2.5GBASE Backplane Baseline Proposal

January 20, 2016

William Lo, Marvell

Supporters

- Anthony Calbone (Seagate)
- Peter Wu (Marvell)

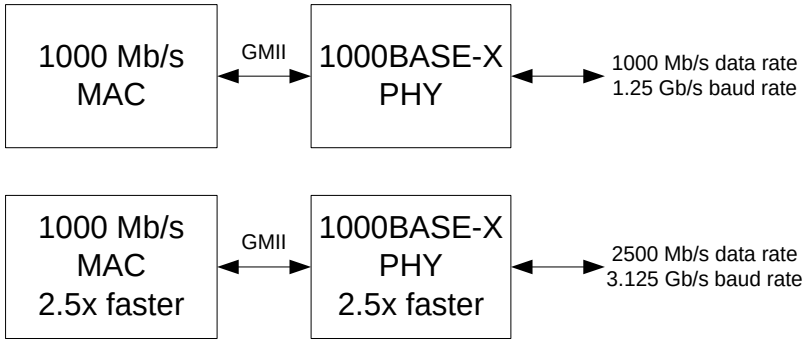
Agenda

- Define Baseline Proposal for 2.5G Backplane
- Covers Reconciliation Sublayer, PCS, PMA, Auto-Negotiation, EEE
- Does not cover PMD
- Does not cover registers

- Goal: Adopt set of baselines so we can get bulk of text started

Define 2.5G PHY for wide adoption

- **Implementations in the field from various suppliers running 1000BASE-X at 2.5 times speed.**



- **Can leverage existing equipment if we define 2.5G backplane such that existing equipment can operate in compatible fashion.**
  - **Existing implementations would be compatible, but do not need to be compliant**

- **Plenty of margin with 1000BASE-X running 2.5 times faster**
  - **No RX DFE and no TX Equalization - wu_CU4HDDSG_01_1115.pdf**
  - **using channels from Calbone_CU4HDDsg_02_0915.pdf**

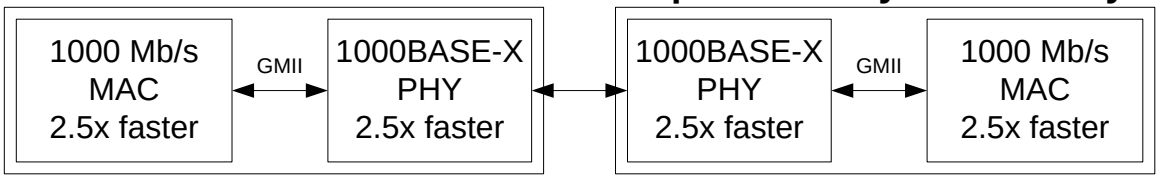
What is needed for compatibility

- **Adopt subset of 1000BASE-X PCS and run 2.5 times faster**
  - No need to support Clause 37 Auto-Negotiation
  - Clause 73.1 – recommends disabling Clause 37 AN if Clause 73 AN is used
  - Let's enforce this by making it mandatory to disable (or not implement) portion of the PCS that supports Clause 37 AN.
- **Make Clause 73 Parallel Detect support mandatory for 2.5G Backplane**
  - i.e. One PHY AN on and other PHY AN off. The PHY with AN on detects 2.5G signal from link partner and stops the AN process and proceeds to link in 2.5G
  - Parallel detection currently supported in 1000BASE-KX, 10GBASE-KX4.
- **Energy Efficient Ethernet can be enabled only if both PHYs advertise the capability using Clause 73 Auto-Negotiation**
  - Implication – Parallel Detect link up will not support EEE
  - EEE can be manually turned on without AN but this is outside the scope of the standard.
- **Need to reconcile possibility of GMII based 1G MAC running 2.5G interacting with XGMII based 10G MAC running 2.5G**
  - Existing 2.5G implementations most likely using scaled up 1G MAC
  - 802.3bz chose scaled down 10G MAC running at 2.5G. We are stuck with XGMII.

Cases To Consider

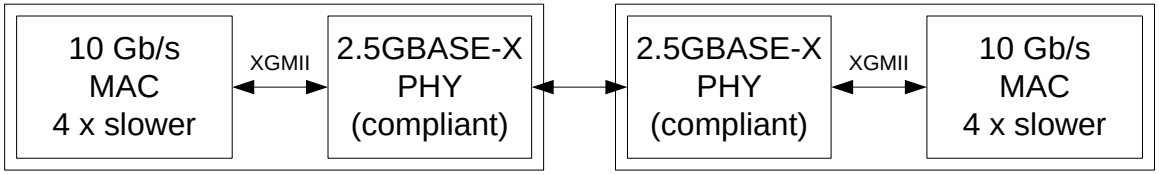
Legacy PHY to Legacy PHY

- No Problem – 1000BASE-X at 2.5x speed already works today



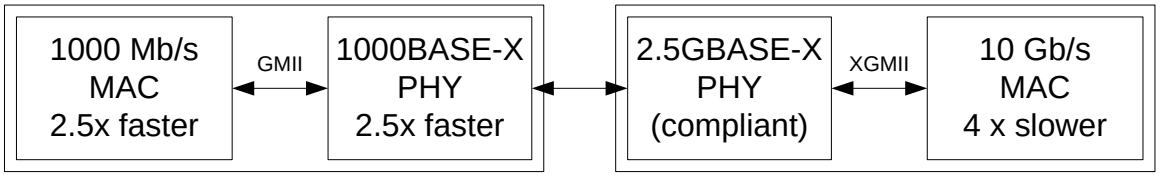
Compliant PHY to Compliant PHY

- No Problem – We can define anything we want
- Need to introduce concept of passing Sequence and Signal ordered set as 10G MAC capable of sending



Legacy PHY to Compliant PHY

- Compliant PHY needs to compensate for non-4 byte alignment issue
- Compliant PHY passing transmitting Sequence and Signal ordered set does not interfere with legacy PHY



Fitting 1000BASE-X PCS using 2.5G XGMII from 802.3bz

- Implementing *MII is optional. But from standards point of view it is necessary as a fixed point of reference to the Reconciliation Sublayer
- 1000BASE-X PCS uses the GMII as reference
- 802.3bz defined the 2.5G reference using the XGMII

- To allow compatibility need to address following 2 issues
- 1 byte vs 4 byte alignment issue of start of packet
  - XGMII is 4 byte interface while GMII is 1 byte interface
- Passing Sequence and Signal ordered set
  - XGMII based MAC can pass |Q| ordered sets
  - No concept of |Q| or |Fsig| ordered sets in GMII based MAC

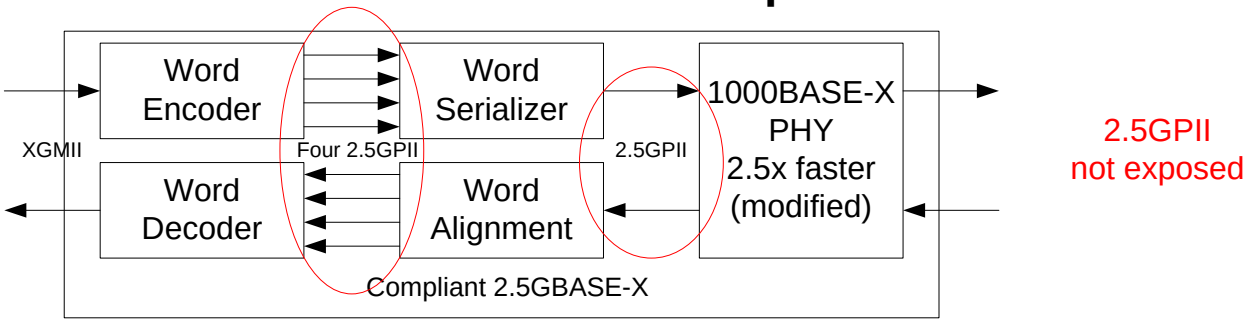
Assumption and Definition

- 802.3bz XGMII running at 2.5Gb/s is used as the reference to the Reconciliation Sublayer

- Define 2.5GPII – (2.5Gb/s PCS Intermediate Interface)
  - An internal functional interface with the exact same pin mapping as full duplex GMII (No CRS or COL signals)
  - Runs 2.5x faster than GMII
  - Adds additional coding to generate/receive single K28.5 symbol

Proposal on PCS Specification

- **Start with base 1000BASE-X PCS running 2.5x faster and make modifications**
  - **Small state machine modification to pass |Q| and |Fsig| ordered sets**
  - **Disable/remove portions of the state machine that supports Clause 37 Auto-Negotiations**
  - **If we decide to block |Q| and |Fsig| and send as idles then no need for changes. Simply set variable to disable Clause 37.**
- **Word serializer/alignment simply handles the 4 byte to 1 byte conversion between XGMII and 2.5GPII**
- **Word encoder/decoder – mapping between XGMII to Internal 2.5GPII**
- **Implementation as shown does not require much incremental logic**



- **Does not preclude implementations that directly map XGMII into PCS**
  - **Diagram above for IEEE functional specification purposes only**

Word Serializer/Alignment Specification

- **Serializer**
  - Simply take four 2.5GPII bytes and send it out one 2.5GPII byte at a time
- **Alignment**
  - XGMII operates 4 bytes at a time and requires Start of Packet be on byte 0
  - 2.5GPII operates 1 byte at a time. Simply grouping 4 bytes will not guarantee Start of Packet will be on byte 0.
  - Use deficit idle counting (DIC) in Clause 46.3.1.4 to align Start of Packet

Deficit	SOP on byte 0	SOP on byte 1	SOP on byte 2	SOP on byte 3
0 byte	Do nothing	Delete 1 idle byte	Delete 2 idle bytes	Delete 3 idle bytes
1 byte	Do nothing	Delete 1 idle byte	Delete 2 idle bytes	Insert 1 idle byte
2 bytes	Do nothing	Delete 1 idle byte	Insert 2 idle bytes	Insert 1 idle byte
3 bytes	Do nothing	Insert 3 idle bytes	Insert 2 idle bytes	Insert 1 idle byte

- **Extend concept of Deficit Idle Counting to align other things to byte 0**
  - Start of Packet
  - First Low power idle when transitioning in from idles
  - Start of ordered sets
    - This is more an error condition case as ordered set can only be generated from XGMII interface which should already be aligned
    - See slide on ordered set on how to align

Word Encoder Specification

- **2.5GPII – Define Sequence code**
  - We are NOT changing definition of GMII of Clause 35, we are only defining a new code for 2.5GPII
  - TX shown, RX similar
- **Simple XGMII to four 2.5GPII mapping except for Sequence ordered set**
- **Sequence ordered set expand from 4 bytes to 8 bytes**
  - Throw away every other Sequence order set on XGMII – ok to do this (10GBASE-X4 throws away more than 90% of Sequence order set seen on XGMII)
  - Use Prev Seq variable to track whether to throw away next Sequence ordered set on XGMII
  - Truncate ordered set transmission if one throw away is not a Sequence ordered set

TX_EN	TX_ER	TXD[7:0]	Description
0	0	xx	Idle
0	1	0x01	Low Power Idle
0	1	0x0F	Carrier Extend - Not used
0	1	0x1F	Carrier Extend Error - Not used
1	0	00 to FF	Data
1	1	xx	Transmit Error
0	1	0x9C	Sequence

XGMII						Four 2.5GPII				
Byte 0	Byte 1	Byte 2	Byte 3	Prev Seq		GPII 0	GPII 1	GPII 2	GPII 3	Next Seq
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	No
Idle	Idle	Idle	Idle	X		Idle	Idle	Idle	Idle	No
LPI	LPI	LPI	LPI	X		LPI	LPI	LPI	LPI	No
SOP	Data A/Err	Data B/Err	Data C/Err	X		0x55 Data	Data A/Err	Data B/Err	Data C/Err	No
Terminate	Idle	Idle	Idle	X		Idle	Idle	Idle	Idle	No
Data A/Err	Terminate	Idle	Idle	X		Data A/Err	Idle	Idle	Idle	No
Data A/Err	Data B/Err	Terminate	Idle	X		Data A/Err	Data B/Err	Idle	Idle	No
Data A/Err	Data B/Err	Data C/Err	Terminate	X		Data A/Err	Data B/Err	Data C/Err	Idle	No
Sequence	Data X	Data Y	Data Z	No		Sequence	Data S0	Sequence	Data S1	Yes
Sequence	Data X	Data Y	Data Z	Yes		Sequence	Prev Data S2	Sequence	Prev Data S3	No
else						Error	Error	Error	Error	No

Word Decoder Specification

- State dependent mapping based on Prev Word and Next Word variables
- False Carrier, Carrier Extend Error, and out of place Carrier Extend converted to errors
- Look ahead needed to check S0, S1, S2, S3 bit 7 consistency (i.e. 0110)
- If link is down then XGMII outputs Local Fault ordered set

Four 2.5GPII					
GPII 0	**GPII 1**	**GPII 2**	**GPII 3**	**Prev Word**	**Next Word**
Data A/Err	Data B/Err	Data C/Err	Data D/Err	Not Idle	X
Data *	Data A/Err	Data B/Err	Data C/Err	Idle	X
Idle	Idle	Idle	Idle	Not Data	X
Idle	Idle	Idle	Idle	Data	X
Data A/Err	Idle or Carrier Extend	Idle	Idle	Data	X
Data A/Err	Data B/Err	Idle	Idle	Data	X
Data A/Err	Data B/Err	Data C/Err	Idle or Carrier Extend	Data	X
LPI	LPI	LPI	LPI	X	X
Idle	Idle	LPI	LPI	X	X
LPI	LPI	Idle	Idle	X	X
Sequence	Data S0	Sequence	Data S1	X	Sequence S2, S3
Sequence	Data S0	Sequence	Data S1	X	Not Sequence S2, S3
Sequence	Data S2	Sequence	Data S3	Sequence	X
else					

XGMII				
Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Prev Word**
Data A/Err	Data B/Err	Data C/Err	Data D/Err	Data
SOP	Data A/Err	Data B/Err	Data C/Err	Data
Idle	Idle	Idle	Idle	Idle
Terminate	Idle	Idle	Idle	Idle
Data A/Err	Terminate	Idle	Idle	Idle
Data A/Err	Data B/Err	Terminate	Idle	Idle
Data A/Err	Data B/Err	Data C/Err	Terminate	Idle
LPI	LPI	LPI	LPI	Idle
LPI	LPI	LPI	LPI	Idle
Idle	Idle	Idle	Idle	Idle
Sequence	Data X	Data Y	Data Z	Sequence
Idle	Idle	Idle	Idle	Idle
Sequence	Data X	Data Y	Data Z	Idle
Error	Error	Error	Error	Error

What Does Sequence Ordered Set Like

- XGMII – Sequence, Data X, Data Y, Data Z (4 bytes)
- 2.5GPII – Sequence, S0, Sequence, S1, Sequence, S2, Sequence, S3
- Output of PCS – K28.5, W0, K28.5, W1, K28.5, W2, K28.5, W3

- Sequence ordered set - |Q| = /K28.5/W/K28.5/W/K28.5/W/K28.5/W/
- Truncated Sequence ordered set when less than 8 symbols output i.e. /K28.5/W/K28.5/W/
- Signal ordered set - |Fsig| = /K28.5/W/K28.5/W/K28.5/W/K28.5/W/
  - Different /W/ sent
  - XGMII cannot indicate |Fsig| but including mechanics to do this here since 10GBASE-X4 and 10GBASE-R have provision to send |Fsig|

- |W| ordered set defined in next slide

What Does Sequence Ordered Set Like

- S0[7] = S3[7] = 0, S1[7] = S2[7] = 1 for |Q| Ordered Set
- S1[7] = 1, S0[7] = S2[7] = S3[7] = 0 for |Fsig| Ordered Set
  - Note: S0[7] and S1[7] opposite values can use for ordered set alignment
  - S0[7],S1[7],S2[7],S3[7] = 0110 or 0100. Look for 01 pattern to align.
- 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

- Six K28.5 Dx.y to avoid
  - Forcing bit 6 to be the same as bit 7 or bit 5 guarantees this

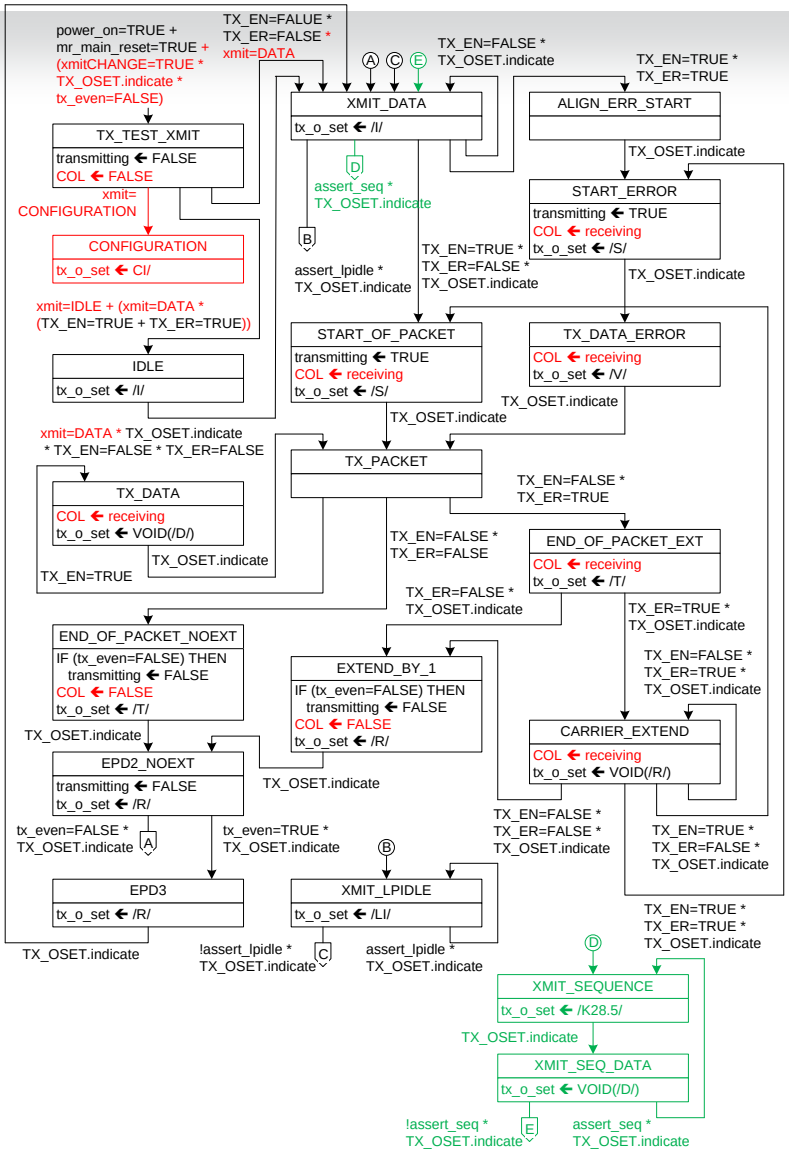
Function	Data Code	Octet	7	6	5	4	3	2	1	0
Idle	D5.6	C5	1	1	0	0	0	1	0	1
Idle	D16.2	50	0	1	0	1	0	0	0	0
LPI	D6.5	A6	1	0	1	0	0	1	1	0
LPI	D26.4	9A	1	0	0	1	1	0	1	0
Config	D21.5	B5	1	0	1	1	0	1	0	1
Config	D2.2	42	0	1	0	0	0	0	1	0

- |W| is the set of 10-bit data ordered sets that can be encoded from the 128 possible values of S.

1000BASE-X State Machine Modifications

- **Disable configuration ordered set used to support Clause 37 Auto-Negotiation**
  - **Can delete the portion of the state machine shown in red**
  - **Alternatively, it can be disabled by forcing xmit = DATA**
- **New changes to support |Q| and |Fsig| ordered set shown in green**
- **Unchanged state machine shown in black**

1000BASE-X PCS Transmit ordered_set State Diagram (Fig 36-5)



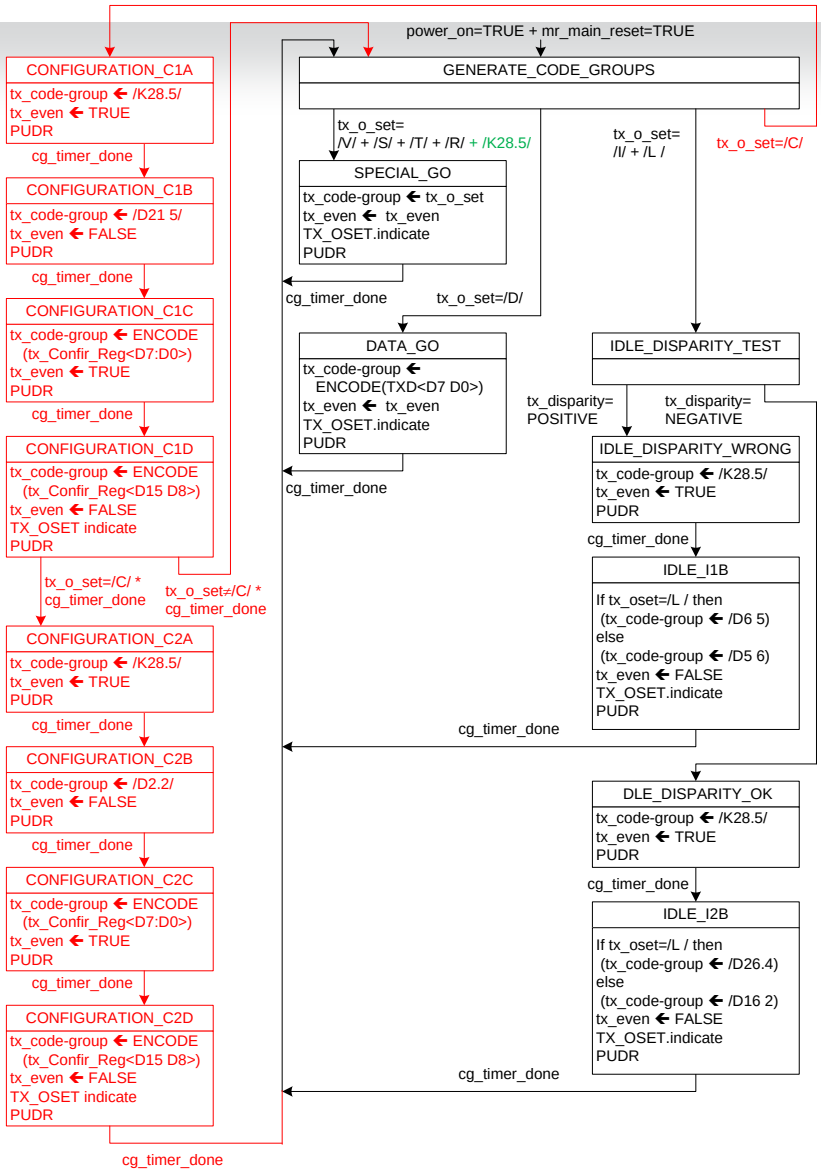
- **Add states and variables to handle Sequence ordered set**

Variable:
assert_seq
Alias used for sequence ordered set, consisting of the following terms:
(TX_EN=FALSE * TX_ER=TRUE * (TXD<7:0>=0x9C)

Constant:
/Q/
Sequence ordered set. 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/.

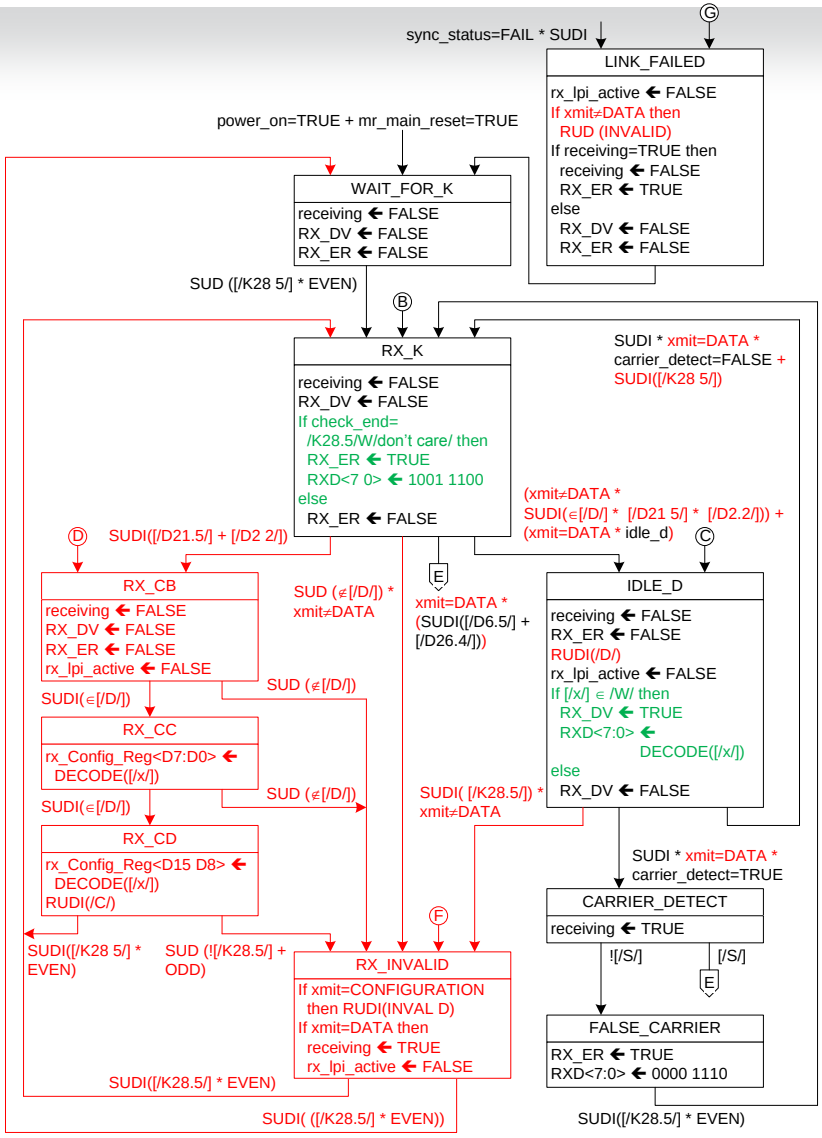
/W/
The set of 128 code-groups that is generated by ENCODE(s<7:0>) where for all 128 possible values of x<6:0>
s<7> = x<6>, s<5:0> = x<5:0>,
s<6> is set to x<5> when x<2> = 1 and s<6> is set to x<6> when x<2> = 0
Note that /W/ is a subset of /D/

1000BASE-X PCS Transmit code_group State Diagram (Fig 36-6)



- **Add ability to send single K28.5**

1000BASE-X PCS Receive State Diagram, part a (Fig 36-7a)



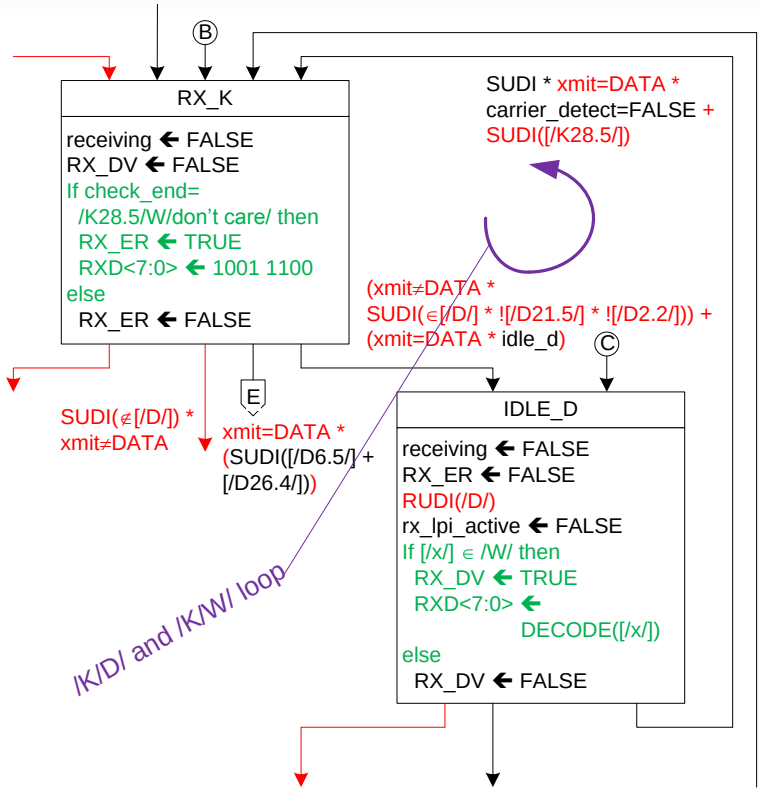
- **Decode Sequence Ordered set**

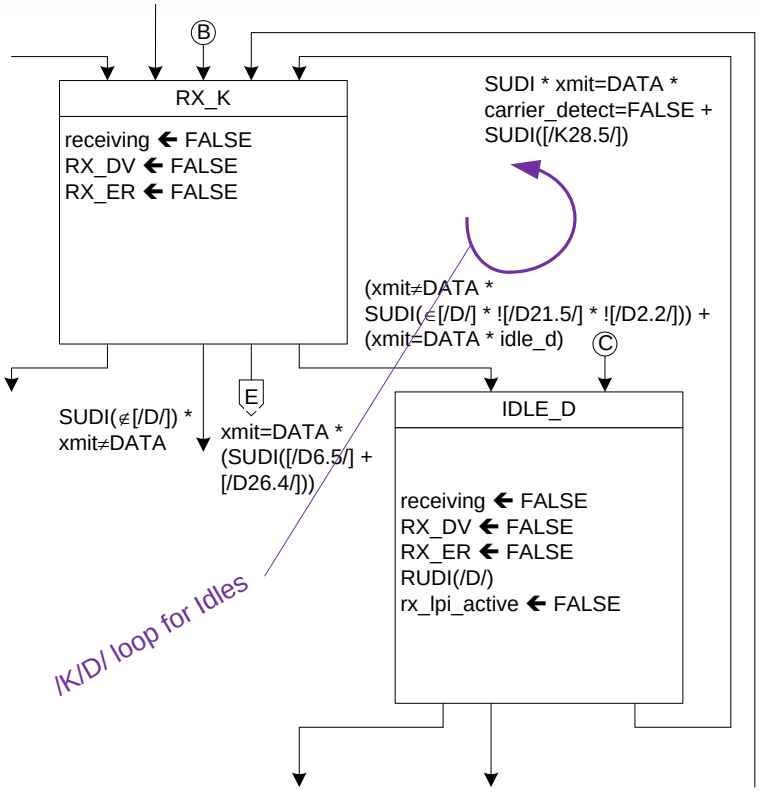
Variable:
idle_d
Alias for the following terms:
SUDI(![/D21.5/] * ![/D2.2/]) UCT
that uses an alternate form to support the EEE capability:
SUDI(![/D21.5/] * ![/D2.2/] * ![/D6.5/] * ![/D26.4/])

Other 1000BASE-X State Diagrams

- **PCS Receive State diagram, Part b (Fig 36-7b)**
  - Packet handling portion of the state machine
  - No changes needed for Sequence ordered set handling
  - If Configuration ordered set handling not implemented then transition to D is removed and transition to C is simply SUDI
- **PCS Receive State diagram, Part c (Fig 36-7c)**
  - Low Power Idle portion of the state machine
  - No changes needed for Sequence ordered set handling
  - If Configuration ordered set handling not implemented then transition to D and F are removed and all xmit variables are forced to DATA
- **Carrier Sense State Diagram (Fig 36-8)**
  - Not needed. We are running full duplex only
- **Synchronization State Diagram (Fig 36-9)**
  - No change
- **LPI Transmit State Diagram (Fig 36-10)**
  - No change

Sequence Ordered Set Compatibility





- **Idle is /K/D/K/D/K/D/K/D/…. while |Q| is /K/W/K/W/K/W/K/W/**
  - **|W| is a subset of |D| and is treated as idles at unmodified 1000BASE-X PCS**
  - **Green logic differentiates whether [/x/] is in the set of |W| or not**

PMA and Clause 73 Auto-Negotiation

- **PMA**
  - Adopt Clause 36.3.1 and 36.3.2 with following modifications
  - Change 125MHz to 312.5MHz
  - Change reference from GMII to XGMII
  - No need to adopt clause 36.3.3 to 36.3.7 since we are not exposing the PMA
  - We may want to specify new PMA test functions in lieu of Clause 36.3.8

- **Auto-Negotiation**
  - Adopt Clause 73 Auto-Negotiation
  - Technology Ability Field bit A11 to advertise 2.5GBASE-KX
  - Require parallel detect support for 2.5GBASE-KX

Energy Efficient Ethernet

- State machines already defined in Clause 36
- 2.5GBASE-KX EEE timing identical to XGXS (XAUI) timing
  - 8/10 coding and raw line rate of 3.125Gb/s

Table 78–2—Summary of the key EEE parameters for supported PHY

Protocol	T_s (µs)		T_q (µs)		T_r (µs)	
	Min	Max	Min	Max	Min	Max
100BASE-TX	200	220	20 000	22 000	200	220
1000BASE-T	182.0	202.0	20 000	24 000	198.0	218.2
1000BASE-KX	19.9	20.1	2 500	2 600	19.9	20.1
XGXS (XAUI)	19.9	20.1	2 500	2 600	19.9	20.1
10GBASE-KX4	19.9	20.1	2 500	2 600	19.9	20.1
10GBASE-KR	4.9	5.1	1 700	1 800	16.9	17.5
10GBASE-T	2.88	3.2	39.68	39.68	1.28	1.28

THANK YOU