(5) Number of octetsTransmittedOK: Recommended, Read only, 32 bit counter.
This contains a count of data and padding octets of frames that are successfully transmitted. This counter is incremented when the TransmitStatus is reported as transmitOK. The update occurs in the LayerMgmtTransmitCounters procedure (5.2.4.2).

(6) Number of Frames with deferredTransmissions: Recommended, Read only, 32 bit counter.
This contains a count of frames whose transmission was delayed on its first attempt because the medium was busy. This counter is incremented when the Boolean variable deferred has been asserted by the TransmitLinkMgmt function (4.2.8). Frames involved in any collisions are not counted. The update occurs in the LayerMgmtTransmitCounters procedure (5.2.4.2).

(7) Number of multicastFramesTransmittedOK: Optional, Read only, 32 bit counter.
This contains a count of frames that are successfully transmitted, as indicated by the status value transmitOK, to a group destination address other than broadcast. The update occurs in the LayerMgmtTransmitCounters procedure (5.2.4.2).

(8) Number of broadcastFramesTransmittedOK: Optional, Read only, 32 bit counter.
This contains a count of the frames that were successfully transmitted as indicated by the TransmitStatus transmitOK, to the broadcast address. Frames transmitted to multicast addresses are not broadcast frames and are excluded. The update occurs in the LayerMgmtTransmitCounters procedure (5.2.4.2).

5.2.2.1.2 MAC Transmit Error Statistics Descriptions. This section defines the MAC sublayer transmission related error statistics.

(1) Number of lateCollision: Recommended Read only, 32 bit counter.
This contains a count of the times that a collision has been detected later than 512 bit times into the transmitted packet. A late collision is counted twice, i.e., both as a collision and as a lateCollision. This counter is incremented when the lateCollisionCount variable is nonzero. The update is incremented in the LayerMgmtTransmitCounters procedure (5.2.4.2).

(2) Number of frames aborted due to excessiveCollision: Recommended, Read only, 32 bit counter.
This contains a count of the frames that due to excessive collisions are not transmitted successfully. This counter is incremented when the value of the attempts variable equals attemptLimit during a transmission. The update occurs in the LayerMgmtTransmitCounters procedure (5.2.4.2).

(3) Number of frames lost due to internalMACTransmitError: Recommended, Read only, 32 bit counter.
This contains a count of frames that would otherwise be transmitted by the station, but could not be sent due to an internal MAC sublayer transmit error. If this counter is incremented, then none of the other counters in this section are incremented. The exact meaning and mechanism for incrementing this counter is implementation-dependent.

(4) Number of carrierSenseErrors: Recommended, Read only, 32 bit counter.
This contains a count of times that the carrierSense variable was not asserted or was deasserted during the transmission of a frame without collision (see 7.2.4.6). This counter is incremented when the carrierSenseFailure flag is true at the end of transmission. The update occurs in the LayerMgmtTransmitCounters procedure (5.2.4.2).

(5) Number of frames with excessiveDeferral: Optional, Read only, 32 bit counter.
This contains a count of frames that were deferred for an excessive period of time. This counter may only be incremented once per LLC transmission. This counter is incremented when the excessDefer flag is set. The update occurs in the LayerMgmtTransmitCounters procedure (5.2.4.2).

5.2.2.1.3 MAC Receive Statistics Descriptions

(1) Number of framesReceivedOK: Mandatory, Read only, 32 bit counter.
This contains a count of frames that are successfully received (receiveOK). This does not include frames received with frame-too-long, FCS, length or alignment errors, or frames lost due to internal MAC sublayer error. This counter is incremented when the ReceiveStatus is reported as receiveOK. The update occurs in the LayerMgmtReceiveCounters procedure (5.2.4.3).

(2) Number of octetsReceivedOK: Recommended, Read only, 32 bit counter.
This contains a count of data and padding octets in frames that are successfully received. This does not include octets in frames received with frame-too-long, FCS, length or alignment errors, or frames lost due to internal MAC sublayer error. This counter is incremented when the result of a

reception is reported as a receiveOK status. The update occurs in the LayerMgmtReceiveCounters procedure (5.2.4.3).

(3) Number of multicastFramesReceivedOK: Optional, Read only, 32 bit counter.
This contains a count of frames that are successfully received and are directed to an active non-broadcast group address. This does not include frames received with frame-too-long, FCS, length or alignment errors, or frames lost due to internal MAC sublayer error. This counter is incremented as indicated by the receiveOK status, and the value in the destinationField. The update occurs in the LayerMgmtReceiveCounters procedure (5.2.4.3).

(4) Number of broadcastFramesReceivedOK: Optional, Read only, 32 bit counter.
This contains a count of frames that are successfully received and are directed to the broadcast group address. This does not include frames received with frame-too-long, FCS, length or alignment errors, or frames lost due to internal MAC sublayer error. This counter is incremented as indicated by the receiveOK status, and the value in the destinationField. The update occurs in the LayerMgmtReceiveCounters procedure (5.2.4.3).

5.2.2.1.4 MAC Receive Error Statistics Descriptions. This section defines the MAC sublayer reception related error statistics. Note that a hierarchical order has been established such that when multiple error statuses can be associated with one frame, only one status is returned to the LLC. This hierarchy in descending order is as follows:

frameTooLong
alignmentError
frameCheckError
lengthError

The following counters are primarily incremented based on the status returned to the LLC, and therefore the hierarchical order of the counters is determined by the order of the status.

(1) Number of frames received with frameCheckSequenceErrors: Mandatory, Read only, 32 bit counter.
This contains a count of frames that are an integral number of octets in length and do not pass the FCS check. This counter is incremented when the ReceiveStatus is reported as frameCheckError. The update occurs in the LayerMgmtReceiveCounters procedure (5.2.4.3).

(2) Number of frames received with alignmentErrors: Mandatory, Read only, 32 bit counter.
This contains a count of frames that are not an integral number of octets in length and do not pass the FCS check. This counter is incremented when the ReceiveStatus is reported as alignmentError. The update occurs in the LayerMgmtReceiveCounters procedure (5.2.4.3).

(3) Number of frames lost due to internalMACReceiveError: Recommended, Read only, 32 bit counter.
This contains a count of frames that would otherwise be received by the station, but could not be accepted due to an internal MAC sublayer receive error. If this counter is incremented, then none of the other counters in this section are incremented. The exact meaning and mechanism for incrementing this counter is implementation-dependent.

(4) Number of frames received with inRangeLengthErrors: Optional, Read only, 32 bit counter.
This contains a count of frames with a length field value between the minimum unpadded LLC data size and the maximum allowed LLC data size, inclusive, that does not match the number of LLC data octets received. The counter also contains frames with a length field value less than the minimum unpadded LLC data size. The update occurs in the LayerMgmtReceiveCounters procedure (5.2.4.3).

(5) Number of frames received with outOfRangeLengthField: Optional, Read only, 32 bit counter.
This contains a count of frames with a length field value greater than the maximum allowed LLC data size. The update occurs in the LayerMgmtReceiveCounters procedure (5.2.4.3).

(6) Number of frames received with frameTooLongErrors: Optional, Read only, 32 bit counter.
This contains a count of frames that are received and exceed the maximum permitted frame size. This counter is incremented when the status of a frame reception is frameTooLong. The update occurs in the LayerMgmtReceiveCounters procedure (5.2.4.3).

5.2.2.2 MAC Actions. This subsection defines the actions offered by the MAC sublayer to the LME client.

These actions enable the LME client to influence the behavior of the MAC sublayer, i.e., to execute "actions" on the MAC sublayer for management purposes. Many of the following actions enable or disable some function; if either the enable or disable action is implemented, the corresponding disable or enable action must also be implemented. If the enable/disable action is supported, then its corresponding read action must also be supported.

In implementing any of the following actions, receptions and transmissions that are in progress are completed before the action takes effect.

The security considerations related to the following actions should be properly addressed by the SMAE.

The items in parenthesis in the descriptions are the procedures that are affected by these actions.

5.2.2.2.1 MAC Action Definitions

(1) initializeMAC: Mandatory
Call the Initialize procedure (4.2.7.5). This action also results in the initialization of the PLS.

(2) enablePromiscuousReceive: Recommended
Cause the LayerMgmtRecognizeAddress function to accept frames regardless of their destination address (LayerMgmtRecognizeAddress function).
Frames without errors received solely because this action is set are counted as frames received correctly; frames received in this mode that do contain errors update the appropriate error counters.

(3) disablePromiscuousReceive: Recommended
Cause the MAC sublayer to return to the normal operation of carrying out address recognition procedures for station, broadcast, and multicast group addresses (LayerMgmtRecognizeAddress function).

(4) readPromiscuousStatus: Recommended
Return true if promiscuous mode enabled, and false otherwise (LayerMgmtRecognizeAddress function).

(5) addGroupAddress: Recommended
Add the supplied multicast group address to the address recognition filter (RecognizedAddress function).

(6) deleteGroupAddress: Recommended
Delete the supplied multicast group address from the address recognition filter (RecognizeAddress function).

(7) readMulticastAddressList: Recommended
Return the current multicast address list.

(8) enableMacSublayer: Optional
Cause the MAC sublayer to enter the normal operational state at idle. The PLS is reset by this operation (see 7.2.2.2.1). This is accomplished by setting receiveEnabled and transmitEnabled to true.

(9) disableMacSublayer: Optional
Cause the MAC sublayer to end all transmit and receive operations, leaving it in a disabled state. This is accomplished by setting receiveEnabled and transmitEnabled to false.

(10) readMACEnableStatus: Optional
Return true if MAC sublayer is enabled, and false if disabled. This is accomplished by checking the values of the receiveEnabled and transmitEnabled variables.

(11) enableTransmit: Optional
Enable MAC sublayer frame transmission (TransmitFrame function). This is accomplished by setting transmitEnabled to true.

(12) disableTransmit: Optional
Inhibit the transmission of further frames by the MAC sublayer (TransmitFrame function). This is accomplished by setting transmitEnabled to false.

(13) readTransmitEnableStatus: Optional
Return true if transmission is enabled and false otherwise. This is accomplished by checking the value of the transmitEnabled variable.

(14) enableMulticastReceive: Optional
Cause the MAC sublayer to return to the normal operation of multicast frame reception.

(15) disableMulticastReceive: Optional
Inhibit the reception of further multicast frames by the MAC sublayer.

(16) readMulticastReceiveStatus: Optional
Return true if multicast receive is enabled, and false otherwise.
(17) modifyMACAddress: Optional
Change the MAC station address to the one supplied (RecognizeAddress function). Note that the supplied station address shall not have the group bit set and shall not be the null address.
(18) readMACAddress: Optional
Read the current MAC station address.
(19) executeSelftest: Optional
Execute a self test and report the results (success or failure). The mechanism employed to carry out the self test is not defined in this standard.

5.2.3 Physical Layer Management Facilities. This section of the standard defines the Layer Management facilities for the Physical Layer.

5.2.3.1 Physical Statistics. The statistics defined in this section are implemented by means of counters.

In the following definition, the term "Read only" specifies that the object cannot be written by the client of the LME.

Note that the carrierSenseFailed statistic is a statistic relating to the physical layer, but is listed and maintained in the MAC sublayer for ease of implementation.

5.2.3.1.1 Physical Statistics Descriptions

(1) Number of SQETestErrors: Recommended, Read only, 32[8] bit counter.
This contains a count of times that the SQE_TEST_ERROR was received. The SQE_TEST_ERROR is set in accordance with the rules for verification of the SQE detection mechanism in the PLS Carrier Sense Function (see 7.2.4.6).

5.2.4 Layer Management Model. The following model provides the descriptions for Layer Management facilities.

5.2.4.1 Common Constants and Types. The following are the common constants and types required for the Layer Management procedures:

```
const
   maxFrameSize = ...; {in octets, implementation-dependent, see 4.4}
   maxDeferTime = ...; {2 × (maxFrameSize × 8), in bits, error timer limit for maxDeferTime}
   maxLarge = 4294967295; {maximum value (2^32 – 1) of wraparound 32 bit counter}
   max64 = xxxxxxxxx; {maximum value (2^64 – 1) of wraparound 64 bit counter}
   oneBitTime = 1; {the period it takes to transmit one bit}
type
   CounterLarge = 0..maxLarge--See footnote.;
```

5.2.4.2 Transmit Variables and Procedures. The following items are specific to frame transmission:

```
var
   excessDefer: Boolean; {set in process DeferTest}
   carrierSenseFailure: Boolean; {set in process CarrierSenseTest}
   transmitEnabled: Boolean; {set by MAC action}
   lateCollisionError: Boolean; {set in Section 4 procedure WatchForCollision}
   deferred: Boolean; {set in Section 4 function TransmitLinkMgmt}
   carrierSenseTestDone: Boolean; {set in process CarrierSenseTest}
```

[8]32 bit counter size specification is not a part of this ISO/IEC standard. Resolution of 32 vs. 64 bit counter size will be addressed during the further work required to develop this section into a specification sufficient for ISO/IEC interoperability requirements.

```
lateCollisionCount: 0..attemptLimit – 1; {count of late collision that is used in Section 4
                                          TransmitLinkMgmt}

{MAC transmit counters}
framesTransmittedOK: CounterLarge; {mandatory}
singleCollisionFrames: CounterLarge; {mandatory}
multipleCollisionFrames: CounterLarge; {mandatory}
collisionFrames: array [1..attemptLimit – 1] of CounterLarge; {recommended}
octetsTransmittedOK: CounterLarge; {recommended}
deferredTransmissions: CounterLarge; {recommended}
multicastFramesTransmittedOK: CounterLarge; {optional}
broadcastFramesTransmittedOK: CounterLarge; {optional}
{MAC transmit error counters}
lateCollision: CounterLarge; {recommended}
excessiveCollision: CounterLarge; {recommended}
carrierSenseErrors: CounterLarge; {optional}
excessiveDeferral: CounterLarge; {optional}
```

Procedure LayerMgmtTransmitCounters is invoked from the TransmitLinkMgmt function in 4.2.8 to update the transmit and transmit error counters.

```
procedure LayerMgmtTransmitCounters;
begin
  while not carrierSenseTestDone do nothing;
  if transmitSucceeding then
  begin
    IncLargeCounter(framesTransmittedOK);
    SumLarge(octetsTransmittedOK, dataSize/8); {dataSize (in bits) is defined in 4.2.7.1}
    if destinationField = ... {check to see if to a multicast destination}
      then IncLargeCounter(multicastFramesTransmittedOK);
    if destinationField = ... {check to see if to a broadcast destination}
      then IncLargeCounter(broadcastFramesTransmittedOK);

    if attempts > 1 then
    begin {transmission delayed by collision}
      if attempts = 2 then
        IncLargeCounter(singleCollisionFrames) {delay by 1 collision}
      else {attempts > 2, delayed by multiple collisions}
        IncLargeCounter(multipleCollisionFrames)
      IncLargeCounter(collisionFrames[attempts – 1]);
    end; {delay by collision}
  end; {transmitSucceeding}

  if deferred and (attempts = 1) then
    IncLargeCounter(deferredTransmissions);
  if lateCollisionCount > 0 then {test if late collision detected}
    SumLarge(lateCollision, lateCollisionCount);
  if attempts = attemptLimit and not transmitSucceeding then
    IncLargeCounter(excessiveCollision);
  if carrierSenseFailure then
    IncLargeCounter(carrierSenseErrors);
  if excessDefer then
    IncrementLargeCounter(excessiveDeferral);
end; {LayerMgmtTransmitCounters}
```

The DeferTest process sets the excessDefer flag if a transmission attempt has been deferred for a period of time longer than maxDeferTime.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.