Proposed Comment Resolution related to 127.2.6, 2.5GBASE-X PCS

Comment #: 335~354 (DL), 228~233 (BM), etc
Pages 62~ 91 of D2.0 PDF (61~92 extracted)

Edited by Yong Kim

Standalone Resolutions (1)

- 332 DL – Word encode/decode description improvements
- 356 YK – Ordered set TX for link status to be optional
- 281 CD –2.5 Gb/s, not 2.5Gb/s
- 333 DL - 2.5GPII symbol rates, etc.
- 334 DL & 219 BM – abbrev is longer. Also state table column
- 218 BM – wencode_state(n) and (n+1)
- 220 BM - ||Q|| , /W/
- 221 – BM – /W/ reference
- 282 – CD – Sequence → sequence
- 283, 286 – CD – 24-bit, 284- CD 2.5GMII → 2.5GPII (got deleted)
- 222 – BM - ||Q|| ||Fsig||
- 223 – BM – Idle → IDLE in Table 127-2 – Proposed Accept now, but should be REJECT due to #334
- 224 – BM – wdecode_state(n) (n+1)
- 225 – BM -- ??? Data* SOP etc. (editors note not relevant)
- 226 – BM – DATA to LPI not allowed. ; 227 – BM DATA to Sequence not allowed (pass through ERR 1st)
- 246 – EB – Link Status optional
- 228 – BM – "/W/" should be in a same line. 229 – BM - /PL_LIMIT/ is not a set

BROADCOM

Standalone Resolutions (2)

- 136 – DS – SIGNAL_DETECT – should be rejected (comment response says “proposed accept”).
- 335 DL – sync_status variable def correction.
- 336 DL – add tx_status to 2.5GPII interface in Fig 127-2 (and remedy seems to be wrong (refers to sync_status). So change to accept in principle.
- 337 – DL – Add values FAIL and OK definitions.
- 339 – DL – reference correction … running disparity.
- 231 – BM – editorial “SUDI…EEE….”
- 232 – BM – 287 – CD - editorial “NEXTSEQ()
- 233 – BM – 137 – DS - editorial “Signal_de..” to “signal_de…”
- 340 - DL – TX_CLK on XGMII
- 341 - DL – Editorial improvements on 127.2.6.1
- 342 – DL – tx_even to 127-2
- 343 – DL – PCS Word Encode SD
- 346 – DL – IF THEN ELSE (capitalize)
- 3 – ML – Editorial Improvements on Fig 127-5 PCS TX State Diagram

Standalone Resolutions (3)

- 135 – DS – Effect/affect ?
- 347 – DL – sync_status and code_sync_status
- 348 – DL – rx_lpi_active
- 349 & 350 – DL – LINK_FAILED state. Delete ; ; .; WAIT_FOR_K state Delete ; .
- 351 & 352 DL – rp-dv rp_dv
- 353 DL – Note 2 to be added
- 4, 5 – ML – Clean up lines, corners, and arrows in Fig 128-8b

Broad blush resolutions

- 338 DL – rename tp_en<3:0>, tp_er<3:0>, tpd<3:0><7:0> to we_tp_en<3:0>, we_tp_er<3:0>, we_tpd<31:0> and
  338’ - likewise to RX path.
  338” – additional editorials to to help to clarify.

b) Serialization (deserialization) of code-groups for transmission (reception) on the underlying serial PMD.
c) Recovery of clock from the 8B/10B-coded data supplied by the PMD.
d) Mapping of transmit and receive bits between the PMA and PMD via the PMD Service Interface.

127.1.3.3 Physical Medium Attachment (PMA) service interface rates

2.5GBASE-X Physical Layer specification has nominal rate at the PMA service interface of 3.125 Gb/s, which provides MAC data rate of 2.5 Gb/s.

127.1.4 Inter-sublayer interfaces

There are a number of interfaces employed by 2.5GBASE-X. Some (such as the PMA Service Interface) use an abstract service model to define the operation of the interface. An optional physical instantiation of the PCS Interface is the XGMII. Figure 127–2 depicts the relationship and mapping of the services provided by all of the interfaces relevant to 2.5GBASE-X.

While this specification defines interfaces in terms of bits, octets, and code-groups, implementers may choose other data path widths for implementation convenience. The only exceptions are a) the XGMII, which, when implemented at an observable interconnection port, uses an word-wide data path as specified in Clause 46.

127.1.5 Functional block diagram

Figure 127–2 provides a functional block diagram of the 2.5GBASE-X PHY.

127.1.6 State diagram conventions

The body of this standard is comprised of state diagrams, including the associated definitions of variables, constants, and functions. Should there be a discrepancy between a state diagram and descriptive text, the state diagram prevails.

The notation used in the state diagrams follows the conventions of 21.5. State diagram timers follow the conventions of 14.2.3.2.

127.2 Physical Coding Sublayer (PCS)

127.2.1 PCS Interface (XGMII)

The PCS Service Interface allows the 2.5GBASE-X PCS to transfer information to and from a PCS client. PCS clients include the MAC (via the Reconciliation sublayer). The PCS Interface is precisely defined as the 10 Gigabit Media Independent Interface (XGMII) in Clause 46.

127.2.2 Functions within the PCS

The PCS includes the Word Encode, Word-to-Octets, Transmit, Synchronization, Receive, Octets-to-Word, and Word Decode processes for 2.5GBASE-X. The PCS shields the RS (and MAC) from the specific nature of the underlying channel.

When communicating with the XGMII, the PCS uses in each direction, 32 data signals (TXD <31:0> and RXD <31:0>), four control signals (TXC <3:0> and RXC <3:0>), and a clock (TX_CLK and RX_CLK).

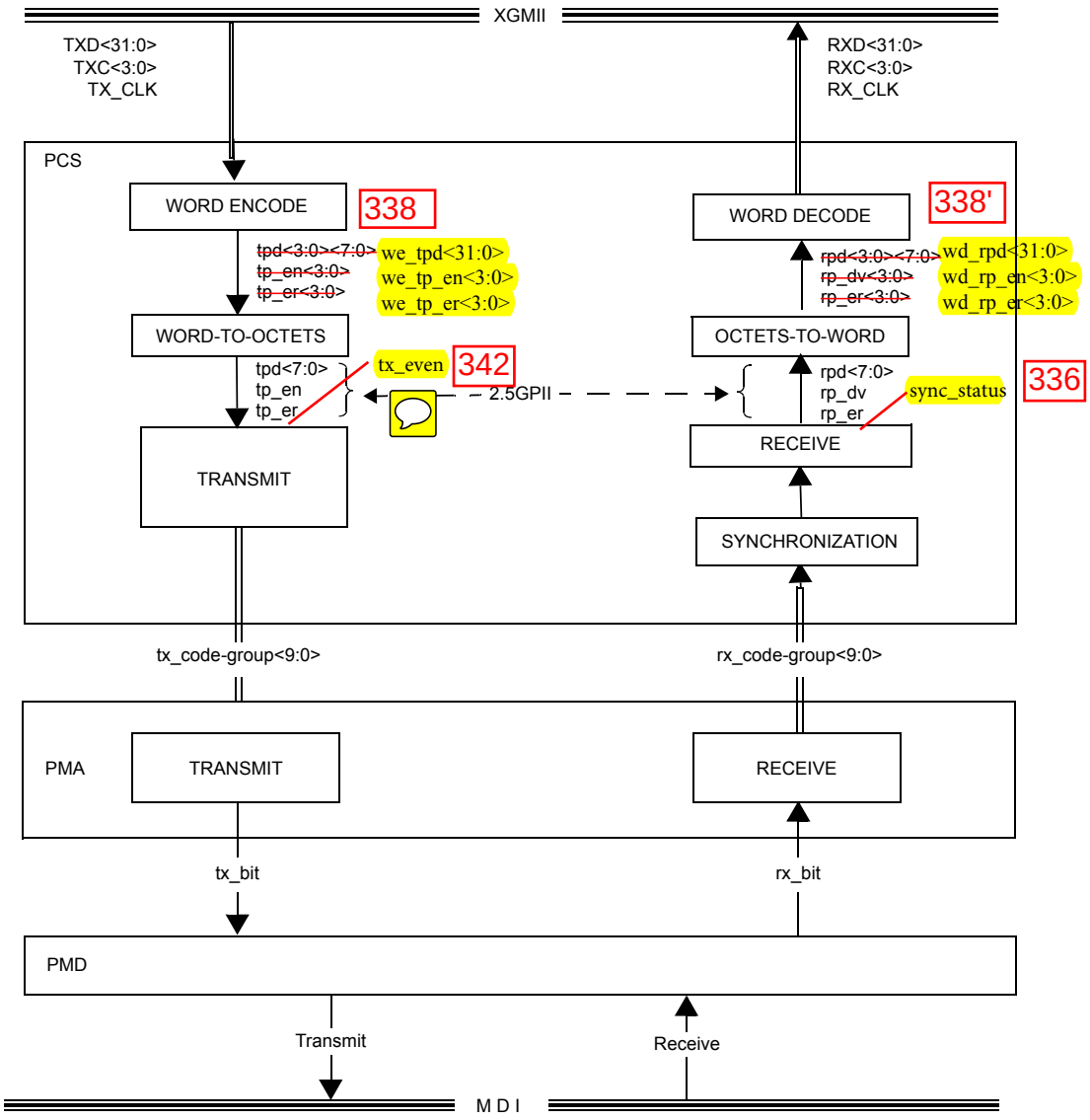


Figure 127–2—Functional block diagram

When communicating with the PMA, the PCS uses a ten-bit wide synchronous data path, tx_code-group<9:0> and rx_code-group<9:0>, which conveys ten-bit code-groups. At the PMA Service Interface, code-group alignment and MAC packet delimiting are made possible by embedding special non-data code-groups in the transmitted code-group stream. The PCS provides the functions necessary to map packets between the XGMII format and the PMA service interface format. 332

(tpd<3:0><7:0>), and associated four bits of transmit enable (tp_en<3:0>) and four bits of transmit error (tp_er<3:0>),

The Word Encode process continuously generates four 2.5GPII symbols based upon the TXD <31:0> and TXC <3:0> signals on the XGMII, sending them to the Word-to-Octets process.

The Word-to-Octets process takes the four 2.5GPII symbols ~~and outputs them one 2.5GPII symbol at a time to the PCS Transmit Process.~~ , and associated transmit enable and transmit error, and transmits one 2.5GPII symbol and its associated transmit enable and transmit error at a time to the PCS Transmit Process across the 2.5GPII.

332

The PCS Transmit process continuously generates code-groups based upon the tpd<7:0>, tp_en, and tp_er signals on the 2.5GPII, sending them immediately to the PMA Service Interface via the PMA_UNITDATA.request primitive.

The PCS Synchronization process continuously accepts code-groups via the PMA_UNITDATA.indication primitive and conveys received code-groups to the PCS Receive process via the SYNC_UNITDATA.indicate primitive. The PCS Synchronization process sets the sync_status flag to indicate whether the PMA is functioning dependably.

The PCS Receive process continuously accepts code-groups via the SYNC_UNITDATA.indicate primitive. The PCS Receive process monitors these code-groups and generates rpd<7:0>, rp_dv, and rp_er on the 2.5GPII.

The Octets-to-Word process queues the received 2.5GPII symbols and aligns them in groups of four 2.5GPII symbols. Symbols may be deleted or idle symbols added in order to do the alignment.

The Word Decode process continuously accepts the four 2.5GPII symbols from the Word Alignment process and generates RXD <31:0> and RXC <3:0> on the XGMII.

All PCS processes are described in detail in the state diagrams in 127.2.6.2.

127.2.3 Use of code-groups

The transmission code used by the PCS, referred to as 8B/10B, is identical to that specified in Clause 36. The PCS maps XGMII characters into an intermediate 2.5GPII which is then mapped to 10-bit code-groups, and vice versa, using the 8B/10B block coding scheme. Implicit in the definition of a code-group is an establishment of code-group boundaries by a PCS Synchronization process. The 8B/10B transmission code as well as the rules by which the PCS ENCODE and DECODE functions generate, manipulate, and interpret code-groups are specified in 127.2.5. The XGMII to 2.5GPII mapping and vice versa are specified in 127.2.4. Code-groups and the 2.5GPII are unobservable and have no meaning outside the PCS.

127.2.4 XGMII to 2.5GPII mapping

The Word Encode, Word-to-Octets, Octets-to-Word, and Word Decode processes together define the XGMII to 2.5GPII mapping. This mapping function formats and serializes/de-serializes the data between the four XGMII lanes and the single lane of the PCS transmit/receive process.

2.5 Gb/s 281

127.2.4.1 ~~2.5Gb/s~~ PCS Internal Interface (2.5GPII)

2.5 Gb/s 281

The ~~2.5Gb/s~~ PCS Internal Interface (2.5GPII) is a logical interface that is internal to the 2.5GBASE-X PCS and exists solely for the purposes of defining the 2.5GBASE-X PCS functionality. Physical implementation of the 2.5GPII is optional and is not exposed outside of the PCS.

338''

The 2.5GPII consists of the following variables: tp_en, tp_er, tpd<7:0>, rp_dv, rp_er, rpd<7:0> and its encoding is similar but not identical to the GMII in clause 35. The permissible encodings are shown in Table 127–1 and Table 127–2.

338''

A 2.5GPII symbol is defined to be a set of tp_en, tp_er, tpd<7:0> variables, rp_dv, rp_er, rpd<7:0> variables.

we_tp_en<3:0>, we_tp_er<3:0>, we_tpd<31:0>, wd_rp_dv<3:0>, wd_rp_er<3:0>, wd_rpd<31:0>

338'

The Word Encode and Word Decode processes maps between the four XGMII lanes to four 2.5GPII symbols. The ~~four~~ 2.5GPII symbols are indexed as ~~tp_en<3:0>, tp_er<3:0>, tpd<3:0><7:0>, rp_dv<3:0>, rp_er<3:0>, rpd<3:0><7:0>~~. The nominal rate of operation is ~~12.8ns +/- 0.01%.~~ 78.125 Msymbols/s +/- 100pm.

333

The Word-to-Octets and Octets-to-Word processes serializes/de-serializes four 2.5GPII symbols to/from four consecutive single 2.5GPII symbols. The nominal rate of operation of the single 2.5GPII ~~symbol is 3.2ns +/- 0.01%.~~ is 312.5 Msymbols/s +/- 100ppm. 333

Table 127–1—Permissible encodings of tpd<7:0>, tp_en, tp_er at 2.5GPII

tp_en	tp_er	tpd<7:0>	Description	~~Abbreviation~~ Mnemonics
0	0	0x00 to 0xFF	Normal Inter-frame	~~IDLE~~ Idle
0	1	0x00	Reserved	
0	1	0x01	Assert LPI	LPI
0	1	0x02 to 0x9B	Reserved	
0	1	0x9C	Sequence	Seq
0	1	0x9D to 0xFF	Reserved	
1	0	0x00 to 0xFF	Data	Data X
1	1	0x00 to 0xFF	Transmit Error	Err

334 334

Table 127–2—Permissible encodings of rpd<7:0>, rp_dv, rp_er at 2.5GPII

rp_dv	rp_er	rpd<7:0>	Description	~~Abbreviation~~ Mnemonics
0	0	0x00 to 0xFF	Normal Inter-frame	~~IDLE~~ Idle
0	1	0x00	Reserved	
0	1	0x01	Assert LPI	LPI
0	1	0x02 to 0x0D	Reserved	
0	1	0x0E	False carrier indication	FCI
0	1	0x0F	Carrier Extend (odd byte packets)	CE
0	1	0x10 to 0x9B	Reserved	
0	1	0x9C	Sequence	Seq
0	1	0x9D to 0xFF	Reserved	
1	0	0x00 to 0xFF	Data	Data X
1	1	0x00 to 0xFF	Receive Error	Err

334 334

127.2.4.2 Word Encode

The Word Encode process maps the four XGMII lanes onto four 2.5GPII symbols as shown in Table 127–3. The XGMII encoding is specified in Table 46-3. The 2.5GPII encoding is specified in Table 127–1. The mapping of the sequence ordered set is dependent on the current state of the wencode_state variable as shown in column 5. The state of wencode_state is updated once the mapping occurs per the last column.

2.5GPII and XGMII columns delimiter to be added

Table 127–3—Word Encode mapping

XGMII						2.5GPII				
Lane 0	**Lane 1**	**Lane 2**	**Lane 3**	**wencode _state(n)**		**2.5GPII<0>**	**2.5GPII<1>**	**2.5GPII<2>**	**2.5GPII<3>**	**wencode _state (n+1)**
Data A/Err	Data B/Err	Data C/Err	Data D/Err	X	=>	Data A/Err	Data B/Err	Data C/Err	Data D/Err	DATA
Idle	Idle	Idle	Idle	X	=>	Idle	Idle	Idle	Idle	IDLE
LPI	LPI	LPI	LPI	X	=>	LPI	LPI	LPI	LPI	DATA
SOP	Data A/Err	Data B/Err	Data C/Err	X	=>	0x55 Data	Data A/Err	Data B/Err	Data C/Err	DATA
Terminate	Idle	Idle	Idle	X	=>	Idle	Idle	Idle	Idle	IDLE
Data A/Err	Terminate	Idle	Idle	X	=>	Data A/Err	Idle	Idle	Idle	DATA
Data A/Err	Data B/Err	Terminate	Idle	X	=>	Data A/Err	Data B/Err	Idle	Idle	DATA
Data A/Err	Data B/Err	Data C/Err	Terminate	X	=>	Data A/Err	Data B/Err	Data C/Err	Idle	DATA
Sequence	Data X	Data Y	Data Z	IDLE	=>	Seq	Data S0	Seq	Data S1	SEQ
Sequence	Data X	Data Y	Data Z	SEQ	=>	Seq	Prev Data S2	Seq	Prev Data S3	IDLE
Sequence	Data X	Data Y	Data Z	DATA	=>	Idle	Idle	Idle	Idle	IDLE
else					=>	Err	Err	Err	Err	DATA

A sequence ordered set ~~|Q|~~ appears to the PMA as /K28.5/W/K28.5/W/K28.5/W/K28.5/W/. On the 2.5GPII this appears as Seq, Data S0, ~~Seq Data S1,~~ Seq, Data S2, Seq, Data S3. The same sequence ordered-set is assumed to be repeating over multiple XGMII cycles. Every other consecutive ~~Sequence~~ ordered-set on the XGMII is ignored. If a ~~Sequence~~ ordered-set occurs over odd number of cycles on the XGMII then the final one will be truncated as Seq, Data S0, Seq Data S1 and Seq, Data S2, Seq, Data S3 are not sent.

The ~~24-bit~~ Data X, Data Y, and Data Z from the sequence ordered set is mapped to Data S0, Data S1, Data S2, Data S3 as shown in Equation (127-1).

S0<7> = S3<7> = 0, S1<7> = S2<7> = 1
S0<5:0> = Data X<5:0>
S1<5:0> = Data Y<3:0>, Data X<7:6>
S2<5:0> = Data Z<1:0>, Data Y<7:4>
S3<5:0> = Data Z<7:2>
Sn<6> = Sn<7> if Sn<2> = 0
Sn<6> = Sn<5> if Sn<2> = 1 (127-1)

The signal ordered-set |Fsig| uses the same equation except S2<7> is set to 0.

Since sequence ordered-set can be sent back to back it is necessary to determine the boundaries of the ordered-set. {S0<7>, S1<7>, S2<7>, S3<7>} can be used to determine the boundary. {S0<7>, S1<7>} will always be 01, while the 01 combination will never occur over {S1<7>, S2<7>}, {S2<7>, S3<7>}, or {S3<7>, S0<7>}.

Only 128 combinations of Sn<7:0> are possible. When encoded to their 10-bit equivalent these 128 code-groups are defined to be in the set of ~~|W|~~ /W/ (see 127.2.6.1.2).

127.2.4.3 Word-to-Octets

The Word-to-Octets process takes the ~~four 2.5GPII symbols (tp_en<3:0>, tp_er<3:0>, tpd<3:0><7:0>) from by the~~ Word Encoder and presents one symbol at a time (~~tp_en, tp_er, tpd<7:0>~~) to the PCS transmit process. ~~Index 0~~ is presented first and ~~index 3~~ is presented last.

The Word-to-Octets process shall be synchronized to the PCS transmit process such that ~~the 2.5GPII index 0 and 2 symbols~~ are presented to the PCS transmit process which will result in the corresponding ordered set to be output to the PMA when the variable tx_even is TRUE and ~~index 1 and 3 variables when~~ tx_even is FALSE.

127.2.4.4 Octets-to-Word

The Octets-to-Word process de-serializes ~~a sequence of 2.5GPII symbols (rp_dv, rp_er, rpd<7:0>) from~~ the PCS receive process to four ~~2.5GPII~~ symbols (~~rp_dv<3:0>, rp_er<3:0>, rpd<3:0><7:0>~~). ~~Index 0~~ is the earliest to arrive and ~~index 3~~ is the ~~latest~~.

The Octets-to-Word process ~~will insert 2.5GPII~~ idle symbols, or ~~delete 2.5GPII~~ symbols from the sequence of ~~2.5GPII symbols~~ received to achieve the following conditions.

a) A transition of a ~~2.5GPII~~ idle symbol to a data or error symbol shall place the data or error symbol on ~~index 0.~~

b) A transition of a 2.5GPII idle symbol to a LPI symbol shall place the LPI symbol on either ~~index 0 or 2.~~

c) A transition of a 2.5GPII LPI symbol to an idle symbol shall place the idle symbol on either ~~index 0 or 2.~~

d) The start of ~~|Q| or |Fsig|~~ set shall always occur on ~~index 0~~. Clause 127.2.4.2 describes how the start of ~~|Q| and |Fsig|~~ can be determined.

The Octets-to-Word process maintains a Deficit Idle Count (DIC) that represent the cumulative count of ~~2.5GMII symbols~~ added or deleted from the sequence of received ~~2.5GPII~~ symbols. The DIC is incremented by one for each ~~2.5GPII~~ symbol deleted and decremented by one for each ~~2.5GPII~~ idle symbol added. The DIC shall be bounded to a minimum of 0 and a maximum of 3.

Note that in a properly behaved ~~system 2.5GPII symbol deletion~~ should only occur at most once at the beginning of link, and afterwards no further insertions or deletions are required. In order to interoperate with the application described in Annex 127B, additional symbol insertions and deletions may be required during normal operation.

The only symbol that may be inserted is a ~~2.5GPII~~ idle symbol. However any ~~2.5GPII~~ symbol may be deleted. Usually this will either be a ~~2.5GPII~~ idle or LPI symbols, though in pathological error conditions (i.e. unterminated packet followed immediately with sequence ordered-set) some other symbol may be deleted.

127.2.4.5 Word Decode

The Word Decode process maps the four 2.5GPII symbols onto the four XGMII lanes as shown in Table 127–4. The XGMII encoding is specified in Table 46-4. The 2.5GPII encoding is specified in Table 127–2. The mapping is dependent on the current state of the wdecode_state and next_seq_s2_s3 variables as shown in columns 5 and 6. The state of wdecode_state is updated once the mapping occurs per the last column.

The ~~24 bit~~ Data X, Data Y, and Data Z from the sequence ordered is reconstructed from Data S0, Data S1, Data S2, Data S3 according to Equation (127-2).

if (S0<7>, S1<7>, S2<7>, S3<7> = 0110) then output
XGMII = Sequence, Data X, Data Y, Data Z where
Data X<7:0> = S1<1:0>, S0<5:0>
Data Y<7:0> = S2<3:0>, S1<5:2>
Data Z<7:0> = S3<5:0>, S2<5:4>
else
XGMII = Idle, Idle, Idle, Idle (127-2)

2.5GPII and XGMII columns delimiter to be added

Table 127–4—Word Decode mapping

2.5GPII				334			XGMII				334
2.5GPII<0>	**2.5GPII<1>**	**2.5GPII<2>**	**2.5GPII<3>**	**wencode_state(n)** 224	**seq_s2s3**		**Lane 0**	**Lane 1**	**Lane 2**	**Lane 3**	**wencode_state(n+1)** 224
Data A/Err	Data B/Err	Data C/Err	Data D/Err	!IDLE	X	=>	Data A/Err	Data B/Err	Data C/Err	Data D/Err	DATA
Data *	Data A/Err	Data B/Err	Data C/Err	IDLE	X	=>	SOP	Data A/Err	Data B/Err	Data C/Err	DATA
Idle	Idle	Idle	Idle	!DATA	X	=>	Idle	Idle	Idle	Idle	IDLE
Idle	Idle	Idle	Idle	DATA	X	=>	Terminate	Idle	Idle	Idle	IDLE
Data A/Err	Idle or CE	Idle	Idle	DATA	X	=>	Data A/Err	Terminate	Idle	Idle	IDLE
Data A/Err	Data B/Err	Idle	Idle	DATA	X	=>	Data A/Err	Data B/Err	Terminate	Idle	IDLE
Data A/Err	Data B/Err	Data C/Err	Idle or CE	DATA	X	=>	Data A/Err	Data B/Err	Data C/Err	Terminate	IDLE
LPI	LPI	LPI	LPI	~~X~~ !IDLE	X	=>	LPI	LPI	LPI	LPI	IDLE
Idle	Idle	LPI	LPI	X 226	X	=>	LPI	LPI	LPI	LPI	IDLE
LPI	LPI	Idle	Idle	~~X~~ !IDLE 226	X	=>	Idle	Idle	Idle	Idle	IDLE
Seq	Data S0	Seq	Data S1	~~X~~ !DATA 227	TRUE	=>	Sequence	Data X	Data Y	Data Z	SEQ
Seq	Data S0	Seq	Data S1	~~X~~ !DATA 227	FALSE	=>	Idle	Idle	Idle	Idle	IDLE
Seq	Data S2	Seq	Data S3	SEQ	X	=>	Sequence	Data X	Data Y	Data Z	IDLE
else						=>	Err	Err	Err	Err	ERR

Note*: Data that corresponds to Start of Packet (SOP) is only placed on 2.5GPII<0> when XGMII is implemented.

255

127.2.5 8B/10B transmission code

The transmission code used by the PCS, referred to as 8B/10B, is identical to that specified in Clause 36. In addition to the requirements in this clause, a 2.5GBASE-X PCS shall also meet the 8B/10B transmission code requirements specified in 36.2.4.1 through 36.2.4.6, 36.2.4.8, and 36.2.4.9. The relationship of code-group bit positions to PMA and other PCS constructs is illustrated in Figure 127–3.

127.2.5.1 Notation conventions

The 8B/10B transmission code uses letter notation for describing the bits of an unencoded information octet and a single control variable according to 36.2.4.1.

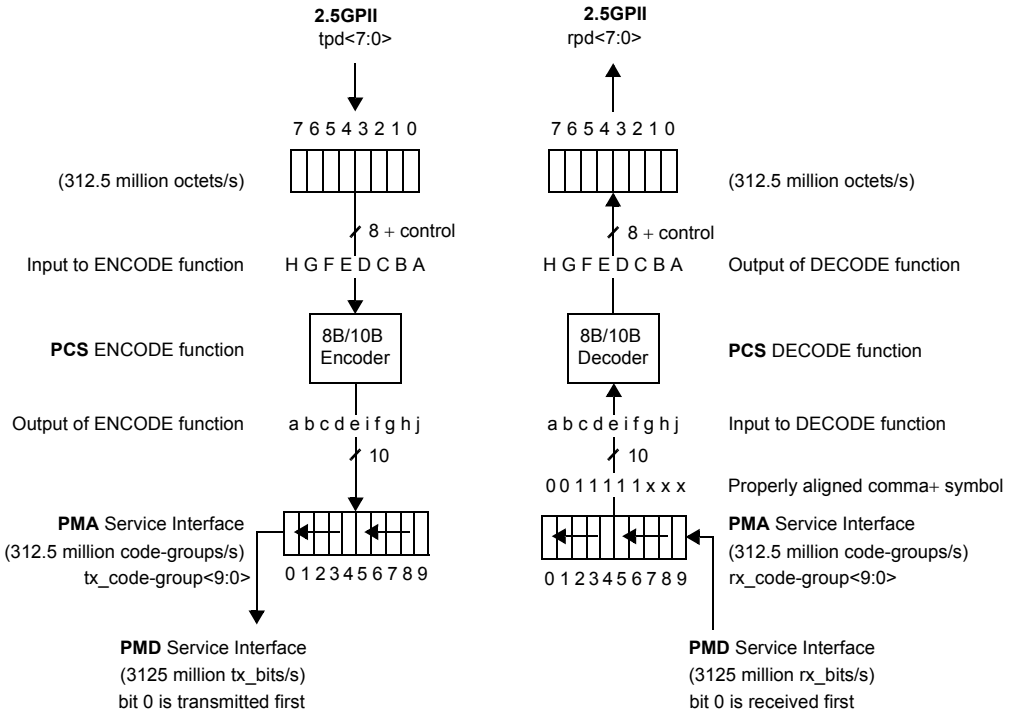


Figure 127–3—PCS 8B/10B reference diagram

127.2.5.2 Transmission order

Code-group bit transmission order is illustrated in Figure 127–3 and defined in 36.2.4.2.

127.2.5.3 Generating code-groups and checking the validity of received code

Valid code-groups are defined in 36.2.4.3.

The running disparity rules are defined in 36.2.4.4.

The code-group generation is defined in 36.2.4.5.

The check on the validity of received code-groups is defined in 36.2.4.6.

127.2.5.4 Ordered sets

Table 127–5 lists the defined ordered_sets, consisting of a single special code-group or combinations of special and data code-groups. Ordered sets which include /K28.5/ provide the ability to obtain bit and code-group synchronization and establish ordered set alignment (see 36.2.4.9 and 127.3.2.4). Ordered sets provide for the delineation of a packet and synchronization between the transmitter and receiver circuits at opposite ends of a link. Certain PHYs include an option (see 78.3) to transmit or receive /LI/, /LI1/ and /LI2/ to support Energy-Efficient Ethernet (see Clause 78).

Ordered sets are specified according to the following rules:

a) Ordered sets consist of either one, two, or four code-groups.
b) The first code-group of all ordered sets is always a special code-group.
c) The second code-group of all multi-code-group ordered sets is always a data code-group. The second code-group is used to distinguish the ordered set from all other ordered sets.

Table 127–5—Defined ordered sets

Code	Ordered Set	Number of Code-Groups	Encoding
/I/	**IDLE**		Correcting /I1/, Preserving /I2/
/I1/	IDLE 1	2	/K28.5/D5.6/
/I2/	IDLE 2	2	/K28.5/D16.2/
	Encapsulation		
/R/	Carrier_Extend	1	/K23.7/
/S/	Start_of_Packet	1	/K27.7/
/T/	End_of_Packet	1	/K29.7/
/V/	Error_Propagation	1	/K30.7/
/LI/	**LPI**		Correcting /LI1/, Preserving /LI2/
/LI1/	LPI 1	2	/K28.5/D6.5/
/LI2/	LPI 2	2	/K28.5/D26.4/
	Link Status		
/Q/	Sequence ordered_set	8	/K28.5/W0/K28.5/W1/K28.5/W2/ K28.5/W3[a]
	Reserved		
/Fsig/	Signal ordered_set	8	/K28.5/W0/K28.5/W1/K28.5/W2/ K28.5/W3[a]

[a] /W0/, /W1/, /W2/, /W3/ are the 10-bit /Dx.y/ version of S0, S1, S2, S3 as defined per clause 127.2.4.2 and will never have a value of /D5.6/, /D16.2/, /D6.5/, /D26.4/.

127.2.5.5 Comma considerations

The comma considerations are described in 36.2.4.8 and 36.2.4.9.

127.2.5.6 Sequence (/Q/)

246 various optional ~~link status~~ 286 24-bit

A sequence ordered_set is used to convey ~~various link status~~ such as local fault or remote fault. The ~~24 bit~~ data of the sequence ordered_set on the ~~XGMII are mapped~~ to S0, S1, S2, S3 (see 127.2.4.2), and /W0/, ~~/~~ /W1/ ~~W1/~~, /W2/, /W3/ are the 8B/10B mapped version. XGMII, when implemented, are mapped 246 228

S0, S1, S2, S3 are constructed such that each /Wn/ can be mapped to only 128 out of 256 possible /Dx.y/ code-groups. /W/ is defined to be the set of 128 /Dx.y/ code-groups that can appear in a sequence ordered_set. /W/ does not contain /D5.6/, /D16.2/, /D6.5/, /D26.4/.

It is possible for the transmitter to send out a truncated sequence ordered_set that appears as /K28.5/W0/ K28.5/W1/. When a truncated sequence ordered_set is received, the Word Decode process will convert it to idles.

127.2.5.7 Data (/D/)

A data code-group, when not used to distinguish or convey information for a defined ordered set, conveys one octet of arbitrary data between the XGMII and the PCS. The sequence of data code-groups is arbitrary, where any data code-group can be followed by any other data code-group. Data code-groups are coded and decoded but not interpreted by the PCS. Successful decoding of the data code-groups depends on proper receipt of the Start_of_Packet delimiter, as defined in 127.2.5.10 and the checking of validity, as defined in 36.2.4.6.

127.2.5.8 IDLE (/I/)

IDLE ordered sets (/I/) are transmitted continuously and repetitively whenever the XGMII is idle. /I/ provides a continuous fill pattern to establish and maintain clock synchronization. /I/ is emitted from, and interpreted by, the PCS. /I/ consists of one or more consecutively transmitted /I1/ or /I2/ ordered sets, as defined in Table 127–5.

The /I1/ ordered set is defined such that the running disparity at the end of the transmitted /I1/ is opposite that of the beginning running disparity. The /I2/ ordered set is defined such that the running disparity at the end of the transmitted /I2/ is the same as the beginning running disparity. The first /I/ following a packet or a sequence order set restores the current positive or negative running disparity to a negative value. All subsequent /I/s are /I2/ to ensure negative ending running disparity.

Distinct carrier events are separated by /I/s.

A received ordered set that consists of two code-groups, the first of which is /K28.5/ and the second of which is a data code-group other than any of the 128 possible /W/, /D21.5/, or /D2.2/ (or /D6.5/ or /D26.4/ to support EEE capability), is treated as an /I/ ordered set.

127.2.5.9 Low Power Idle (/LI/)

LPI is transmitted in the same manner as IDLE. LPI ordered sets (/LI/) are transmitted continuously and repetitively whenever the XGMII is indicating “Assert LPI”.

127.2.5.10 Start_of_Packet (SPD) delimiter

A Start_of_Packet delimiter (SPD) is used to delineate the starting boundary of a data transmission sequence. Upon initiation of packet transmission, the PCS replaces the first octet of the MAC preamble with SPD. Upon initiation of packet reception, the PCS replaces the received SPD delimiter with the data octet value associated with the first preamble octet. A SPD delimiter consists of the code-group /S/, as defined in Table 127–5.

127.2.5.11 End_of_Packet delimiter (EPD)

An End_of_Packet delimiter (EPD) is used to delineate the ending boundary of a packet. The EPD is transmitted by the PCS following the last data octet comprising the FCS of the MAC packet. On reception, EPD is interpreted by the PCS as terminating a packet. A EPD consists of the code-groups /T/R/. If /R/ is transmitted in an even-numbered code-group position, the PCS appends a single additional /R/ to the code-group stream to ensure that the subsequent /I/ is aligned on an even numbered code-group boundary. An /I/ always follows the conclusion of EPD. The /T/R/ or /T/R/R/ occupies part of the region considered by the MAC to be the IPG. The code-group /T/ and /R/ are defined in Table 127–5.

127.2.5.12 Error_Propagation (/V/)

Error_Propagation (/V/) indicates that the PCS client wishes to indicate a transmission error to its peer entity. /V/ is emitted from the PCS when the XGMII indicates transmit error propagation, or when the XGMII presents a mapping that is undefined in Table 127–3 to the Word Encode process. The code group /V/ is defined in Table 127–5.

The presence of Error_Propagation or any invalid code-group on the medium denotes an error condition. Invalid code-groups are not intentionally transmitted onto the medium.

127.2.6 Detailed functions and state diagrams

The body of this subclause is comprised of state diagrams, including the associated definitions of variables, constants, and functions. Should there be a discrepancy between a state diagram and descriptive text, the state diagram prevails. The notation used in the state diagrams follows the conventions of 21.5. State diagram timers follow the conventions of 14.2.3.2.

127.2.6.1 State variables

127.2.6.1.1 Notation conventions

/x/
: Denotes the constant code-group specified in 127.2.6.1.2 (valid code-groups must follow the rules of running disparity as per 127.2.5.3).

[/x/]
: Denotes the latched received value of the constant code-group (/x/) specified in 127.2.6.1.2 and conveyed by the SYNC_UNITDATA.indicate message described in 127.2.6.1.6.

127.2.6.1.2 Constants

/COMMA/
: The set of special code-groups which include a comma as specified in 36.2.4.9 and listed in Table 36-2.

/D/
: The set of 256 code-groups corresponding to valid data, as specified in 127.2.5.7.

/Dx.y/
: One of the set of 256 code-groups corresponding to valid data, as specified in 127.2.5.7.

/I/
: The IDLE ordered set group, comprising either the /I1/ or /I2/ ordered sets, as specified in 127.2.5.8.

/INVALID/
: The set of invalid data or special code-groups, as specified in 36.2.4.6.

/Kx.y/
: One of the set of 12 code-groups corresponding to valid special code-groups, as specified in Table 36-2.

/Q/
: Sequence ordered set as specified in 127.2.5.6. A properly formed sequence order set appears as /K28.5/W/K28.5/W/K28.5/W/K28.5/W/. A truncated sequence order set appears as /K28.5/W/K28.5/W/.

/R/

The code-group used as either: End_of_Packet delimiter part 2; End_of_Packet delimiter part 3.

/S/

The code-group corresponding to the Start_of_Packet delimiter (SPD) as specified in 127.2.5.10.

/T/

The code-group used for the End_of_Packet delimiter part 1.

/V/

The Error_Propagation code-group, as specified in 127.2.5.12.

/W/

The set of 128 code-groups that is generated by ENCODE(s<7:0>) where for all 128 possible values of x<6:0>
s<7> = x<6>, s<5:0> = x<5:0>,
s<6> is set to x<5> when x<2> = 1; and s<6> is set to x<6> when x<2> = 0
NOTE— /W/ is a subset of /D/.

~~/PL_LIMIT/~~ PL_LIMIT 229

The number of 2.5GPII symbols to preload in the Octets-to-Word process after release from the RESET state. The number of symbols to preload is implementation dependent and should be minimized to reduce latency. The minimum number must be 3 to account for the deficit idle counting in 127.2.4.4.

The following constant is used only for the EEE capability:

/LI/

The LP_IDLE ordered set group, comprising either the /LI1/ or /LI2/ ordered sets, as specified in 127.2.5.9.

127.2.6.1.3 Variables

assert_seq

Alias used for sequence ordered set, consisting of the following terms:
tp_en=0 * tp_er=1 * (tpd<7:0>=0x9C)

cgbad

Alias for the following terms: ((rx_code-group∈/INVALID/) + (rx_code-group=/COMMA/ *rx_even=TRUE)) * PMA_UNITDATA.indication

cggood

Alias for the following terms: !((rx_code-group∈/INVALID/) + (rx_code-group=/COMMA/ *rx_even=TRUE)) * PMA_UNITDATA.indication

EVEN

The latched state of the rx_even variable, when rx_even=TRUE, as conveyed by the SYNC_UNITDATA.indicate message described in 127.2.6.1.6.

mr_loopback

A Boolean that indicates the enabling and disabling of data being looped back through the PHY. Loopback of data through the PHY is enabled when Control register bit 3.0.14 is set to one.

Values: FALSE; Loopback through the PHY is disabled.
TRUE; Loopback through the PHY is enabled.

mr_main_reset

Controls the resetting of the PCS via Control Register bit 3.0.15.

Values: FALSE; Do not reset the PCS.
TRUE; Reset the PCS.

ODD

The latched state of the rx_even variable, when rx_even=FALSE, as conveyed by the SYNC_UNITDATA.indicate message described in 127.2.6.1.6.

power_on

Condition that is true until such time as the power supply for the device that contains the PCS has reached the operating region. The condition is also true when the device has low power mode set via Control register bit 3.0.11.

Values: FALSE; The device is completely powered (default).
TRUE; The device has not been completely powered.

NOTE—Power_on evaluates to its default value in each state where it is not explicitly set.

rpd<7:0>

Single lane 2.5GPII receive data from the PCS receive process.

338'

~~rpd<x><7:0>~~ wd_rpd<31:0>

Receive ~~2.5GPII receive~~ data from the Octets-to-Word process. ~~x= 0, 1, 2, 3 for the four sets of 2.5GPII~~.

rp_dv

Single lane 2.5GPII receive data valid from the PCS receive process.

Values: 0 or 1.

~~rp_dv<x>~~ wd_rp_dv<3:0>

338'

Receive ~~2.5GPII receive~~ data valid from the Octets-to-Word process. ~~x= 0, 1, 2, 3 for the four sets of 2.5GPII.~~

Values: 0 or 1.

rp_er

Single lane 2.5GPII receive error from the PCS receive process.

Values: 0 or 1.

338'

~~rp_er<x>~~ wd_rp_er<3:0>

Receive ~~2.5GPII receive~~ error from the Octets-to-Word process. ~~x= 0, 1, 2, 3 for the four sets of 2.5GPII.~~

Values: 0 or 1.

rx_bit

A binary parameter conveyed by the PMD_UNITDATA.indication service primitive, as specified in 128.2.2, to the PMA.

Values: ZERO; Data bit is a logical zero.
ONE; Data bit is a logical one.

rx_code-group<9:0>

A 10-bit vector represented by the most recently received code-group from the PMA. The element rx_code-group<0> is the least recently received (oldest) rx_bit; rx_code-group<9> is the most recently received rx_bit (newest). When code-group alignment has been achieved, this vector contains precisely one code-group.

rx_even

A Boolean set by the PCS Synchronization process to designate received code-groups as either even- or odd-numbered code-groups as specified in 127.2.5.2.

Values: TRUE; Even-numbered code-group being received.
FALSE; Odd-numbered code-group being received.

RXC<3:0>

The RXC<3:0> signal of the XGMII as specified in Clause 46. Set by the Word Decode process.

RXD<31:0>

The RXD<31:0> signal of the XGMII as specified in Clause 46. Set by the Word Decode process.

signal_detect

A Boolean set by the PMD continuously via the PMD_SIGNAL.indication(signal_detect) message to indicate the status of the incoming link signal.

Values: FAIL; A signal is not present on the link.
OK; A signal is present on the link.

sync_status

335 ~~A parameter set by the PCS Synchronization process to reflect the status of the link as viewed by the receiver. The values of the parameter are defined for code_sync_status. The equation for this parameter is~~ Alias used by the PCS receive state diagram, consisting of the following terms:

sync_status = code_sync_status + rx_lpi_active.

337 Values: FAIL; A signal is not present on the link.
OK; A signal is present on the link

NOTE—If EEE is not supported, the variable rx_lpi_active is always false, and this variable is identical to code_sync_status controlled by the synchronization state diagram.

tpd<7:0>

Single lane 2.5GPII transmit data to the PCS transmit process.

~~tpd<x><7:0>~~ we_tpd<31:0>

338 ~~2.5GPII transmit data to the Word-to-Octets process. x= 0, 1, 2, 3 for the four sets of 2.5GPII~~. Transmit data output of the WORD ENCODE process.

tp_en

Single lane 2.5GPII transmit data enable to the PCS transmit process.

Values: 0 or 1.

~~tp_en<x>~~ we_tp_en<3:0>

338 ~~2.5GPII transmit data valid to the Word-to-Octets process. x= 0, 1, 2, 3 for the four sets of 2.5GPII.~~ Transmit data valid output of the WORD ENCODE process.

Values: 0 or 1.

tp_er

Single lane 2.5GPII transmit error to the PCS transmit process.

Values: 0 or 1.

~~tp_er<x>~~ we_tp_er<3:0>

338 ~~2.5GPII transmit error to the Word-to-Octets process. x= 0, 1, 2, 3 for the four sets of 2.5GPII~~. Transmit error output of the WORD ENCODE process.

Values: 0 or 1.

tx_bit

A binary parameter used to convey data from the PMA to the PMD via the PMD_UNITDATA.request service primitive as specified in 128.2.1.

Values: ZERO; Data bit is a logical zero.
ONE; Data bit is a logical one.

tx_code-group<9:0>

A vector of bits representing one code-group, as specified in Tables 36-1a through 36-1e, or 36-2, which has been prepared for transmission by the PCS Transmit process. This vector is conveyed to the PMA as the parameter of a PMD_UNITDATA.request(tx_bit) service primitive. The element tx_code-group<0> is the first tx_bit transmitted; tx_code-group<9> is the last tx_bit transmitted.

tx_disparity

A Boolean set by the PCS Transmit process to indicate the running disparity at the end of code-group transmission as a binary value. Running disparity is described in ~~36.2.4.3.~~ 36.2.4.4 339

Values: POSITIVE
NEGATIVE

tx_even

A Boolean set by the PCS Transmit process to designate transmitted code-groups as either even or odd numbered code-groups as specified in 127.2.5.2.

Values: TRUE; Even-numbered code-group being transmitted.
FALSE; Odd-numbered code-group being transmitted.

tx_o_set

One of the following defined ordered sets: /T/, /R/, /I/, /S/, /V/, /LI/, /K28.5/, or the code-group /D/.

TXC<3:0>

The TXC<3:0> signal of the XGMII as specified in Clause 46.

TXD<31:0>

The TXD<31:0> signal of the XGMII as specified in Clause 46.

wdecode_state

Word Decoder State used by the Word Decode process to properly assemble the next XGMII value to output.

Values: DATA;
IDLE;
SEQ;

wencode_state

Word Encoder State used by the Word Encode process to properly assemble the Sequence ordered-set.

Values: DATA;
IDLE;
SEQ;

The following variables are used only for the EEE capability:

assert_lpidle

Alias used for the optional LPI function, consisting of the following terms:

(tp_en=0 * tp_er=1 * (tpd<7:0>=0x01))

code_sync_status

A parameter set by the PCS Synchronization process to reflect the status of the link as viewed by the receiver.

Values: FAIL; The receiver is not synchronized to code-group boundaries.
OK; The receiver is synchronized to code-group boundaries.

idle_d

Alias for the following terms:

SUDI(![/D21.5/] * ![/D2.2/]) when EEE is not supported, or 231
~~that uses an alternate form to support the EEE capability:~~
SUDI(![/D21.5/] * ![/D2.2/] * ![/D6.5/] * ![/D26.4/]) when EEE is supported.

rx_lpi_active

A Boolean variable that is set to TRUE when the receiver is in a low power state and set to FALSE when it is in an active state and capable of receiving data.

rx_quiet

A Boolean variable set to TRUE while in the RX_QUIET state and set to FALSE otherwise.

tx_quiet

A Boolean variable set to TRUE when the transmitter is in the TX_QUIET state and set to FALSE otherwise. When set to TRUE, the PMD will disable the transmitter as described in 128.6.5.

127.2.6.1.4 Functions

carrier_detect

In the PCS Receive process, this function uses for input the latched code-group ([/x/]) and latched rx_even (EVEN/ODD) parameters of the SYNC_UNITDATA.indicate message from the PCS Synchronization process. When SYNC_UNITDATA.indicate message indicates EVEN, the carrier_detect function detects carrier when either:

a) A two or more bit difference between [/x/] and both /K28.5/ encodings exists (see Table 36-2); or

b) A two to nine bit difference between [/x/] and the expected /K28.5/ (based on current running disparity) exists.

Values: TRUE; Carrier is detected.
FALSE; Carrier is not detected.

check_end

Prescient End_of_Packet function used by the PCS Receive process. The check_end function returns the current and next two code-groups in rx_code-group<9:0>.

DECODE ([/x]/)

In the PCS Receive process, this function takes as its argument the latched value of rx_code-group<9:0> ([/x/]) and the current running disparity, and returns the corresponding 2.5GPII rpd<7:0>, as per Table 36-1a–e. DECODE also updates the current running disparity per the running disparity rules outlined in 36.2.4.4.

ENCODE(x)

In the PCS Transmit process, this function takes as its argument (x), where x is a 2.5GPII tpd<7:0> octet, and the current running disparity, and returns the corresponding ten-bit code-group as per Table 36-1a–e. ENCODE also updates the current running disparity as per Table 36-1a–e.

NEXTSEQ~~()~~ 232

Prescient function used by the Word Decode process that returns whether the next four 2.5GPII symbols presented to the Word Decode process is of the form Sequence, Data, Sequence, Data.

Values: TRUE; Next four 2.5GPII symbols are Sequence, Data, Sequence, Data.
FALSE; Next four 2.5GPII symbols are not Sequence, Data, Sequence, Data.

signal_detectCHANGE

In the PCS Synchronization process, this function monitors the signal_detect variable for a state change. The function is set upon state change detection.

Values: TRUE; A signal_detect variable state change has been detected.
FALSE; A signal_detect variable state change has not been detected (default).

233 signal_detect...

NOTE—~~Signal_detect~~CHANGE is set by this function definition; it is not set explicitly in the state diagrams. ~~Signal_detect~~CHANGE evaluates to its default value upon state entry.
signal_detect...

SINSERT(x)

Add a single 2.5GPII symbol to the end of a queue that stores the 2.5GPII symbol presented by the receive process. The variable x is the 2.5GPII symbol (rp_dv, rp_er, rpd<7:0>). If x is a null set then all content in the queue is emptied. The depth of the queue is implementation dependent.

VOID(x)

x ∈ /D/, /T/, /R/, /K28.5/. Substitutes /V/ on a per code-group basis as requested by the 2.5GPII.

If [tp_en=0 * tp_er=1 * tpd<7:0>≠(0000 1111)],

then return /V/;

Else if [tp_en=1 * tp_er=1],

then return /V/;

Else return x.

338 we_rp_dv<3:0>, we_rp_er<3:0>, we_rpd<31:0>

WALIGN~~()~~ 232

In the PCS Octets-to-Word process, this function performs the alignment according to 127.2.4.4. Four 2.5GPII symbols (~~rp_dv<3:0>, rp_er<3:0>, rpd<3:0><7:0>~~) are returned by this function. The SINSERT(x) function adds 2.5GPII symbols to the queue. The WALIGN~~()~~ functions removes one to seven 2.5GPII symbols from the front of the queue every time it is called 232

When no 2.5GPII symbols are inserted or deleted, this function will return the first four 2.5GPII symbols in the queue and remove them from the queue.

When X 2.5GPII idles symbols are inserted then X 2.5GPII idles symbols and the first 4 – X 2.5GPII symbols are returned by the function, and the first 4 - X 2.5GPII symbols are removed

from the queue.

When X 2.5GPII symbols are deleted then the first X 2.5GPII symbols are removed from the queue and the next four in the queue are returned by the function and then removed from the queue.

WDECODE(x, y, z)

wd_rp_dv<3:0>, wd_rp_er<3:0>, wd_rpd<31:0>

338'

In the PCS Word Decode process, this function performs the mapping according to 127.2.4.5. The variable x is four sets of 2.5GPII variables ~~rp_dv<3:0>, rp_er<3:0>, rpd<3:0><7:0>,~~ the variable y is the current state of the wdecode_state variable, and the variable z indicates whether the next four 2.5GPII variables are the final four 2.5GPII symbols of the |Q| or |Fsig| ordered-set.

The output is XGMII RXC<3:0>, RXD<31:0>, wdecode_state.

WENCODE(x, y)

338'

In the PCS Word Encode process, this function performs the mapping according to clause 127.2.4.2. The variable x is the XGMII TXC<3:0>, TXD<31:0>, and the variable y is the current state of the wencode_state variable.

The output is four sets of 2.5GPII variables and the updated state of the wencode_state variable and is ordered as follows: ~~tp_en<3:0>, tp_er<3:0>, tpd<3:0><7:0>,~~ wencode_state.

we_tp_en<3:0>, we_tp_er<3:0>, we_tpd<31:0>

127.2.6.1.5 Counters

good_cgs

Count of consecutive valid code-groups received.

plcnt

Count of number the number of 2.5GPII symbols to preload in the Octets-to-Word process after release from the RESET state. The number of symbols to preload is implementation dependent and should be minimized to reduce latency.

The following counter is used only for the EEE capability:

wake_error_counter

A counter that is incremented each time that the LPI receive state diagram enters the RX_WTF state indicating that a wake time fault has been detected. The counter is reflected in register 3.22 (see 45.2.3.10).

127.2.6.1.6 Messages

PMA_UNITDATA.indication(rx_code-group<9:0>)

A signal sent by the PMA Receive process conveying the next code-group received over the medium (see 127.3.1.2).

PMA_UNITDATA.request(tx_code-group<9:0>)

A signal sent to the PMA Transmit process conveying the next code-group ready for transmission over the medium (see 127.3.1.1).

PMD_SIGNAL.indication(signal_detect)

A signal sent by the PMD to indicate the status of the signal being received on the MDI.

PUDI

Alias for PMA_UNITDATA.indication(rx_code-group<9:0>).

PUDR

Alias for PMA_UNITDATA.request(tx_code-group<9:0>).

SUDI

Alias for SYNC_UNITDATA.indicate(parameters).

SYNC_UNITDATA.indicate(parameters)

A signal sent by the PCS Synchronization process to the PCS Receive process conveying the following parameters:

Parameters: [/x/]; the latched value of the indicated code-group (/x/);
EVEN/ODD; the latched state of the rx_even variable;

Value: EVEN; passed when the latched state of rx_even=TRUE.
ODD; passed when the latched state of rx_even=FALSE.

TX_OSET.indicate

A signal sent to the PCS Transmit ordered set process from the PCS Transmit code-group process signifying the completion of transmission of one ordered set.

The following messages are used only for the EEE capability:

PMD_RXQUIET.request(rx_quiet)

A signal sent by the PCS/PMA LPI receive state diagram to the PMD. This message is ignored by devices that do not support EEE capability.

Values: TRUE: The receiver is in a quiet state and is not expecting incoming data.
FALSE: The receiver is ready to receive data.

PMD_TXQUIET.request(tx_quiet)

A signal sent by the PCS/PMA LPI transmit state diagram to the PMD. This message is ignored by devices that do not support the optional LPI mechanism.

Values: TRUE: The transmitter is in a quiet state and may cease to transmit a signal on the medium.
FALSE: The transmitter is ready to transmit data.

127.2.6.1.7 Timers

cg_timer

340

A continuous free-running timer. If XGMII is implemented, cg_timer shall expire synchronously with the rising and falling edges of TX_CLK (see tolerance required for TX_CLK in 46.3.1.1). In the absence of XGMII, cg_timer shall expire every 3.2 ns ± 100ppm.

Values: The condition cg_timer_done becomes true upon timer expiration.

Restart when: immediately after expiration; restarting the timer resets the condition cg_timer_done.

Duration: 3.2 ns nominal.

The following timers are used only for the EEE capability:

rx_tq_timer

This timer is started when the PCS receiver enters the START_TQ_TIMER state. The timer terminal count is set to T_{QR}. When the timer reaches terminal count, it will set the rx_tq_timer_done = TRUE.

rx_tw_timer

This timer is started when the PCS receiver enters the RX_WAKE state. The timer terminal count shall not exceed the maximum value of T_{WR} in Table 127–7. When the timer reaches terminal count, it will set the rx_tw_timer_done = TRUE.

rx_wf_timer
: This timer is started when the PCS receiver enters the RX_WTF state, indicating that the receiver has encountered a wake time fault. The rx_wf_timer allows the receiver an additional period in which to synchronize or return to the quiescent state before a link failure is indicated. The timer terminal count is set to T_{WTF}. When the timer reaches terminal count, it will set the rx_wf_timer_done = TRUE.

tx_tq_timer
: This timer is started when the PCS transmitter enters the TX_QUIET state. The timer terminal count is set to T_{QL}. When the timer reaches terminal count, it will set the tx_tq_timer_done = TRUE.

tx_tr_timer
: This timer is started when the PCS transmitter enters the TX_REFRESH state. The timer terminal count is set to T_{UL}. When the timer reaches terminal count, it will set the tx_tr_timer_done = TRUE.

tx_ts_timer
: This timer is started when the PCS transmitter enters the TX_SLEEP state. The timer terminal count is set to T_{SL}. When the timer reaches terminal count, it will set the tx_ts_timer_done = TRUE.

127.2.6.2 State diagrams

341

127.2.6.2.1 Word Encode and Word-to-Octets

(see 127.2.4.3) (see 127.2.4.3)

The Word Encode process and Word-to-Octets process are merged into one state diagram depicted in Figure 127–4, including compliance with the associated state variables as specified in 127.2.6.1.

The Word Encode process continuously maps the four XGMII lanes to four 2.5GPII symbols via the WENCODE function in the TX_XGMII state. The four 2.5GPII symbols are then serialized and output one at a time by the Word-to-Octets process. The presentation of the 2.5GPII symbols are synchronized to the PCS transmit process such that the 2.5GPII index 0 and 2 symbols are presented to the PCS transmit process which will result in the corresponding ordered set to be output to the PMA when the variable tx_even is TRUE and index 1 and 3 variables when tx_even is FALSE.

127.2.6.2.2 Transmit

The PCS Transmit process is depicted in two state diagrams: PCS Transmit ordered set and PCS Transmit code-group. The PCS shall implement its Transmit process as depicted in Figure 127–5 and Figure 127–6, including compliance with the associated state variables as specified in 127.2.6.1.

The Transmit ordered_set process continuously sources ordered_sets to the Transmit code-group process. Upon the assertion of tp_en by the 2.5GPII, the SPD ordered_set is sourced. Following the SPD, /D/ code-groups are sourced until tp_en is deasserted. Following the de-assertion of tp_en, EPD ordered_sets are sourced. If tp_en and tp_er are both deasserted, the /R/ ordered_set may be sourced, after which the sourcing of /I/ is resumed. If, while tp_en is asserted, the tp_er signal is asserted, the /V/ ordered_set is sourced except when the SPD ordered set is selected for sourcing. If the 2.5GPII indicates sequence then /Q/ ordered_sets are sourced. If the optional EEE is enabled and the 2.5GPII indicates low power idles then /LI/ ordered_sets are sourced.

The Transmit code-group process continuously sources tx_code-group<9:0> to the PMA based on the ordered_sets sourced to it by the Transmit ordered_set process. The Transmit code-group process determines the proper code-group to source based on even/odd-numbered code-group alignment, running disparity requirements, and ordered_set format.

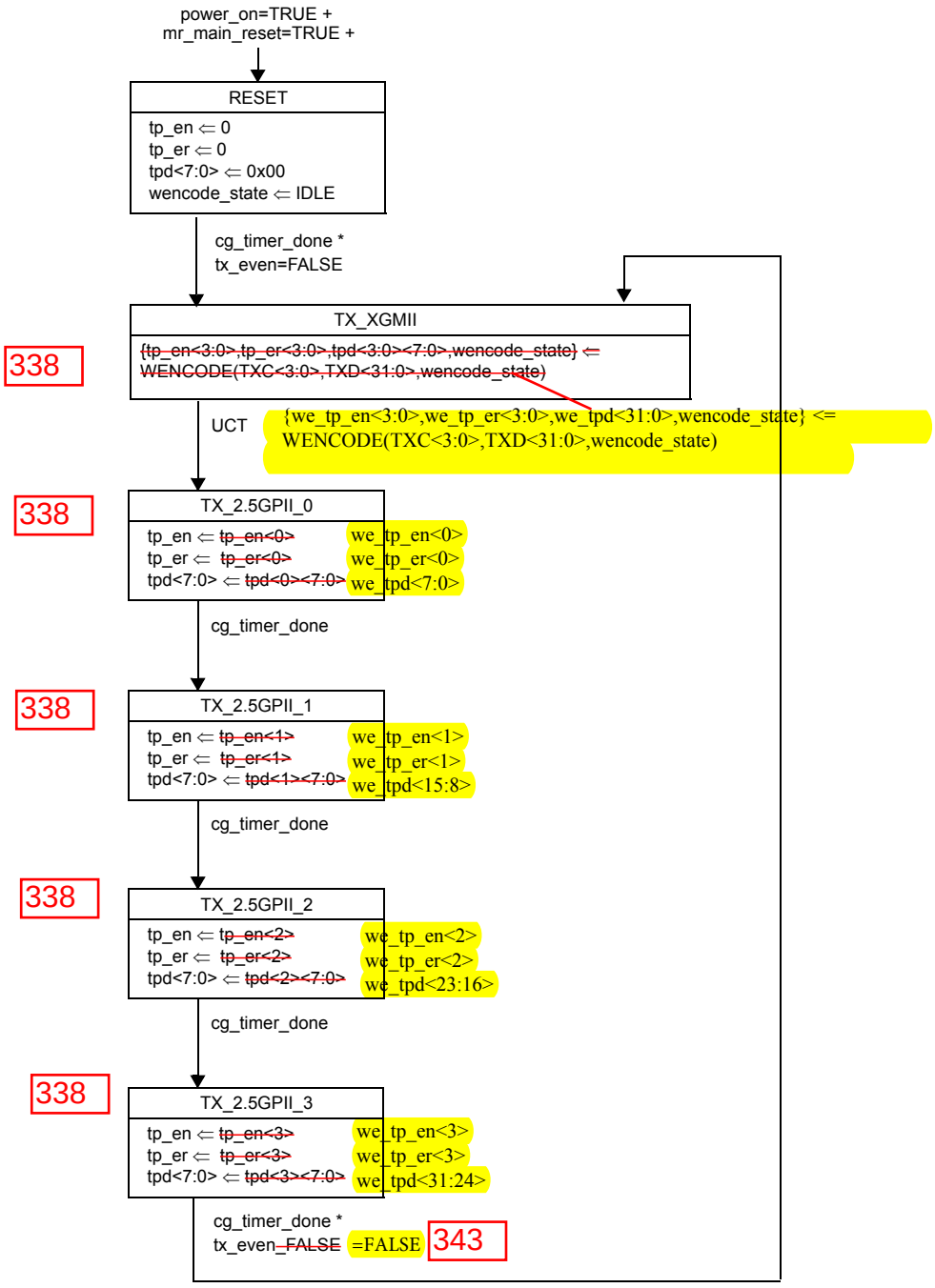
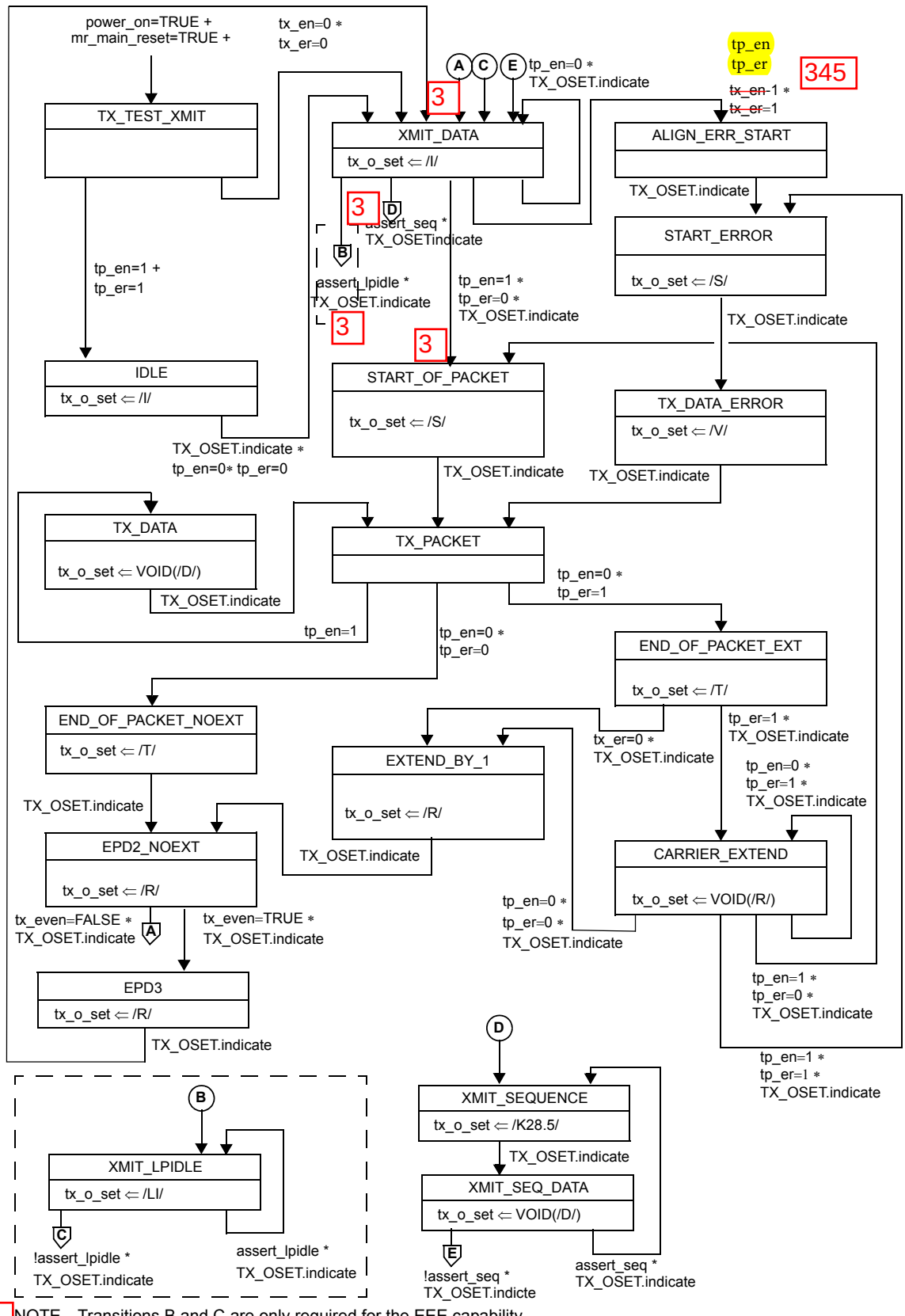


Figure 127–4—PCS Word Encode and Word-to-Octets state diagram

127.2.6.2.3 Synchronization

The PCS shall implement the Synchronization process as depicted in Figure 127–7 including compliance with the associated state variables as specified in 127.2.6.1. The Synchronization process is responsible for determining whether the underlying receive channel is ready for operation. Failure of the underlying channel typically causes the PMA client to suspend normal actions.



3 NOTE—Transitions B and C are only required for the EEE capability.

Figure 127–5—PCS transmit ordered set state diagram

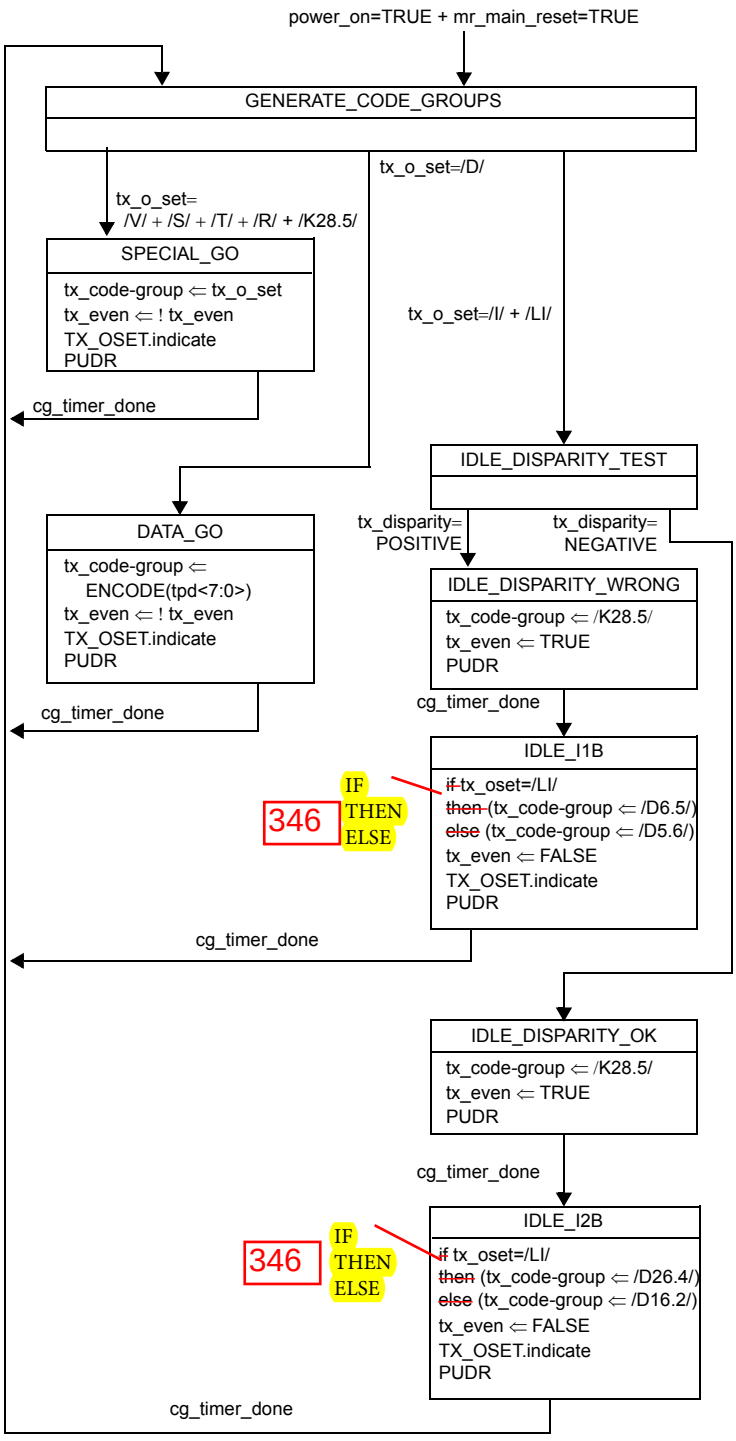


Figure 127–6—PCS transmit code-group state diagram

A receiver that is in the LOSS_OF_SYNC state and that has acquired bit synchronization attempts to acquire code-group synchronization via the Synchronization process. Code-group synchronization is acquired by the detection of three ordered sets containing commas in their leftmost bit positions without intervening invalid code-group errors. Upon acquisition of code-group synchronization, the receiver enters the SYNC_ACQUIRED_1 state. Acquisition of synchronization ensures the alignment of multi-code-group ordered sets to even-numbered code-group boundaries.

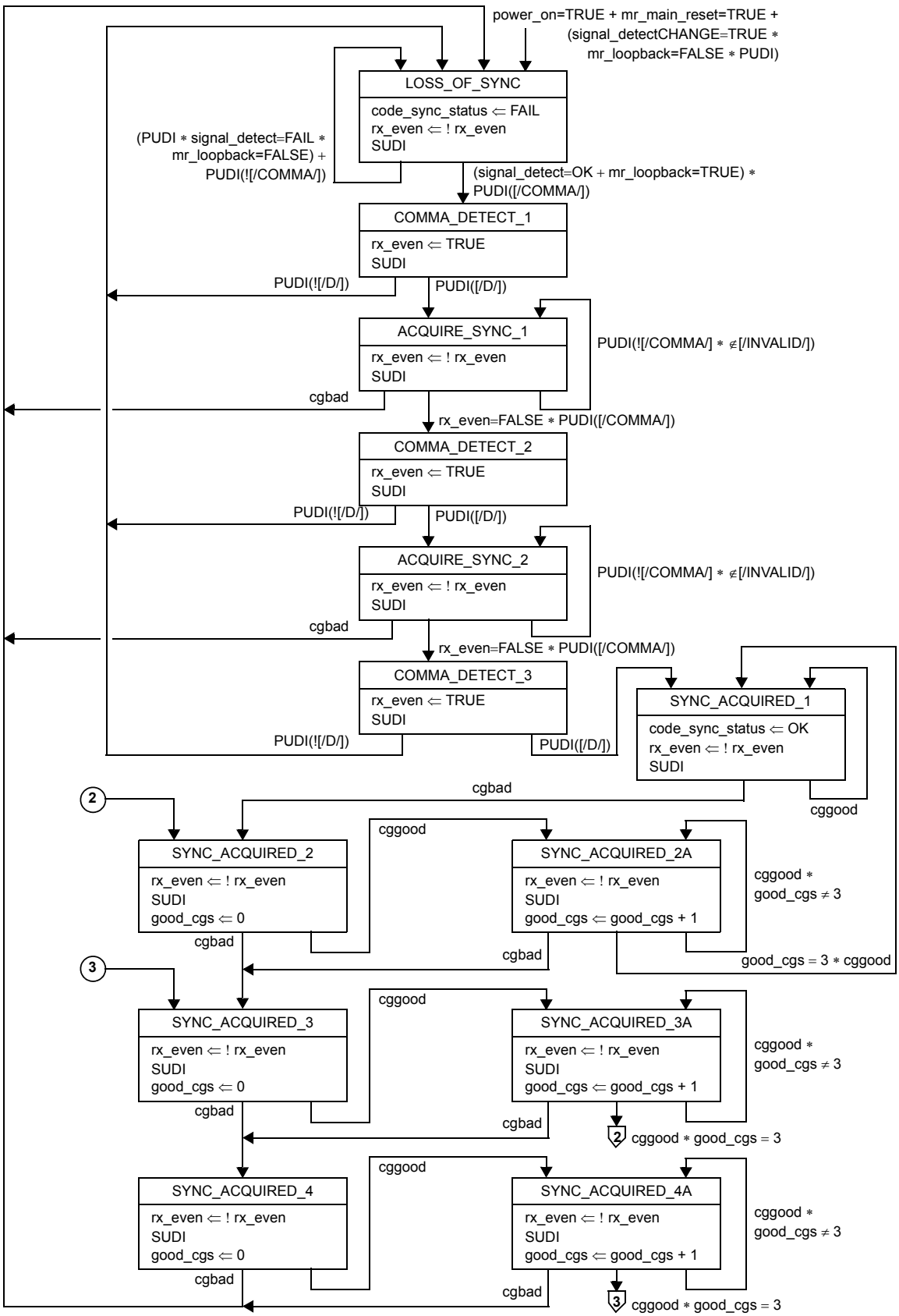


Figure 127–7—Synchronization state diagram

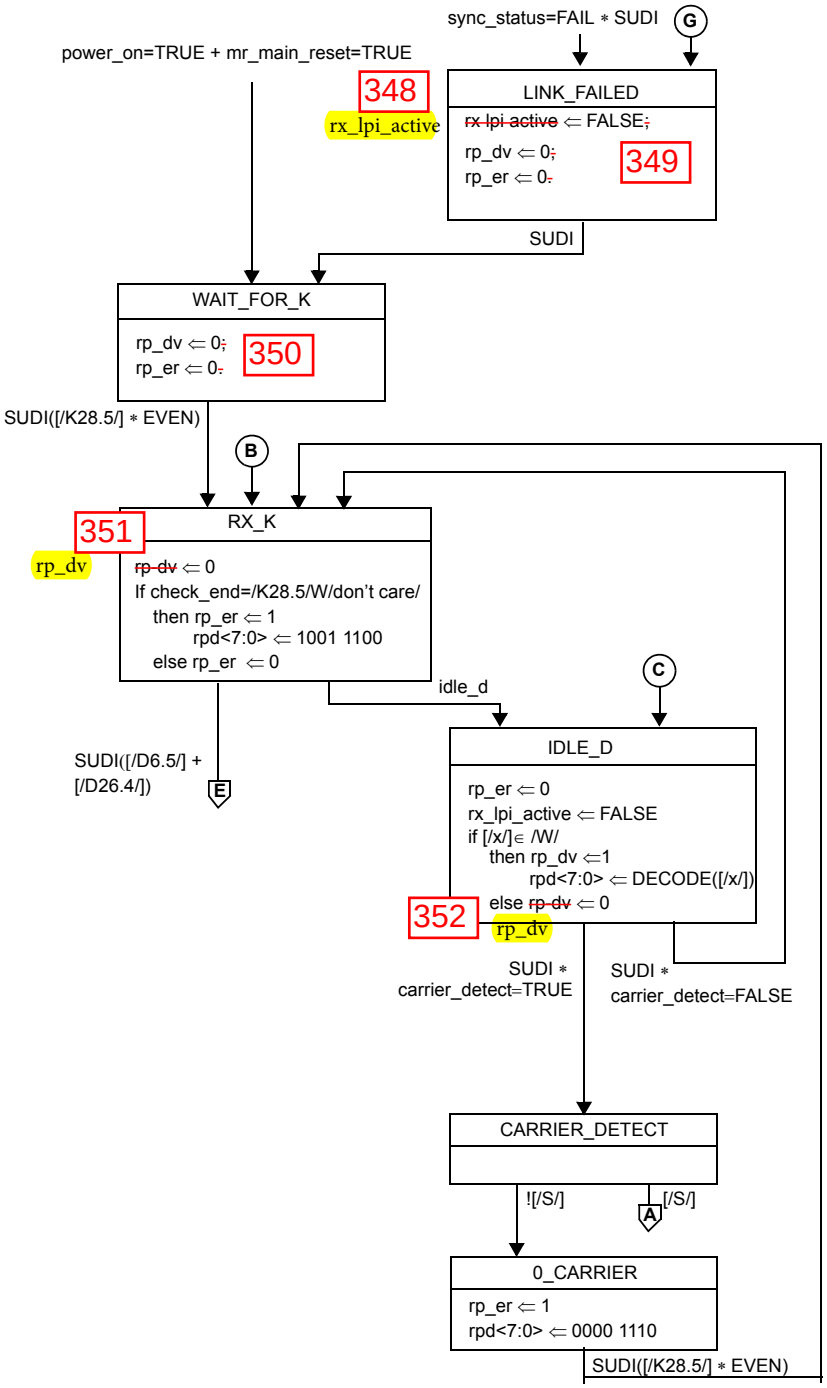
Once synchronization is acquired, the Synchronization process tests received code-groups in sets of four code-groups and employs multiple sub-states, ~~effecting~~ affecting hysteresis, to move between the SYNC_ACQUIRED_1 and LOSS_OF_SYNC states. 135

For EEE capability the relationship between sync_status and code_sync_status is given by ~~Figure 127–8c; otherwise sync_status is identical to code_sync_status.~~ the definition of the sync_status variable in 127.2.6.1.3; otherwise sync_status is identical to code_sync_status. 347

127.2.6.2.4 Receive

The PCS shall implement its Receive process as depicted in Figure 127–8a and Figure 127–8b, including compliance with the associated state variables as specified in 127.2.6.1. The PCS shall implement Figure 127–8c if the optional EEE is present and enabled.

The PCS Receive process continuously passes rpd<7:0> and sets the rp_dv and rp_er signals to the 2.5GPII based on the received code-group from the PMA.



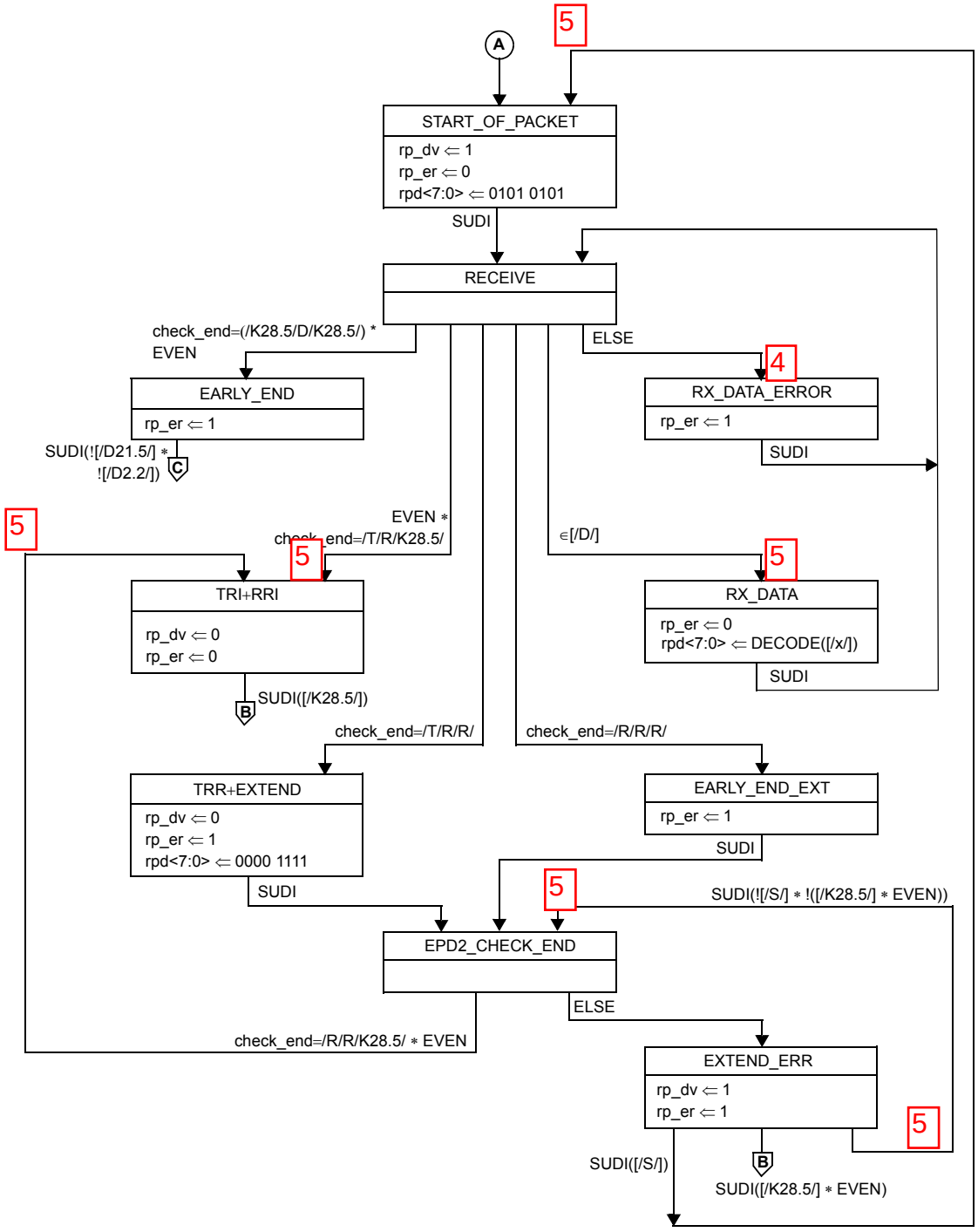
NOTE 1

NOTE—Outgoing arcs leading to labeled polygons flow off page to corresponding incoming arcs leading from labeled circles on Figure 127–8b and Figure 127–8c, and vice versa.

353

NOTE 2 - The transitions from the CARRIER_DETECT state is a test against the codegroup obtained from the SUDI that caused the transition to CARRIER_DETECT state.

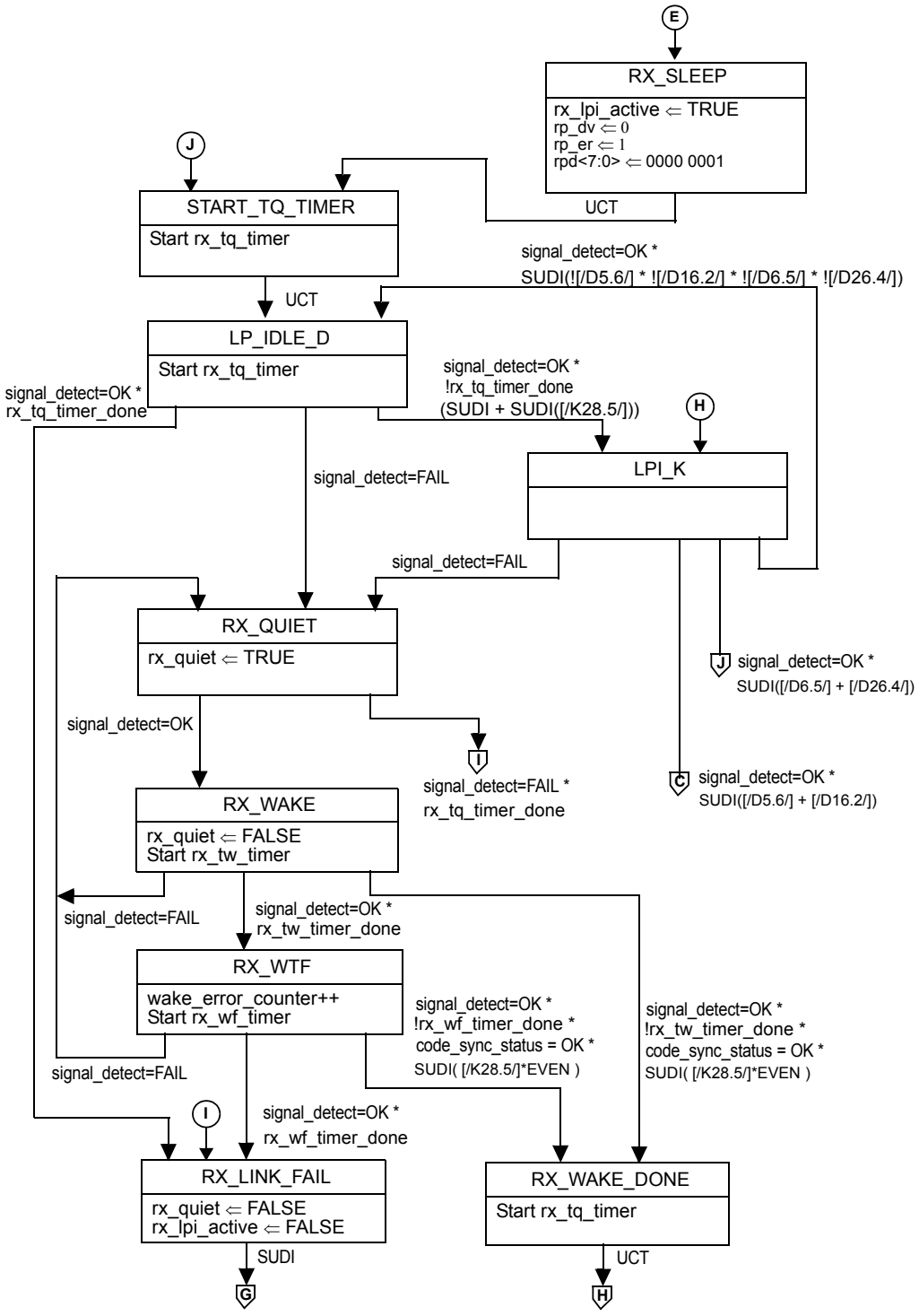
Figure 127–8a—PCS receive state diagram, part a



NOTE 1—Outgoing arcs leading to labeled polygons flow off page to corresponding incoming arcs leading from labeled circles on Figure 127–8a, and vice versa.

NOTE 2—In the transition from RECEIVE to RX_DATA state the transition condition is a test against the code-group obtained from the SUDI that caused the transition to RECEIVE state.

Figure 127–8b—PCS receive state diagram, part b



NOTE—Outgoing arcs leading to labeled polygons flow off page to corresponding incoming arcs leading from labeled circles on Figure 127–8a, and vice versa.

Figure 127–8c—PCS Receive state diagram, part c
(only required for the optional EEE capability)

127.2.6.2.5 Octets-to-Word and Decode

The Octets-to-Word process and Word decode process are merged into one state diagram depicted in Figure 127–9, including compliance with the associated state variables as specified in 127.2.6.1.

The Octets-to-Word process continuously queues the incoming 2.5GPII symbols from the PCS receive process and outputs four 2.5GPII symbols at a time using the WALIGN function. Symbols may be dropped or idles symbols added by the WALIGN function. The Word Decode process continuously maps the four 2.5GPII symbols and presents them on the XGMII using the WDECODE function in the RX_XGMII state.

127.2.6.2.6 LPI state diagram

A PCS that supports the EEE capability shall implement the LPI transmit process as shown in Figure 127–10. The transmit LPI state diagram controls tx_quiet, which disables the transmitter when true**.**

The timer values for these state diagrams are shown in Table 127–6 for transmit and Table 127–7 for receive.

Table 127–6—Transmitter LPI timing parameters

Parameter	Description	Min	Max	Units
T_{SL}	Local Sleep Time from entering the TX_SLEEP state to when tx_quiet is set to TRUE	19.9	20.1	µs
T_{QL}	Local Quiet Time from when tx_quiet is set to TRUE to entry into the TX_REFRESH state	2.5	2.6	ms
T_{UL}	Local Refresh Time from entry into the TX_REFRESH state to entry into the TX_QUIET state	19.9	20.1	µs

Table 127–7—Receiver LPI timing parameters

Parameter	Description	Min	Max	Units
T_{QR}	The time the receiver waits for signal detect to be set to OK while in the LP_IDLE_D, LPI_K and RX_QUIET states before asserting a rx_fault	3	4	ms
T_{WR}	Time the receiver waits in the RX_WAKE state before indicating a wake time fault (WTF)		11	µs
T_{WTF}	Wake time fault recovery time		1	ms

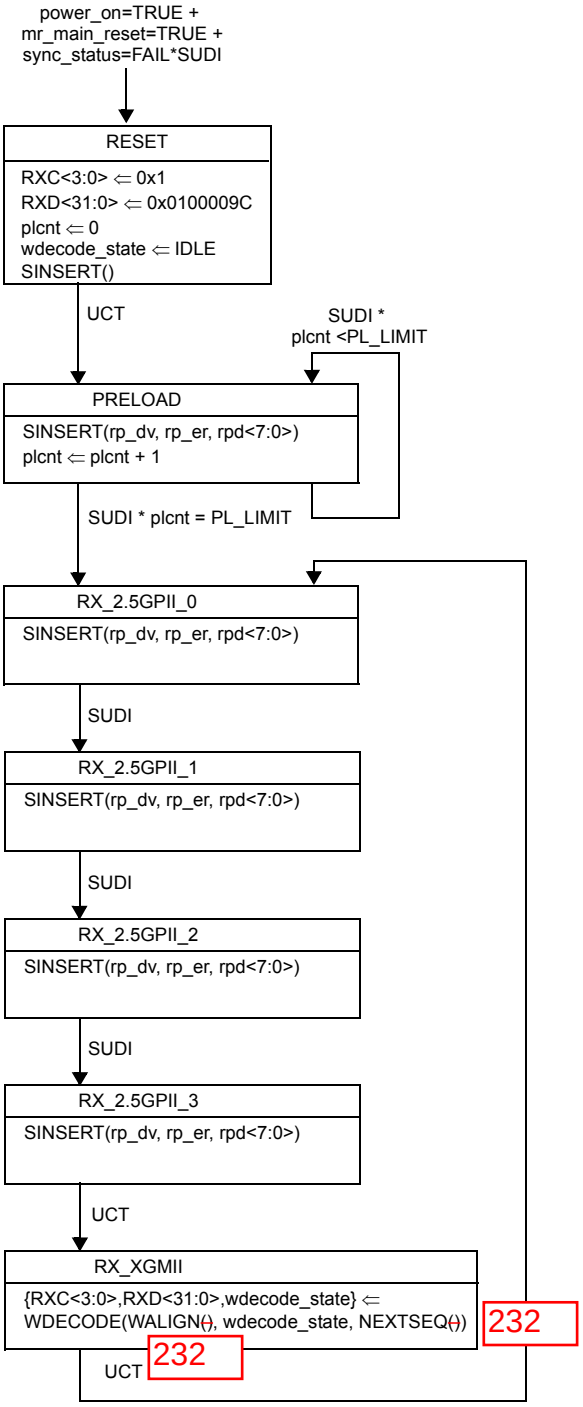


Figure 127–9—Octets-to-Word and Decode state diagram

127.2.6.2.7 LPI status and management

For EEE capability, the PCS indicates to the management system that LPI is currently active in the receive and transmit directions using the status variables shown in Table 36-10.

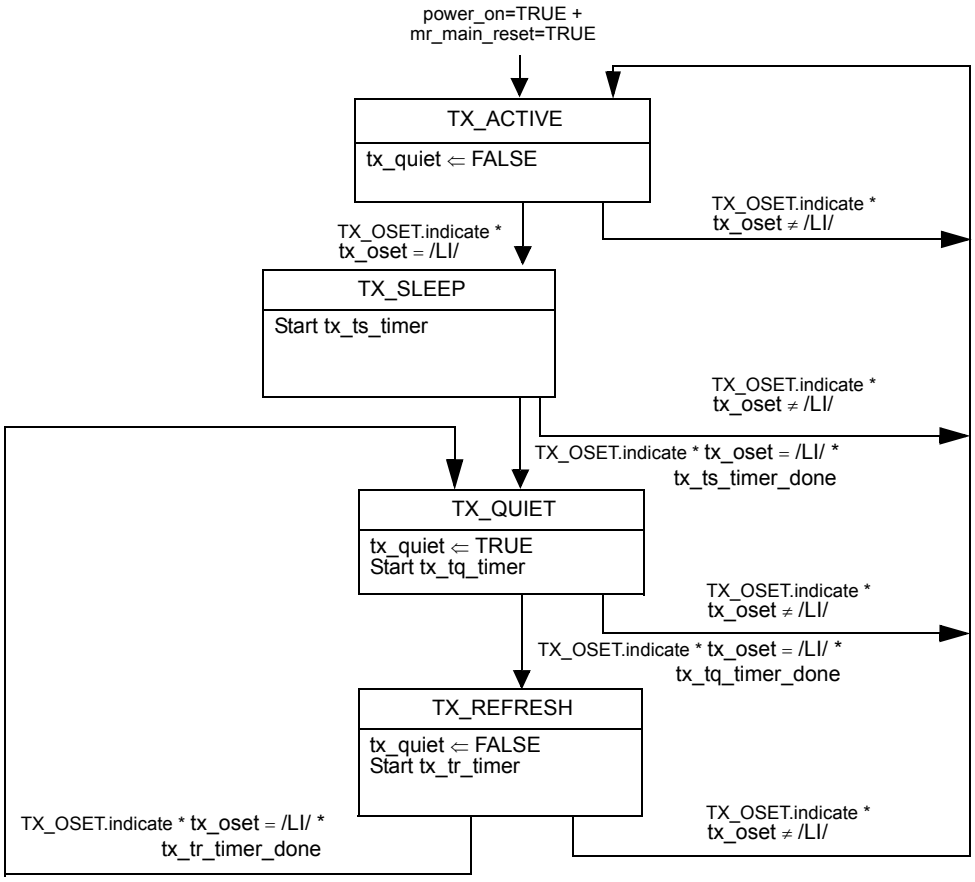


Figure 127–10—LPI Transmit state diagram

127.3 Physical Medium Attachment (PMA) sublayer

127.3.1 Service Interface

The PMA provides a Service Interface to the PCS. These services are described in an abstract manner and do not imply any particular implementation. The PMA Service Interface supports the exchange of code-groups between PCS entities. The PMA converts code-groups into bits and passes these to the PMD, and vice versa. It also generates an additional status indication for use by its client.

The following primitives are defined:

PMA_UNITDATA.request(tx_code-group<9:0>)

PMA_UNITDATA.indication(rx_code-group<9:0>)

127.3.1.1 PMA_UNITDATA.request

This primitive defines the transfer of data (in the form of code-groups) from the PCS to the PMA. PMA_UNITDATA.request is generated by the PCS Transmit process.

127.3.1.1.1 Semantics of the service primitive

PMA_UNITDATA.request(tx_code-group<9:0>)

The data conveyed by PMA_UNITDATA.request is the tx_code-group<9:0> parameter defined in 127.2.6.1.3.

127.3.1.1.2 When generated

The PCS continuously sends, at a nominal rate of 312.5 MHz, tx_code-group<9:0> to the PMA.

127.3.1.1.3 Effect of receipt

Upon receipt of this primitive, the PMA generates a series of ten PMD_UNITDATA.request primitives, requesting transmission of the indicated tx_bit to the PMD.

127.3.1.2 PMA_UNITDATA.indication

This primitive defines the transfer of data (in the form of code-groups) from the PMA to the PCS. PMA_UNITDATA.indication is used by the PCS Synchronization process.

127.3.1.2.1 Semantics of the service primitive

PMA_UNITDATA.indication(rx_code-group<9:0>)

The data conveyed by PMA_UNITDATA.indication is the rx_code-group<9:0> parameter defined in 127.2.6.1.3.

127.3.1.2.2 When generated

The PMA continuously sends one rx_code-group<9:0> to the PCS corresponding to the receipt of each code-group aligned set of ten PMD_UNITDATA.indication primitives received from the PMD. The nominal rate of the PMA_UNITDATA.indication primitive is 312.5 MHz, as governed by the recovered bit clock.

127.3.1.2.3 Effect of receipt

The effect of receipt of this primitive by the client is unspecified by the PMA sublayer.

127.3.2 Functions within the PMA

Figure 127–2 and Figure 127–3 depicts the mapping of the four octet-wide data path of the XGMII to the ten-bit-wide code-groups of the PMA Service Interface, and on to the serial PMD Service Interface. The PMA comprises the PMA Transmit and PMA Receive processes for 2.5GBASE-X.

The PMA Transmit process serializes tx_code-groups into tx_bits and passes them to the PMD for transmission on the underlying medium, according to Figure 127–3. Similarly, the PMA Receive process deserializes rx_bits received from the PMD according to Figure 127–3. The PMA continuously conveys ten bit code-groups to the PCS, independent of code-group alignment. After code-group alignment is achieved,