**Table 22-1 Permissible encodings of TXD<3:0>, TX\_EN, and TX\_ER**

Make the following changes



**Table 22-2 Permissible encoding of RXD<3:0>, RX\_ER, and RX\_DV**

Make the following changes



**22.2.2.13 Link Fault Signaling**

Link fault signaling is optional on the MII and is active only when both devices on the link advertises it.

Link fault signaling operates as defined in Clause 46.3.4. The 4-byte sequence ordered set is formed on the MII as two Sequence nibbles followed by the three data bytes in lane 1, 2, and 3 as shown in Table 46-5. The data bytes are the six nibbles on TXD<3:0> or RXD<3:0> that immediately follows the two sequence ordered set symbols where the lower 4 bits of each byte are presented on the MII before the upper 4 bits and the byte in lane 1 is transmitted first followed by lane 2 and then lane 3. Unlike data nibbles in a frame, these data nibbles are presented with TX\_EN = 0, TX\_ER = 0, and RX\_DV = 0 and RX\_ER = 0.

Since there is no concept of lanes on the MII, the Sequence ordered set may be presented on the MII at any MII transfer outside of the packet.

Editor’s Note: Items in yellow is subject to further discussion. This draft is loosely based on the text in Clause 97.

**ZZZ.3 Physical Coding Sublayer (PCS)**

 **ZZZ.3.1 PCS service interface (MII)**

The PCS service interface allows the 100BASE-T1L PCS to transfer information to and from a PCS client.
The PCS Interface is defined as the Media Independent Interface (MII) in Clause 22.

**ZZZ.3.2 PCS functions**

The PCS comprises one PCS Reset function and two simultaneous and asynchronous operating functions.
The PCS operating functions are: PCS Transmit and PCS Receive. All operating functions start immediately
after the successful completion of the PCS Reset function.

The PCS reference diagram, Figure ZZZ–4, shows how the two operating functions relate to the messages of the PCS-PMA interface. Connections from the management interface (signals MDC and MDIO) to other
layers are pervasive and are not shown in Figure ZZZ–4.

Editor’s Note: Need to draw something like Figure 97-4 here.

**Figure ZZZ–4—PCS reference diagram**

**ZZZ.3.2.1 PCS Reset function**

PCS Reset initializes all PCS functions. The PCS Reset function shall be executed whenever one of the
following conditions occur:

a) Power for the device containing the PMA has not reached the operating state

b) The receipt of a request for reset from the management entity

PCS Reset sets pcs\_reset = ON while any of the above reset conditions hold true. All state diagrams take the open-ended pcs\_reset branch upon execution of PCS Reset. The reference diagrams do not explicitly show the PCS Reset function.

The control and management interface shall be restored to operation within 10 ms from the setting of
bit 3.2304.15.

Editor’s Note: need to assign a similar register to the one above.

**ZZZ.3.2.2 PCS Transmit function**

The PCS Transmit function shall conform to the PCS 80B/81B Transmit state diagram in Figure ZZZ–14 and
the PCS Transmit bit ordering in Figure ZZZ–5 and Figure ZZZ–7.

When communicating with the MII, the PCS uses a nibble-wide, synchronous data path, with packet
delimiting being provided by transmit control signals and receive control signals. Two nibbles are combined to form either a control or data octet symbol. Alignment of the octet symbols to 8N/(8N+1)
blocks is performed in the PCS. The PMA sublayer operates independently of block and packet boundaries.
The PCS provides the functions necessary to map packets between the MII format and the PMA service
interface format.

When the transmit channel is in the data mode, the PCS Transmit process continuously generates 8N/(8N+1)blocks based upon the TXD <3:0>, TX\_EN and TX\_ER signals on the MII. The subsequent functions of the PCS Transmit process then pack the resulting blocks plus one pad bit. This is passed as is in the low latency mode, or processed by a Reed-Solomon (RS-FEC) encoder in the long reach mode. It is then scrambled and subsequently 8B6T mapped into a transmit PHY frame of PAM3 symbols. Transmit data-units are sent to the PMA service interface via the PMA\_UNITDATA.request primitive. A symbol period, T, is 80 ns.

Editor’s Note: need text to discuss training and EEE – currently not baselined.

**ZZZ.3.2.2.1 Use of blocks**

Editor’s Note: need text to discuss training frame and block alignment – currently not baselined.

**ZZZ.3.2.2.2 Transmission code**

Editor’s Note: The following paragraph needs to be re-written since some of this applies to the low latency mode with the RS FEC.

In the long reach mode the PCS uses a transmission code to improve the transmission characteristics of information to be transferred across the link and to support transmission of control and data characters. The encoding defined by the transmission code ensures that sufficient information is present in the PHY bit stream to make clock recovery possible at the receiver. The encoding also preserves the likelihood of detecting any PHY frame errors that may occur during transmission and reception of information. In addition, the code enables the receiver to achieve PCS synchronization alignment on the incoming PHY bit stream.

Editor’s Note: There may be more than one diagram since there are 2 modes.

The relationship of block bit positions to GMII, PMA, and other PCS constructs is illustrated in Figure ZZZ–5 for transmit and Figure ZZZ–6 for receive. These figures illustrate the processing of a multiplicity of blocks containing N data octets. See 97.3.2.2.5 for information on how blocks containing control characters are mapped.

**ZZZ.3.2.2.3 Notation conventions**

For values shown as binary, the leftmost bit is the first transmitted bit.

**ZZZ.3.2.2.4 Transmission order**

The PCS Transmit bit ordering shall conform to Figure ZZZ–5 and Figure ZZZ–7.

Editor’s Note: Need to supply 3 diagrams. Maybe 2 additional ones are needed since we have 2 modes of operation.

**Figure ZZZ–5—PCS Transmit bit ordering**

**Figure ZZZ–6—PCS Receive bit ordering**

**Figure ZZZ–7—PCS detailed transmit bit ordering with RS-FEC**

**ZZZ.3.2.2.5 Block structure**

The encoding of data on the MII into a 8N/(8N+1) block is a two step process. The first step is the nibbles to octet symbol conversion. The second step packs the N octet symbols into an 8N+1 bit block.

Data presented on the MII are delineated as alternating even nibble and odd nibble. A state variable Nalign when set to even uses the even nibble as the first nibble and the odd nibble as the second nibble when converting to an octet symbol. Nalign when set to odd uses the odd nibble as the first nibble and the even nibble as the second nibble when converting to an octet symbol. The conversion of the first and second nibble into an octet symbol, and the conditions to set the Nalign to even or odd shall follow Table ZZZ-A.

Support of sequence ordered set is optional and is recognized only if both devices on the link advertise it during training. If unrecognized and the sequence nibble is presented on the MII, it shall be converted to an idle nibble. Note that the six nibbles that follows a sequence ordered set symbol would be recognized as idle nibbles since TX\_EN = 0 and TX\_ER = 0.

If sequence ordered set is recognized, then the 6 data nibbles with TX\_EN = 0 and TX\_ER = 0 are recognized as data symbols with TXD<3:0> presented as Data in Table ZZZ-A.

The skip cycle in Table ZZZ-A means no symbol octet is presented to the 8N/(8N+1) encoder.

When Nalign transitions from even to odd or odd to even, the second nibble of the first nibble to octet symbol conversion is used as the first symbol of the next octet symbol conversion.

Editor’s Note: Do we want to include the 2 notes below?

Note:

If the MII is not exposed and the MAC always outputs in byte increments and not in nibble increments, it is strongly recommended that the first nibble of the byte be placed on even nibble and the second nibble of the byte on the odd nibble.

Note:

If sequence ordered set support is not implemented, Nalign is always even.

**Table ZZZ-A – MII nibble to octet symbol mapping**



N octets symbols are grouped together and presented to the 8N/(8N+1) encoding process. Blocks consist of 17 bits (N = 2) for the low latency mode and 65 bits (N = 8) for the long reach mode. The first bit of a block is the data/ctrl header. Blocks are either data blocks or control blocks. The data/ctrl header is 0 for data blocks and 1 for control blocks. The remainder of the block contains the payload.

Data blocks contain N data octets. Control blocks begin with a 3-bit pointer field that indicates the location
of the first control code within the block.

If the first octet in the block is a control character, the pointer field is followed by a 2-bit Mode code and a 3-bit Control code. Otherwise, the pointer field is followed by one or more data octets until the position of the first Mode and Control code. The 2-bit Mode code and 3-bit Control code combined indicates type of control character. The encoding is shown in Table ZZZ-B.

If Mode[0] field maps to 1, it implies the next octet is a control symbol.

If Mode[0] field maps to 0, Mode[1] shall be set to 0 if the current octet is the final control symbol, otherwise it is set to 1.

If there are additional control symbols the current Control code is followed by a 3-bit pointer field to the next control symbol. The pointer field may be followed by a data octet or Mode and Control code depending on the value of the pointer field. In this way any combination of N data octets and
control characters may be encapsulated within an 8N/(8N+1) block.

**Table ZZZ-B Octet symbol to Mode and Control code mapping**



The N octets are mapped to 8N+1 bits as described in the following pseudo code, where N = 2 for low latency mode and N = 8 for long reach mode.

N = number of octets encoded into block.

Octets numbered n = 0,1,2,…,N–1.

octet 0 is the first one presented to the encoder.

TC[–1] = 1 by definition

if octet n is a data symbol then

 TC[n] = 0

else

 TC[n] = 1

NEXT(n)[0:2] = bit position of lowest bit in TC[n:N–1] that is a 1. Bit 2 is MSB.

NEXT(n)[4] = 0 if Bitwise SUM of TC[n:N–1] = 1, else 1

if TC[n] = 1 then

 TD[0:2] is undefined

 if octet n is one of the Tu\* symbols then

 TD[n][3:7] = {Mode[0:1], Control[0:2] of the corresponding control

 symbol as defined in Table ZZZ-B.

 else

 TD[n][4] = NEXT(n)[4]

 TD[n][3, 5:7] = {Mode[0], Control[0:2]} of the corresponding

 control symbol as defined in Table ZZZ-B.

else

 TD[n][0:7] = octet n {first nibble TXD[0:3], second nibble TXD[0:3]}

B[0:8N] is the 8N+1 block. Bit 0 transmitted first.

OR(n) = Bitwise OR of TC[n:N–1]

B[0] = OR(0)

B[8n+1:8n+3] = TD[n][0:2] – if OR(n) = 0

 NEXT(n)[0:2] – if OR(n) = 1 AND TC[n–1] = 1

 TD[n–1][5:7] – if OR(n) = 1 AND TC[n–1] = 0

B[8n+4:8n+8] = TD[n][3:7] – if OR(n) = 0

 TD[n][3:7] – if OR(n) = 1 AND TC[n] = 1

 TD[n][0:4] – if OR(n) = 1 AND TC[n] = 0

Editor’s Note:

I do not think an equivalent to clauses 97.3.2.2.6 is needed since this is concisely covered as a combination in Tables ZZZ-A and ZZZ-B.

**ZZZ.3.2.2.7 Idle**

Idle (Normal Inter-frame) control characters are transmitted when TX\_EN is not asserted and no other
supported control code is present at the MII. Idle characters may be added or deleted by the PCS to adapt
between clock rates. Idle characters shall not be added within a MAC frame

**ZZZ.3.2.2.8 LP\_IDLE**

The low power idle control characters (LP\_IDLEs) are transmitted when TX\_EN is not asserted, TX\_ER is
asserted, and TXD<3:0> = 0x1. A continuous stream of LPI control characters is used to maintain a link in
the LPI transmit mode. Idle control characters are used to transition from the LPI transmit mode to the
normal power mode. EEE compliant PHYs respond to the Assert Low Power Idle condition on the MII
using the procedure outlined in ZZZ.TBD. LP\_IDLE characters may be added or deleted by the PCS to adapt
between clock rates. LP\_IDLE characters shall not be added within a MAC frame.

Where the MII and PMA sublayer data rates are not synchronized, the transmit process needs to insert
LPI\_IDLEs, or delete LPI\_IDLEs to adapt between the rates.

If EEE is not supported, then LP\_IDLE shall be converted to IDLE.

**ZZZ.3.2.2.9 Error**

The Error control code is sent when TX\_ER and TX\_EN are both asserted. Error allows physical sublayers
such as the PCS to propagate received errors.

Editor’s Note:

Equivalent to clauses 97.3.2.2.10 to 97.3.2.2.15 is needed after details on the RS-FEC, scrambler, EEE, and PAM modulation are settled.

**ZZZ.3.2.3.3 Octet symbols to MII nibbles and valid and invalid blocks**

The received octets symbols on the 8N/(8N+1) block are unpacked and mapped according to Table ZZZ-C subject to the following exceptions below.

Editor’s Note: An undefined control code is not listed as a condition below since all bit permutations are defined.

An 8N/(8N+1) block is invalid if any of the following conditions exist:

1. The block contains an invalid pointer
2. The PHY frame containing this 64B/65B block is uncorrectable.

Editor’s Note: The following paragraphs needs further discussion on whether this is an acceptable error handling.

If Su or Sp is received then a packet has commenced and invalid blocks, or any control symbol that is not E, Tp, or Tu\* that is received shall be converted to error nibbles on the MII.

If Tu\* or Tp is received indicating an end of packet, then invalid blocks, or any control symbol that is E, Tp, Tu\*, or data that is not part of the sequence ordered set shall be converted to false carrier nibbles on the MII. If sequence ordered set is not recognized then the sequence symbol shall be converted to false carrier nibbles on the MII. Once false carrier nibbles are output on the MII it shall be output for all subsequent symbols received until either I, Ix, or L symbol is received.

If sequence ordered set is recognized, then the 3 data bytes following the sequence ordered set symbol shall be presented as nibbles on RXD<3:0> with RX\_DV = 0 and RX\_ER = 0.

**Table ZZZ-C Octet symbol to MII nibble mapping**

