Possible clarifications to make reserved only for future definition in standard

NOTE—Headings are provided rather than page/line numbers specific to a draft. Paragraphs are included to provide context so readers may determine if the proposed change is technically accurate. The provided context may be separated significantly from the heading in the actual draft. No attempt has been made to preserve indentation and other formatting. I didn't check if any of the following possible changes are in deprecated clauses.

NOTE – There are a lot of various cases (e.g., text and table entry descriptions) where I also did not propose modifying any of the "reserved for future use" or similar text (e.g., "reserved for IEEE 802.3 future use"). Similarly, I did not highlight things like "reserved for future speeds". I viewed these as redundant with defining "reserved" as being for future specification by 802.3.

NOTE – I have highlighted cases where values or ranges of values have been reserved for use or specification outside the standard. Examples include reserved for INCITS T11 Fiber Channel use, or more problematic, reserved for use by the owner of the OUI or similar text. IMHO, this is the biggest challenge for getting a definition of reserved correct if we want to make it specifically for future specification in Std 802.3.

NOTE – I have not attempted to reconcile the capitalization of "Reserved" versus "reserved". Most common use appears to be "Reserved" is for things like filed names (or perhaps because it was the first word in the table cell in some clauses) and "reserved" in text, table descriptions, etc. This would be a lot of changes to make consistent, though we could do something in the 802.3 style guide for future clauses.

1.4.352 P2MP Discovery window: A time period in a given wavelength band ~~reserved~~ used by the OLT exclusively for the discovery process.

B.2.3 Jitter budget

The remainder of the jitter that can be tolerated by the Manchester decoder in a receiver is ~~reserved to allow~~ allocated for distortion of the signal due to noise, receiver threshold offset, receiver skew, and receiver sampling timing error.

NOTE—All of the clause 30 changes can be fixed by search and manual replace "reserved" with "assigned" or other synonym. (There are occurrences of "preserved", one " Reserved for future use.

30.3.3.4 aMACControlFramesReceived

BEHAVIOUR DEFINED AS:
A count of MAC Control frames passed by the MAC sublayer to the MAC Control

sublayer. This counter is incremented when a ReceiveFrame function call returns a valid frame with a lengthOrType field value equal to the ~~reserved~~ assigned Type for 802.3_MAC_Control as specified in 31.4.1.3.;

30.3.3.5 aUnsupportedOpcodesReceived

BEHAVIOUR DEFINED AS:
A count of MAC Control frames received that contain an opcode from Table 31A–1 that is not supported by the device. This counter is incremented when a ReceiveFrame function call returns a valid frame with a lengthOrType field value equal to the ~~reserved~~ assigned Type for 802.3_MAC_Control as specified in 31.4.1.3, and with an opcode for a function that is not supported by the device.;

30.3.4.3 aPAUSEMACCtrlFramesReceived

BEHAVIOUR DEFINED AS:
A count of MAC Control frames passed by the MAC sublayer to the MAC Control sublayer. This counter is incremented when a ReceiveFrame function call returns a valid frame with: (1) a lengthOrType field value equal to the ~~reserved~~ assigned Type for 802.3_MAC_Control as specified in 31.4.1.3, and (2) an opcode indicating the PAUSE operation.;

30.3.5.1.5 aMPCPRemoteMACAddress

BEHAVIOUR DEFINED AS:

This value is updated on reception of a valid frame with (1) a destinationField equal to the ~~reserved~~ assigned multicast address for MAC Control specified in 31A, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for MAC Control as specified in 31A, (3) an opcode value ~~reserved~~ assigned for one of MPCP messages, as specified in 31A.;

30.3.5.1.8 aMPCPMACCtrlFramesReceived

BEHAVIOUR DEFINED AS:

Increment counter by one when a ReceiveFrame function call returns a valid frame with: (1) a lengthOrType field value equal to the ~~reserved~~ assigned Type for 802.3_MAC_Control as specified in 31.4.1.3, and (2) an opcode indicating an MPCP frame.;

30.3.5.1.14 aMPCPRxGate

BEHAVIOUR DEFINED AS:

Increment the counter by one when a ReceiveFrame function call returns a valid frame with: (1) a destinationField equal to the ~~reserved~~ assigned multicast address for MAC Control specified in 31A, or 1 unique physical address associated with this station, (2) a lengthOrType field value equal to the ~~reserved~~ assigned Type for 802.3_MAC_Control as specified in 31.4.1.3, (3) an opcode indicating a GATE 3 MPCPDU.;

30.3.5.1.15 aMPCPRxRegAck

BEHAVIOUR DEFINED AS:
A count of the number of times a REGISTER_ACK MPCP frames reception occurs.

Increment the counter by one when a ReceiveFrame function call returns a valid frame with: (1) a destinationField equal to the ~~reserved~~ assigned multicast address for MAC Control specified in 31A, (2) a lengthOrType field value equal to the ~~reserved~~ assigned Type for 802.3_MAC_Control as specified in 31.4.1.3, (3) an opcode indicating a REGISTER_ACK MPCPDU.;

30.3.5.1.16 aMPCPRxRegister

BEHAVIOUR DEFINED AS:
A count of the number of times a REGISTER MPCP frames reception occurs.

Increment the counter by one when a ReceiveFrame function call returns a valid frame with: (1) a destinationField equal to the unique physical address associated with this station, (2) a lengthOrType field value equal to the ~~reserved~~ assigned Type for 802.3_MAC_Control as specified in 31.4.1.3, (3) an opcode indicating a REGISTER MPCPDU.;

30.3.5.1.17 aMPCPRxRegRequest

BEHAVIOUR DEFINED AS:

Increment the counter by one when a ReceiveFrame function call returns a valid frame with: (1) a destinationField equal to the ~~reserved~~ assigned multicast address for MAC Control specified in 31A, (2) a lengthOrType field value equal to the ~~reserved~~ assigned Type for 802.3_MAC_Control as specified in 31.4.1.3, (3) an opcode indicating a REGISTER_REQ MPCPDU.;

30.3.5.1.18 aMPCPRxReport

BEHAVIOUR DEFINED AS:

Increment the counter by one when a ReceiveFrame function call returns a valid frame with: (1) a destinationField equal to the ~~reserved~~ assigned multicast address for MAC Control specified in 31A, (2) a lengthOrType field value equal to the ~~reserved~~ assigned Type for 802.3_MAC_Control as specified in 31.4.1.3, (3) an opcode indicating a REPORT MPCPDU.;

30.3.6.1.5 aOAMRemoteMACAddress

BEHAVIOUR DEFINED AS:

This value is updated on reception of a valid frame with (1) a destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for-Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3.;

30.3.6.1.9 aOAMRemotePDUConfiguration

BEHAVIOUR DEFINED AS:

This value is updated on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for-Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAM code equals the OAM Information code as specified in Table 57–4, (5) the frame contains a Local Information TLV (see 57.5.2.1).;

30.3.6.1.11 aOAMRemoteFlagsField

BEHAVIOUR DEFINED AS:

A string of seven bits corresponding to the Flags field (see Table 57–3) in the most recently received OAMPDU.

The first bit corresponds to the Link Fault bit in the Flags field. The second bit corresponds to the Dying Gasp bit in the Flags field. The third bit corresponds to the Critical Event bit in the Flags field. The fourth bit corresponds to the Local Evaluating bit in the Flags field. The fifth bit corresponds to the Local Stable bit in the Flags field. The sixth bit corresponds to the Remote Evaluating bit in the Flags field. The seventh bit corresponds to the Remote Stable bit in the Flags field.

This value is updated on reception of a valid frame with (1) a destinationField equal to the multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAM code equals one of the codes as specified in Table 57–4.;

30.3.6.1.13 aOAMRemoteRevision

BEHAVIOUR DEFINED AS:

The value of the Revision field (see 57.5.2.1) in the Local Information TLV of the most recently received Information OAMPDU.

This value is updated on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equal to the Information code as specified in Table 57–4, (5) the frame contains a Local Information TLV (see 57.5.2.1).;

30.3.6.1.15 aOAMRemoteState

BEHAVIOUR DEFINED AS:

This value is updated on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equal to the Information code as specified in Table 57–4, (5) the frame contains a Local Information TLV (see 57.5.2.1).;

30.3.6.1.16 aOAMRemoteVendorOUI

BEHAVIOUR DEFINED AS:

This value is updated on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) a OAMPDU code equal to the Information code as specified in Table 57–4, (5) the frame contains a Local Information TLV (see 57.5.2.1).;

30.3.6.1.17 aOAMRemoteVendorSpecificInfo

BEHAVIOUR DEFINED AS:

This value is updated on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equal to the Information code as specified in Table 57–4, (5) the frame contains a Local Information TLV (see 57.5.2.1).;

30.3.6.1.19 aOAMUnsupportedCodesRx

BEHAVIOUR DEFINED AS:
A count of OAMPDUs received that contain an OAM code from Table 57–4 that are not supported by the device. This counter is incremented on reception of a valid frame with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) an OAMPDU code for a function that is not supported by the device.;

30.3.6.1.21 aOAMInformationRx

BEHAVIOUR DEFINED AS:
A count of OAMPDUs received that contain the OAM Information code specified in Table 57–4. This counter is incremented on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equals the OAM Information code and is supported by the 1 device.;

30.3.6.1.22 aOAMUniqueEventNotificationTx

BEHAVIOUR DEFINED AS:
A count of OAMPDUs passed to the OAM subordinate sublayer for transmission that contain the Event Notification code specified in Table 57–4. This counter is incremented when a CTL:OAMI.request service primitive is generated within the OAM sublayer, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols multicast address specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equals the Event Notification code, (5) the Sequence Number field is not equal to the Sequence Number field of the last transmitted Event Notification OAMPDU.;

30.3.6.1.23 aOAMDuplicateEventNotificationTx

BEHAVIOUR DEFINED AS:
A count of OAMPDUs passed to the OAM subordinate sublayer for transmission that contain the Event Notification code specified in Table 57–4. This counter is incremented when a CTL:OAMI.request service primitive is generated within the OAM sublayer, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equals the Event Notification code, (5) the Sequence Number field is equal to the Sequence Number field of the last transmitted Event Notification OAMPDU.;

30.3.6.1.24 aOAMUniqueEventNotificationRx

BEHAVIOUR DEFINED AS:
A count of the OAMPDUs received that contain the Event Notification code specified in Table 57–4. This counter is incremented on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equals the Event Notification code, (5) the Sequence Number field is not equal to the Sequence Number field of the last received Event

30.3.6.1.25 aOAMDuplicateEventNotificationRx

BEHAVIOUR DEFINED AS:
A count of the OAMPDUs received that contain the Event Notification code specified in Table 57–4. This counter is incremented on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as 15 specified in Table 57A–3, (4) the OAMPDU code equals the Event Notification code, (5) the Sequence Number field is equal to the Sequence Number field of the last received Event Notification OAMPDU.;

30.3.6.1.27 aOAMLoopbackControlRx

BEHAVIOUR DEFINED AS:
A count of OAMPDUs received that contain the Loopback Control code specified in Table 57–4. This counter is incremented on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equals the Loopback Control code and is supported by the device.;

30.3.6.1.29 aOAMVariableRequestRx

BEHAVIOUR DEFINED AS:
A count of OAMPDUs received that contain the Variable Request code specified in Table 57–4. This counter is incremented on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equals the Variable Request code and is supported by the device.;

30.3.6.1.31 aOAMVariableResponseRx

BEHAVIOUR DEFINED AS:
A count of OAMPDUs received that contain the Variable Response code specified in Table 57–4. This counter is incremented on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to

the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equals the Variable Response code and is supported by the device.;

30.3.6.1.33 aOAMOrganizationSpecificRx

BEHAVIOUR DEFINED AS:
A count of OAMPDUs received that contain the Organization Specific code specified in Table 57–4. This counter is incremented on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) a Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) the OAMPDU code equals the Organization Specific code and is supported by the device.;

30.3.6.1.42 aOAMRemoteErrSymPeriodEvent

BEHAVIOUR DEFINED AS:
A sequence of six integers corresponding to the respective fields in the most recently received Errored Symbol Period Event TLV in an Event Notification OAMPDU (see 57.4.3.2). This sequence is updated on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) OAMPDU Code field value equal to the Event Notification code as specified in Table 57–4, (5) an Event TLV Type field equal to the Errored Symbol Period Event value as specified in Table 57–12. If more than one Event TLV of the same Event Type is present within an Event Notification OAMPDU, the Event with the most recent timestamp should be used.;

30.3.6.1.43 aOAMRemoteErrFrameEvent

BEHAVIOUR DEFINED AS:
A sequence of six integers corresponding to the respective fields in the most recently received Errored Frame Event TLV in an Event Notification OAMPDU (see 57.4.3.2). This sequence is updated on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to

the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) OAMPDU Code field value equal to the Event Notification code as specified in Table 57–4, (5) an Event TLV Type field equal to the Errored Frame Event value as specified in Table 57–12. If more than one Event TLV of the same Event Type is present within an Event Notification OAMPDU, the Event with the most recent timestamp should be used.;

30.3.6.1.44 aOAMRemoteErrFramePeriodEvent

BEHAVIOUR DEFINED AS: A sequence of six integers corresponding to the respective fields in the most recently received Errored Frame Period Event TLV in an Event Notification OAMPDU (see 57.4.3.2). This sequence is updated on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) OAMPDU Code field value equal to the Event Notification code as specified in Table 57–4, (5) an Event TLV Type field equal to the Errored Frame Period Event value as specified in Table 57–12. If more than one Event TLV of the same Event Type is present within an Event Notification OAMPDU, the Event with the most recent timestamp should be used.;

30.3.6.1.45 aOAMRemoteErrFrameSecsSummaryEvent

BEHAVIOUR DEFINED AS:
A sequence of six integers corresponding to the respective fields in the most recently received Errored Frame Seconds Summary Event TLV in an Event Notification OAMPDU (see 57.4.3.2). This sequence is updated on reception of a valid frame, with (1) destinationField equal to the ~~reserved~~ assigned multicast address for Slow_Protocols specified in Table 57A–1, (2) lengthOrType field value equal to the ~~reserved~~ assigned Type for Slow_Protocols as specified in Table 57A–2, (3) Slow_Protocols subtype value equal to the subtype ~~reserved~~ assigned for OAM as specified in Table 57A–3, (4) OAMPDU Code field value equal to the Event Notification code as specified in Table 57–4, (5) an Event TLV Type field equal to the Errored Frame Seconds Summary Event value as specified in Table 57–12. If more than one Event TLV of the same Event Type is present within an Event Notification OAMPDU, the Event with the most recent timestamp should be used.;

30.3.8.2 aEXTENSIONMACCtrlFramesReceived

BEHAVIOUR DEFINED AS:
A count of MAC Control frames passed by the MAC sublayer to the MAC Control sublayer. This counter is incremented when a ReceiveFrame function call returns a valid frame with: (1) a lengthOrType field value equal to the ~~reserved~~ assigned Type for 802.3_MAC_Control as specified in 31.4.1.3, and (2) an opcode indicating the EXTENSION operation.;

31.5.3.1 Constants

802.3_MAC_Control
The 16-bit Length/Type field value ~~reserved~~ used for CSMA/CD MAC Control ~~usage~~, specified in 31.4.1.3.

32.1.2.1 Physical coding sublayer (PCS)

NOTE—I don't have a good of confidence as for other possible changes. This one appears with my superficial knowledge of 100BASE-T2 to be a synonym for "assigned.

The functions performed by the PCS comprise the generation of continuous quinary symbol sequences to be transmitted over each wire pair. During *data mode*, i.e., when a data stream from the MII is transmitted, the four bits representing the TXD<3:0> data nibble are scrambled by a side-stream scrambler and encoded into a pair of quinary symbols. During *idle mode*, i.e., between transmission of consecutive data streams, the sequences of quinary symbols are generated with an encoding rule that differs from the encoding rule used in data mode. Through this technique, sequences of arbitrary quinary symbols that represent data can easily be distinguished from sequences that represent the idle mode. Furthermore, idle mode encoding takes into account the information of whether the local PHY is operating reliably or not and allows conveying this information to the remote station. A transition from the idle to the data mode is signaled by inserting a Start- of-Stream delimiter that consists of a pattern of two consecutive pairs of quinary symbols. Similarly, the end of a data stream transmission is signaled by inserting an End-of-Stream delimiter that also consists of a pat- tern of two consecutive pairs of quinary symbols. Further patterns are ~~reserved~~assigned for signaling a transmit error during transmission of a data stream.

32.3.1.2 PCS Transmit function

NOTE—I don't have a good of confidence as for other possible changes. This one appears with my superficial knowledge of 100BASE-T2 to be a synonym for "assigned.

In normal mode of operation, the tx_mode parameter, which is transferred from PHY Control to the PCS via the PHYC_TXMODE.indication message, assumes the value tx_mode=SEND_N, and the PCS Transmit function generates at each symbol period pairs of quinary symbols that represent data or the idle mode. A symbol period T is equal to 40 ns. A time index n, where n is an integer, is introduced to establish a temporal relationship between different symbol periods. The tx_symb_vector parameter at time n is a two-element vector of quinary symbols (A_n, B_n) that is transferred to the PMA via PMA_UNITDATA.request. The PMA shall transmit symbols A_n and B_n over wire pairs BI_DA and BI_DB, respectively. During transmission of data, the four bits representing the TXD<3:0> data nibble are scrambled by the OCS using a side-stream scrambler then encoded into a pair of quinary symbols and transferred to the PMA. The idle mode is signaled by a sequence of pairs of quinary symbols that are also generated using the side-stream transmit scrambler. However, the encoding rules by which the quinary symbols are obtained are different for the data and the idle modes. This allows, at the receiver, sequences of quinary symbol pairs that represent data to be distinguished from sequences of quinary symbol pairs that represent the idle mode. A transition from the idle mode to the data mode is signalled by inserting a Start-of-Stream delimiter that consists of a pattern of two consecutive pairs of quinary symbols. Similarly, the end of transmission of data is signalled by an End-of-Stream delimiter that also consists of a pattern of two consecutive pairs of quinary symbols. Further patterns are ~~reserved~~assigned for signaling the assertion of TX_ER within a stream of data.

33.5.1.1.4 Pair Control (11.3:2)

NOTE—I want this one reviewed by POE folk.

If bit 12.0 is one, writing to these register bits shall set mr_pse_alternative to the corresponding value: ‘01’ = A and ‘10’ = B. The combinations ‘00’ and ‘11’ for bits 11.3:2 are ~~reserved and will never be assigned~~ defined values. Reading bits 11.3:2 returns an unambiguous result of ‘01’ or ‘10’ that may be used to determine the presence of the PSE Control register.

31B.3.2.1 Constants

NOTE—While not changing this name could be argued as inconsistent with the definition of reserved, because it is a name, it probably does not need to be changed.

reserved_multicast_address
The 48-bit address specified in 31B.1 (a).

31B.1 PAUSE description

The globally assigned 48-bit multicast address 01-80-C2-00-00-01 has been reserved for use in MAC Control PAUSE frames for inhibiting transmission of data frames from a DTE in a full duplex mode IEEE 802.3 LAN. IEEE 802.1D-conformant bridges will not forward frames sent to this multicast destination address, regardless of the state of the bridge's ports, or whether or not the bridge implements the MAC Control sublayer. To allow generic full duplex flow control, stations implementing the PAUSE operation shall instruct the MAC (e.g., through layer management) to enable reception of frames with destination address equal to this multicast address.

31B.3.2.1 Constants

reserved_multicast_address
The 48-bit address specified in 31B.1 (a).

33B.3.3 Receive Operation

NOTE—Highlighted text should be the constant name with underscores between words.

Upon receipt of a valid MAC Control frame with the opcode indicating PAUSE and the destination address indicating either: (1) the reserved multicast address specified in 31B.1, or (2) the unique physical address associated with this station, the MAC Control sublayer starts a timer for the length of time specified by the pause_time request_operand in the Control frame (see 31B.2). The value of the variable pause_timer_Done is set to false upon the setting of the pause_timer to a non-zero value by the MAC Control PAUSE Operation Receive state diagram. The value of the variable pause_timer_Done is set to true when the timer value reaches zero (i.e., the timer expires). If the received PAUSE operation indicates a pause_timer value of zero, the value of pause_timer_Done is set to true immediately.

The receipt of a valid MAC Control frame with the opcode indicating PAUSE and the destination address indicating the reserved multicast address specified in 31B.1 or the unique physical address associated with this station will cause the pause_timer to be set according to the pause_time request_operand in the newly received frame, regardless of the current setting of the pause_timer, i.e., new PAUSE operations override earlier PAUSE operations.

31D.1 PFC description

The globally assigned 48-bit multicast address 01-80-C2-00-00-01 has been ~~reserved~~assigned for use in MAC Control frames. Bridges conformant to IEEE Std

802.1D or IEEE Std 802.1Q will not forward frames sent to this multicast destination address, regardless of the state of the bridge’s ports, and whether or not the bridge implements the MAC Control sublayer. To allow PFC full duplex flow control, stations implementing the PFC operation shall instruct the MAC (e.g., through layer management) to enable reception of frames with destination address equal to this multicast address.

31D.5 PFC receive

NOTE—Highlighted text should be the constant name with underscores between words.

The opcode-independent MAC Control sublayer Receive state diagram accepts and parses valid frames received from the MAC sublayer. The functions specified in this subclause are performed upon receipt of a valid Control frame containing the PFC opcode, and define the function called by the INITIATE MAC CONTROL FUNCTION state of Figure 31–4. (See 31.5.3.) Upon receipt of a valid MAC Control frame with the opcode indicating PFC and the destination address indicating the reserved multicast address specified in 31D.1, the MAC Control sublayer generates the MA_CONTROL.indication to the MAC Control Client.

34.1.4 Auto-Negotiation, type 1000BASE-X

Auto-Negotiation (Clause 37) provides a 1000BASE-X device with the capability to detect the abilities (modes of operation) supported by the device at the other end of a link segment, determine common abilities, and configure for joint operation. Auto-Negotiation is performed upon link startup through the use of a special sequence of ~~reserved~~ link codewords. Clause 37 adopts the basic architecture and algorithms from Clause 28, but not the use of fast link pulses. Auto-Negotiation for 1000BASE-KX is defined in Clause 73.

38.1.1 Physical Medium Dependent (PMD) sublayer service interface

NOTE—Delay requirements from the MDI to GMII that include the PMD layer are specified in Clause 36. Of this budget, 4 ns is ~~reserved~~ allocated for each of the transmit and receive functions of the PMD.

39.5.1.2 Style-2 connector specification

NOTE—Not totally comfortable with these uses. If a different word is used, then Table 39-8 would need to be changed.

NOTE 1—Style-1 pins 2 and 8 (Style-2 pins 7 and 2) are reserved for applications that assign these pins to power and ground.

NOTE 2—Style-1 pin 3 (Style-2 pin 4) is reserved for applications that assign this pin to a Fault Detect function.

NOTE 3—Style-1 pin 7 (Style-2 pin 5) is reserved for applications that assign this pin to an Output Disable function.

40.3.1.3 PCS Transmit function

The transition from idle or carrier extension to data is signalled by inserting a SSD, and the end of transmission of data is signalled by an ESD. Further code-groups are reserved for signaling the assertion of TX_ER within a stream of data, carrier extension, CSReset, and other control functions. During idle and carrier extension encoding, …

NOTE—Clauses 48, 49, 55 have text of the general form “Reserved for INCITS T11 Fibre Channel use. This could require looking at the definition “in this standard” part of definition, or change to Assigned INCITS T11 for their use.

NOTE—Clause 57 seems to be unique in placing *Reserved* in italics (mostly in tables).

Table 57-4

Reserved for Organization Specific Extensions, distinguished by Organizationally Unique Identifier.

58.1.4.1 Delay constraints

Delay requirements which affect the PMD layer are specified in 24.6. Of the budget, up to 12 ns is ~~reserved~~ allocated for each of the transmit and receive functions of the PMD to account for those cases where the PMD includes a pigtail.

59.1.5 Delay constraints

Delay requirements from the MDI to the GMII which include the PMC layer are specified in Clause 36. Of the budget, up to 20 ns is ~~reserved~~ allocated for each of the receive and transmit functions of the PMD to account for those cases where the PMD includes a pigtail.

60.1.5 Delay constraints

Delay requirements from the MDI to the GMII which include the PMD layer are specified in Clause 36. Of the budget, up to 20 ns is ~~reserved~~ allocated for each of the transmit and receive functions of the PMD to account for those cases where the

PMD includes a pigtail.

Table 61-10

02_{16}–$7B_{16}$ — reserved for allocation by IEEE 802.3
$7C_{16}$–$7F_{16}$ — ~~reserved~~ assigned for allocation by ATIS T1E1.4

82_{16}–FB_{16} — reserved for allocation by IEEE 802.3
FC_{16}–FF_{16} — ~~reserved~~ assigned for allocation by ATIS T1E1.4

64.2.2.1 Constants

tailGuard
This constant holds the value used to reserve space at the end of the upstream transmission at the ONU in addition to the size of last MAC service data unit (m_sdu) in units of octets. Space is ~~reserved~~ allocated for the MAC overheads including: preamble, SFD, DA, SA, Length/ Type, FCS, and the End of Packet Delimiter (EPD). The sizes of the above listed MAC overhead items are described in 3.1.1. The size of the EPD is described in 36.2.4.14.

Table 64-3

NOTE—Inconsistent capitalization of Reserved.

76.2.6.1.3.2 LLID

If the packet is transferred, the one octet preceding the LLID is passed without modification. A number of LLIDs have been ~~reserved~~ allocated (see Figure 76–4) for various purposes including downstream broadcast, discovery messages, and upstream registration request messages. An additional block of LLIDs has been set aside for future use and definition. A registered ONU shall not transmit frames with one of these reserved LLIDs.

Table 76–4~~—Reserved~~ LLID values

LLID value	Used in RS	Purpose
0x7FFF	1000BASE-PX	Downstream: 1 Gb/s SCB Upstream: ONU reg Gb/s
0x7FFE	10/1GBASE-PRX	Downstream: 10 Gb/s SCB Upstream: ONU regi Gb/s

	10GBASE-PR	Downstream: 10 Gb/s SCB Upstream: ONU re 10 Gb/s
	10GPASS-XR	Downstream: EPoC SCB Upstream: CNU registr
0x7FFD–0x7F00	—	Reserved for future use

77.2.2.1 Constants

tailGuard
TYPE: integer
This constant holds the value used to reserve space at the end of the upstream transmission at the ONU in addition to the size of last MAC service data unit (m_sdu) in units of octets. Space is ~~reserved~~ allocated for the MAC overheads including: preamble, SFD, DA, SA, Length/Type, FCS, and minimum interpacket gap. The sizes of the above listed MAC overhead items are described in 3.1.1. The size of the minimum IPG is described in 4A.4.2.
VALUE: 38

57A.3 Addressing

NOTE 1—This address is within the range reserved by IEEE Std 802.1D for link-constrained protocols. As such, frames sent to this address will not be forwarded by conformant MAC Bridges; its use is restricted to a single link.

100.3.7.2 Downstream exclusion band rules

-- The 6 MHz of contiguous spectrum ~~reserved~~ used for the PHY link cannot have any exclusion bands or excluded subcarriers.

101.4.3.4.5 PHY Link managed variables

0001 = reserved (used by PHY for contiguous pilots only, if set via MDIO to this value the PHY treats the subcarrier as null)

NOTE—I don't know what to propose for a change here, but based on the parenthetical, it clearly has an assigned use that indicates the value is allocated, not reserved.

101.4.4.5.1 Variables

NOTE—Two cases that resist copy from the pdf.
PILOT_MAP – "this Resource Block is reserved for use by the PHY Link".

RB_Frame – "is reserved for special use in coordinating symbol mapper idle and fill processing".

Table 102-6

NOTE—Inconsistent capitalization of Reserved.

Table 102-11

NOP Instruction negative acknowledge returned in response to a unsuccessfully received NOP Instruction ~~{might just want to keep this as a reserved value}~~

NOTE—How did this get through ballot and publication?

103.2.2.1 Constants

This constant holds the value used to reserve space at the end of the upstream transmission at the CNU in addition to the size of last MAC service data unit (m_sdu) in units of octets. Space is ~~reserved~~ allocated for the MAC overheads including: preamble, SFD, DA, SA, Length/Type, FCS, and minimum interpacket gap. The sizes of the above listed MAC overhead items are described in 3.1.1. The size of the minimum IPG is described in 4A.4.2.

Table 113-1 and Table 113-2

[a]Reserved for INCITS T11 Fibre Channel use.

113.3.2.2.7 Ordered sets

NOTE—I think this one is okay.

Ordered sets are used to extend the ability to send control and status information over the link such as remote fault and local fault status. Ordered sets consist of a control character followed by three data characters. Ordered sets always begin on the first octet of the 25GMII or XLGMII. 25 Gigabit and 40 Gigabit Ethernet use one kind of ordered set: the sequence ordered set (see 81.3.4). The sequence ordered set control character is denoted /Q/. An additional ordered set, the signal ordered set, has been reserved and it begins with another control code. The four-bit O field encodes the control code. See Table 113–1 for the mappings for 25GBASE-T, and Table 113–2 for the mappings for 40GBASE-T.

113.3.2.2.12 ordered set (/O/)

The ordered set control characters (/O/) indicate the start of an ordered set. There are two kinds of ordered sets: the sequence ordered set and the signal ordered set (which is reserved). When it is necessary to designate the control character for the sequence ordered set specifically, /Q/ is used. /O/ is only valid on the first octet of the 25GMII/XLGMII. Receipt of an /O/ on any other octet of TXD indicates an error. For 40 Gb/s transmission, block type field values implicitly encode an /O/ as the first character of the block. The four-bit O code encodes the specific /O/ character for the ordered set.

Table 98B-1

NOTE—This table uses capitalization of RESERVED, contrary to the rest of the document.

Table 126-1

[a]Reserved for INCITS T11 Fibre Channel use.

Amendments 10 and 11

NOTE—I found no problematic uses of reserved in either published Amendment 10 or approved Amendment 11 (D3.3).