



Document Number	EDCS-1517762
Based on Template	EDCS-189229 Rev 20
Created By	Amrik Bains
Contributors	Arvind Kansal

Universal SXGMII PHY-MAC Interface for

Multiple Network Port

The Universal Serial Media Independent Interface for carrying multiple network ports over a single SERDES (USXGMII-M) is specified in this document to meet the following requirements:

- Convey **MULTIPLE** network ports over an USXGMII MAC-PHY interface
- Utilize a 64/66 PCS to minimize power and serial bandwidth
- Use modified 802.3by section 108.5.2.4, to add Alignment Markers to support multiple ports over single SERDES
- System Interface operates in full duplex mode only
- Ability to send PTP time stamp from PHY to MAC to improve accuracy/jitter on encrypted/non-encrypted PTP packet with MACSec is in the ASIC
- Hardware assisted auto-negotiation for all supported speeds
- Flexibility to add new features using Extension Field in pre-amble

USXGMII isn't a single protocol, but rather an architecture that allows for the definition of specific interfaces in a way that maximizes reuse and reduces risk. There are other implementation targeted at different applications and contact Cisco for details. A PHY can implement one or more options specified above based on a particular application, cost and power optimization.

Cisco Systems Intellectual Property

Cisco encourages others to adopt this specification. To the extent that Cisco has proprietary rights to information contained in this specification, any company wishing to use this specification may do so under reasonable, non-discriminatory terms, with reciprocity, to implement and fully comply with the specification.

The reasonable non-discriminatory terms are:

Cisco will not assert any claims of any patents that are owned or controlled by Cisco essential for implementing this specification against any party for making, using, selling, importing or offering for sale a product that implements the specification, provided, however, that Cisco retains the right to assert its patents (including the right to claim past royalties) against any party that asserts a patent it owns or controls (either directly or indirectly) against any product that implements this specification. Cisco retains the right to assert its patents against any product or portion thereof that is not necessary for compliance with the specification.

Royalty-bearing licenses are available to anyone who prefers that option.

For information contact:

Standards-ipr@cisco.com

Modification History

Revision	Date	Originator	Comments
2.0	11/20/2015	Amrik Bains	First official Cisco release derived from rev1.1
2.1	02/12/2016	Amrik Bains/Arvind Kansal	Added Alignment Markers to support 2.5G, clarification and other typo fixes.
2.2	03/08/2016	Amrik Bains	Address comments received, corrected replication and other typo.
2.3	03/31/2016	Amrik Bains	Clarification of port muxing markers vs. RS-FEC markers
2.4	04/03/2016	Amrik Bains	Added replication details at XGMII on 4-byte boundary in section 2.6
2.5	05/11/2016	Amrik Bains	Remove reference to USXGMII flow control
2.6	11/02/2016	Amrik Bains	Typo correction for Extension Field Type to: 01: Extension Field contains Tag/Signature associated with for time-stamp to be taken by the PHY
2.7	11/18/2016	Amrik Bains	Added Pre-emption features to PCH
2.8	12/20/2016	Amrik Bains	More clarification added for non FEC mode alignment makers and replication
2.9	01/28/2016	Amrik Bains/Arvind Kansal	Added for details of FEC mode
2.10	02/02/2017	Amrik Bains	Typo fixes and additional information added to EoP processing
2.11	05/11/2017	Amrik Bains	With pre-emption, additional PTP enable is required on egress – see table 6
2.12	01/03/2018	Amrik Bains	Typo fix for Frame Count vs. Fragment count in table 6
2.13	06/15/2018	Amrik Bains	AM clarification for Octal USXGMII
2.14	12/03/2018	Amrik Bains	Pre-amble clarification for 1G/2.5G
2.15	07/30/2019	Amrik Bains	Added more details on IPG and Pre-amble processing and interoperability with legacy 10M/100M/1G using 8b/10b coding
2.16	08/17/23	Dale Mohlenhoff	Clarification on Intellectual Property and remove Cisco Confidential footer

Table of Contents

1	Overview	6
1.1	In-band Control and Status Signaling	10
1.1.1	Auto-neg Mechanism	10
1.1.2	Auto-negotiation Message	11
1.1.3	Packet Control Header	14
2	Implementation Specification.....	17
2.1	XGMII Mapping	19
2.2	GMII Mapping	19
2.3	MII Mapping.....	19
2.4	Pause Frame Support	19
2.5	Auto-neg Mechanism.....	19
2.5.1	Transmitting Configuration Words	20
2.6	Rate Adaptation - Replicating Transmit Bytes	21
2.6.1	4x2.5G Tx Mode (10G MAC/PHY Interface)	21
2.6.2	4x2.5G Sampling Received Bytes (Rx)	24
2.6.3	2x5G Tx Mode	26
2.6.4	Rx 2x5G Sampling Received Bytes	27
2.7	Multiple Network Port over Single SERDES	28
2.7.1	Network Port Muxing	28
2.7.2	Tx Mux – Alignment Markers in non-FEC mode.....	31
2.7.3	Rx Demux in non-FEC mode.....	31
2.7.4	Tx Muxing in FEC mode	32
2.7.5	Rx Demux in FEC mode	33
2.7.6	Port De-Mux Framing	33
2.7.7	Clocking	33
2.7.8	Hardware Auto-negotiation Programming Sequence.....	34
2.7.9	Port ASIC Software Controlled Negotiation Programming Sequence.....	34
2.8	USXGMII Packet Control Header Implementation	36
2.9	PHY Implementation	41
2.9.1	Rx PHY Block	41
2.9.2	Tx PHY Block.....	42
2.9.3	MDI: Tx IPG and Pre-amble Processing on MDI.....	43
2.10	Electrical Specification	44
3	Appendix	45

Table of Figures

Figure 1: USXGMII Port ASIC Functional Block Diagram.....	8
Figure 2 USXGMII PHY Functional Block Diagram.....	9
Figure 3: Packet Control Header (PCH) Format	14
Figure 4: USXGMII TX/RX Implementation – Non FEC.....	17
Figure 5: USXGMII TX/RX Implementation – With FEC.....	18
Figure 6: Receiver Data sampling for 1G over 2.5G	25
Figure 7: 4x2.5G Frame Format for non-FEC mode	29
Figure 8: 4x2.5G Frame Format for FEC mode.....	29
Figure 9: 8x2.5G Frame Format with FEC	30
Figure 10: Packet Information Message Mapping in Pre-amble.....	36
Figure 11: Bit order for Serial CRC Computation	38
Figure 12: CRC Calculation	39

Table of Tables

Table 1: USXGMII-M Options	6
Table 2: Definition of channel control information passes between links	11
Table 3: Auto-neg Message aligned to Ordered Set followed Data and then by 7-bit Control Characters	12
Table 4 Auto-neg Message aligned to Control Code followed by Ordered Set.....	13
Table 5: Auto-neg Message aligned to Ordered Set and NO Control Codes.....	13
Table 6: Definition of channel control information passed between links via UsxgmiiPCH [47:0]	16
Table 7: UsxgmiiPCH Placement when SOP in LANE 0.....	40
Table 8: UsxgmiiPCH Placement when SOP in LANE 4 – Part1	40
Table 9: UsxgmiiPCH Placement when SOP in LANE 4 – Part 2	40

Definitions

MII - Media Independent Interface: A digital interface that provides a 4-bit wide datapath between a 10/100 Mbit/s PHY and a MAC sublayer. Since MII is a subset of GMII, in this document, we will use the term “GMII” to cover all of the specification regarding the MII interface.

GMII- Gigabit Media Independent Interface: A digital interface that provides an 8-bit wide datapath between a 1000 Mbit/s PHY and a MAC sublayer. It also supports the 4-bit wide MII interface as defined in the IEEE 802.3z specification. In this document, the term “GMII” covers all 10/100/1000 Mbit/s interface operations.

LPI- Low Power Idle: An alternative form of idle signaling that is used by the MAC to indicate that the PHY may enter a low power state and signal this change of state to the link partner; and is used by the PHY to signal to the MAC that the link partner has entered a low power state. The functions are defined by IEEE 802.3az in IEEE 802.3 clauses 22, 24, 25 (for 100Mb/s); 35, 36, 40, 70 (for 1Gb/s); 46, 48, 49, 55, 71, 72 (for 10GE); and 78 (for overall descriptions).

SGMII- Serial Gigabit Media Independent Interface: A digital interface that provides a 1.25 Gbps serial dual-data-rate datapath between a 1000 Mbit/s PHY and a MAC sublayer. Refer to ENG-46158

QSGMII- Quad Serial Gigabit Media Independent Interface: A digital interface that provides a 5.0 Gbps serial datapath between four 1000 Mbit/s PHY ports and a MAC sublayer. Refer to EDCS-540103

USGMII - Universal Serial Gigabit Media Independent Interface: A digital interface that provides capability to carry multi-port/multi-rate serial datapath between PHY ports and a MAC sublayer using 8B/10B coding. Refer to EDCS 1155168

USXGMII - Universal Serial 10 Gigabit Media Independent Interface: A digital interface that provides capability to carry multiport/multi-rate serial datapath between PHY ports and a MAC sublayer using 64/66b coding.

1 Overview

USXGMII uses two data signals in each direction to convey frame data and link rate information between a single or multi-port PHY and the Ethernet MAC(s). This document specifies requirements for carrying multiple networks ports over a single PHY-MAC Interface. The maximum MAC/PHY SERDES speed is configured based on the maximum network port speed and number of network ports. Table 1, shows the different combinations of USXGMII speed, number of ports, rate adaptation, alignment marker and PCS requirements. A PHY can support appropriate USXGMII mode depending on number of network ports and SERDES power/cost optimization.

MAC-PHY IF Type	Number of Ports	Network Port Types	Replications – Lowest to Highest data speed	Port Mux/ Alignment Marker	PCS	SERDES Speed (Gbps)
10G-SXGMII	1	10M/100M/1G/2.5 G/5G/10G	1000/100/10/4/2/1	No	Clause 49	10.3125
5G-SXGMII	1	10M/100M/1G/2.5 /5G	500/50/5/2/1	No	Clause 49	5.15625
10G-DXGMII	2	10M100M/1G/2.5 G/5G	500/50/5/2/1	Yes	Clause 49	10.3125
5G-DXGMII	2	10M/100M/1G/2.5 G	250/25/2.5/1	Yes	Clause 49	5.156
20G-QXGMII	4	10M/100M/1G/2.5 G/5G	500/50/5/2/1	Yes	Clause 49	20.625
20G-DXGMII	2	10M/100M/1G/2.5 G/5G/10G	1000/100/10/4/2/1	Yes	Clause 49	20.625
2.5G-SXGMII	1	10M/100M/1G/2.5G	250/25/2.5/1	No	Clause 49	2.578125
10G-QXGMII	4	10M/100M/1G/2.5G	250/25/2.5/1	Yes	Clause 49	10.3125
20G-OXGMII	8	10M/100M/1G/2.5G	250/25/2.5/1	Yes	Clause 49	20.625

Table 1: USXGMII-M Options

All the options above can be implemented, but REFER to the RFI for the latest priority for a PHY. 10Mbps is a supported option unless stated otherwise in the RFI.

Other features:

- System Interface operates in full duplex mode only
- Ability to send PTP time stamp from PHY to MAC to improve accuracy/jitter on encrypted/non-encrypted PTP packet with MACSec is in the ASIC
- Hardware assisted auto-negotiation for all supported speeds
- Flexibility to add new features using Extension Field in the pre-amble

This document uses 10.3125Gbps SERDES to describe multiple network ports over single PHY-MAC interface, but same features applies to the other SERDES speeds and configurations unless stated otherwise.

Due to the high speed of operation, each of these signal pairs are realized as differential pairs thus optimizing signal integrity while minimizing system noise.

USXGMII leverages the 64/66b PCS defined in IEEE 803.2ae Clause 49. The PCS is unchanged and additional functionality is achieved via new “ordered set” mechanism defined by IEEE.

Figure 1 and 2, illustrates the high level view of Port ASIC and PHY with USXGMII interface respectively.

ASIC USXGMII block consists of

- N 10G and N 100M/1G MACs, were N is the number of ports supported on USXGMII-M interface.
- There are N PCS Clause 49 blocks, one per port with additional ordered sets
- Port MUX/De-Mux with Alignment Markers for port identification
- Rate adaption (Replication) for data rates less than maximum port speed, e.g. 100M/1G/2.5/5G
- Auto-neg messages usign 16-bit configuration word
- 10.325Gbps SERDES

For 20G-USXGMII or other speeds, ASIC consists of same functions but MAC, Rate Adaptation and speed changes – refer to table 1.

NOTE: If the incoming data (Rx PHY) has less then 8-byte of pre-amble include SFD, the PHY MUST add approriate bytes to ensure 7-bytes +SFD to meet XGMII/Clause 49 requirements. This applies to all data rates and PCH and non-PCH mode. See section 2.9 for more details.

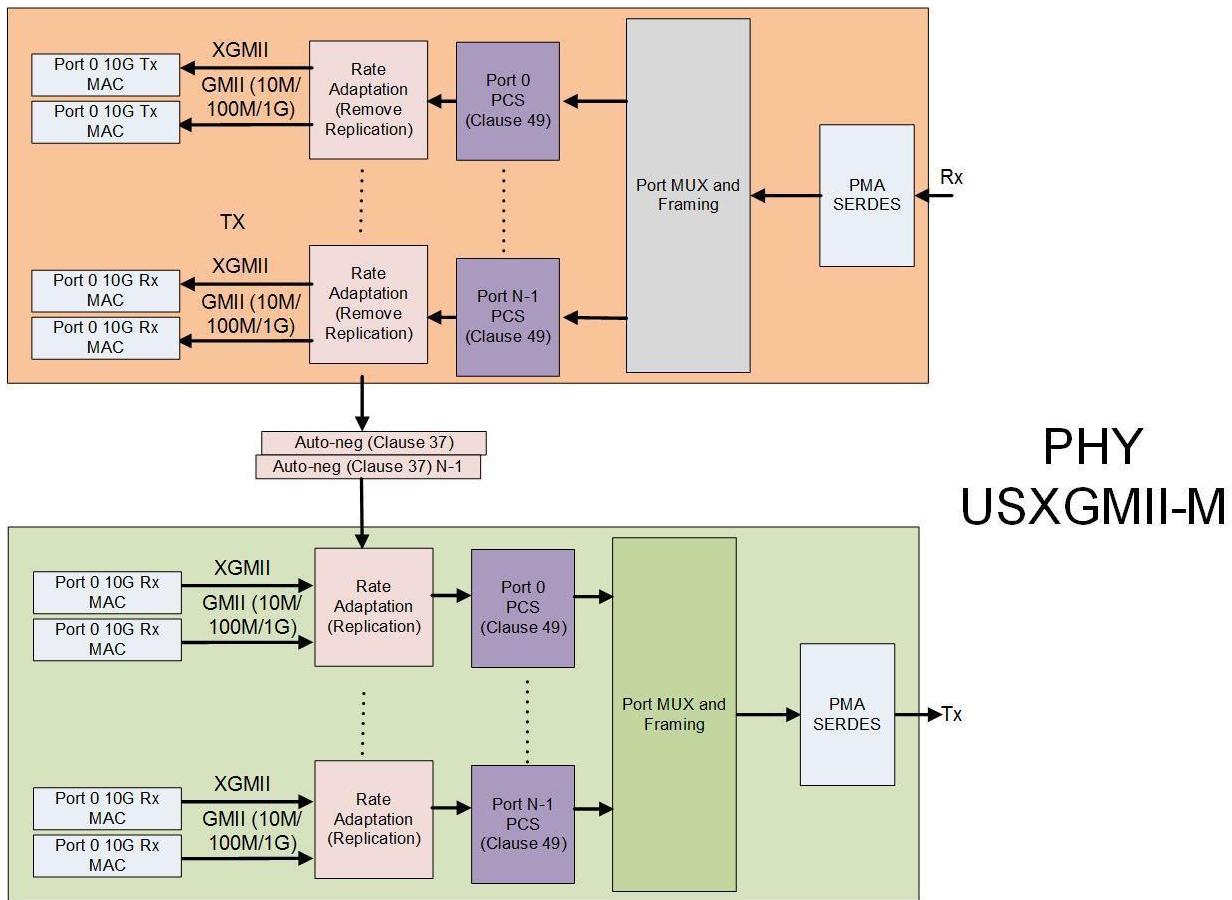
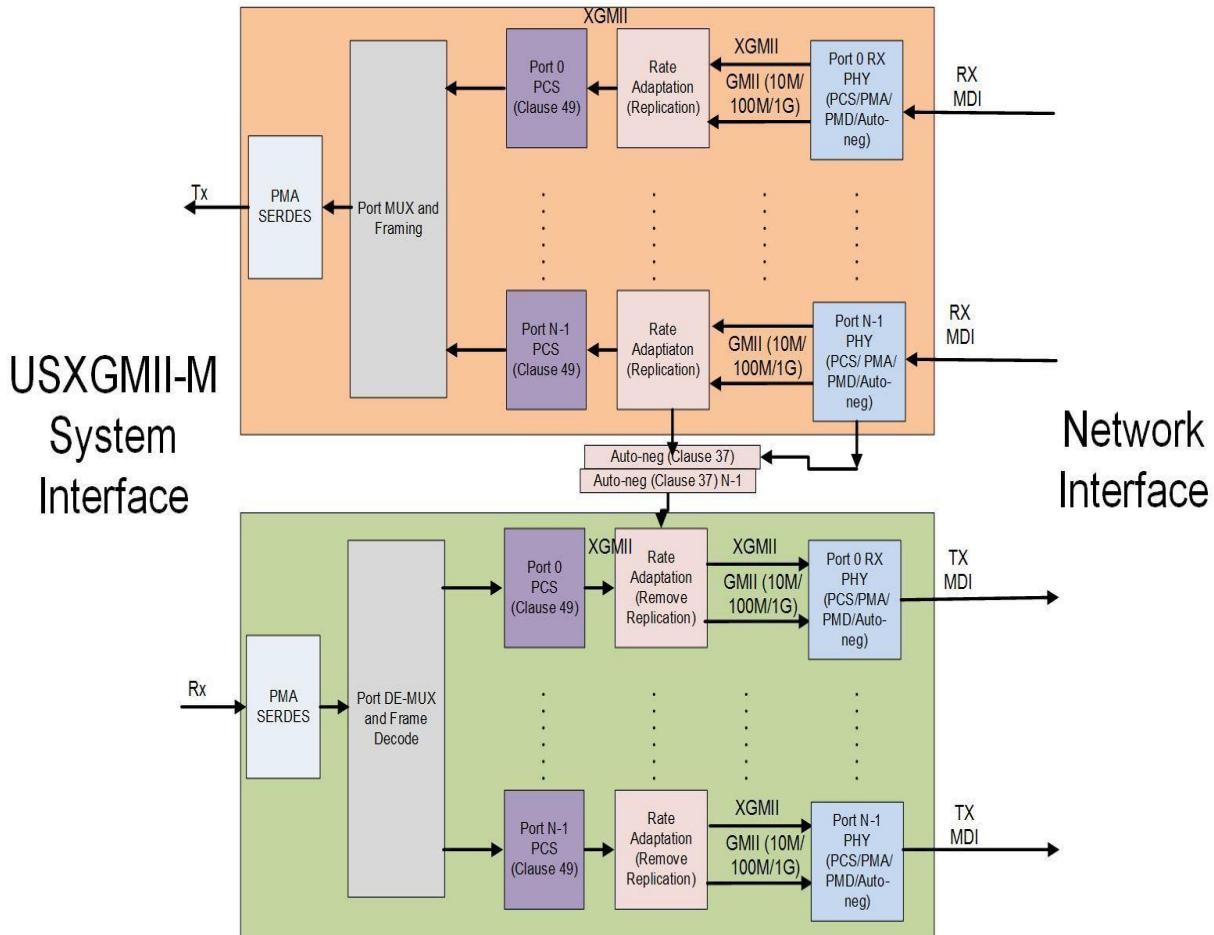


Figure 1: USXGMII Port ASIC Functional Block Diagram

PHY USXGMII block for 10G-USXGMII consists of

- N 10G and N 100M/1G MACs, were N is the number of ports supported on USXGMII-M interface.
- There are N PCS Clause 49 blocks, one per port with additional ordered sets
- Port MUX/De-Mux with Alignment Markers for port identification
- Auto-neg messages usign 16-bit configuration word
- Rate adaption (Replication) for data rates less than maximum port speed, e.g. 100M/1G/2.5/5G
- 10.325Gbps SERDES
- PHY PCS/PMA/PMD as appriate for network interface type

**Figure 2 USXGMII PHY Functional Block Diagram**

When network is 100M/1GE mode, the traditional GMII data signals (TXD/RXD), data valid signals (TX_EN/RX_DV), and error signals (TX_ER/RX_ER) are encoded into XGMII words. For 2.5G, 5G and 10G mode XGMII is used.

1.1 In-band Control and Status Signaling

In-band signaling is carried via ordered sets, which are defined in IEEE 802.3ae Clause 46 (XGMII). Ordered sets provide the capability to carry messages of 3 bytes; two such messages are defined in 802.3ae -- *LocalFault* (0x00, 0x00, and 0x01) and *RemoteFault* (0x00, 0x00, and 0x02). These messages are used in USXGMII to signal faults within the channel carrying the message. The current implementation defines new ordered sets to support following features:

- Auto-negotiation Message - Ability to send auto-neg message from PHY to/from port ASIC- similar to SGMII auto-neg message.
- Packet Information Message: Ability to send Packet Type, SubportID, and Extension Field Type/Extension Field

1.1.1 Auto-neg Mechanism

MDI based auto-neg is performed based on the media type. For example for 10GBASE-T is based on IEEE 802.3 Clause 55 and Clause 28. Rx PHY block sends MDI auto-negotiated parameter to Auto-neg block.

USXGMII Auto-neg mechanism is based on Clause 37 (Figure 37-6) plus additional management control to select USXGMII mode. The PHY must provide a USXGMII enable control configuration via MDIO.

On power reset, USXGMII enable bit is de-asserted (logic “0”) and system interface on Port ASIC and PHY must assume normal XGMII/XFI (Clause 46/49) operation for 10Gbps. Once USXGMII enable bit is enabled via MDIO, auto-neg operation should follow Clause 37-6 functions with following modifications:

- an_sync_status=fail changed to block_lock=false (restart Autoneg FSM)
- Autoneg FSM will restart whenever the (network) link changes is down
- rudi.invalid changed to idle received during an_restart, ability_detect, acknowledge_detect
- Link_timer changed to be configurable from 1msec to 2 msec in steps of 0.1msec.
- Ability_match and acknowledge_match as per figure 37-6

1.1.2 Auto-negotiation Message

USXGMII also utilizes ordered sets to convey channelization and auto-negotiation information. The *UsxgmiiChannelInfo* message carries the status pertinent to the channel as per Table 2.

Bit	UsxgmiiChannelInfo[15:0] sent from the PHY to the MAC	UsxgmiiChannelInfo[15:0] sent from the MAC to the PHY	Default
15	Link: 1 = link up, 0 = link down	Same as PHY to MAC	0
14	Reserved for Auto-Negotiation acknowledge	1	0
13	0: Reserved for future use	0: Reserved for future use	0
12	Duplex mode: 1 = full duplex, 0 = half duplex	Duplex mode: 1 = full duplex, 0 = half duplex	1
11:9	Speed: Bit 11, 10, 9 : This should be simple network port speed 000 = 10Mbps 001 = 100 Mbps 010 = 1000 Mbps 011 = 10 Gbps 100 = 2.5 Gbps 101 = 5 Gbps 110 = Reserved 111 = Reserved	Same as PHY to MAC Speed: Bit 11, 10, 9: 000 = 10Mbps 001 = 100 Mbps 010 = 1000 Mbps 011 = 10 Gbps 100 = 2.5 Gbps 101 = 5 Gbps 110 = Reserved 111 = Reserved	010
8	EEE capability: 1= supported, 0 = not supported	EEE capability: 1= supported, 0 = not supported	0
7	EEE clock stop capability: 1= supported, 0 = not supported	EEE clock stop capability: 1= supported, 0 = not supported	0
6:1	0: Reserved for future use	0: Reserved for future use	0
0	Set to 1	1	1

Table 2: Definition of channel control information passes between links

Like SGMII, *UsxgmiiChannelInfo* uses a 1.6ms link timer. Any change in the status of the link requires the PHY to re-signal *UsxgmiiChannelInfo* words until message is received with Auto-negotiation acknowledge bit set to 1 or for the duration of the link timer.

At the XGMII interface, Auto-neg message uses the format below:

TXC [3:0]	Lane 0 [31:24]	Lane 1 [23:16]	Lane 2 [15:8]	Lane 3 [7:0]
0x1	Character Control Code=0x9C	Config[15:8]	Config[7:0]	Opcode for auto-neg = 0x03 (Cisco specific)

The above definition is compatible with the Clause 46 definition of sequence ordered sets. Cisco specific opcode is defined with a value of 0x03 in order to meet Lane 3 >= 0x03 specified in Table 46-5 of 802.3 Clause 46.

The tables below shows the possible insertion of the auto-neg message based on different position of ordered sets. The value of ordered set equal to 0x0, D3/D7 and 0x03 identifies the message as auto-neg message.

Input Data		Syn c	Payload (63:0)									
	Bit Position	0 1	65									
Data Block Format			Block Type									
O0 D1 D2 D3/C4 C5 C6 C7		10	0x4B	Config [15:8]	Config [7:0]	0x03	O4=0x0 (4-bit)	C4 (7-bit)	C5 (7-bits)	C6 (7-bit)	C7 (7-bit)	

Table 3: Auto-neg Message aligned to Ordered Set followed Data and then by 7-bit Control Characters

Input Data		Syn c	Payload (63:0)								
	Bit Position	0 1	2	65							
Data Block Format			Block Type								
Co C1 C2 C3/O4 D5 D6 D7		10	0x2D	C0 (7-bit)	C1 (7-bits)	C2 (7-bit)	C3 (7-bit)	O4=0x0 (4-bit)	Config [15:8]	Config [7:0]	0x03

Table 4 Auto-neg Message aligned to Control Code followed by Ordered Set

Input Data		Syn c	Payload (63:0)								
	Bit Position	0 1	2	65							
Data Block Format			Block Type								
O0 D1 D2 D3/O4 D5 D6 D7		10	0x55	Config[15:8]	Config[7:0]	0x03	O0 (4-bit)	O4=0x0 (4-bit)	Config [15:8]	Config [7:0]	0x03

Table 5: Auto-neg Message aligned to Ordered Set and NO Control Codes

1.1.3 Packet Control Header

The *UsxgmiiPCH* message conveys information that may be needed on a packet-by-packet basis. This message is optional, and is only needed when the Usxgmii instance is using features that require this message for PTP Time Stamps, pre-emption etc.

PHY communicates with a port ASIC through Packet Control Header (PCH). PCH is 8 bytes and it replaces the preamble of the frame.

The details of PCH and fields are shown in *Figure 3*.

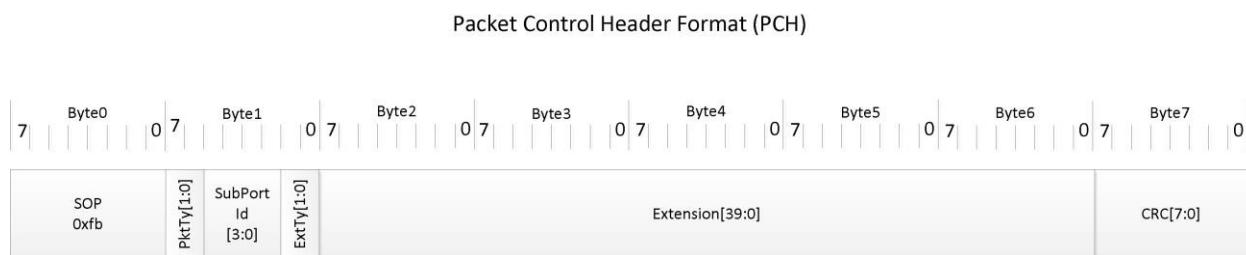


Figure 3: Packet Control Header (PCH) Format

Bit	UsgmiiPCH[47:0] sent from the PHY to the MAC	UsgmiiPCH[47:0] sent from the MAC to the PHY	Default
47:46	Packet type: 00: Ethernet Packet with PCH (packet information) 01: Ethernet packet, without PCH (packet information) 10: Idle Packet – Contains status data for a port – no packet data (set to all zero's) 11: Reserved	Packet type: 00: Ethernet Packet with PCH 01: Ethernet packet, without PCH (packet information) 10: Idle Packet – Contains status data for a port – no packet data (set to all zero's) 11: Reserved	0
45:42	Subport ID (Channel number) 3:0: Port 0 to 15	Subport ID (Channel number): MAC to PHY 3:0: Port 0 to 15	0
41:40	Extension Field Type 00: Ignore Extension Field 01: Extension Field contains 8-bit Reserved + 32-bit Timestamp 10: Extension Field: For devices that support Preemption aka Interspersing Express Traffic (IET) frames, otherwise	Extension Field Type 00: Ignore Extension Field 01: Extension Field contains Extension Field contains Tag/Signature associated with for time-stamp to be taken by the PHY 10: Extension Field: For devices that support Preemption aka Interspersing Express Traffic (IET) frames, otherwise	0

Bit	UsgmiiPCH[47:0] sent from the PHY to the MAC	UsgmiiPCH[47:0] sent from the MAC to the PHY	Default
	<p>Reserved. It also contains fields related to PTP Timestamp</p> <p>11: Reserved</p>	<p>Reserved. It also contains Tag/Signature associated with for time-stamp to be taken by the PHY.</p> <p>11: Reserved</p>	
39:0	<p>Extension Field</p> <p>Extension Type = 00: Extension Field is ignored</p> <p>Extension Type = 01:</p> <ul style="list-style-type: none"> • 39:32: Reserved. • 31:0: PTP Timestamp <p>Extension Type = 10: Preemption and PTP</p> <ul style="list-style-type: none"> • 39:38: Preemption (IET) Frame Type/PTP 00: High-priority frame (eMAC) 01: Low-priority preemptable frame or initial fragment (pMAC) 10: Low-priority intermediate fragment or last fragment (pMAC) 11: Verify/Respond frame • 37:36: Type of Verify/Respond frame 00: Verify frame received 01: Respond frame received 10/11: Reserved • 35:34: Frame Count is a 2-bit binary Fragment Count of low-priority • 33:32: Fragment Count is a 2-bit binary Fragment Count of low-priority Intermediate or Last Fragment • 31:0: PTP Timestamp <p>Extension Type = 11 are reserved</p>	<p>Extension Field</p> <p>Extension Type = 00: Extension Field is ignored</p> <p>Extension Type = 01:</p> <ul style="list-style-type: none"> • 39:32: Reserved • 31:16: Reserved • 15:0 Signature associated with the egress PTP Timestamp to be done in PHY <p>Extension Type = 10: Preemption and PTP</p> <ul style="list-style-type: none"> • 39:38: Preemption (IET) Frame Type/PTP 00: High-priority frame (eMAC) 01: Low-priority pre-emptable frame or initial fragment (pMAC) 10: Low-priority intermediate fragment or last fragment (pMAC) 11: Verify/Respond frame • 37:36: Type of Verify/Respond 00: Verify 01: Respond 10/11: Reserved • 35:34: Frame Count is a 2-bit binary of low-priority Fragmented frames • 33:32: Fragment Count is a 2-bit binary count of low-priority Intermediate or Last Fragment frame • 31: PTPTimestampEnable – when enabled, PHY adds egress PTP Timestamp to the associated frame • 30:16: Reserved • 15:0 Signature associated with the 	Refer to PHY spec for full details

Bit	UsgmiiPCH[47:0] sent from the PHY to the MAC	UsgmiiPCH[47:0] sent from the MAC to the PHY	Default
		egress PTP Timestamp to be done in PHY Extension Type = 11 are reserved	

Table 6: Definition of channel control information passed between links via UsxgmiiPCH [47:0]

Six byte UsxgmiiPCH message is sent in the pre-amble of the packet. It is inserted between SOP (0xfb) and Header CRC. Refer to section 2.6 for more details.

2 Implementation Specification

This section describes example of a possible implementation, but vendors can use other implementation as long as it meets required specification and interoperability. This section describes MII, GMII, XGMII byte replication, mapping to 64b data + 8b control USXGMII words for translation into a USXGMII stream via the Clause 49 PCS, Block Interleave, Alignment Markers for multiple port support and optional RS-FEC for speeds greater than 15G with channel loss >15dB.

Figure 4, show the block diagram of USXGMII Tx/Rx without RS-FEC and Figure 5 shows USXGMII with RS-FEC.

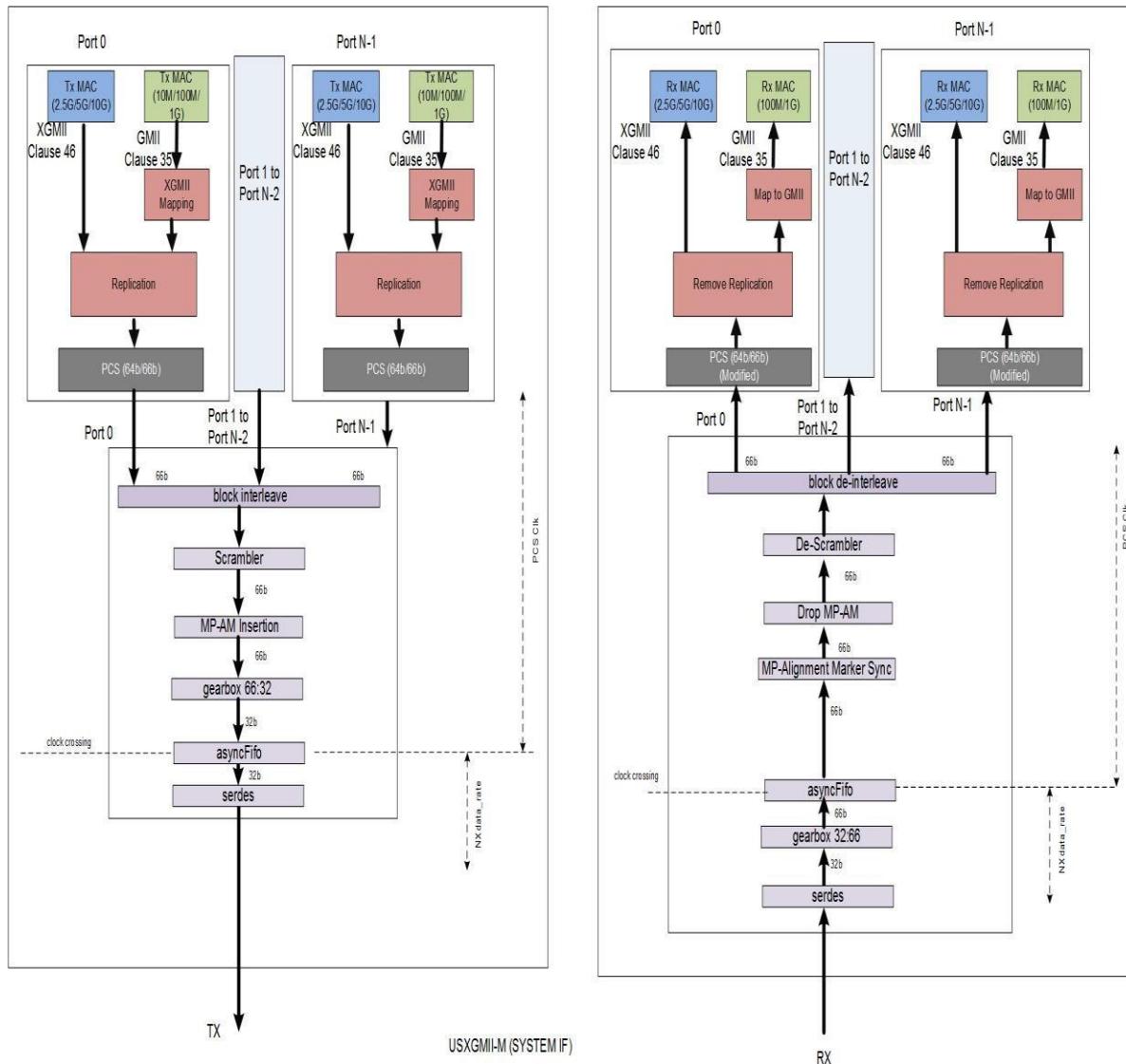


Figure 4: USXGMII TX/RX Implementation – Non FEC

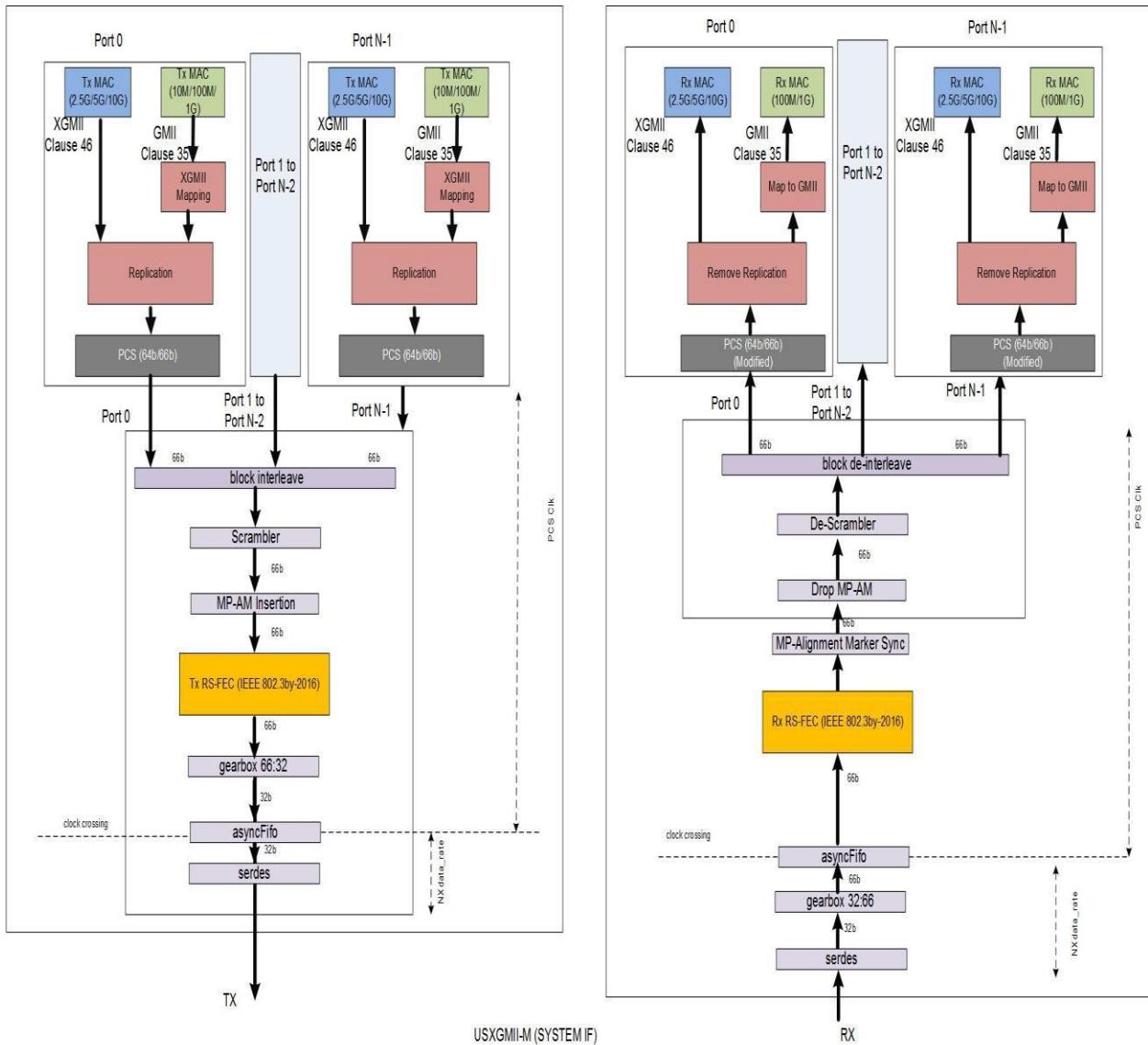


Figure 5: USXGMII TX/RX Implementation – With FEC

As shown in figure 5, since the MP-USXGMII alignment markers are very similar to codeword markers used by RS-FEC block (IEEE 802.3by-2016), they replace the codeword makers. For detailed specification, refer to Clause 108 of IEEE 802.3by-2016.

Following sections describes MP-USXGMII without FEC. FEC option is simply additional logic block without impact to MP-USXGMII implementation.

2.1 XGMII Mapping

XGMII naturally maps very cleanly into USXGMII words. Packet SOPs are already aligned into lane 0 or 4.

UsxgmiiChannelInfo is required to convey speed negotiation between PHY and MAC.

UsxgmiiPCH is required for feature using the Extension Field for features such as passing PTP Time Stamp between PHY and MAC Device. Extension Field could also be used for additional feature in the future.

2.2 GMII Mapping

GMII, being a byte-wide interface, requires additional translation to map into USXGMII words. The SOP Adjust feature is required. During USXGMII encoding, when an SOP is generated, if it is not naturally aligned on lane 0 or 4, it is left-shifted or right-shifted with appropriate IPG adjustment to the nearest legal SOP lane.

NOTE: If the incoming data (Rx PHY) has less than 8-byte of pre-amble include SFD (GMII), the PHY MUST add appropriate bytes to ensure 7-bytes +SFD to meet XGMII/Clause 49 requirements. This applies to all data rates and PCH and non-PCH mode. Refer to section 2.9 for more details.

2.3 MII Mapping

Same as GMII

2.4 Pause Frame Support

There is no PAUSE frame support at USXGMII level, port level pause frames are sent over USXGMII interface as payload. Channel level pause frame behavior should be as per 802.3 Annex 31B.

2.5 Auto-neg Mechanism

MDI based auto-neg is performed based on the media type. For example for 10GBASE-T is based on IEEE 802.3 Clause 55 and Clause 28. Rx PHY block sends MDI auto-negotiated parameter to Auto-neg block.

USXGMII Auto-neg mechanism is based on Clause 37 (Figure 37-6) plus additional management control to select USXGMII mode. The PHY must provide a USXGMII enable control configuration via MDIO.

On power reset, USXGMII enable bit is de-asserted (logic “0”) and system interface on Port ASIC and PHY must assume normal XGMII/XFI (Clause 46/49) operation for 10Gbps. Once USXGMII enable bit is enabled via MDIO, auto-neg operation should follow Clause 37-6 functions with following modifications:

- an_sync_status=fail changed to block_lock=false (restart Autoneg FSM)
- Autoneg FSM will restart whenever the link changes
- rudi.invalid) changed to idle received during an_restart, ability_detect, acknowledge_detect
- Link_timer changed to be configurable from 1msec to 2 msec in steps of 0.1msec.
- Ability_match and acknowledge_match as per figure 37-6

2.5.1 Transmitting Configuration Words

The transmit configuration words are enabled by software. Configuration words are transmitted during hardware auto-negotiation or software controlled negotiation on USXGMII interface.

During hardware auto-negotiation (see section 1.1.2) the configuration words are transmitted in accordance to IEEE 802.3 Clause 37 auto-negotiation. During auto-negotiation, the packet data or idles on the MII interface of Tx MAC are discarded and not sent to Tx PCS. Instead, the auto-negotiation block drives the configuration words to Tx PCS on the MII interface. However if Tx MAC transmits errors or link faults, the auto-negotiation is interrupted and error/link fault information is passed on the MII interface from Tx MAC to Tx PCS. Other than data/idles all other control words are pass-through e.g. errors and link faults.

Once auto-negotiation is completed, the Tx MAC output drives the Tx PCS using the MII interface. USXGMII Auto-neg is NOT re-started due to error/link fault conditions, Auto-neg is started due to network interface link changes or forced via software control.

Software may choose to use software controlled negotiation (section 2.5.7) instead of hardware auto-negotiation. This allows for diagnostic debug or “force link speed” under software control.

2.6 Rate Adaptation - Replicating Transmit Bytes

Once software programs the link speed, rate adaptation logic replicates the 4-byte words on the transmit MII interface based on 10G and 5G SERDES modes in the following manner for multi-port USXGMII. For single port USXGMII refer to EDCS 1150953.

2.6.1 4x2.5G Tx Mode (10G MAC/PHY Interface)

- SOP word (4-bytes) is transmitted only once; in remainder of the replicated words, SOP byte is replaced by 0xAA – 1 or 2 times (alternating between replication of 2 and 3 for 1G), 24 times (100M), 249 times (10M). This is compatible with the checks of the Tx PCS state machine, which disallows back-to-back SOPs.
- Each 4-byte word of the preamble and payload is replicated 2or3/50/500 times for 1G/100M/10M respectively.
- The 4-byte word containing EOP is transmitted once and for the remaining 1or2/24/499 4-byte words, EOP byte is replaced with idle (0x07). This is compatible with the checks of the Tx PCS state machine, which disallows back-to-back EOPs. This requires the receiver to search for EoP in 4-byte words, independent of Rx sampling.
- Each 4-byte idle word is replicated 2or3/25/500 times. This ensures that the data rate in case of 1G/100M/10M does not exceed the frame rate.
- All other sequence ordered sets and control sets are replicated 2or3/25/500 times, on a 4-byte word basis.

NOTE: From a data path point of view, replication is done before PCS (Tx) and sampling (aka de-replication) is done after the PCS (Rx) on 4-byte XGMII words.

The procedure for rate adaptation (replication) with multiple network ports over a single SERDES is more complex than the case of a single network port over a single SERDES. For multiple network ports over a single SERDES, the following parameters need to be taken into account:

- Maximum SERDES speed
- Number of network ports
- Maximum speed of a network port

The example below shows replication of a transmit frame on the 4-byte MII interface.

Transmit frame:

Without PCH: fb555555 555555d5 01020304 05060708 ... fd070707

With PCH: {0xfb PCH-Byte1 PCH-Byte2 PCH-Byte3 PCH-Byte4 PCH-Byte5 PCH-Byte6 PCH-Byte7(CRC)} 01020304 05060708 ...fd070707

Tx MII Interface Data [31:0] view of the transmit frame:

Without PCH: 555555fb d5555555 04030201 08070605 ... 070707fd

With PCH: {PCH-Byte3 PCH-Byte2 PCH-Byte1 0xfb} {PCH-Byte7(CRC) PCH-Byte6 PCH-Byte5 PCH-Byte4} 04030201 08070605 ...070707fd

2.6.1.1 Replication for 1G port over 4x2.5G (Tx)

Replicated frame on the 4-byte MII interface for 1G mode (x2.5 replication using ((2+3)/2)):

A Replication Counter on Tx, keeps track of x2 and x3 replication: when two 4-byte words are replicated the replication cycle is 0, and when three 4-byte words are replicated, the replication cycle is 1.

Once the link is up and link speed is known (via auto-negotiation), the transmitter set the replication counter to 0 for the first packet (SOP)

Case-1 with Replication Cycle = 0 at SOP

SOP (replication cycle = 0):

555555fb 555555aa (x2 without PCH)

{PCH-Byte3 PCH-Byte2 PCH-Byte1 0xfb} {PCH-Byte3 PCH-Byte2 PCH-Byte1 0xaa} (x2 with PCH) {

Preamble (replication cycle = 1):

d5555555 d5555555 d5555555 (x3 without PCH)

{PCH-Byte7(CRC) PCH-Byte6 PCH-Byte5 PCH-Byte4} {PCH-Byte7(CRC) PCH-Byte6 PCH-Byte5 PCH-Byte4} {PCH-Byte7(CRC) PCH-Byte6 PCH-Byte5 PCH-Byte4} (x3 with PCH)

Data (replication cycle = 0): 04030201 04030201 (first 4-bytes x2)

(replication cycle = 1): 08070605 08070605 08070605 (second 4-bytes x3)

...

EOP (replication cycle = 0): 070707fd 07070707 (x2) OR

(replication cycle = 1): 070707fd 07070707 07070707 (x3)

Case-2 with Replication Cycle = 1 (this can happen at next 4-byte boundary when replication is 3 or next packet SOP), example below shows SOP with replication of 3

SOP (replication cycle = 1):

555555fb 555555aa 555555aa (x3 without PCH)

{PCH-Byte7(CRC) PCH-Byte6 PCH-Byte5 PCH-Byte4} {PCH-Byte7(CRC) PCH-Byte6 PCH-Byte5 PCH-Byte4} {PCH-Byte7(CRC) PCH-Byte6 PCH-Byte5 PCH-Byte4} (x3 with PCH)

Preamble (replication cycle = 0):

d5555555 d5555555 (x2 without PCH)

{PCH-Byte7(CRC) PCH-Byte6 PCH-Byte5 PCH-Byte4} {PCH-Byte7(CRC) PCH-Byte6 PCH-Byte5 PCH-Byte4} (x2 with PCH)

Data (replication cycle = 1): 04030201 04030201 04030201 (first 4-bytes x3)

(replication cycle = 1): 08070605 08070605 (second 4-bytes x2)

...

EOP (replication cycle = 0): 070707fd 07070707 (x2) OR

(replication cycle = 1): 070707fd 07070707 07070707 (x3)

*NOTE: EoP byte can be in one 4-byte word, receiver must search for EoP in all replicated 4-byte words.

2.6.1.2 Replication for 10M/100M/2.5G over 4x2.5G (Tx)

Replicated frame on the 4-byte MII interface for 10M/100M/2.5G mode requires replication by 250/25/1 times respectively and Replication Counter is set to 0 for all cases of these speeds.

Following examples are shown using standard Pre-amble and SFD, which can be replaced with PCH if configured.

The following example shows a frame for 100M, where 4-byte MII words are replicated 25 times:

SOP (replication cycle = 0): 555555fb 555555aa 555555aa

555555aa 555555aa 555555aa 555555aa 555555aa 555555aa 555555aa 555555aa 555555aa
555555aa

Preamble (replication cycle = 0): d5555555 d5555555 d5555555 d5555555 d5555555 d5555555
d5555555 d5555555 d5555555 d5555555 d5555555 d5555555 d5555555 d5555555
d5555555 d5555555 d5555555 d5555555 d5555555 d5555555 d5555555 d5555555
d5555555

• • •

2.6.2 4x2.5G Sampling Received Bytes (Rx)

The logic receives SOP and depending on the software-programmed or hardware auto-negotiated link speed, it samples (aka de-replicates) on the 4-byte MII interface.

For 2.5G/100M/10M speeds,

- Receive logic samples the 1st 4-byte word containing SOP
 - For 2.5G, it samples the subsequent 4-byte words to form the Rx MII data: 2nd, 3rd, ...
 - For 100M, it samples the following 4-byte words: 26th, 51st, ...
 - For 10M, it samples the following 4-byte words: 251st, 501st, ...

For 1G speed, the receive logic is required to change the sampling rate due to 2/3 4-byte word replication at the transmitter. The receiver makes use of the fact Tx is sending fixed pattern of 2 4-byte word followed by 3 4-byte words. This cycle repeat every 5 4-byte words. On a SOP, receiver:

- Sample 1st 4-byte word,
 - followed by sampling 5th 4-byte words
 - followed by 1st -byte word
 - followed by sampling 5th 4-byte words etc... repeat this pattern
 - EoP detection is performed across all 4-byte words independent of 1st or 5th sampling.
Examples of 4-byte word with EoP are

- 07 07 07 fd
- 07 07 fd Da
- 07 fd Da Db
- fd Da Db Dc

Figure 6 show the two replication that can occur on an SOP detection and sampling point for each is the same.

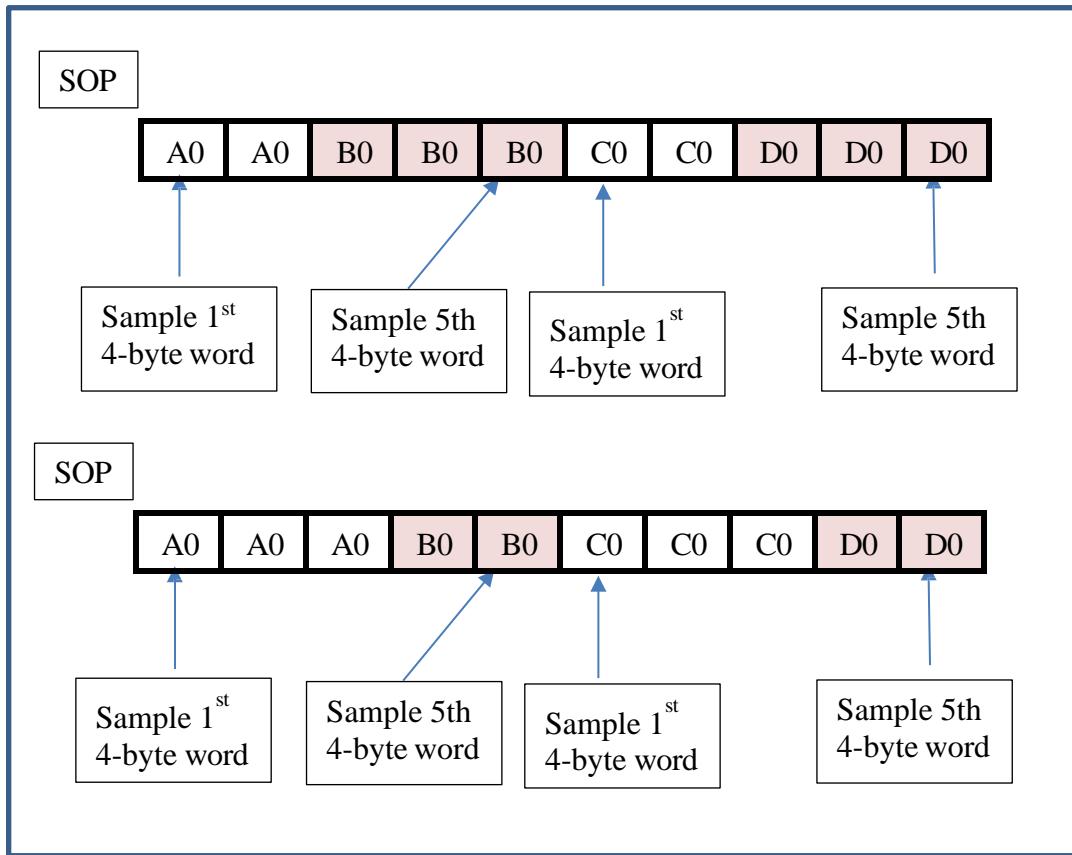


Figure 6: Receiver Data sampling for 1G over 2.5G

The sampled 4-byte MII word is then passed to Rx MAC, which processes the packet, checks CRC and other errors and ultimately passes the assembled frame to user interface.

Any configuration words, link fault words or errors received are similarly sampled at the MII interface as described above.

2.6.3 2x5G Tx Mode

- SOP word (4-bytes) is transmitted only once; in remainder of the 1 (2.5G), 4(1G), 49 (100M) and 249 (10M) 4-byte words are replaced by 0xAA. This is compatible with the checks of the Tx PCS state machine, which disallow back-to-back SOPs.
- Each word of the preamble and payload is replicated 2/5/50/500 times for 2.5G/1G/100M/10M respectively
- EOP is transmitted once and the remaining 1/4/49/499 bytes are idles. This is compatible with the checks of the Tx PCS state machine, which disallow back-to-back EOPs.
- Each idle byte is replicated 2/4/50/500 times. This ensures that the data rate in case of 2.5/1G/100M/10M does not exceed the frame rate.
- All other sequence ordered sets and control sets are replicated 2/4/50/500 times.

Following examples are shown using standard Pre-amble and SFD, which can be replaced with PCH if configured.

The example below shows replication of a transmit frame on the 4-byte MII interface.

Transmit frame:

```
fb555555 555555d5 01020304 05060708 ... fd070707
```

Tx MII Interface Data [31:0] view of the transmit frame:

```
555555fb d5555555 04030201 08070605 ... 070707fd
```

2.6.3.1 Replication for 2.5G over 2x5G

Replication for 2x5G is similar to single port USXGMII. Replication Counter is set to 0 for all cases, since replication is always 2x for 2.5G over 2x5G interface

Following examples are shown using standard Pre-amble and SFD, which can be replaced with PCH if configured.

Following shows an example for 2.5G

SOP (replication cycle = 0): 555555fb 555555aa (x2)

Preamble (replication cycle = 0): d5555555 d5555555 (x2)

Data (replication cycle = 0): 04030201 04030201 08070605 08070605 (x2)

...

EOP (replication cycle = 0): 070707fd 07070707 (x2)

2.6.4 Rx 2x5G Sampling Received Bytes

The logic receives SOP and depending on the software-programmed or hardware auto-negotiated link speed, samples every 2nd, 5th, 50th or 500th subsequent 4-byte word to form the Rx MII data for 2.5G, 1G, 100M, 10M network port speed respectively. This is then passed to Rx MAC which processes the packet, checks CRC and other errors and ultimately passes the assembled frame to user interface.

Any configuration words, link fault words or errors received are similarly sampled at the MII interface and the 2nd, 5th, 50th or 500th 4-byte word is passed to the Rx MAC for 2.5G, 1G 100M and 10M network port speed respectively.

For 5G/2.5G/1G/100M/10M speeds,

- Receive logic samples the 1st 4-byte word containing SOP
- For 5G, it samples the subsequent 4-byte words to form the Rx MII data: 2nd, 3rd, ...
- For 2.5G, it samples the following 4-byte words: 3rd, 5th, ...
- For 1G, it samples the following 4-byte words: 6th, 11th, ...
- For 100M, it samples the following 4-byte words: 51st, 101st, ...
- For 10M, it samples the following 4-byte words: 501st, 1001st, ...

2.7 Multiple Network Port over Single SERDES

2.7.1 Network Port Muxing

An USXGMII interface may be configured to carry multiple network ports over a single MAC/PHY interface. In such a configuration, the network ports are multiplexed and transmitted over a single SERDES.

In this configuration, 2, 4 or 8 network ports may be carried over a single SERDES. This is accomplished by multiplexing the 64/66b encoded blocks of each network port in a round-robin fashion and then scrambling the multiplexed stream. At the receiver, the 64/66b encoded blocks are first descrambled and then de-multiplexed to recover the underlying network port streams. Each of these network port over USXGMII consists of full independent PCS and MAC functionality.

The port mux/de-mux logic is required only when a single SERDES contains multiple network ports; for a single network port, the port mux/de-mux logic is not needed.

Port Mux/De-Mux logic consists of 64/66b encoded blocks and Alignment Marker to identify port0/Port Cycle 0. These markers are independent of RS-FEC. If RS-FEC option is supported then appropriate RS-FEC alignment markers should be supported as per IEEE 802.3 Clause 108. Note that the scrambler, gearbox and SERDES are required in all USXGMII modes, as shown in figure 4 and 5.

2.7.1.1 Quad-USXGMII

The following description discusses the example of 4x2.5G network ports over a single 10G SERDES:

- Create time slots based on the number of network ports: i.e. 4 time slots (a.k.a. port cycle)
- Divide SERDES speed by the number of network ports: $10G/4 = 2.5G$ per time slot
- Each time slot sends a 64-bit encoded block (6.4ns @ 10G) in a round-robin scheme for each port (Port 0, 1, 2, 3, 0, 1, 2, 3, ...)
- A port cycle is a group of time slots that repeats itself periodically, in this example, every 4 time slots. The number of port cycles between consecutive Alignment Markers is an integer value.
- Non_FEC: Alignment Markers identify time slot 0 (Port 0) and port cycle 0. Their frequency is programmable to support different number of port and SERDES speed options. For example:
 - 4x2.5G (10G), 2x5G (10G) and 2x2.5G (5G) require markers to be inserted at “5 x number of port cycles” required to meet replication factors of 2.5, 5, 25 and 50 for 1G/100M network ports. Standard marker insertion of 16384 block is

NOT multiple of 5. For this reason default markers insertion is every 16400 64-bit encoded blocks. This results in the port cycles of 4100 as shown in figure 7.

- NOTE: Multiple of 5 requirement is not necessary, but for non-FEC mode we are not changing specification to avoid issue with devices already available.

Figure 7 shows the 4 time slots and the network port sequence for non-FEC mode. Figure 7 applies to 2, 4 and 8 port frame format.

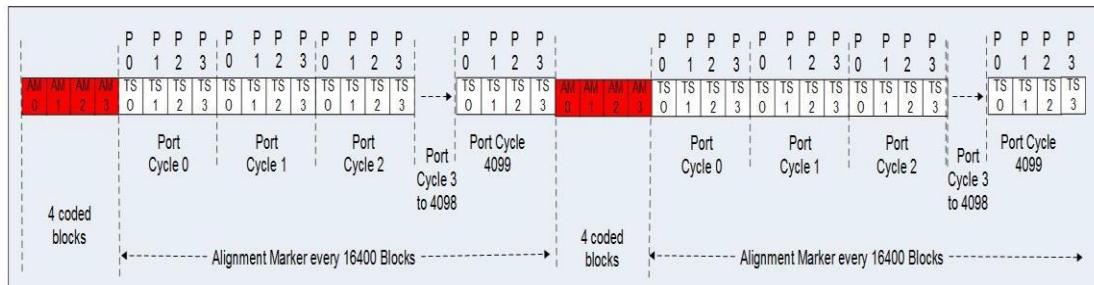


Figure 7: 4x2.5G Frame Format for non-FEC mode

- With FEC: FEC requires multiple of 80 blocks, the MP-AM insertion will need to be programmed: $80 * n - 4$, where n is an integer and 4 represents the 4 MP-AM. Figure 8 shows frame format with $n=200$ for 2 and 4 port configuration and port 0 is always after MP-AM marker.

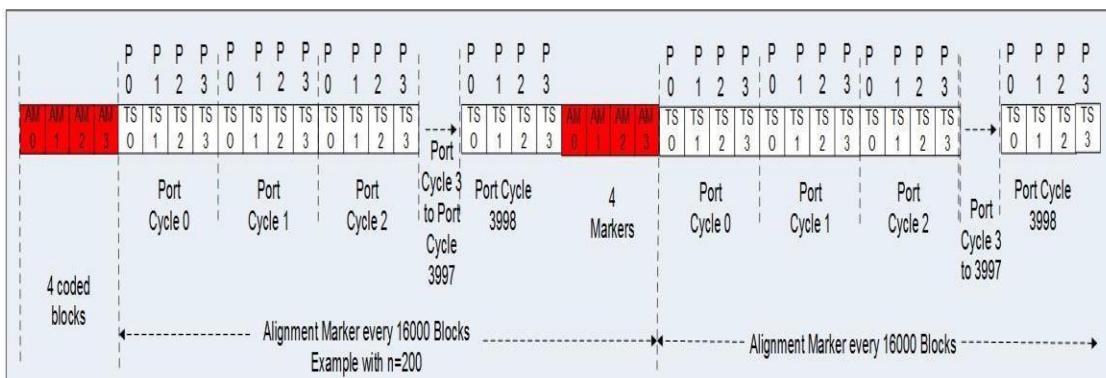


Figure 8: 4x2.5G Frame Format for FEC mode

- Port Cycle is set after initialization, depending on the number of ports (time slots). The speed of any port can change in response to the network port speed. For example, in a 4x2.5G configuration, one or more network ports can change to 1G speed. This is accomplished without changing the Alignment Markers.
- Alignment Markers are used to mark the sequence and start of each Port Cycle. If the number of network port changes, Alignment Markers will need to be re-initialized.
- Each port within a time-slot is independent of other ports in terms of speed and replication.

2.7.1.2 Octal-USXGMII

The following description discusses the example of 8x2.5G network ports over a single 20G SERDES:

- Create time slots based on the number of network ports: i.e. 8 time slots (a.k.a. port cycle)
- Divide SERDES speed by the number of network ports: $20G/8 = 2.5G$ per time slot
- Each time slot sends a 64-bit encoded block (3.2ns @20G) in a round-robin scheme for each port (Port 0, 1, 2, 3, 4, 5, 6, 7, 0, 1, 2, 3, 4, 5, ...)
- A port cycle is a group of time slots that repeats itself periodically, in this example, every 8 time slots. The number of port cycles between consecutive Alignment Markers is an integer value for non-FEC but is NOT an integer with FEC (blocks of 257-bit). To have same behavior for non-FEC and FEC modes, port number will be 0 or 4 after the markers
- Alignment Markers identify time slot 0 (Port 0) or 4 after the markers as in figure 9.

Figure 9 shows the frame format for 8 port frame format with $n=200$. With 8 ports, port number after the MP-AM alternate between 0 and 4 to maintain $80*n - 4$ blocks. See section 2.7.4 for details.

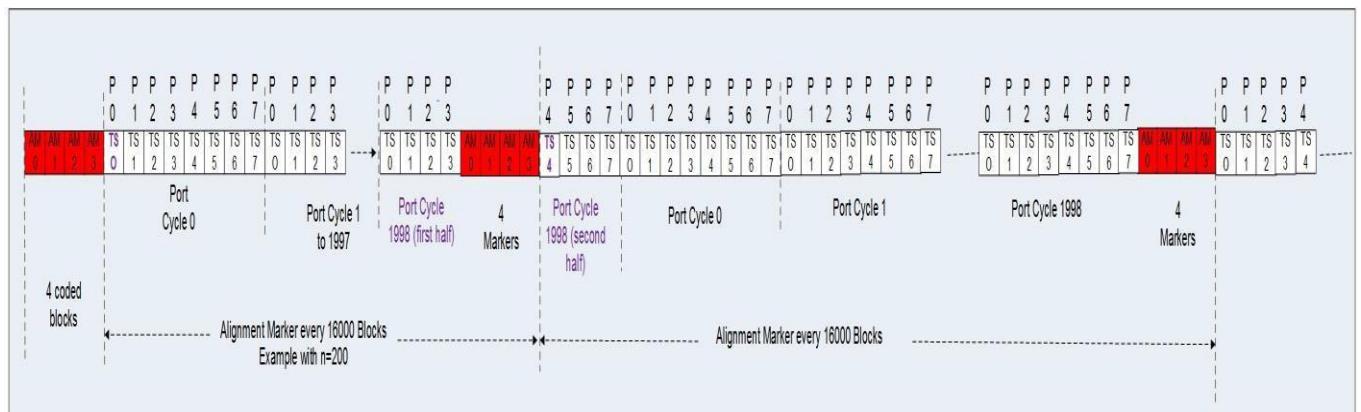


Figure 9: 8x2.5G Frame Format with FEC

- Port Cycle is set after initialization, depending on the number of ports (time slots). The speed of any port can change in response to the network port speed. For example, in a 8x2.5G configuration, one or more network ports can change to 1G speed. This is accomplished without changing the Alignment Markers.
- Alignment Markers are used to mark the sequence and start of each Port Cycle. If the number of network port changes, Alignment Markers will need to be re-initialized.
- Each port within a time-slot is independent of other ports in terms of speed and replication.

2.7.2 Tx Mux – Alignment Markers in non-FEC mode

Four Alignment Markers are used to convey the position of the 1st port block after the alignment marker. The alignment marker is followed with blocks of port numbers as indicated below – the sequence of blocks after the alignment marker is 0, 1, ..., N-1.

In non-FEC mode, the 4 Multi-Port Alignment Markers (MP-AM) are same as the alignment markers used to construct the codeword marker (cwm) in Clause 108 of IEEE 802.3by-2016. (IEEE 802.3by-2016 re-uses makers specified in IEEE-802.3-2015 Clause 82 Table 82-2 PCS Lane Number 0 as alignment marker 0 and Table 82-3 PCS Lane Number 1, 2 and 3 as alignment marker 1, 2 and 3 respectively.)

MP-USXGMII is re-uses IEEE-802.3by-2016 (Clause 108) alignment makers with following modifications to create MP-AM alignment markers.

1. The BIP3 field in each MP-AM, originally 0x33, is replaced by 0x00 to indicate that the block after the four Alignment Markers corresponds to Port 0 (0x00) for 2 and 4 port modes. For 8 port mode BIP filed in each MP-AM will be 0 or 4 to provide integer blocks for marker insertion. Similarly, the BIP7 field in each MP-AM, originally 0xCC, is the bit-complement of BIP3 field, i.e. 0xFF.
2. The system software programs the number of ports muxed. This number is a power of 2 integer.
3. The transmitter will maintain an 8-bit counter to control the muxing of blocks of different ports. This free-running counter will increment from 0 to N-1, then wrap to 0 and continue incrementing, where N is the number of ports muxed (programmed by system software, as listed above).
4. The alignment marker insertion frequency SHOULD BE PROGRAMMABLE. macTx will perform IDLE deletion to make room for MP-AM markers and it will need to be configurable to support the programmable alignment marker frequency as shown in figure 7.

Block periodicity = programmable alignment maker frequency + 4 blocks.

2.7.3 Rx Demux in non-FEC mode

The following points describe the Rx Demux function:

1. PhyRx “Block Sync”. When there is no block lock, Rx PCS module will send local fault to Rx MAC. Block Sync is present only in non-FEC mode. The programmable AM Frequency indicates the number of data blocks. So block periodicity is AM Frequency + 4 blocks as shown in figure 7.
2. After reset, the RxDemux counter’s max value is programmed by system software. If N ports are demuxed (value programmed by system software), then the RxDemux counter will be free running, incrementing from 0 to (N-1), wrapping back to 0 and then continuing to increment again.

3. The BIP3 and BIP7 fields of MP-AM are not checked. Otherwise they are handled similar to the alignments markers for 40G.
4. The RxDemux counter will determine the demux steering of the received 64/66b blocks.
5. The alignment markers will be dropped prior to the macRx module.
6. For each of the 4 MP-AM received, the Port Number field and its complement field pair will be compared. If match is successful, the RxDemux counter will be initialized to indicate that the block received after the 4 MP-AM will correspond to Port 0 for 2 and 4 port mode, while 0 or 4 for and 8 port mode. If there is a mismatch, it will indicate an error condition which should be logged and should follow the rules of AM lock state machine figure 82-12 in IEEE 802.3-15.
7. **Design allows optional feature to add RS-FEC (e.g. Clause 108 KR4 RS 528:514) and associated markers..** See section 2.7.4

2.7.4 Tx Muxing in FEC mode

In FEC enabled mode, the 4 Multi-Port Alignment Markers (MP-AM) are same as the alignment markers used to construct the codeword marker (cwm) in Clause 108 of IEEE 802.3by-2016. (IEEE 802.3by-2016 re-uses makers specified in IEEE-802.3-2015 Clause 82 Table 82-2 PCS Lane Number 0 as alignment marker 0 and Table 82-3 PCS Lane Number 1, 2 and 3 as alignment marker 1, 2 and 3 respectively.)

Since the FEC function requires multiple of 80 blocks, the MP-AM insertion will need to be programmed: $80 * n - 4$, where n is an integer and 4 represents the 4 MP-AM markers.

MP-USXGMII is re-uses IEEE-802.3by-2016 (Clause 108) alignment makers with following modifications to create MP-AM alignment markers

1. The BIP3 field in each MP-AM, originally 0x33, is replaced by the port number whose block immediately follows the set of MP-AM. Consider the examples below, when marker insertion is every 16000 blocks:
 - a. When the number of muxed ports is either 2 or 4, the block immediately following the set of 4 MP-AM is that of port 0 and the BIP7 field is 0xFF (the bit-complement of BIP3 field). See figure 8.
 - b. When the number of muxed ports is 8, the block immediately following the set of 4 MP-AM alternates between 0 and 4. So the block immediately following MP-AM set is that of port 0 (so BIP3 = 0x00 and BIP7 = 0xFF) and after the next set of 4 MP-AM it is of port 4. (so BIP3 = 0x04 and BIP7 = 0xFB). See Figure 9.
2. The system software programs the number of ports muxed. This number is a power of 2 integer.
3. The transmitter will maintain an 8-bit counter to control the muxing of blocks of different ports. This free-running counter will increment from 0 to N-1, then wrap to 0 and continue incrementing, where N is the number of ports muxed (programmed by system software, as listed above). For the example above when a maximum of 8 ports are supported, a 3-bit counter is sufficient.

The alignment marker insertion frequency SHOULD BE PROGRAMMABLE. macTx will perform IDLE deletion to make room for MP-AM markers and it will need to be configurable to support the programmable alignment marker frequency.

2.7.5 Rx Demux in FEC mode

The following points describe the Rx Demux function:

1. PhyRx “CWM Lock”. In the absence of cwm lock, Rx PCS will send local fault to Rx MAC. The programmable AM Frequency indicates the number of data blocks. Since block periodicity needs to be a multiple of 80 for FEC, block periodicity = AM Frequency - 4 blocks, as shown in Fig. 8 and 9.
2. After reset, the RxDemux counter’s max value is programmed by system software. If N ports are demuxed (value programmed by system software), then the RxDemux counter will be free running, incrementing from 0 to (N-1), wrapping back to 0 and then continuing to increment again.
3. The RxDemux counter will determine the demux steering of the received 64/66b blocks.
4. The BIP3 and BIP7 fields of MP-AM are checked in the following manner: any 3 out of 4 BIP3 fields AND any 3 out of 4 BIP7 fields must match. The RxDemux counter is updated only when this condition is satisfied: the BIP3 field indicates the port number of the block immediately after the set of 4 MP-AM markers.
If the match specified above is not satisfied, the RxDemux counter keeps incrementing as before and it is not updated in response to the 4 MP-AM markers received.
5. The alignment markers will be dropped prior to the macRx module.

2.7.6 Port De-Mux Framing

After Rx alignment maker and de-mux synchronization, data 64-bit coded data is sent to PCS Decode. Refer to section 2.6 for removing replicated data on 4-byte boundary.

2.7.7 Clocking

The auto-negotiation block runs on PCS/MAC clock.

2.7.8 Hardware Auto-negotiation Programming Sequence

Software will follow the programming sequence below to achieve auto-negotiation with the link partner. If hardware auto-negotiation doesn't converge to a link speed (step 8 below), software may timeout and restart hardware auto-negotiation. Following uses 10G as an example.

1. Reset.
2. 10G link up. At this time the link is active at 10G, no replication or sampling is being done. Device can receive and transmit packets on the native 10G link.
3. Programs the link timer registers, if desired. Otherwise the default value is used.
4. Programs transmit configuration word.
5. Enables hardware auto-negotiation and disables software negotiation in a register.
6. Device starts transmitting configuration words in accordance to the auto-negotiation state diagram of IEEE 802.3z Clause 37.
7. Device transitions through the auto-negotiation states in hardware, without any software involvement. Since the link partner may start transmitting configuration words before, the reception and transmission of configuration words happen independently.
8. Upon completion of auto-negotiation, hardware sets the link speed register. At this point Device starts transmitting the replicated idles and packets and sampling the received data. Upon receiving SOP, Device recalibrates the sampling start point.
9. If 10G link is lost or regained, the software is expected to disable auto-negotiation and re-enable auto-negotiation.

2.7.9 Port ASIC Software Controlled Negotiation Programming Sequence

If software wishes to disable hardware auto-negotiation and manage the negotiation process itself, it follows the sequence below.

If software wishes to read the PHY directly (using MDIO, for example) it may skip steps 4 through 6 below and program the Link Status register directly.

1. Reset.
2. 10G link up. At this time the link is active at 10G/5G, no replication or sampling is being done. Device can receive and transmit packets on the native 10G/5G link.
3. Disable hardware auto-negotiation and enables software negotiation in Auto-neg Control register.
4. Programs transmit configuration word in Tx Config Control register and enables transmission of configuration words.
5. Check for the valid bit of Rx Config Word register. When set, it reads the configuration word received.
6. Enable transmission of idles in Tx Config Control register.
7. Programs the link speed and negotiation complete fields in Link Speed Control Status register. At this point Port ASIC starts transmitting the replicated idles and packets and

sampling the received data. Upon receiving SOP, Port ASIC recalibrates the sampling start point.

8. If 10G/5G link is lost or regained, the software is expected to restart the negotiation sequence.

2.8 USXGMII Packet Control Header Implementation

When Packet Control Header is used, the PHY and ASIC must use fixed offset of 7-bytes from 0xfb to match 0xd5 on the 8th byte to detect start of packet data. The data after 0xfb and before 0xd5 is the Packet Information Message and packet data is after 0xd5.

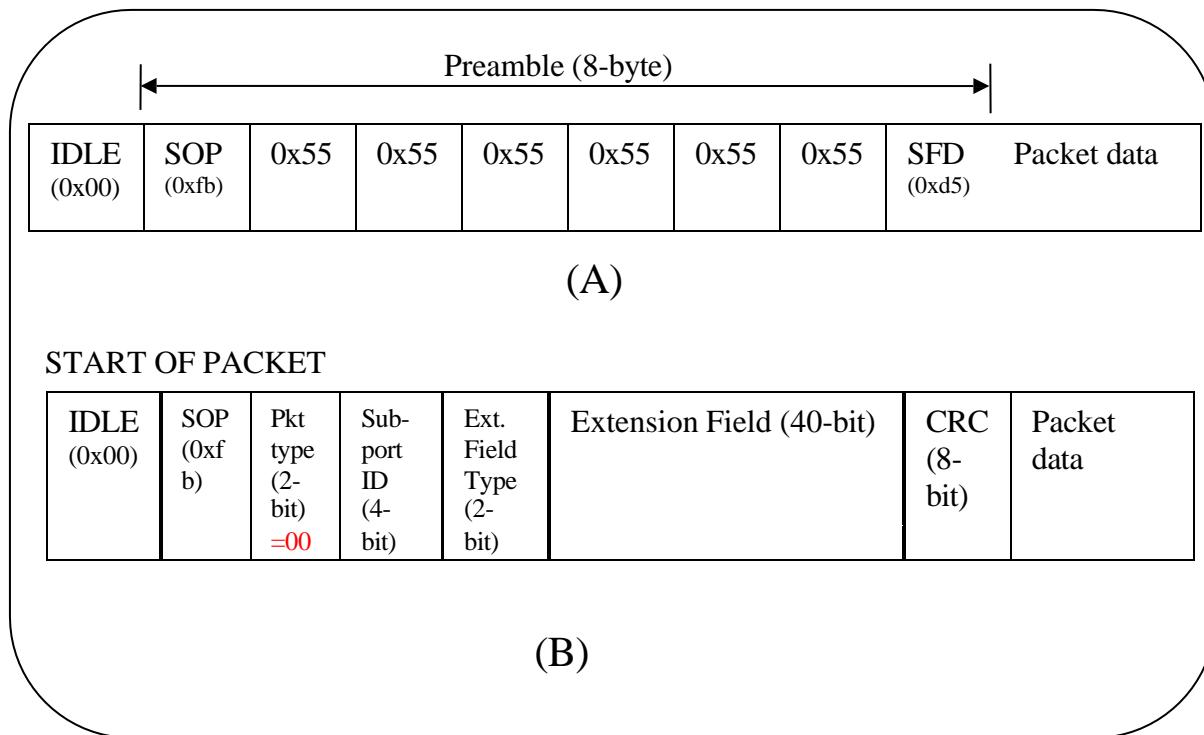


Figure 10: Packet Information Message Mapping in Pre-amble

2.8.1.1 Start of Packet (SOP)

The SOP symbol indicates the start of packet. The value of SOP MUST be 0xFB (Hex) as per IEEE 802.3 standard.

2.8.1.2 Packet Type

Packet Type identifies type of packet:

- 00: Ethernet Packet with status header
- 01: Ethernet packet, no Status Header (packet information)
- 10: Idle Packet – Contains status data for a port – no packet data
- 11: Unused (Reserved)

2.8.1.3 Sub-portID

Sub-portID is provided as additional information to be used at layer protocol layers. For multiple port PHY, Sub-port ID is same as network port number on PHY e.g. 0, 1, 2, 3. For single port, Sub-port ID is 0.

2.8.1.4 Extension Field Type

This 2-bit field type defines the content of Extension Field. Refer to appropriate PHY for the definition.

2.8.1.5 Extension Field

This 40-bit Extension Filed carries data defined by Extension Field Type, e.g. Time-Stamp and Pre-emptions. Refer to PHY specification for detail creating the PCH header.

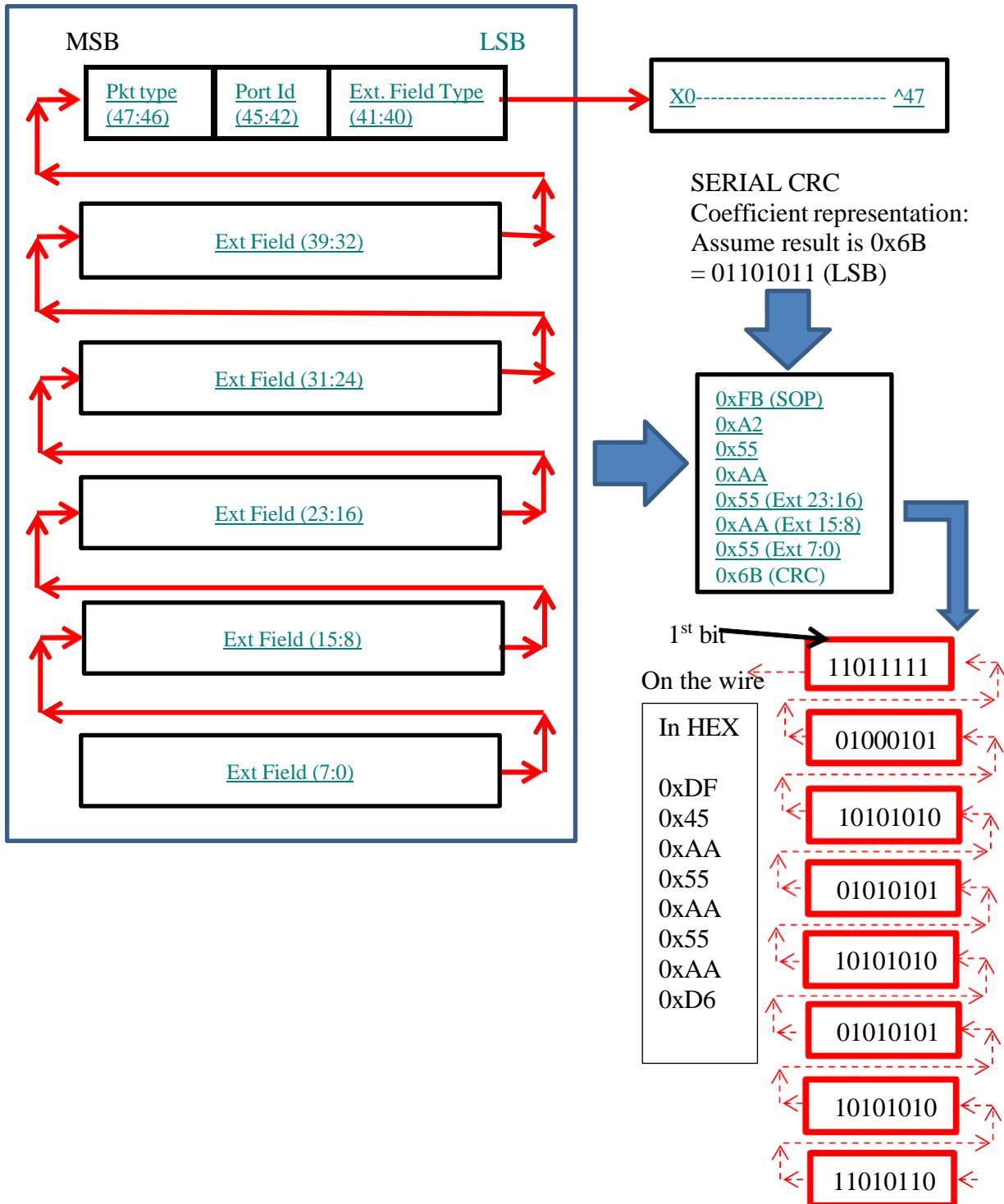
2.8.1.6 CRC computation

Header CRC is computed using CRC-8 algorithm with polynomial x^8+x^2+x+1 over the 6-bytes header data (inclusive of Packet Type to Extension Field). The initial value of CRC-8 computation is 0. The CRC remainder is exclusive-or'ed with 0x55 to header.crc value. The result is copied to the header.crc field on header generation and compared to header.crc on header reception. Header CRC generation is based on ITU-T I.432

Readers must pay attention to the differences in bit numbering conventions and bit CISCO Systems Confidential Information [Page 15]. CDL Working Group CDL Specification 17 May 2002 transmission order between ITU-T standard I.432 and this specification.

The I.432 standard numbers the LSB of a multi-bit field as zero and bits within a byte are transmitted MSB first. See the Section "Definitions" for bit numbering conventions and bit transmission order applicable to this specification.

The Header CRC is an 8-bit sequence calculated over 6 bytes PCH but excluding the SOP symbol and the Header CRC. The 48-bits long relevant portion of the CDL header is taken to represent a polynomial of order 47. The coefficients can only have the value 0 or 1. The least significant bit of the first byte of the header represents the coefficient of the highest order (x^{47}) term. The polynomial operations are performed modulo-2.

**Figure 11: Bit order for Serial CRC Computation**

Other examples for CRC:

PCH[47:0] = 0x2910_4602_7710. PCHCRC = 0x0B

PCH[47:0] = 0x2910_4602_7720. PCHCRC = 0x07

PCH[47:0] = 0x2910_4602_7730. PCHCRC = 0x0F

PCH[47:0] = 0x2910_4602_7740. PCHCRC = 0x01

The Header CRC MUST be recomputed whenever any of the fields in the header is changed and MUST be passed transparently whenever the fields of the header do not change.

The receiver in to-CDL-network direction must perform HCRC and maintain count of packets with failed HCRC for performance monitoring purposes. CDL information MUST NOT be extracted from, made use of or inserted into packets that failed HCRC. Such packets MAY also be discarded by the transmitter in from-CDL-network direction.

The linear feedback shift register below can be used to calculate the 8-bit CRC in a bit serial fashion. 8, 16 or 32 bit parallel implementations can be deduced from the bit serial implementation.

NOTE: The feedback on this implementation is different than 32-bit frame CRC and Clause 55

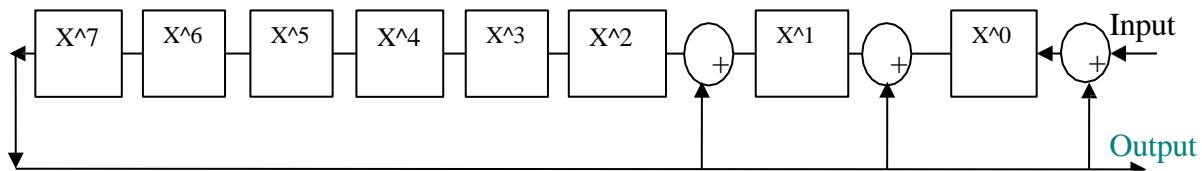


Figure 12: CRC Calculation

2.8.1.7 Mapping Packet Information Massage to 64/66b PCS Layer

UsxgmiiPCH message is carried in the preamble of the SOP.

Input Data		Sync	Payload (63:0)										
	Bit Position	0 1	2	65									
Data Block Format			Block Type										
S0 D1 D2 D3/D4 D5 D6 D7	10	0x78	<i>UsxgmiiPCH(47:0)</i>										CRC (8bit)

Table 7: UsxgmiiPCH Placement when SOP in LANE 0

Input Data		Sync	Payload (63:0)										
	Bit Position	0 1	2	65									
Data Block Format			Block Type										
O0 D1 D2 D3/S4 D5 D6 D7	10	0x66	0x00	0x00	0x00	O0	O4	<i>UsxgmiiPCH(47:24)</i>					

Table 8: UsxgmiiPCH Placement when SOP in LANE 4 – Part1

Input Data		Sync	Payload (63:0)										
	Bit Position	0 1	2	65									
Data Block Format			Block Type										
D0D1D2D3D4D5 D6D7	0 1	<i>UsxgmiiPCH (23:0)</i>			CRC (8-bit)	Pkt Data0	Pkt Data1	Pkt Data2	Pkt Data3				

Table 9: UsxgmiiPCH Placement when SOP in LANE 4 – Part 2

2.9 PHY Implementation

This is vendor dependent, but basic functions are described. Figure 2 show major Rx and Tx blocks required in PHY to support USXGMII features.

2.9.1 Rx PHY Block

Rx PHY consists of:

- Media Dependent Interface – 802.3an for Copper
- Tx Rate Adaptation – Rate adaptation for 100M/1G/2.5G and or 5G, UsxgmiiPCH and GMII to XGMII mapping
- Tx PCS – 802.3 – 2008 Clause 49 (10GBASE-R) with modifications
- Tx System Interface - PMA/PMD SERDES (10.3125Gbps)

For 10G/5G/2.5G data rates, MDI interface provides XGMII (Clause 46) interface to Tx Rate Adaptation block, while in 100/1000M rate, data/control information is provided over GMII (Clause 35).

Refer to section 2.6.1 for replications/deletions for each MAC-PHY rate based on network port combinations.

NOTES:

- *When MDI is in auto-neg, PHY to send data from user programmable register. Data could be IDLE or Local Fault.*
- *When txer=1 and txen=1 from mii copper receiver interface, xgmii will send control character /E/ which is 0xfe*
- *Carrier extension for 1GE is not supported (No Half-Duplex support)*

2.9.1.1 MDI: Receive IPG and Pre-amble Processing on MDI

PHY must be able to meet IPG and Pre-ambles received on the MDI based on the speed:

- 10M/100M/1G/2.5G/5G with Clause 36 (8b/10b coding) – USXGMII PHY is required to receive Ethernet packets from remote PHY using 8b/10b PCS coding with IPG of 8-bytes and 7-byte (6 + SFD) Preamble as per IEEE 802.3. But legacy 10M/100M/1G implementation are not always following IEEE 802.3 IPG and pre-amble rules, and hence USXGMII PHY needs to be more flexible in terms of receiver design. It is possible the legacy equipment may send few packets with IPG less than 8-bytes and Preamble of more than 8-bytes. USXGMII PHY should avoid exact match of 8-byte IPG and 8-byte

of pre-amble. This can cause overflow of the PHY which should be handled gracefully and provide overflow status accessible via MDIO/management interface

- 802.3Clause 46/49 and 126: PHY must be able to receive minimum of 5-byte IPG (4-byte Idle + T) and pre-amble of 8-bytes

2.9.1.2 USXGMII Transmit to MAC (ASIC)

PHY must be able to receive a packet with minimum of 5-byte IPG. For debug, it's recommended that the PHY detect number of IPG bytes received and maintain overflow status of its transmit buffer.

2.9.2 Tx PHY Block

Tx PHY consists of (from Port ASIC to PHY):

- Rx System Interface - PMA/PMD SERDES (10.3125Gbps)
- Rx PCS – Clause 49 (10GBASE-R) with modifications
- Rx Rate Adaptation – Rate adaptation for 10/100/1000M and XGMII to GMII mapping
- Media Dependent Interface – e.g. 802.3an for Copper
- PHY MUST ensure that MDI meet Tx IPG for 100M/1G and 2.5G/5G and 10G. IPG for 100M/1G must be minimum of 12-bytes, while minimum of 9-bytes for 2.5G/5G/10G.

Phys Rx System Interface from Port ASIC receives data using clause 49 as shown in figure 2. Data from Rx PCS is USXGMII format. In 100M/1000M, Rx Rate Adaptation removes replicated data and regenerates GMII data/control information with preamble and SFD as appropriate. In 10G mode, XGMII is sent to PHY.

When in auto-neg or fault status, Rx Adaptation block removes special sequence ordered-set and forward them to USXGMII Auto-neg block. Only the valid data/control Clause 35/46 is sent to PHY MDI block. Refer to section 2.6.2 for replications/deletions for each MAC-PHY rate based on network port combinations.

Rx Rate Adaptation passes the Rx –configuration/error status to Auto-neg block for clause 37 Rx processing.

For other PHY features, please refer to appropriate PHY specification.

2.9.2.1 USXGMII Receive from MAC (ASIC)

PHY must be able to receive a packet with minimum of 5-byte IPG from the MAC. For debug, it's recommended the PHY to detect number of IPG bytes received and maintain overflow status.

2.9.3 MDI: Tx IPG and Pre-amble Processing on MDI

PHY must be able to meet IPG and Pre-ambles received on the MDI based on the speed:

- 802.3bz Clause 46/49 and 126: PHY must send minimum IPG of 8-bytes with average of 12-bytes. Transmit Pre-amble must be 8-bytes

2.10 Electrical Specification

CML, (Current Mode Logic) is by far the most common Serdes IO standard in use today. The signal swing provided by the CML output is small, resulting in low power consumption. The driver and receiver are often self-terminated, eliminating external components and minimizing transmission line impedance discontinuity effects on timing and signal integrity.

The details of the signal integrity and FEC requirements depends on the type of PHY and system requirements for a particular application. Electrical specifications are based on IEEE 802.3 standard and OFI, some of the example are:

- 5.15625G/10.3125 Gbps - electrical specification use 10GBASE-KR electrical characteristics as defined in section 72.7 and Annex 69B of the IEEE 802.3-2008. No FEC
- 20.625Gbps – electrical specification uses IEEE 25GBASE-R. Some applications may require Clause 108 RS-FEC (528, 514)

3 Appendix

Following pseudo code is provided to clarify bit order in computation of CRC-8. This used be used with reference to section 2.6.16. This example is included for reference/validation only – actual implementation is left for the implementer.

```

logic [47:40] pchHdr;
logic [63:0] preamble;
logic [7:0] pchCrc;

// Preamble contains SOP, PCH, PCH-CRC
assign preamble = {8'hfb, pchHdr, pchCrc}; // SOP + PCH (6-bytes) + PCH-
// CRC (1-byte)

// Generate PCH-CRC
Pch48bCrc8 pch48bCrc8
(
    .sysClk      (sysClk),
    .sysClkReset (sysClkReset),
    .data        ({pchHdr[7:0], pchHdr[15:8], pchHdr[23:16],
    pchHdr[31:24], pchHdr[39:32], pchHdr[47:40]}),
    .crc8        (pchCrc)
);

// 
// PCH CRC Generator/Checker
//
module NifPch48bCrc8 (
    input logic          sysClk,
    input logic          sysClkReset,
    input logic [5:0][7:0] data,    // 48-bit PCH: data[0][7:0] is the
                                    // first PCH byte rcvd/transmitted
    output logic [7:0]    crc8
);

parameter      crcPolynomial = 8'b10000011; // per USXGMII spec:
 $x_8+x_2+x_1+1$ 
integer        pchByteNum, bitNum;

logic [5:0][7:0] feedbk;
logic [5:0][7:0] transTerm;
logic [5:0][7:0] nextCrc;
logic [5:0]       dataXorBitIn;

// 8-bit CRC output
assign crc8 = nextCrc[5] ^ 8'h55; // per USXGMII spec

always_comb begin

```

```

        for (pchByteNum=0; pchByteNum < 6; pchByteNum=pchByteNum+1) begin
// for each of the 6 PCH bytes

        if (pchByteNum==0)
            nextCrc[pchByteNum] = 8'h00;
        else
            nextCrc[pchByteNum] = nextCrc[pchByteNum-1];

            for (bitNum=0; bitNum < 8; bitNum=bitNum+1) begin           //
for each bit in a PCH byte
            feedbk[pchByteNum]      = {8{nextCrc[pchByteNum][7]}};
// feedback term
            transTerm[pchByteNum]   = (feedbk[pchByteNum] &
crcPolynomial) ^ nextCrc[pchByteNum];
            dataXorBitIn[pchByteNum] = data[pchByteNum][bitNum] ^
nextCrc[pchByteNum][7]; // LSB in first
            nextCrc[pchByteNum]     = {transTerm[pchByteNum][6:0],
dataXorBitIn[pchByteNum]};
            end
        end
    end

endmod

```