballot Processing

Authored by Most of the MAC Group

# 8. MAC Layer Management Entity

## 8.1. Synchronization

All nodes within a single BSS shall be synchronized to a common clock using the mechanisms defined in this section.

### 8.1.1. Basic Approach

All stations shall maintain a local Synchronization Timer. A Timing Synchronization Function (TSF) keeps the timers for all stations in the same BSS synchronized.

#### 8.1.1.1. TSF for Infrastructure Networks

In an infrastructure network, the AP shall be the timing master and shall perform the Timing Synchronization Function. The AP shall initialize the TSF timer such that simultaneously started APs are not synchronized. To synchronize the other stations in a BSS, the AP shall periodically transmit special frames called Beacons that contain a copy of its Synchronization Timer. Receiving stations shall always accept the timing information in Beacons sent from the AP servicing their BSS. If the station's Synchronization Timer is different from the timestamp in the received Beacon, they shall set their local timer to the received timestamp value.

Beacons shall be generated for transmission by the AP once every aBeacon_Interval time units.

#### 8.1.1.2. TSF for Ad Hoc Networks

The Timing Synchronization Function in an ad hoc network is implemented via a distributed algorithm that is performed by all of the members of the BSS. All stations in the BSS shall transmit Beacons according to an algorithm to be specified below. Stations receiving a Beacon from another station in the same BSS shall adjust their Synchronization Timers towards the Beacon's timestamp value in a manner to be specified below.

It is permitted that a station within an ad-hoc BSS may scan for a better BSS within the same ESS. Within an ad-hoc network, all Beacons and probe-responses carry a TSF time element. A station receiving such a frame from another BSS with the same ESS ID will compare the TSF time with its own TSF time. If the TSF time of the received frame is later than its own TSF time, it will adopt the BSS-ID, channel synchronization information and TSF time contained in that received frame.

### 8.1.2. Maintaining Synchronization

Each station shall maintain a TSF timer with modulus 2e64 counting in increments of microseconds. Stations expect to receive Beacons at a nominal rate. The interval between Beacons is defined by the aBeacon_Interval parameter of the station. A station sending a Beacon shall set the value of the Beacon's timestamp so that it equals the value of the station's TSF timer at the time that the first ~~MAC~~ bit of the timestamp~~Beacon~~ is transmitted to the PHY adjusted by adding the transmitting station's delays through its local PHY. The algorithms below define a mechanism that maintains the synchronization of the TSF timers in a BSS within 4 microseconds plus the maximum propagation delay of the PHY.

#### 8.1.2.1. Beacon Generation in Infrastructure Networks

The access point shall define the timing for the entire BSS by transmitting Beacons according to the aBeacon_Interval parameter within the AP. This defines a series of Target Beacon Transmission Times (TBTTs) exactly aBeacon_Interval time units apart, time zero is defined to be a TBTT. At each TBTT, the

AP shall schedule a Beacon as the next frame for transmission. If the medium is sensed to be unavailable, the AP shall delay the actual transmission of a Beacon according to the CSMA medium access rules specified in Section 5.

NOTE: Though the transmission of a Beacon may be delayed because of CSMA deferrals, subsequent Beacons will be scheduled at the nominal beacon interval. This is shown in Figure 8-1

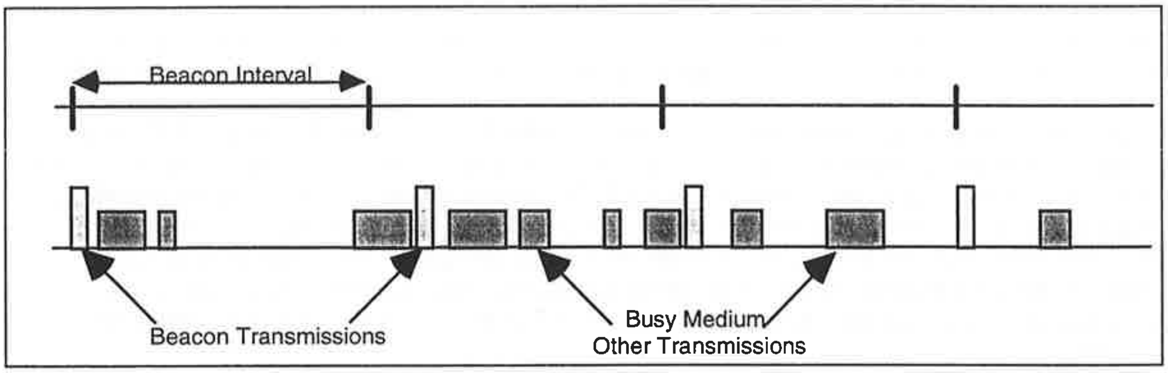

**Figure 8-1 — Beacon transmission on a busy network.**

### 8.1.2.2. Beacon Generation in Ad Hoc Networks

Beacon generation in an ad hoc network is distributed. All members of the BSS participate in Beacon generation. Each station shall maintain its own TSF timer which is used for aBeacon_Interval timing. This defines a series of Target Beacon Transmission Times (TBTTs) exactly aBeacon_Interval time units apart, time zero is defined to be a TBTT. At each TBTT the station shall 1) calculate a random delay, 2) wait for the period of the random delay, 3) if no Beacon has arrived during the delay period, send a Beacon. See Figure 8-2.

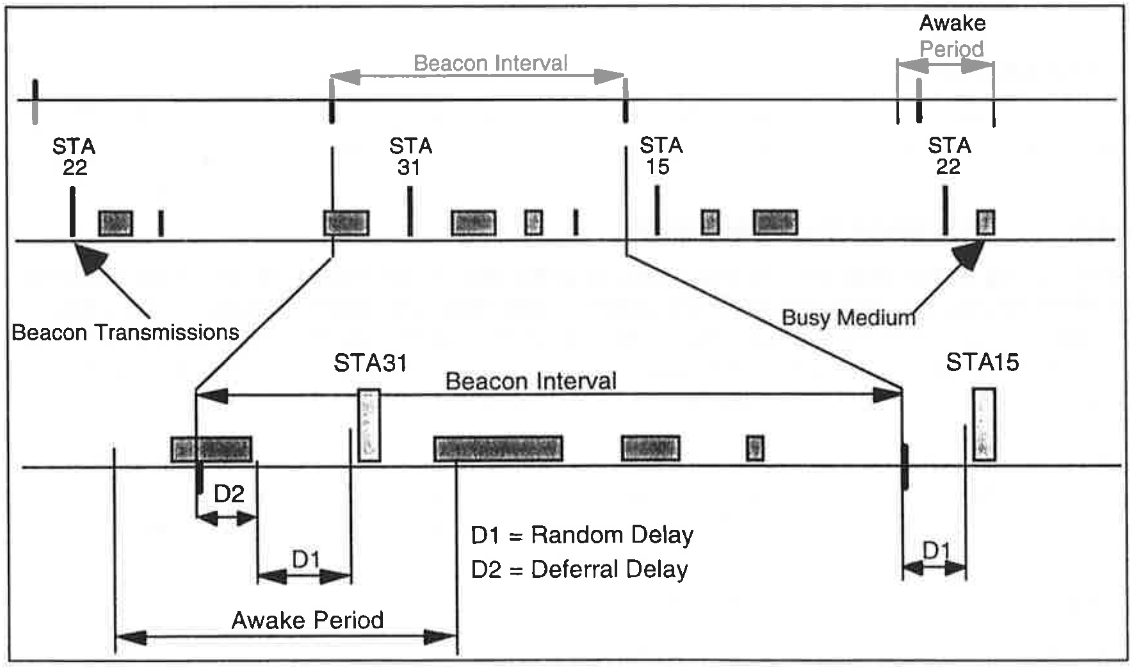

**Figure 8-2 — Beacon transmission in an Ad Hoc network.**

The Beacon transmission will always occur during the Awake Period of stations that are operating in a low power mode. This is described in more detail in Section 7.2.

#### 8.1.2.3. Synchronization Timer Accuracy

The Beacon's timestamp field shall not be filled in until after the CSMA deferral on the Beacon transmission. ~~The start of the MAC frame is used as the timing reference.~~The timestamp value in the Beacon frame shall be~~is~~ the value of the TSF timer at the instant that the first bit of the timestamp~~MAC frame~~ is transmitted to the PHY adjusted by adding the transmitting station's delays through its local PHY.

Upon receiving a Beacon frame~~BSS~~ with a valid CRC and BSS-ID, a Station shall update its TSF timer according to the following algorithm: The received timestamp value shall be adjusted by adding an amount equal to the receiving station's delay through its local PHY components plus the time since the first ~~MAC~~ bit of the timestamp was received at the MAC/PHY interface. In the case of an infrastructure BSS, the station's TSF timer shall then be set to the adjusted value of the timestamp. In the case of an ad hoc BSS, the station's TSF timer shall be set to the value of the adjusted received timestamp, if the value of the adjusted timestamp is greater than the value of the station's TSF timer. The accuracy of the TSF timer shall be +/- 0.01%.

### 8.1.3. Acquiring Synchronization, Scanning

A Station shall perform a scan whenever its aScan_State variable is SCAN.

A Station shall operate in either a Passive Scanning mode or an Active Scanning mode depending on the current value of the system variable aScan_Mode, which can take the values PASSIVE or ACTIVE.

#### 8.1.3.1. Passive Scanning

If a Station's aScan_Mode variable is PASSIVE, the station shall listen to each channel scanned for a maximum duration aPassive_Scan_Duration.

#### 8.1.3.2. Active Scanning

Active scanning involves the generation of Probe frames and the subsequent processing of received Probe Response Frames. The details of the active scanning procedures are described below.

##### 8.1.3.2.1. Sending a Probe Response

Stations receiving probes shall respond with a probe response only if the ESSID is the broadcast ESSID or if the ESSID matches the specific ESSID of the station. Probe responses shall be sent as directed messages to the address of the station that generated the probe. The probe response shall be sent using normal frame transmission rules. An access point shall respond to all probes meeting the criteria above. In an ad hoc network, the station that generated the last Beacon shall respond to a probe.

In a network there shall be at least one node that is awake at any given time to respond to probes. The station that sent the most recent Beacon shall remain awake and shall be the only station to respond to Probes until a Beacon frame is received. If the station is an access point, it shall always respond to probes.

##### 8.1.3.2.2. Active Scanning Procedure

A station using active scanning shall use the following procedure.

For each channel to be scanned:

a) Wait until a aProbe_Delay time has expired,

b~~a~~) wait until CCA indicates the medium is clear,
c~~b~~) Send Probe with Broadcast Destination, ESSID, and broadcast BSSID,
d~~c~~) Clear and s~~S~~tart Probe_Timer~~_1~~,
e~~d~~) If CCA indicates no activity Probe_Timer reaches aMin_Probe_Response_Time~~prior to expiration of Probe_Timer_1~~,
then clear NAV and Scan next channel,~~start Probe_Timer_2~~
~~Else If CCA indicates no activity before the expiration of Probe_Timer_1~~
~~then Clear NAV, Scan next channel.~~
f~~e~~) When Probe_Timer~~_2~~ reaches aMax_Probe_Response_Time~~expires~~, process all received Probe Responses,~~.~~
g~~f~~) Clear NAV and Scan next channel.

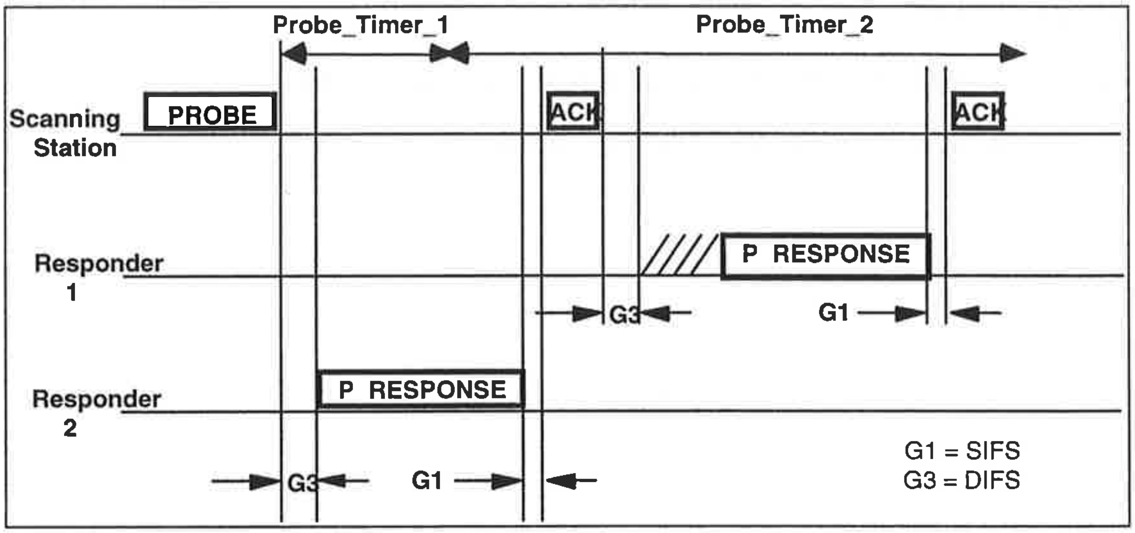

**Figure 8-3 - Probe Response**

### 8.1.3.3. Initializing or Synchronizing with a BSS

An access point shall select a BSSID, select channel synchronization information, select a beacon interval, initialize its TSF timer, and begin transmitting Beacons.

Stations which are not access points shall:

a) Scan for the presence of an existing BSS with a specific ESSID
b) If a BSS with the specific ESSID is found, adopt the BSSID, channel synchronization information, TSF timer value of the BSS.
Else if the ESSID designates an ad hoc network, select an ad hoc BSSID, select channel synchronization, select a beacon interval, initialize and start the TSF timer, and begin transmitting Beacons.
Else indicate failure to find a network matching the ESSID.

### 8.1.4. Adjusting Station Timers and Coalescing

In the infrastructure network, stations shall always adopt the timer in a Beacon or Probe Response coming from the AP in their BSS.

In an ad hoc network, a station shall always adopt the information in the contents of a Beacon or probe response frame when those frames contain a matching ESSID and the value of the time stamp is greater than the station's TSF timer. A station may return to its previous BSS, if any, and transmit a Beacon with the newly adopted information.

### 8.1.5. Timing Synchronization for Frequency Hopping PHYs

NOTE: This section only pertains to stations using a Frequency Hopped PHY.

The TSF described here provides a mechanism for stations in a frequency hopping system to synchronize their transitions from one channel to another (their hops). Every station shall maintain a table of all of the hopping sequences that are used in the system. All of the stations in a BSS shall use the same hopping sequence. Each Beacon and probe response includes the channel synchronization information necessary to determine the hop pattern and timing for the BSS.

Stations shall use their TSFTIMER to time the aDwell_Interval. The aDwell_Interval is the length of time that stations will stay on each frequency in their hopping sequence. Once stations are synchronized, they have the same TSFTIMER value.

Stations in the BSS shall tune to the next frequency in the hopping sequence whenever:

[TSFTIMER + aDwell_Offset] MOD aDwell_Interval = 0

## 8.2. Power Management

### 8.2.1. Power Management in an Infrastructure Network

#### 8.2.1.1. Overview

Stations changing power management mode shall inform the AP of this fact (via the Power Management bits within transmitted frames). The AP shall not arbitrarily transmit frames to station operating in a power saving mode, but shall buffer frames and only transmit them at designated times.

The stations which currently have buffered frames within the AP are identified in a **Traffic Indication Map (TIM)**, which shall be included as an element within all Beacons generated by the AP. A station shall determine that a frame is buffered for it by receiving and interpreting a TIM.

Stations operating in power save modes shall periodically listen for Beacons, as determined by the Station's **aListen_Interval** parameter.

Upon determining that a frame is currently buffered in the AP, a Station operating in the **Power Save Polling** mode (PSP) shall transmit a short Poll frame to the AP, which will respond with the corresponding buffered frame. If any station in its BSS is in power saving mode, the AP shall buffer all broadcast and multicast frames and deliver them to all stations following a **Delivery TIM** (DTIM) transmission.

A station shall remain in its current power management mode until it informs the AP of a power management mode change via a successful frame exchange. Power management mode shall not change during any single frame exchange sequence, as described in section 4.3.

#### 8.2.1.2. Station Power Management Modes

A station can be in two different power states:

**Awake**: Station is fully powered.

**Doze**: Station is not able to transmit or receive and consumes very low power. Some circuitry (such as timers) may still be active.

The manner in which a station transitions between these two power states shall be determined by the station's Power Management Mode (aPower_Management_Mode). These modes are summarized in the table below.

Active Mode or AM

Station may receive frames at any time, no AP buffering of frames. In Active Mode, a station shall be in the Awake state. A station on the polling list of a PCF shall be in Active Mode for the duration of the contention free period.

Power Save Polling or PSP

Station listens to selected TIMs (based upon its aListen_Interval) and polls the AP if the TIM indicates a frame is buffered for that station (except for broadcasts). The AP shall transmit buffered unicast frames only in response to a poll. In PSP mode, a station shall be in the Doze state and shall enter the Awake state to listen for selected TIMs and to transmit.

To change power management modes, a station shall inform the AP through a successful frame exchange initiated by the station.

### 8.2.1.3. Access Point TIM Transmissions

The TIM will identify the stations for which traffic is pending and buffered in the AP. This information is coded in a *virtual bitmap*, as described in Section 4. In addition the TIM contains an indication whether Broadcast/multicast traffic is pending. Every station is assigned a Station ID code (SID) by the AP as part of the association process (see Section 7.3). SID 0 (zero) shall be reserved to indicate the presence of buffered broadcast/multicast frames. The AP shall identify those stations for which frames are buffered by setting bits in the TIM's virtual bitmap that correspond to the appropriate SIDs.

### 8.2.1.4. TIM Types

Two different TIM types are distinguished: TIM and DTIM. After a DTIM, the AP shall send out the stored Broadcast/Multicast frames using normal frame transmission rules.

The AP shall transmit TIMs with every Beacon. Every aDTIM_Interval, a TIM of type "DTIM" is transmitted within a Beacon rather than an ordinary TIM.

The following figure illustrates the AP and station activity under the assumption that a DTIM is transmitted once every three TIMs. The top line in the figure represents the time axis, with the Beacon Interval shown together with a DTIM Interval of three Beacon Intervals. The second line depicts AP activity. The AP schedules Beacons for transmission every Beacon Interval, but the Beacons may be delayed if there is traffic. This is indicated as "busy medium" on the second line. For the purposes of this figure, the important fact about Beacons is that they contain TIMs, some of which may be DTIMs.

The third and fourth lines in the figure depict the activity of two stations operating with different power management requirements. Both stations power on their receivers whenever they need to listen for a TIM. This is indicated in the figure as a ramp-up of the receiver power prior to the target beacon transmission time. The first station, for example, powers up its receiver and receives a TIM in the first beacon which indicates the presence of a buffered frame for it. It generates a subsequent PS-Poll frame, which elicits the transmission of the buffered Data frame from the AP. Broadcast frames are sent by the AP subsequent to the transmission of a Beacon containing a DTIM.

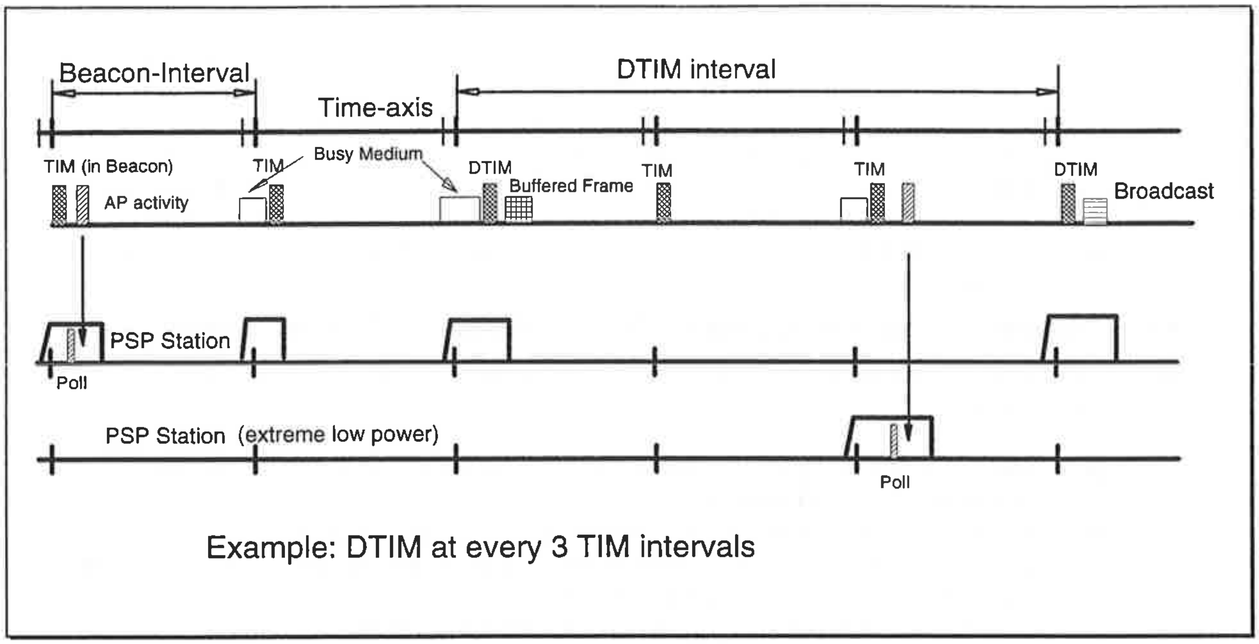

**Figure 8-4: Infrastructure Power Management Operation (No PCF Operating)**

### 8.2.1.5. Access Point Operation During the Contention Period

Access Points shall maintain a Power Management status for each currently associated station that indicates in which Power Management mode the station is currently operating. An AP shall, depending on the Power Management mode of the station, buffer the frame destined to the station temporarily. No frames received for stations operating in the Active mode shall be buffered.

a) Frames destined for PSP stations shall be temporarily buffered in the AP.
b) Frames destined to stations in the Active mode shall be directly transmitted.
c) At every Beacon Interval, the AP shall assemble the virtual bitmap containing the buffer status per destination for stations in the PSP mode, and shall send this out in the TIM field of the Beacon. The bit for SID 0 (zero) shall be set whenever broadcast or multicast traffic is buffered.
d) All Broadcast/Multicast frames shall be buffered if any associated stations are in PSP mode.
e) After every DTIM, the AP shall transmit all buffered broadcast/multicast frames.
f) Buffered frames for stations in the PSP mode shall be forwarded to the station after a Poll has been received from that station.
g) An AP shall have an aging function to delete pending traffic when it was buffered for an excessive time period.
h) Whenever an AP is informed that a station changes to the Active Mode, then the AP shall send buffered frames (if any exist) to that station immediately.

### 8.2.1.6. Access Point Operation During the Contention Free Period

Access Points shall maintain a Power Management status for each currently associated CF Aware station that indicates in which Power Management mode the station is currently operating. An AP shall, for stations in PSP Mode, buffer the frame destined to the station temporarily.

a) Frames destined for PSP stations shall be temporarily buffered in the AP.
b) Frames destined to stations in the Active mode shall be transmitted as defined in Section 5.

c) Prior to every Contention Free Period, the AP shall assemble the virtual bitmap containing the buffer status per destination for stations in the PSP mode, set the bits in the virtual bitmap for stations the Point Coordinator is intending to poll in this Contention Free Period, and shall send this out in the TIM field of the DTIM. The bit for SID 0 (zero) shall be set whenever broadcast or multicast traffic is buffered.

d) Buffered frames for stations in the PSP mode shall be forwarded to the CF Aware stations under control of the Point Coordinator.

e) An AP shall have an aging function to delete pending traffic when it was buffered for an excessive time period.

### 8.2.1.7. Receive Operation for Stations in PSP Mode During the Contention Period

Stations in PSP mode shall operate as follows to receive a frame from the AP when not participating in the contention free period:

a) Stations shall wake-up so as to receive the next scheduled Beacon after Listen Interval from the last TBTT~~received Beacon~~.

b) When a station detects that the bit corresponding to its SID is set in the TIM, the station shall issue a Poll to retrieve the buffered frame. If more than one bit is set in the TIM, the Poll shall be transmitted after a random delay.

c) If the Power Management bits in the received frame indicate that more traffic for that station is buffered, the station shall Poll until no more frames are buffered for that station.

d) To receive broadcast frames, the station shall wake up so as to receive every DTIM.

### 8.2.1.8. Receive Operation for Stations in PSP Mode During the Contention Free Period

Stations in PSP mode that are associated as CF Aware shall operate as follows to receive a frame from the AP during the contention free period:

a) Stations shall wake-up so as to receive the next scheduled DTIM.

b) When a station detects that the bit corresponding to its SID is set in the DTIM, the station shall remain awake for the duration of the contention free period to receive the frame(s) buffered for it in the AP or until a frame is received with the Power Management bits indicating no further traffic is buffered.

c) If the Power Management bits in the received frame indicate that more traffic for that station is buffered and the contention free period ends, the station may send a Poll to the AP to request the delivery of additional buffered frames.

### 8.2.1.9. Stations Operating in the Active Mode

Stations operating in this mode shall have their transceivers activated continuously, so they do not need to interpret the traffic announcement part of the Beacons.

## 8.3. Association and Reassociation

This section defines how a station associates and reassociates with an access point.

### 8.3.1. Station Association Procedures

A station shall associate with an access point via the following procedure:

a) The station shall transmit a ASSOCIATION Request frame to an access point.
b) If the ASSOCIATION Request frame transmission failsafter the appropriate number of retries, the station shall scan for a different access point with which to attempt association.
c) If an ASSOCIATION Responseframe is received with status value of "successful", the station is now associated with the access point.
d) If an ASSOCIATION Response frame is received with status value of "failed", the station shall scan for a different access point with which to attempt association.

### 8.3.2. Access Point Association Procedures

An access point shall operate as follows in order to support the association of stations.

a) Whenever an ASSOCIATION Request frame is received from a station, , the access point shall transmit a ASSOCIATION response with a status value of "successful" or "failed". If the status value is "successful", the assigned Station ID to the station is included in the response..
b) When the ASSOCIATION Response with a status value of "successful" frame is acknowledged by the station, the station is considered to be associated with this access point.
c) The AP shall inform the Disribution System of the association.

### 8.3.3. Station Reassociation Procedures

A station shall reassociate with an access point via the following procedure:

a) The station shall transmit a REASSOCIATION Request frame to an access point.
b) If transmission fails, the station shall scan for a different access point with which to attempt reassociation.
c) If an REASSOCIATION Responseframe is received with status value of "successful", the station is now associated with the access point.
d) If an REASSOCIATION Response frame is received with status value of "failed", the station shall scan for a different access point with which to attempt reassociation.

### 8.3.4. Access Point Reassociation Procedures

An access point shall operate as follows in order to support the reassociation of stations.

a) Whenever an REASSOCIATION Request frame is received from a station, , the access point shall transmit a REASSOCIATION response with a status value of "successful" or "failed". If the status value is "successful", the assigned Station ID to the station is included in the response..
b) When the REASSOCIATION Response with a status value of "successful" frame is acknowledged by the station, the station is considered to be associated with this access point.
c) The AP shall inform the Disribution System of the reassociation.

## 8.4. Managment Information Definitions

### 8.4.1. MIB Summary

The following sections summarize the 802.11 Management Information Base (MIB). Each group, attribute, action and notification is listed. This summary is for information purposes only. If any errors exist, the formal definitions have precedence.

### 8.4.1.1. Station Management Attributes

#### 8.4.1.1.1. agStation_Config_grp

aActing_as_AP_Status,
aActing_as_Wireless_AP_Status,
aAssociated_State,
aBeacon_Period,
aPower_Mgt_State,
~~aPower_Mgt_Capability~~aPassive_Scan_Duration,
aListen_Interval,~~;~~
aListen_Interval,
aScan_Mode,
aPassive_Scan_Duration;

#### 8.4.1.1.2. agAuthentication_grp

aAuthentication_Algortihms,
aSelect~~ed~~_Authentication_Algorithm,
~~aAuthentication_Handshake_State,~~
~~aAuthentication_State,~~
~~aMin_Authentication_Required;~~

#### 8.4.1.1.3. agPrivacy_grp

aPrivacy_Option_Implemented~~Algortihms~~,
~~aSelected_Privacy_Algorithm,~~
~~aPrivacy_Handshake_State,~~
aPrivacy_Invoked~~State~~,
~~aMin_Privacy_Required;~~

#### 8.4.1.1.4. Not Grouped

aStation_ID
aCurrent_BSS_ID
aCurrent_ESS_ID
aKnown_APs

### 8.4.1.2. MAC Attributes

#### 8.4.1.2.1. agAddress_grp

aMAC_Address.
aGroup_Addresses;

### 8.4.1.2.2. agOperation_grp

~~aNAV,~~
~~aNAV_max,~~
aRate_Factor,
aHandshake_Overhead,
aSIFS,
aPIFS,
aDIFS,
aRTS_Threshold,
~~aSlot_Time,~~
aCW_max,
aCW_min,
aCTS_Time,
aACK_Time,
aRetry_max,
aMax_Frame_Length,
aFragmentation_Threshold;

### 8.4.1.2.3. agCounters_grp

aTransmitted_MPDU~~Frame~~_Count,
aTransmitted_MSDU_Count,
aOctets_Transmitted_Count,
aMulticast_Transmitted_Frame_Count,
aBroadcast_Transmitted_Frame_Count,
aFailed_Count,
aRetry
~~Collision~~_Count,
aSingle_Retry~~Collision~~_Count,
aMultiple_Retry_Count~~Multiple_Collision_Count~~,
aFrame_Duplicate_Count,
aRTS_Success_Count,
aRTS_Failure_Count,
aReceived_Frame_Count,
aOctets_Received_Count,
aMulticast_Received_Count,
aBroadcast_Received_Count,
aError_Count,
aFCS_Error,Count,
~~aLength_Mismatch_Count,~~
aFrame_Too_Long_Count,
aTotal_Backoff_Time;

### 8.4.1.2.4. agStatus_grp

~~aMAC_Enable_Status,~~
aTransmit_Enable_Status,
aPromiscuous_Status,
aScan_State;

### 8.4.1.2.5. Not Grouped

aManufacturer_ID
aProduct_ID

### 8.4.1.3. ResourceTypeID Attributes

#### 8.4.1.3.1. Not Grouped

aResourceTypeIDName
aResourceInfo

### 8.4.1.4. Actions

#### 8.4.1.4.1. SMT Actions

acStation_init
acStation_reset

#### 8.4.1.4.2. MAC Actions

acMAC_init
acMAC_reset

#### ~~8.4.1.4.3. PHY Actions~~

~~acPHY_init~~
~~acPHY_reset~~

### 8.4.1.5. Notifications

#### 8.4.1.5.1. SMT Notifications

nAssociate
nDissociate

#### 8.4.1.5.2. MAC Notifications

nFrame_Error_Rate_Exceeded

### 8.4.2. Managed Object Class Templates

### 8.4.2.1. SMT Object Class

#### 8.4.2.1.1. oSMT

```
SMT MANAGED OBJECT CLASS
DERIVED FROM "ISO/IEC 10165-2":top;
CHARACTERIZED BY
    pSMT_base                                PACKAGE
        BEHAVIOUR
            bSMT_base BEHAVIOUR
                DEFINED AS "The SMT object class provides the necessary support at the station to
                manage the processes in the station such that the station may work cooperatively as a part
                of an 802.11 network.";
            ATTRIBUTES
                aStation_ID                      GET,
                aActing_as_AP_Status             GET,
```

aActing_as_Wireless_AP_Status GET,
aCurrent_BSS_ID GET,
aCurrent_ESS_ID GET-REPLACE,
aKnown_APs GET, (1 to N deep)
aAuthentication_Algorithms GET,
aPrivacy_Option_Implemented~~aPrivacy_Algorithms~~ GET,
aSelect_Authentication GET-REPLACE,
~~aSelected_Privacy_Algorithm GET,~~
~~aAuthentication_Handshake_State GET,~~
~~aPrivacy_Handshake_State GET,~~
~~aAuthentication_State GET,~~
aPrivacy_Invoke~~State~~ GET-REPLACE,
~~aMin_Authentication_Required GET,~~
~~aMin_Privacy_Required GET,~~
aAssociated_State GET,
aBeacon_Period GET-REPLACE,
aPower_Mgt_State GET-REPLACE,
~~aPower_Mgt_Capability GET~~aPassive_Scan_Duration GET-REPLACE
aListen_Interval GET-REPLACE,
aScan_Mode GET-REPLACE;

ATTRIBUTE GROUPS
agStation_Config_grp,
agAuthentication_grp,
agPrivacy_grp;

ACTIONS
acSMT_init,
acSMT_reset;

REGISTERED AS { iso(1) member-body(2) us(840) ieee802dot11(10036) smt(0) };

### 8.4.2.2. MAC Object Class

#### 8.4.2.2.1. oMAC

MAC MANAGED OBJECT CLASS
DERIVED FROM "ISO/IEC 10165-2":top;
CHARACTERIZED BY
pMAC_base PACKAGE
BEHAVIOUR
bMAC_base BEHAVIOUR

DEFINED AS "The MAC object class provides the necessary support for the access control, generation and verification of frame check sequences, and proper delivery of valid data to upper layers.";

ATTRIBUTES
aMAC_Address GET,
aGroup_Addresses GET-REPLACE,
aPromiscuous_Status GET,
aScan_State GET,
aTransmitted_MPDU~~Frame~~_Count GET-REPLACE,
aTransmitted_MSDU_Count GET-REPLACE,
aOctets_Transmitted_Count GET-REPLACE,
aMulticast_Transmitted_Frame_Count GET-REPLACE,
aBroadcast_Frame_Count GET-REPLACE,
aFailed_Count GET-REPLACE,
aFrame_Retry~~Exchange_Error~~_Count GET-REPLACE,

```
                aSingle_Retry~~Frame_Exchange_Error~~_Count        GET-REPLACE,
                aMultiple_Retry~~Frame_Exchange_Error~~_Count GET-REPLACE,
                aRTS_Success_Count,                    GET-REPLACE,
                aRTS_Failure_Count,                    GET-REPLACE,
                aReceived_Frame_Count                  GET-REPLACE,
                aOctets_Received_Count                 GET-REPLACE,
                aMulticast_Received_Frame_Count        GET-REPLACE,
                aBroadcast_Received_Frame_Count        GET-REPLACE,
                aReceived_Frame_Error_Count            GET-REPLACE,
                aFCS_Error_Count                       GET-REPLACE,
                aFrame_Too_Long_Count                  GET-REPLACE,
                aFrame_With_Protocol_Error_Count       GET-REPLACE,
                aMAC_Enable_Status                     GET,
                aRate_Factor                           GET,
                aHandshake_Overhead                    GET,
                aSIFS                                  GET,
                aPIFS                                  GET,
                aDIFS                                  GET,
                aRTS_Threshold                         GET-REPLACE,
                aTotal_Accumulated_Backoff_Time        GET-REPLACE,
                ~~aSlot_Time                             GET,~~
                CW_max                                 GET-REPLACE,
                aCW_min                                GET-REPLACE,
                aCTS_Time                              GET,
                aACK_Time                              GET,
                aRTS_Retry_max                         GET-REPLACE,
                aDATA_Retry_max                        GET-REPLACE
                aMax_Frame_Length                      GET,
                aFragmentation_Threshold               GET-REPLACE,
                aManufacturer_ID                       GET,
                aProduct_ID                            GET;
            ATTRIBUTE GROUPS
                agCapabilities_grp,
                agConfig_grp,
                agAddress_grp,
                agOperation_grp,
                agCounters_grp,
                agFrame_Error_Condition_grp,
                agStatus_grp;
            ACTIONS
                acMAC_init,
                acMAC_reset;
            NOTIFICATIONS
                nFrame_Error_Rate_Exceeded;
REGISTERED AS { iso(1) member-body(2) us(840) ieee802dot11(10036) mac(1) };
```

### 8.4.2.3. Resource Type Object Class

#### 8.4.2.3.1. oResourceTypeID

```
ResourceTypeID MANAGED OBJECT CLASS
DERIVED FROM IEEE802CommonDefinitions.oResourceTypeID;
CHARACTERIZED BY
    pResourceTypeID                       PACKAGE
            ATTRIBUTES
```

aResourceTypeIDName GET,
aResourceInfo GET;
REGISTERED AS { iso(1) member-body(2) us(840) ieee802dot11(10036) resourcetypeid(3) };

### 8.4.3. Attribute Group Templates

#### 8.4.3.1. Station Management Attribute Group Templates

#### 8.4.3.1.1. agStation_Config_grp

Station_Config_grp ATTRIBUTE GROUP
GROUP ELEMENTS
aActing_as_AP_Status,
aActing_as_Wireless_AP_Status,
aAssociated_State,
aBeacon_Period,
aPower_Mgt_State,
~~aPower_Mgt_Capability~~aPassive_Scan_Duration
aListen_Interval,
aScan_Mode;
REGISTERED AS { iso(1) member-body(2) us(840) ieee802dot11(10036) smt(0) station_config_grp(0) };

#### 8.4.3.1.2. agAuthentication_grp

Authentication_grp ATTRIBUTE GROUP
GROUP ELEMENTS
aAuthentication_Algortihms,
aSelected_Authentication_Algorithm,
~~aAuthentication_Handshake_State,~~
aAuthentication_State,
~~aMin_Authentication_Required;~~
REGISTERED AS { iso(1) member-body(2) us(840) ieee802dot11(10036) smt(0) authentication_grp(1) };

#### 8.4.3.1.3. agPrivacy_grp

Privacy_grp ATTRIBUTE GROUP
GROUP ELEMENTS
aPrivacy_Option_Implemented~~Algortihms~~,
aSelected_Privacy_Algorithm,
~~aPrivacy_Handshake_State,~~
aPrivacy_Invoked~~State~~,
~~aMin_Privacy_Required;~~
REGISTERED AS { iso(1) member-body(2) us(840) ieee802dot11(10036) smt(0) privacy_grp(2) };

#### 8.4.3.2. MAC Attribute Group Templates

#### 8.4.3.2.1. agAddress_grp

Address_grp ATTRIBUTE GROUP
GROUP ELEMENTS
aMAC_Address,
aGroup_Addresses;
REGISTERED AS { iso(1) member-body(2) us(840) ieee802dot11(10036) mac(0) address_grp(0) };

### 8.4.3.2.2. agOperation_grp

Operation_grp ATTRIBUTE GROUP
GROUP ELEMENTS
~~aNAV,~~
~~aNAV_max,~~
aRate_Factor,
aHandshake_Overhead,
aSIFS,
aPIFS,
aDIFS,
aRTS_Threshold,
~~aSlot_Time,~~
aCW_max,
aCW_min,
aCTS_Time,
aACK_Time,
aRetry_max,
aMax_Frame_Length,
aFragmentation_Threshold;
REGISTERED AS { iso(1) member-body(2) us(840) ieee802dot11(10036) mac(0) operation_grp(1) };

### 8.4.3.2.3. agCounters_grp

Counters_grp ATTRIBUTE GROUP
GROUP ELEMENTS
aTransmitted_MPDU~~Frame~~_Count,
aTransmitted_MSDU_Count,
aOctets_Transmitted_Count,
aMulticast_Transmitted_Frame_Count,
aBroadcast_Transmitted_Frame_Count,
aFailed_Count,
aRetry~~Collision~~_Count,
aSingle_Retry~~Collision~~_Count,
aMultiple_Retry_Count~~Multiple_Collision_Count~~,
aRTS_Success_Count,
aRTS_Failure_Count,
aFrame_Duplicate_Count,
aReceived_Frame_Count,
aOctets_Received_Count,
aMulticast_Received_Count,
aBroadcast_Received_Count,
aError_Count,
aFCS_Error,Count,
~~aLength_Mismatch_Count,~~
aFrame_Too_Long_Count,
aTotal_Backoff_Time;
REGISTERED AS { iso(1) member-body(2) us(840) ieee802dot11(10036) mac(0) counters_grp(2) };

### 8.4.3.2.4. agStatus_grp

Status_grp ATTRIBUTE GROUP
GROUP ELEMENTS
~~aMAC_Enable_Status,~~
aTransmit_Enable_Status,

```
        aPromiscuous_Status,
        aScan_State;
REGISTERED AS { iso(1) member-body(2) us(840) ieee802dot11(10036) mac(0) status_grp(3) };
```

### 8.4.4. Attribute Templates

### 8.4.4.1. SMT Attribute Templates

#### 8.4.4.1.1. aStation_ID

```
Station_ID ATTRIBUTE
DERIVED FROM
    IEEE802CommonDefinitions.MACAddress;
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) station_id(0) };
```

#### 8.4.4.1.2. aActing_as_AP_Status

```
Acting_as_AP_Status ATTRIBUTE
WITH APPROPRIATE SYNTAX
    boolean;
BEHAVIOUR DEFINED AS
    "True if this station is acting as an access point, false otherwise.";
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) acting_as_ap_status(4) };
```

#### 8.4.4.1.3. aActing_as_Wireless_AP_Status

```
aActing_as_Wireless_AP_Status ATTRIBUTE
WITH APPROPRIATE SYNTAX
        boolean;
BEHAVIOUR DEFINED AS
        "This attribute specifies that the station is acting as a wireless AP";
REGISTERED AS
        { iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7)
        aActing_as_Wireless_AP_Status(5) };
```

#### 8.4.4.1.4. aCurrent_AP_MAC_Address

```
Current_AP_MAC_Address ATTRIBUTE
DERIVED FROM
    IEEE802CommonDefinitions.MACAddress;
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) ap_address(5) };
```

#### 8.4.4.1.5. aCurrent_BSS_ID

Current_BSS_ID ATTRIBUTE
DERIVED FROM
    IEEE802CommonDefinitions.MACAddress;
~~WITH APPROPRIATE SYNTAX~~
    ~~integer;~~
BEHAVIOUR DEFINED AS
    "This attribute shall identify the basic service set (BSS) with which the station is currently associated.";
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) current_bss_id(6) };

#### 8.4.4.1.6. aCurrent_ESS_ID

Current_ESS_ID ATTRIBUTE
WITH APPROPRIATE SYNTAX
octet string~~integer~~;
BEHAVIOUR DEFINED AS
"This attribute shall identify the extended service set (ESS) with which the station is associated, if any.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) current_ess_id(7) };

#### 8.4.4.1.7. aKnown_APs

Known_APs ATTRIBUTE
WITH APPROPRIATE SYNTAX
set-of AP_ID.type;
BEHAVIOUR DEFINED AS
"This attribute shall be a set of the identities of the most recently known Access Points. The Access Point with which the station is currently associated, if any, shall always be the first element of the set. Access Points may be included in this list even if the station did not associate with them. A station may delete AP identities from this set using any algorithm of its choosing. The set may include fewer AP identities than the number of APs the station has encountered.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) known_aps(8) };

#### 8.4.4.1.8. aAuthentication_Algorithms

Authentication_Algorithms ATTRIBUTE
WITH APPROPRIATE SYNTAX
set-of integer;
BEHAVIOUR DEFINED AS
"This attribute shall be a set of all the authentication algorithms supported by the stations. The values of the numbers in the list are as defined in IEEE Standard 802.10.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) authentication_algorithms(9) };

#### 8.4.4.1.9. aPrivacy_Option_Implemented~~Algorithms~~

Privacy_Option_Implemented~~Algorithms~~ ATTRIBUTE
WITH APPROPRIATE SYNTAX
Boolean~~set-of integer~~;
BEHAVIOUR DEFINED AS
"This attribute shall indicate that the 802.11 WEP option is implemented~~be a set all of the privacy algorithms supported by the stations. The values of the numbers in the list are as defined in IEEE Standard 802.10~~.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) privacy_option_implemented~~algorithms~~(10) };

#### 8.4.4.1.10. aSelecte~~d~~_Authentication_Algorithm

Selected_Authentication_Algorithm ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute shall indicate the authentication algorithm used~~identifer selected~~ during the authentication sequence~~negotiation~~. The value of this attribute shall be selected from the set in the

aAuthentication_Algorithms attribute. ~~The value of this attribute shall reference one of the authentication algorithm identifiers defined in IEEE Standard 802.10.~~";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7)
selected_authentication_algorithm(11) };

### ~~8.4.4.1.11. aSelected_Privacy_Algorithm~~

~~Selected_Privacy_Algorithm ATTRIBUTE~~
~~WITH APPOPRIATE SYNTAX~~
~~integer;~~
~~BEHAVIOUR DEFINED AS~~
~~"This attribute shall indicate the privacy algorithm identifer selected during the privacy negotiation. The value of this attribute shall be selected from the set in the aPrivacy_Algorithms attribute. The value of this attribute shall reference one of the privacy algorithm identifiers defined in IEEE Standard 802.10.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7)~~
~~selected_privacy_algorithm(12) };~~

### ~~8.4.4.1.12. aAuthentication_Handshake_State~~

~~Authentication_Handshake_State ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~authentication_handshake.type~~
~~BEHAVIOUR DEFINED AS~~
~~"This attribute shall identify the current state of the station in the authentication process.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7)~~
~~authentication_handshake_state(13) };~~

### ~~8.4.4.1.13. aPrivacy_Handshake_State~~

~~Privacy_Handshake_State ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~privacy_hanshake.type;~~
~~BEHAVIOUR DEFINED AS~~
~~"This attribute shall identify the current state of the station in the privacy negotiation process.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7)~~
~~privacy_handshake_state(14) };~~

### ~~8.4.4.1.14. aAuthentication_State~~

~~Authentication_State ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~authentication_state.type;~~
~~BEHAVIOUR DEFINED AS~~
~~"This attribute shall indicate the authentication state.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) authentication_state(15) };~~

### 8.4.4.1.15. aPrivacy_InvokeState

Privacy_Invoke~~State~~ ATTRIBUTE
WITH APPROPRIATE SYNTAX
privacy_invoke~~state~~.type;
BEHAVIOUR DEFINED AS
"This attribute shall indicate whether the 802.11 WEP mechanism is invoked~~the current privacy state~~.";

REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) privacy_state(16) };

### ~~8.4.4.1.16. aMin_Authentication_Required~~

~~Min_Authentication_Required ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~Authentication_Required.type;~~
~~BEHAVIOUR DEFINED AS~~
~~;~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7)~~
~~min_authentication_required(17) };~~

### ~~8.4.4.1.17. aMin_Privacy_Required~~

~~Min_Privacy_Required ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~Privacy_Required.type;~~
~~BEHAVIOUR DEFINED AS~~
~~;~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) min_privacy_required(18)~~
~~};~~

### 8.4.4.1.18. aAssociated_State

Associated_State ATTRIBUTE
WITH APPROPRIATE SYNTAX
Associated_State.type;
BEHAVIOUR DEFINED AS
"An enumerated type that describes the current associated state of the station.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) associated_state(19) };

### 8.4.4.1.19. aBeacon_Period

Beacon_Period ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR
"The beacon period shall indicate the time, in micr~~nan~~oseconds, between the transmission of beacon frames if the station is acting as an Access Point. If the station is not an Access Point but is associated with one, the beacon period shall indicate the time, in nanoseconds, between the expected arrival of beacon frames. If the station is not an Access Point and is not associated with one, the beacon period shall indicate the time, in nanoseconds, between the transmission of beacon frames.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) beacon_period(20) };

### 8.4.4.1.20. aPower_Mgt_State

Power_Mgt_State ATTRIBUTE
WITH APPROPRIATE SYNTAX
Power_Mgt_State.type;
BEHAVIOUR DEFINED AS
"An enumerated type that describes the current power management state of the station.";
REGISTERED AS

{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) power_mgt_state(21)
};

### ~~8.4.4.1.21. aPower_Mgt_Capability~~

~~Power_Mgt_Capability ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~set of Power_Mgt_State.type;~~
~~BEHAVIOUR DEFINED AS~~
~~"An enumerated type that describes all the possible power management states of which the station is capable.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) power_mgt_capability(22) };~~

### 8.4.4.1.22. aPassive_Scan_Duration

aPassive_Scan_Duration ATTRIBUTE
WITH APPROPRIATE SYNTAX
Integer;
BEHAVIOUR DEFINED AS
"This attribute defines the maximum time that a station will remain on a single channel during a passive scan of that channel.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7)
aPassive_Scan_Duration(23) };

### 8.4.4.1.23. aListen_Interval

aListen_Interval ATTRIBUTE
WITH APPROPRIATE SYNTAX
Integer;
BEHAVIOUR DEFINED AS
"This attribute specifies the number of Beacon intervals which may pass before the station awakens and listens for the next beacon";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) aListen_Interval(24)
};

### 8.4.4.1.24. aScan_Mode

Scan_Mode ATTRIBUTE
WITH APPROPRIATE SYNTAX
Enumerated Type;
BEHAVIOUR DEFINED AS
"Scan_Mode is an enumerated type that can take on the values ACTIVE or PASSIVE";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) aScan_Mode(24) };

## 8.4.4.2. MAC Attribute Templates

### 8.4.4.2.1. aMAC_Address

MAC_Address ATTRIBUTE
DERVIVED FROM
IEEE802CommonDefinitions.MACAddress;
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) mac_address(0) };

### 8.4.4.2.2. aGroup_Addresses

Group_Addresses ATTRIBUTE
WITH APPROPRIATE SYNTAX
set-of IEEE802CommonDefinitions.MACAddress;
BEHAVIOUR DEFINED AS
"A set of ~~xxx~~-MAC_Addresses identifying the multicast addresses for which this station will receive frames."
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) group_addresses(1) };

### 8.4.4.2.3. aPromiscuous_Status

Promiscuous_Status ATTRIBUTE
WITH APPROPRIATE SYNTAX
boolean;
BEHAVIOUR DEFINED AS
"This attribute is true when the station is enabled to receive all frames promiscuously. It is false otherwise.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) promiscuous_status(2) };

### 8.4.4.2.4. aTransmitted_MPDU~~Frame~~_Count

Transmitted_Frame_Count ATTRIBUTE
DERIVED FROM
"ISO/IEC 10165-2":pdusSentCounter;
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) transmitted_MPDU~~frame~~_count(3) };

### 8.4.4.2.5. aTransmitted_MSDU_Count

aTransmitted_MSDU_Count ATTRIBUTE
DERIVED FROM
"ISO/IEC 10165-2":pdusSentCounter;
BEHAVIOUR DEFINED AS
"This attribute counts the MSDUs that have been transmitted successfully.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7) aTransmitted_MSDU_Count(4) };

### 8.4.4.2.6. aOctets_Transmitted_Count

Octets_Transmitted_Count ATTRIBUTE
DERIVED FROM
"ISO/IEC 10165-2":octetsSentCounter;
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) octets_transmitted_count(4) };

### 8.4.4.2.7. aMulticast_Transmitted_Frame_Count

Multicast_Transmitted_Frame_Count ATTRIBUTE
DERIVED FROM
"ISO/IEC 10165-2":pdusSentCounter;
BEHAVIOUR DEFINED AS
"This counter shall increment only when the multicast/broadcast bit is set in the destination MAC address and the destination MAC address is not the broadcast address.";

```
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7)
    multicast_transmitted_frame _count(5) };
```

### 8.4.4.2.8. aBroadcast_Transmitted_Frame_Count

```
Broadcast_Transmitted_Frame_Count ATTRIBUTE
DERIVED FROM
    "ISO/IEC 10165-2":pdusSentCounter;
BEHAVIOUR DEFINED AS
    "This counter shall increment only when the destination MAC address is the broadcast address.";
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7)
    broadcast_transmitted_frame_count(6) };
```

### 8.4.4.2.9. aFailed_Count

```
Failed_Count ATTRIBUTE
DERIVED FROM
    "ISO/IEC 10165-2":counter;
BEHAVIOUR DEFINED AS
    "This counter shall increment when a frame is not transmitted due to the number of transmit attempts
    exceeding the retry_max value.";
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) failed_count(7) };
```

### 8.4.4.2.10. aCollision_Count

```
Collision_count ATTRIBUTE
DERIVED FROM;
    "ISO/IEC 10165-2":counter
BEHAVIOUR DEFINED AS
    "This counter shall increment when a collision is detected.";
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) collision_count(8) };
```

### 8.4.4.2.11. aSingle_Collision_Count

```
Single_Collision_Count ATTRIBUTE
DERIVED FROM
    "ISO/IEC 10165-2":counter;
BEHAVIOUR DEFINED AS
    "This counter shall increment when a frame is successfully transmitted after a single collision.";
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) single_collision_count(9)
    };
```

### 8.4.4.2.12. aMultiple_Retry~~Collision~~_Count

```
Multiple_Retry_Count~~Multiple_Collision_Count~~ ATTRIBUTE
DERIVED FROM
    "ISO/IEC 10165-2":counter;
BEHAVIOUR DEFINED AS
    "This counter shall increment when a frame is successfully transmitted after more than one
    retransmission~~collision~~.";
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7)
    multiple_retry~~collision~~_count(10) };
```

### 8.4.4.2.13. aRTS_Success_Count

```
RTS_Success_Count ATTRIBUTE
DERIVED FROM;
    "ISO/IEC 10165-2":counter
BEHAVIOUR DEFINED AS
    "This counter shall increment when a CTS is received to a RTS.";
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) rts_sucess_count(8) };
```

### 8.4.4.2.14. RTS_Failure_Count

```
RTS_Failure_Count ATTRIBUTE
DERIVED FROM
    "ISO/IEC 10165-2":counter;
BEHAVIOUR DEFINED AS
    "This counter shall increment when a CTS is not received to a RTS.";
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) rts_failure_count(9) };
```

### 8.4.4.2.15. aReceived_Frame_Count

```
Received_Frame_Count ATTRIBUTE
DERIVED FROM
    "ISO/IEC 10165-2":pdusReceivedCounter;
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7)
    received_frame_count(11) };
```

### 8.4.4.2.16. aOctets_Received_Count

```
Octets_Received_Count ATTRIBUTE
DERIVED FROM
    "ISO/IEC 10165-2":octetsReceivedCounter;
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7)
    octets_received_count(12) };
```

### 8.4.4.2.17. aMulticast_Received_Frame_Count

```
Multicast_Received_Frame_Count ATTRIBUTE
DERIVED FROM
    "ISO/IEC 10165-2":pdusReceivedCounter;
BEHAVIOUR DEFINED AS
    "This counter shall increment when a frame is received with the multicast/broadcast bit set in the
    destination MAC address, the destnation MAC address is not the broadcast address and the destination
    address is in the set of Group_Addresses.";
REGISTERED AS
    { iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7)
    multicast_received_frame_count(13) };
```

### 8.4.4.2.18. aBroadcast_Received_Frame_Count

```
Broadcast_Received_Frame_Count ATTRIBUTE
DERIVED FROM
    "ISO/IEC 10165-2":pdusReceivedCounter;
BEHAVIOUR DEFINED AS
    "This counter shall increment when a frame is received with the destination MAC address equal to the
    broadcast address.";
REGISTERED AS
```

{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7)
broadcast_received_frame_count(14) };

#### 8.4.4.2.19. aError_Count

Error_Count ATTRIBUTE
DERIVED FROM
"ISO/IEC 10165-2":courruptedPDUsReceivedCounter;
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) error_count(15) };

#### 8.4.4.2.20. aFCS_Error_Count

FCS_Error_Count ATTRIBUTE
DERIVED FROM
"ISO/IEC 10165-2":counter;
BEHAVIOUR DEFINED AS
"This counter shall increment when an FCS error is detected in a received frame.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) fcs_error_count(16) };

#### ~~8.4.4.2.21. aLength_Mismatch_Count~~

~~Length_Mismatch_Count ATTRIBUTE~~
~~DERIVED FROM~~
~~"ISO/IEC 10165-2":counter;~~
~~BEHAVIOUR DEFINED AS~~
~~"This counter shall increment when a frame is received and the number of bytes in the frame does not equal the value in the length field of the frame.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7)~~
~~length_mismatch_count(17) };~~

#### 8.4.4.2.22. aFrame_Too_Long_Count

Frame_Too_Long_Count ATTRIBUTE
DERIVED FROM
"ISO/IEC 10165-2":counter;
BEHAVIOUR DEFINED AS
"This counter shall increment when a received frame that exceeds Max_Frame_Length is detected.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7)
frame_too_long_count(18) };

#### ~~8.4.4.2.23. aMAC_Enable_Status~~

#### 8.4.4.2.24. aFrame_Duplicate_Count

aFrame_Duplicate_Count ATTRIBUTE
WITH APPROPRIATE SYNTAX
Integer;
BEHAVIOUR DEFINED AS
"This counter shall increment when a frame is received which the Sequence Control field indicates is a duplicate.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) SMT(0) attribute(7)
aFrame_Duplicate_Count(19) };

~~MAC_Enable_Status ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~boolean;~~
~~BEHAVIOUR DEFINED AS~~
~~"This attribute is true when the MAC sublayer is enabled. It is false otherwise. Setting this attribute true causes the MAC to become operational in the idle state.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) mac_enable_status(19) };~~

### ~~8.4.4.2.25. aTransmit_Enable_Status~~

~~Transmit_Enable_Status ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~boolean;~~
~~BEHAVIOUR DEFINED AS~~
~~"This attribute is true when transmission is enabled. It is false otherwise. Setting this attribute to true allows the MAC to transmit frames.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) transmit_enable_status(20) };~~

### ~~8.4.4.2.26. aNAV~~

~~NAV ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~integer;~~
~~BEHAVIOUR DEFINED AS~~
~~"This attribute indicates the amount of time remaining that the station will consider the medium to be in use by another station. This attribute is updated whenever there is a change in the MAC network allocation vector.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) nav(21) };~~

### ~~8.4.4.2.27. aNAV_max~~

~~NAV_max ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~integer;~~
~~BEHAVIOUR DEFINED AS~~
~~"This is the maximum allowable value for the NAV.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) nav_max(22) };~~

### 8.4.4.2.28. aRate_Factor

Rate_Factor ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute indicates the current rate (in bytes per second) at which data is transferred across the medium.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) rate_factor(23) };

### 8.4.4.2.29. aHandshake_overhead

Handshake_overhead ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;

BEHAVIOUR DEFINED AS
"This attribute is the amount of time required to complete an RTS/CTS handshake. This value, along with the Rate_Factor, may be used to determine the desirable setting of the RTS_Threshold to maximize data throughput.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) handshake_overhead(24) };

### 8.4.4.2.30. aSIFS

SIFS ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute indicates the length of the short interframe space.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) sifs(25) };

### 8.4.4.2.31. aPIFS

PIFS ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute indicates the length of the priority interframe space.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) pifs(26) };

### 8.4.4.2.32. aDIFS

DIFS ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute indicates the length of the distributed interframe space.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) difs(27) };

### 8.4.4.2.33. aRTS_Threshold

RTS_Threshold ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute indicates the number of bytes in an M~~S~~PDU, below which an RTS/CTS handshake will not be performed. An RTS/CTS handshake shall be performed for all frames where the length of the M~~S~~PDU is larger than this threshold. Setting this attribute to be larger than the maximum MSDU size will have the effect of turning off the RTS/CTS handshake.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) rts_threshold(28) };

### 8.4.4.2.34. aTotal_Backoff_Time

Total_Backoff_Time ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute indicates the length of time, in number of slots, the MAC has spent in a backoff condition.";

REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) total_backoff_time(29) };

### ~~8.4.4.2.35. aSlot_time~~

~~Slot_time ATTRIBUTE~~
~~WITH APPROPRIATE SYNTAX~~
~~integer;~~
~~BEHAVIOUR DEFINED AS~~
~~"This attribute is the length of a single slot, in nanoseconds.";~~
~~REGISTERED AS~~
~~{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) slot_time(30) };~~

### 8.4.4.2.36. aCW_max

CW_max ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute indicates the maximum size of the contention window, in slots.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) cw_max(31) };

### 8.4.4.2.37. aCW_min

CW_min ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute indicates the minimum size of the contention window, in slots.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) cw_min(32) };

### 8.4.4.2.38. aCTS_Time

CTS_Time ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute indicates the length of time it takes to transmit a CTS frame.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) cts_time(33) };

### 8.4.4.2.39. aACK_Time

ACK_Time ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute indicates the length of time it takes to transmit an ACK frame.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) rts_time(34) };

### 8.4.4.2.40. aRetry_max

Retry_max ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS

"This attribute indicates tha maximum number of transmission attempts that will be made before a failure condition is indicated.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) retry_max(35) };

### 8.4.4.2.41. aMax_Frame_Length

Max_Frame_Length ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR DEFINED AS
"This attribute specifies the maximum MSDU length that will be accepted for transmission. If a frame is received with a length that exceeds this value, a Frame_Too_Long error will be reported.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) max_frame_length(36) };

### 8.4.4.2.42. aFragmentation_Threshold

aFragmentation_Threshold ATTRIBUTE
WITH APPROPRIATE SYNTAX
integer;
BEHAVIOUR
"This attribute specifies the current maximum size, in bytes, of the MPDU that will be delivered to the PHY. An MSDU will be broken into fragments if its size exceeds the value of this attribute after adding MAC headers and trailers. The default value for this attribute shall be equal to the maximum size PSDU of the attached PHY and shall never exceed the maximum size PSDU of the attached PHY. The minimum value of this attribute shall never be less than 256.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) fragmentation_threshold(37) };

### 8.4.4.2.43. aManufacturer_ID

Manufacturer_ID ATTRIBUTE
WITH APPROPRIATE SYNTAX
octet string;
BEHAVIOUR DEFINED AS
"The Manufacturer_ID shall include, at a minimum, the name of the manufacturer. It may include additional information at the manufacturer's discretion.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) manufacturer_id(38) };

### 8.4.4.2.44. aProduct_ID

Product_ID ATTRIBUTE
WITH APPROPRIATE SYNTAX
octet string;
BEHAVIOUR DEFINED AS
"The Product_ID shall include, at a minimum, an identifier that is unique to the manufacturer. It may include additional information at the manufacturer's discretion.";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) product_id(39) };

### 8.4.4.2.45. aScan_State

Scan_State ATTRIBUTE
WITH APPROPRIATE SYNTAX
Enumerated Type;
BEHAVIOUR DEFINED AS

"An enumerated type that describes whether the station is scanning";
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) MAC(1) attribute(7) Scan_State(40) };

### 8.4.4.3. Resource Type Attribute Templates

#### 8.4.4.3.1. aResourceTypeIDName

ResourceTypeIDName ATTRIBUTE
DERIVED FROM
IEEE802CommonDefinitions.ResourceTypeIDName;
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) ResourceTypeID(3) attribute(7) resourcetypeidname(0) };

#### 8.4.4.3.2. aResourceInfo

ResourceInfo ATTRIBUTE
DERIVED FROM
IEEE802CommonDefinitions.ResourceInfo;
REGISTERED AS
{ iso(1) member-body(2) us(840) ieee802dot11(10036) ResourceTypeID(3) attribute(7) resourceinfo(1) };

## 8.4.5. Action Templates

### 8.4.5.1. SMT Action Templates

#### 8.4.5.1.1. acInitialize_SMT

Initialize_SMT ACTION

### 8.4.5.2. MAC Action Templates

#### 8.4.5.2.1. acInitialize_MAC

#### 8.4.5.2.2. acAdd_Group_Address

#### 8.4.5.2.3. acDelete_Group_Address

#### ~~8.4.5.2.4. acExecute_Self_Test~~

## 8.4.6. Notification Templates

### 8.4.6.1. SMT Notification Templates

#### 8.4.6.1.1. nAssociate

#### 8.4.6.1.2. nDisassociate

### 8.4.6.2. MAC Notification Templates

#### 8.4.6.2.1. nFrame_Error_Rate_Exceeded

## 8.4.7. ASN.1 Definitions

### 8.4.7.1. Common Definitions

### 8.4.7.2. SMT Definitions

### 8.4.7.3. MAC Definitions

## 8.4.8. Name Binding

### 8.4.8.1. MAC Naming