# ZL30310

## Combined Synchronous Ethernet and IEEE1588 Network Synchronization

Data Sheet

October 2012

### Features

- Supports the combination of SyncE for frequency synchronization and IEEE 1588 for phase alignment (referred to as Hybrid Mode)
- Recovers and transmits network synchronization over Ethernet, IP and MPLS Networks
- Simultaneously supports both the Synchronous Ethernet (Option 1 and Option 2 and the IEEE 1588 industry standard timing protocols
- Capable of server, client repeater, and boundary clock operation
- Integrates two separate digital phase locked loops, with hitless switching between packet and electrical clock references
- Primary DPLL meets Stratum 2 and 3E quality
- Targeted for synchronization distribution to better than ITU-T G.8261, G,8262, G.823, G.824 and ANSI T1.101 synchronization interface standards
- Average frequency accuracy better than ±10 ppb
- Aligns to a low frequency input signal at server (e.g., 1 Hz) with targeted accuracy better than ±1 µs
- Recovers clocks from two independent servers, with hitless switching between packet streams for redundancy
- Supports holdover if the server stream is lost

**Ordering Information**

| | |
|---|---|
| ZL30310GKG | 256 TEPBGA, 17 x 17 mm |
| ZL30310GKG2* | 256 TEPBGA, 17 x 17 mm |

* PB Free Tin/Silver/Copper

**-40°C to +85°C**

- Accepts eight input references, and up to three associated low frequency alignment or framing pulses
- Generates up to four separate output clocks at frequencies between 8 kHz and 100 MHz
- Generates two separate Synchronous Ethernet clocks to drive industry standard Ethernet PHY devices at either 25 MHz or 125 MHz
- Fully configurable solution, enabling performance to be tailored to application/network requirements
- Two independently configurable MAC interfaces, supporting MII, RMII, GMII and TBI standards
- Wire-speed Ethernet Bridge pass through function between the MAC interfaces
- Synchronous serial control interface
- Full demonstration & evaluation platform available

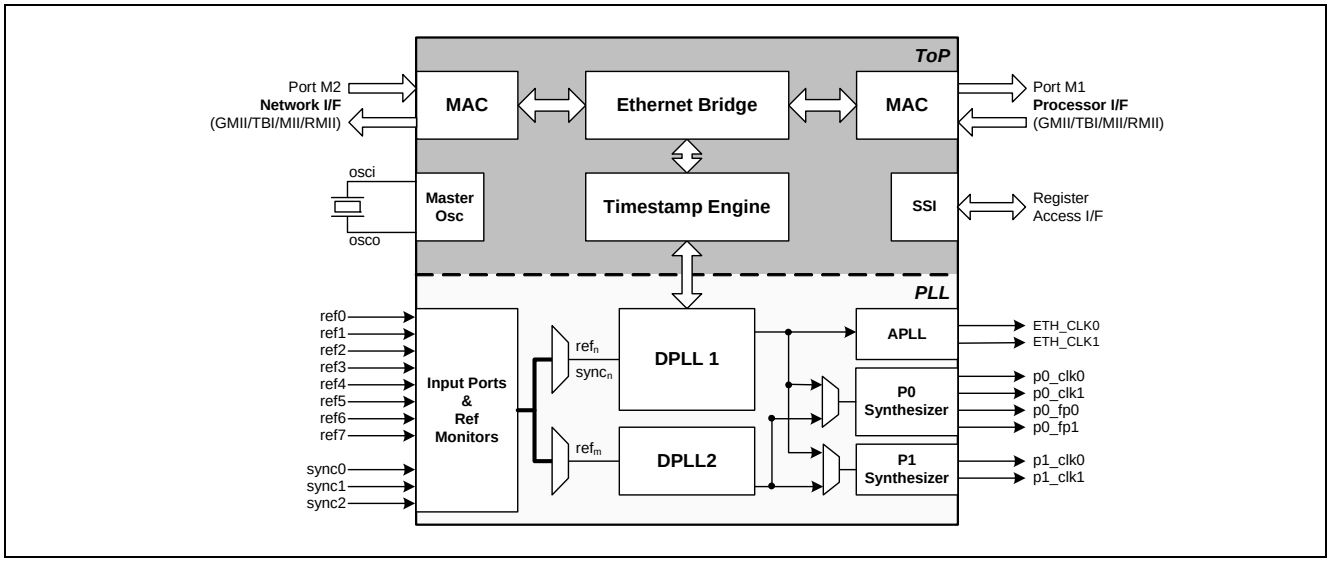



**Figure 1 - ZL30310 Functional Block Diagram**



Copyright 2012, Microsemi Corporation. All Rights Reserved.

## Applications

- IEEE 1588 and Synchronous Ethernet timing
- GSM and UMTS air interface synchronization over a packet network
- Circuit Emulation Services over Packets
- IP-PBX and VoIP Gateways
- Video Conferencing
- Broadband Video Distribution

## Change Summary

Changes from the September 2011 issue to the October 2012 issue.

| Page | Item | Description |
|---|---|---|
| 1 | Ordering Information | A note was added for the ZL30310GKG2 256 TEPBGA package: "Pb Free Tin/Silver/Copper." |

## Description

Network infrastructures are gradually converging onto a packet-based architecture. With this convergence, there are a significant number of synchronous applications that require accurate timing to be distributed over the packet networks. Examples of precision timing sensitive applications that need the transport of synchronization over packet networks include transport of TDM over packet networks, connections to 2 G and 3 G cellular base stations, Voice over IP, IP PBXs, video-conferencing and broadband video.

There are two main ways to enable synchronization over a packet network, synchronizing the packet network itself, as in the Synchronous Ethernet approach, or distributing the timing using the packets as in Microsemi's Timing over Packet (ToP) technology. The two techniques can also be combined to provide a very powerful hybrid solution. Synchronous Ethernet delivers a very accurate frequency reference, but doesn't address phase and time synchronization. ToP can be used to supplement the excellent frequency distribution of Synchronous Ethernet with accurate phase and time information. Alternatively, ToP can be used to extend the reach of the Synchronous Ethernet reference across an asynchronous network, such as a LAN connected to a synchronous WAN.

Microsemi has combined both methods into a single device. The ZL30310 incorporates an extremely low-jitter frequency synthesizer, capable of generating all the frequencies required for Synchronous Ethernet operation, together with Microsemi's patent-pending Timing over Packet (ToP) technology based on the industry-standard IEEE1588™ "PTP" (Precision Time Protocol). Not only can it function as a fully-featured Digital PLL, it also supports the distribution of time, phase and frequency across both layer 2 and layer 3 networks, using both Synchronous Ethernet and IEEE1588 protocols, either alone or in combination.

The ZL30310 is a member of a family of footprint-compatible devices offering the full range of features required for timing and synchronization across the packet network. These devices facilitate design of a flexible card that can be upgraded as required by simply placing another member of the same family.

The family members include:

| | |
|---|---|
| ZL30310 | ZL30310; Combined IEEE1588™ ToP and Synchronous Ethernet, coupled with a GR-1244 Stratum 2/3E/3/4/4E GR-253 SONET, G.812 (types 2 and 3) G.813, and G.8262 quality phase locked loop for timing card applications, plus a second independent PLL for rate conversion or generation of additional derived clocks. |
| ZL30312 | Combined IEEE1588™ ToP and Synchronous Ethernet, coupled with a GR-1244 Stratum 3/4/4E and GR-253 SEONET and G.813 quality phase locked loop for timing card applications, plus a second independent PLL for rate conversion or generation of additional derived clocks. |
| ZL30314 | Combined IEEE1588™ ToP and Synchronous Ethernet, coupled with a GR1244 Stratum 3/4/4E and G.813 Option 1 quality phase locked loop for timing card applications, plus a second independent PLL for rate conversion or generation of additional derived clocks. |
| ZL30316 | Combined IEEE1588™ ToP and Synchronous Ethernet, coupled with two independent, flexible phase locked loops for line card applications |
| ZL30320 | Combined IEEE1588™ ToP and Synchronous Ethernet for line card applications |
| ZL30321 | Synchronous Ethernet line card device in a ToP compatible footprint, containing two independent DPLLs |

The Microsemi device offers the following clock routing options:

| Input | Output | Description |
|---|---|---|
| clock reference | clock | conventional PLL behaviour,<br>*e.g., Synchronous Ethernet node* |
| clock reference | packet stream | server behaviour,<br>*e.g., IEEE1588 server* |
| clock reference | clock and/or packet stream | conventional PLL behaviour coupled with packet time server,<br>*e.g., combined Synchronous Ethernet and IEEE1588 server* |
| packet reference | clock | client behaviour,<br>*e.g., IEEE1588 client* |
| clock and/or packet reference | clock | conventional PLL behaviour, coupled with packet time client, either as fail-over from one to the other, or in combination<br>*e.g., combined Synchronous Ethernet and IEEE1588 client* |
| packet reference | clock and/or packet stream | combination of client and boundary clock behaviour,<br>*e.g., IEEE1588 Boundary Clock* |

When operating as a server, the Microsemi device locks onto the incoming clock reference as a conventional PLL, filtering any jitter that may be present. It also synchronizes to any low-frequency alignment signal, e.g., an 8 kHz TDM frame pulse, or a 1 Hz alignment input. The device delivers streams of packets, each containing a timestamp indicating the precise time that the packet was launched into the network, relative to the acquired reference. It also receives packets from clients, and returns a message indicating the exact time that the client message was received at the server. Using this information, clients are able to align their own timebase with that of the server.

As a client, the Microsemi device can track two independent servers, and determine which one is providing the best time reference. If either the primary reference or the network between the server and client fails, the device can switch to the alternative reference without introducing a phase discontinuity. Alternatively, the client can switch to a conventional clock reference.

The solution timing recovery algorithm continuously tracks the frequency offset and phase drift between the clocks located at the server and the client nodes connected via the packet switched network. The algorithm is tolerant of lost packets, and of packet delay variation caused by packet queuing, route changes and other effects. In the event of a failure in the packet network, or the advent of severe congestion preventing or seriously delaying the delivery of timing packets, the device will put the recovered clocks into holdover until the flow of timing packets is restored. When the device is in holdover mode the drift of the local oscillator directly affects the accuracy of the output clocks.

When using ToP technology, the device is designed to meet ANSI standard T1.101 and ITU-T standards G.823 and G.824 for synchronization distribution. It maintains a mean frequency accuracy of better than ±10 ppb and time alignment of better than ±1 μs when operated over a suitable network.

# Table of Contents

**1.0 Physical Specification . . . 12**
**2.0 External Interface Description . . . 15**
2.1 Clock Interface . . . 15
2.1.1 Output Clock Impedance Matching . . . 16
2.2 Ethernet Output Clock Loop Filter . . . 16
2.2.1 Ethernet Filter Components . . . 16
2.3 Local Oscillator . . . 17
2.3.1 Use of a Clock Oscillator . . . 18
2.4 Packet Interfaces . . . 18
2.4.1 GMII and MII Signal Routing . . . 20
2.5 CPU and Control Interface . . . 21
2.5.1 Reset Circuit . . . 21
2.6 JTAG and Test Interfaces . . . 22
2.7 Power and Ground Connections . . . 23
2.7.1 Power Up/Down Sequence . . . 23
2.7.2 Power Supply Decoupling and Layout Practices . . . 24
2.8 Non-connects . . . 26
**3.0 Modes of Operation . . . 27**
3.1 Time Server Operation . . . 27
3.2 Client Operation . . . 28
3.3 Boundary Clock Mode of Operation . . . 29
3.4 Timing Redundancy and Holdover . . . 30
**4.0 Functional Description . . . 31**
4.1 Ethernet Bridge and MAC Interfaces . . . 31
4.1.1 Overview of Bridge Operation . . . 31
4.1.2 Handling of Timing Packets . . . 32
4.1.2.1 Classification . . . 32
4.1.2.2 Timestamp Insertion . . . 33
4.1.2.3 Timing Packet Scheduling . . . 33
4.1.3 Handling of Non-Timing Packets . . . 34
4.1.3.1 Broadcast and ARP filtering . . . 34
4.1.4 Queuing System and Buffer Management . . . 34
4.1.4.1 Packet Dropping . . . 34
4.1.4.2 Flow Control . . . 35
4.1.5 MAC Interfaces . . . 35
4.1.5.1 GMII/TBI/MII/RMII Connectivity . . . 35
4.1.5.2 MAC Configuration . . . 40
4.1.5.3 Link Up/Down Status . . . 40
4.1.5.4 Management Link . . . 40
4.1.5.5 Statistics . . . 41
4.1.5.6 Layout Guidelines . . . 42
4.2 Timestamp Engine . . . 43
4.2.1 Server Operation . . . 43
4.2.2 Client Operation . . . 44
4.2.3 Boundary Clock Operation . . . 45
4.2.4 Combined Synchronous Ethernet and Timing over Packet Operation . . . 45
4.3 DPLL Functional Description . . . 46
4.3.1 DPLL Features . . . 46
4.3.2 DPLL Mode Control . . . 47
4.3.2.1 DPLL1 Mode Of Operation . . . 48
4.3.2.2 DPLL2 Mode of Operation . . . 48
4.3.3 Loop Bandwidth . . . 48

# Table of Contents


4.3.4 Pull-in/hold-in Range . . . 50
4.3.5 Phase slope Limiting . . . 50
4.3.6 Hitless Reference Switching . . . 50
4.3.7 Holdover . . . 50
4.3.8 Phase Build-Out . . . 51
4.3.9 Reference and Sync Inputs . . . 51
4.3.9.1 Receiving Composite Clocks . . . 54
4.3.10 Reference Input Selection . . . 55
4.3.11 Reference Monitoring . . . 56
4.3.12 Sync Monitoring . . . 59
4.3.13 Reference Monitoring for Custom Configurations . . . 59
4.3.14 Output Clocks and Frame Pulses . . . 62
4.3.14.1 Output Clock and Frame Pulse Squelching . . . 64
4.3.14.2 Disabling Output Clocks and Frame Pulses . . . 65
4.3.14.3 Disabling Output Synthesizers . . . 65
4.3.15 Configurable Input-to-Output and Output-to-Output Delays . . . 65
4.3.16 Active/Redundant Hardware Configuration . . . 66
4.4 Synchronous Serial Interface . . . 68
4.4.1 Transmission Modes . . . 68
4.4.2 Page Addressing . . . 69
4.4.3 Accessing Multi-byte Registers . . . 69
4.5 Timing Software . . . 71
4.5.1 Hardware Requirements . . . 72
4.5.2 User Application . . . 72
4.5.2.1 User-Provided Software Components . . . 72
4.5.3 Microsemi-Provided Software Components . . . 73
4.5.3.1 Solution API . . . 73
4.5.3.2 IEEE 1588 Stack . . . 73
4.5.3.3 Other Software . . . 73
4.5.4 Software Performance . . . 74
4.5.4.1 Use of Multicast vs. Unicast Packets . . . 74
**5.0 DC Characteristics . . . 75**
5.1 Absolute Maximum Ratings* . . . 75
5.2 Recommended Operating Conditions . . . 75
5.3 DC Electrical Characteristics . . . 76
**6.0 AC Characteristics . . . 77**
6.1 Clock Interface Timing . . . 77
6.1.1 Input Timing For Sync References* . . . 77
6.1.2 Input To Output Timing For Ref<n> References* . . . 77
6.1.3 Output Clock Duty Cycle1 . . . 78
6.1.4 Output Clock and Frame Pulse Fall and Rise Times1 . . . 78
6.1.5 E1 Output Frame Pulse Timing* . . . 79
6.1.6 Measured Output Jitter On Ethernet Clock CMOS Outputs with Active MII Interface (ETH_CLK[0,1]) 80
6.1.7 Measured Output Jitter On Ethernet Clock CMOS Outputs with Active GMII Interface (ETH_ CLK[0,1])80
6.1.8 Measured Output Jitter On Programmable CMOS Outputs (P0/1_CLK[0,1]) . . . 80
6.2 Packet Interface Timing . . . 81
6.2.1 Typical Reset Timing Diagram . . . 81
6.2.2 MII Transmit Timing: MAC to PHY Connections . . . 82
6.2.3 MII Receive Timing: MAC to PHY Connections . . . 83
6.2.4 MII Transmit Timing: PHY Emulation Mode (MAC to MAC connections) . . . 84

# Table of Contents

6.2.5 MII Receive Timing: PHY Emulation Mode (MAC to MAC connections) . . . . . 85
6.2.6 RMII Transmit Timing. . . . . 86
6.2.7 RMII Receive Timing . . . . . 87
6.2.8 GMII Transmit Timing. . . . . 88
6.2.9 GMII Receive Timing . . . . . 89
6.2.10 TBI Interface Timing. . . . . 90
6.2.11 Management Interface Timing . . . . . 91
6.3 Synchronous Serial Interface Timing. . . . . 92
6.4 JTAG Interface Timing. . . . . 93
**7.0 Package Dimensions . . . . . 94**

# List of Figures

Figure 1 - ZL30310 Functional Block Diagram . . . 1
Figure 2 - ZL30310 Package View and Ball Positions . . . 12
Figure 3 - Ethernet Loop Filter Component Values . . . 16
Figure 4 - Ethernet Loop Filter Layout Example . . . 17
Figure 5 - Clock Oscillator Circuit. . . . 18
Figure 6 - Typical Power-Up Reset Circuit . . . 22
Figure 7 - Power Supply Decoupling for the ZL30310 . . . 25
Figure 8 - ZL30310 Operating Modes . . . 27
Figure 9 - Example ZL30310 Time Server driven from a GPS or BITS/SSU reference. . . . 28
Figure 10 - Example ZL30310 Timing Client. . . . 29
Figure 11 - ZL30310 Boundary Clock . . . 30
Figure 12 - Ethernet Bridge Structure . . . 31
Figure 13 - GMII Connection (MAC Mode) . . . 36
Figure 14 - GMII Connection (PHY Mode). . . . 37
Figure 15 - TBI Connection . . . 37
Figure 16 - MII Connection (MAC Mode) . . . 38
Figure 17 - MII Connection (PHY Mode) . . . 38
Figure 18 - RMII Connection (MAC Mode). . . . 39
Figure 19 - RMII Connection (PHY Mode) . . . 39
Figure 20 - Relationship of Timestamp Engine to DPLL . . . 43
Figure 21 - Operation as a Timing over Packet Server . . . 44
Figure 22 - Operation as a Timing over Packet Client . . . 45
Figure 23 - Automatic Mode State Machine . . . 47
Figure 24 - Telcordia GR-1244 Stratum 3E Phase Build-Out Requirements . . . 51
Figure 25 - Reference and Sync Inputs . . . 52
Figure 26 - Output Frame Pulse Alignment . . . 53
Figure 27 - GR-378/G.703 Composite Clock Format . . . 54
Figure 28 - External Circuit for Converting a Bipolar CC Signal into Two Unipolar Signals. . . . 54
Figure 29 - G.703 Japan Composite Clock (Appendix II) . . . 55
Figure 30 - Behaviour of the Guard Soak Timer during CFM or SCM Failures . . . 57
Figure 31 - Frequency Acceptance and Rejection Ranges . . . 58
Figure 32 - Reference Monitoring Block Diagram . . . 59
Figure 33 - Sync Monitoring. . . . 59
Figure 34 - Defining SCM Limits for Custom Configurations . . . 60
Figure 35 - Custom CFM Configuration for 25 MHz. . . . 61
Figure 36 - Output Clock Configuration . . . 62
Figure 37 - Phase Delay Adjustments . . . 65
Figure 38 - Typical Master/Slave Configuration . . . 67
Figure 39 - Serial Peripheral Interface Functional Waveforms - LSB First Mode . . . 68
Figure 40 - Serial Peripheral Interface Functional Waveforms - MSB First Mode . . . 69
Figure 41 - Page Addressing Scheme . . . 70
Figure 42 - High Level Software Architecture . . . 71
Figure 41 - Sync Input Timing . . . 77
Figure 42 - Input To Output Timing . . . 77
Figure 43 - Output Duty Cycle . . . 78
Figure 44 - Output Clock Fall and Rise Times . . . 78
Figure 45 - E1 Output Frame Pulse Timing . . . 79
Figure 46 - Typical Reset Timing Diagram. . . . 81

# List of Figures

Figure 47 - MII Transmit Timing Diagram . . . 82
Figure 48 - MII Receive Timing Diagram . . . 83
Figure 49 - MII Transmit Timing Diagram - PHY Emulation Mode . . . 84
Figure 50 - MII Receive Timing Diagram . . . 85
Figure 51 - RMII Transmit Timing Diagram . . . 86
Figure 52 - RMII Receive Timing Diagram . . . 87
Figure 53 - GMII Transmit Timing Diagram . . . 88
Figure 54 - GMII Receive Timing Diagram . . . 89
Figure 55 - TBI Transmit Timing Diagram . . . 90
Figure 56 - TBI Receive Timing Diagram . . . 90
Figure 57 - Management Interface Timing for Ethernet Port - Read . . . 91
Figure 58 - Management Interface Timing for Ethernet Port - Write . . . 91
Figure 59 - Serial Peripheral Interface Timing - LSB First Mode . . . 92
Figure 60 - Serial Peripheral Interface Timing - MSB First Mode . . . 92
Figure 61 - JTAG Signal Timing . . . 93
Figure 62 - JTAG Clock and Reset Timing . . . 93

# List of Figures

Table 1 - ZL30310 Ball Assignments . . . 13
Table 2 - Gigabit Ethernet Ports Signal Mapping in Different Operation Mode . . . 35
Table 3 - MDIO Control . . . 41
Table 4 - DPLL Features . . . 47
Table 5 - DPLL1 Loop Bandwidth Settings . . . 49
Table 6 - DPLL1 Pull-in Range . . . 50
Table 7 - DPLL Phase Slope Limiting . . . 50
Table 8 - Set of Pre-Defined Auto-Detect Clock Frequencies . . . 52
Table 9 - Set of Pre-Defined Auto-Detect Sync Frequencies . . . 53
Table 10 - Frequency Out of Range Limits . . . 57
Table 11 - Supported Output Frequencies . . . 62
Table 12 - APLL LVCMOS Output Clock Frequencies . . . 63
Table 13 - P0 Frame Pulse Frequencies . . . 64
Table 14 - P0 Frame Pulse Widths . . . 64
Table 15 - Reset Timing . . . 81
Table 16 - MII Transmit Timing - 100 Mbps . . . 82
Table 17 - MII Receive Timing - 100 Mbps . . . 83
Table 18 - MII Transmit Timing - 100 Mbps - PHY Emulation Mode . . . 84
Table 19 - MII Receive Timing - 100 Mbps . . . 85
Table 20 - RMII Transmit Timing - 100 Mbps . . . 86
Table 21 - RMII Receive Timing - 100 Mbps . . . 87
Table 22 - GMII Transmit Timing - 1000 Mbps . . . 88
Table 23 - GMII Receive Timing - 1000 Mbps . . . 89
Table 24 - TBI Timing - 1000 Mbps . . . 90
Table 25 - MAC Management Timing Specification . . . 91
Table 26 - JTAG Interface Timing . . . 93

## 1.0 Physical Specification

The package for the ZL30310 is a 256-ball TEBGA

**Features:**

- Body Size: 17 mm x 17 mm (typ.)
- Ball Count: 256
- Ball Pitch: 1.00 mm (typ.)
- Ball Matrix: 16 x 16
- Ball Diameter: 0.50 mm (typ.)
- Total Package Thickness: 1.76 mm (typ.)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | NC | NC | NC | NC | ETH_CLK[0] | IC | IC | ETH_CLK[1] | $V_{DD33}$ | IC | P1_CLK[0] | P1_CLK[1] | NC | P0_CLK[0] | P0_FP[0] | P0_FP[1] | A |
| B | $AV_{DD18}$ | $AV_{DD33}$ | $AV_{DD33}$ | $V_{DD33}$ | NC | NC | NC | NC | $V_{DD33}$ | NC | NC | NC | NC | $V_{DD33}$ | $V_{SS}$ | P0_CLK[1] | B |
| C | NC | $AV_{DD18}$ | $AV_{DD18}$ | NC | $V_{DD18}$ | $V_{DD33}$ | NC | NC | $V_{DD33}$ | NC | NC | NC | NC | $V_{DD33}$ | $V_{DD33}$ | IC | C |
| D | ETH_FILTER | FILTER_REF[0] | FILTER_REF[1] | $AV_{SS}$ | $AV_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | NC | $V_{SS}$ | $V_{DD18}$ | IC | D |
| E | $V_{DD33}$ | $V_{DD33}$ | NC | $AV_{SS}$ | NC | NC | NC | NC | NC | NC | NC | NC | $V_{DD18}$ | IC (tie low) | RST_B | M1_TXER | E |
| F | M2_MDC | M2_MDIO | $V_{SS}$ | $AV_{SS}$ | NC | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | NC | M1_TXEN | M1_TXD[6] | M1_TXD[7] | M1_GTXCLK | F |
| G | M1_MDC | M1_MDIO | PHY_RST_B | $AV_{SS}$ | NC | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | M1_TXD[5] | M1_TXD[4] | $V_{DD33}$ | M1_TXD[3] | M1_TXCLK | G |
| H | $V_{SS}$ | $V_{DD18}$ | NC | NC | $V_{DD18}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | M1_TXD[0] | M1_TXD[1] | M1_TXD[2] | M1_COL | M1_RXCLK | H |
| J | IC | IC | NC | $V_{DD18}$ | NC | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | M1_RXER | $V_{DD33}$ | M1_CRS | M1_REFCLK | J |
| K | REF[5] | REF[6] | REF[7] | NC | NC | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | M1_RXD[4] | M1_RXD[5] | M1_RXD[6] | M1_RXD[7] | M1_RXDV | K |
| L | REF[2] | SYNC[2] | REF[4] | $V_{SS}$ | NC | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | $V_{SS}$ | M1_RXD[0] | M1_RXD[1] | M1_RXD[2] | M1_RXD[3] | $V_{SS}$ | L |
| M | REF[1] | SYNC[1] | REF[3] | $V_{DD18}$ | NC | IC | IC | M2_TXD[3] | M2_TXD[7] | IC | IC | NC | TDI | TDO | NC | OSC_O | M |
| N | SYNC[0] | NC | $V_{DD33}$ | NC | NC | M2_RXD[4] | M2_RXDV | M2_TXD[2] | M2_TXD[6] | $V_{DD33}$ | INT0_B | IC | NC | TMS | NC | OSC_I | N |
| P | REF[0] | NC | NC | M2_RXD[1] | M2_RXD[5] | M2_RXER | M2_COL | M2_TXD[1] | M2_TXD[5] | $V_{SS}$ | IC | INT1_B | NC | IC (tie low) | TCK | TRST_B | P |
| R | NC | NC | $V_{SS}$ | M2_RXD[2] | M2_RXD[6] | M2_CRS | $V_{DD33}$ | M2_TXD[0] | M2_TXD[4] | M2_TXER | CS_B | SO | $V_{DD18}$ | $AV_{SS}$ | NC | NC | R |
| T | DPLL2_IN_REF | $V_{DD18}$ | M2_RXD[0] | M2_RXD[3] | M2_RXD[7] | M2_REFCLK | M2_RXCLK | M2_TXCLK | M2_GTXCLK | M2_TXEN | SCLK | SI | $V_{SS}$ | $AV_{DD18}$ | NC | NC | T |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | |

**Figure 2 - ZL30310 Package View and Ball Positions**

| Ball # | ZL30310 Signal Name |
|---|---|
| A1 | NC |
| A2 | NC |
| A3 | NC |
| A4 | NC |
| A5 | ETH_CLK[0] |
| A6 | IC |
| A7 | IC |
| A8 | ETH_CLK[1] |
| A9 | $V_{DD33}$ |
| A10 | IC |
| A11 | P1_CLK[0] |
| A12 | P1_CLK[1] |
| A13 | NC |
| A14 | P0_CLK[0] |
| A15 | P0_FP[0] |
| A16 | P0_FP[1] |
| B1 | $AV_{DD18}$ |
| B2 | $AV_{DD33}$ |
| B3 | $AV_{DD33}$ |
| B4 | $V_{DD33}$ |
| B5 | NC |
| B6 | NC |
| B7 | NC |
| B8 | NC |
| B9 | $V_{DD33}$ |
| B10 | NC |
| B11 | NC |
| B12 | NC |
| B13 | NC |
| B14 | $V_{DD33}$ |
| B15 | $V_{SS}$ |
| B16 | P0_CLK[1] |
| C1 | NC |
| C2 | $AV_{DD18}$ |
| C3 | $AV_{DD18}$ |
| C4 | NC |
| C5 | $V_{DD18}$ |
| C6 | $V_{DD33}$ |
| C7 | NC |
| C8 | NC |
| C9 | $V_{DD33}$ |
| C10 | NC |
| C11 | NC |

**Table 1 - ZL30310 Ball Assignments**

| Ball # | ZL30310 Signal Name |
|---|---|
| C12 | NC |
| C13 | NC |
| C14 | $V_{DD33}$ |
| C15 | $V_{DD33}$ |
| C16 | IC |
| D1 | ETH_FILTER |
| D2 | FILTER_REF[0] |
| D3 | FILTER_REF[1] |
| D4 | $AV_{SS}$ |
| D5 | $AV_{SS}$ |
| D6 | $V_{SS}$ |
| D7 | $V_{SS}$ |
| D8 | $V_{SS}$ |
| D9 | $V_{SS}$ |
| D10 | $V_{SS}$ |
| D11 | $V_{SS}$ |
| D12 | $V_{SS}$ |
| D13 | NC |
| D14 | $V_{SS}$ |
| D15 | $V_{DD18}$ |
| D16 | IC |
| E1 | $V_{DD33}$ |
| E2 | $V_{DD33}$ |
| E3 | NC |
| E4 | $AV_{SS}$ |
| E5 | NC |
| E6 | NC |
| E7 | NC |
| E8 | NC |
| E9 | NC |
| E10 | NC |
| E11 | NC |
| E12 | NC |
| E13 | $V_{DD18}$ |
| E14 | IC (tie low) |
| E15 | RST_B |
| E16 | M1_TXER |
| F1 | M2_MDC |
| F2 | M2_MDIO |
| F3 | $V_{SS}$ |
| F4 | $AV_{SS}$ |
| F5 | NC |
| F6 | $V_{SS}$ |

**Table 1 - ZL30310 Ball Assignments**

| Ball # | ZL30310 Signal Name |
|---|---|
| F7 | $V_{SS}$ |
| F8 | $V_{SS}$ |
| F9 | $V_{SS}$ |
| F10 | $V_{SS}$ |
| F11 | $V_{SS}$ |
| F12 | NC |
| F13 | M1_TXEN |
| F14 | M1_TXD[6] |
| F15 | M1_TXD[7] |
| F16 | M1_GTXCLK |
| G1 | M1_MDC |
| G2 | M1_MDIO |
| G3 | PHY_RST_B |
| G4 | $AV_{SS}$ |
| G5 | NC |
| G6 | $V_{SS}$ |
| G7 | $V_{SS}$ |
| G8 | $V_{SS}$ |
| G9 | $V_{SS}$ |
| G10 | $V_{SS}$ |
| G11 | $V_{SS}$ |
| G12 | M1_TXD[5] |
| G13 | M1_TXD[4] |
| G14 | $V_{DD33}$ |
| G15 | M1_TXD[3] |
| G16 | M1_TXCLK |
| H1 | $V_{SS}$ |
| H2 | $V_{DD18}$ |
| H3 | NC |
| H4 | NC |
| H5 | $V_{DD18}$ |
| H6 | $V_{SS}$ |
| H7 | $V_{SS}$ |
| H8 | $V_{SS}$ |
| H9 | $V_{SS}$ |
| H10 | $V_{SS}$ |
| H11 | $V_{SS}$ |
| H12 | M1_TXD[0] |
| H13 | M1_TXD[1] |
| H14 | M1_TXD[2] |
| H15 | M1_COL |
| H16 | M1_RXCLK |
| J1 | IC |

**Table 1 - ZL30310 Ball Assignments**

| Ball # | ZL30310 Signal Name |
|---|---|
| J2 | IC |
| J3 | NC |
| J4 | $V_{DD18}$ |
| J5 | NC |
| J6 | $V_{SS}$ |
| J7 | $V_{SS}$ |
| J8 | $V_{SS}$ |
| J9 | $V_{SS}$ |
| J10 | $V_{SS}$ |
| J11 | $V_{SS}$ |
| J12 | $V_{SS}$ |
| J13 | M1_RXER |
| J14 | $V_{DD33}$ |
| J15 | M1_CRS |
| J16 | M1_REFCLK |
| K1 | REF[5] |
| K2 | REF[6] |
| K3 | REF[7] |
| K4 | NC |
| K5 | NC |
| K6 | $V_{SS}$ |
| K7 | $V_{SS}$ |
| K8 | $V_{SS}$ |
| K9 | $V_{SS}$ |
| K10 | $V_{SS}$ |
| K11 | $V_{SS}$ |
| K12 | M1_RXD[4] |
| K13 | M1_RXD[5] |
| K14 | M1_RXD[6] |
| K15 | M1_RXD[7] |
| K16 | M1_RXDV |
| L1 | REF[2] |
| L2 | SYNC[2] |
| L3 | REF[4] |
| L4 | $V_{SS}$ |
| L5 | NC |
| L6 | $V_{SS}$ |
| L7 | $V_{SS}$ |
| L8 | $V_{SS}$ |
| L9 | $V_{SS}$ |
| L10 | $V_{SS}$ |
| L11 | $V_{SS}$ |
| L12 | M1_RXD[0] |

**Table 1 - ZL30310 Ball Assignments**

| Ball # | ZL30310 Signal Name |
|---|---|
| L13 | M1_RXD[1] |
| L14 | M1_RXD[2] |
| L15 | M1_RXD[3] |
| L16 | $V_{SS}$ |
| M1 | REF[1] |
| M2 | SYNC[1] |
| M3 | REF[3] |
| M4 | $V_{DD18}$ |
| M5 | NC |
| M6 | IC |
| M7 | IC |
| M8 | M2_TXD[3] |
| M9 | M2_TXD[7] |
| M10 | IC |
| M11 | IC |
| M12 | NC |
| M13 | TDI |
| M14 | TDO |
| M15 | NC |
| M16 | OSC_O |
| N1 | SYNC[0] |
| N2 | NC |
| N3 | $V_{DD33}$ |
| N4 | NC |
| N5 | NC |
| N6 | M2_RXD[4] |
| N7 | M2_RXDV |
| N8 | M2_TXD[2] |
| N9 | M2_TXD[6] |
| N10 | $V_{DD33}$ |
| N11 | INT0_B |
| N12 | IC |
| N13 | NC |
| N14 | TMS |
| N15 | NC |
| N16 | OSC_I |
| P1 | REF[0] |
| P2 | NC |
| P3 | NC |
| P4 | M2_RXD[1] |
| P5 | M2_RXD[5] |
| P6 | M2_RXER |
| P7 | M2_COL |

**Table 1 - ZL30310 Ball Assignments**

| Ball # | ZL30310 Signal Name |
|---|---|
| P8 | M2_TXD[1] |
| P9 | M2_TXD[5] |
| P10 | $V_{SS}$ |
| P11 | IC |
| P12 | INT1_B |
| P13 | NC |
| P14 | IC (tie low) |
| P15 | TCK |
| P16 | TRST_B |
| R1 | NC |
| R2 | NC |
| R3 | $V_{SS}$ |
| R4 | M2_RXD[2] |
| R5 | M2_RXD[6] |
| R6 | M2_CRS |
| R7 | $V_{DD33}$ |
| R8 | M2_TXD[0] |
| R9 | M2_TXD[4] |
| R10 | M2_TXER |
| R11 | CS_B |
| R12 | SO |
| R13 | $V_{DD18}$ |
| R14 | $AV_{SS}$ |
| R15 | NC |
| R16 | NC |
| T1 | DPLL2_IN_REF |
| T2 | $V_{DD18}$ |
| T3 | M2_RXD[0] |
| T4 | M2_RXD[3] |
| T5 | M2_RXD[7] |
| T6 | M2_REFCLK |
| T7 | M2_RXCLK |
| T8 | M2_TXCLK |
| T9 | M2_GTXCLK |
| T10 | M2_TXEN |
| T11 | SCLK |
| T12 | SI |
| T13 | $V_{SS}$ |
| T14 | $AV_{DD18}$ |
| T15 | NC |
| T16 | NC |

**Table 1 - ZL30310 Ball Assignments**

**NC** - not connected - leave open circuit.
**IC** - internally connected - leave open circuit.
**IC (tie low)** - internally connected, must be tied to $V_{SS}$.

## 2.0 External Interface Description

The following key applies to all tables:

| | |
|---|---|
| I | Input |
| O | Output |
| D | Internal 100 kΩ pull-down resistor present |
| U | Internal 100 kΩ pull-up resistor present |
| T | Tri-state Output |

### 2.1 Clock Interface

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| **Input Reference Clocks** | | | | |
| P1, M1 | REF[0:1] | I U | 2 | **Master Input Reference clocks (LVCMOS, Schmitt Trigger)**<br>Input reference clocks, available to internal DPLL for synchronization of output clocks and timestamp generation. Accepts input references from 8 kHz to 77.76 MHz in 8 kHz increments, including 25 MHz, 50 MHz. Additional pre-dividers are available to allow input frequencies of 62.5 and 125 MHz.<br>These pins are internally pulled up to $V_{DD}$. |
| L1, M3, L3, K1, K2, K3 | REF[2:7] | I U | 5 | **Master Input Reference clocks (LVCMOS, Schmitt Trigger)**<br>Further input reference clocks, available to internal DPLL for synchronization of output clocks and timestamp generation. Accept input references from 8 kHz to 77.76 MHz in 8 kHz increments, including 25 MHz, 50 MHz.<br>These pins are internally pulled up to $V_{DD}$. |
| N1, M2, L2 | SYNC[0:2] | I U | 3 | **Frame Pulse Synchronization References (LVCMOS, Schmitt Trigger)**<br>Frame pulse synchronization or low-frequency alignment signals associated with inputs REF[0:2]. Accepts frame pulses in clock format (50% duty cycle) or a basic frame pulse format with a minimum pulse width of 5 nsec. While the DPLL is locked to the input reference the output frame pulses are synchronized to this input.<br>These pins are internally pulled up to $V_{DD}$. |
| **Output Clocks** | | | | |
| A14, B16 | P0_CLK[0:1] | O | 2 | **Programmable Frequency Synthesizer 0 - Output Clocks (LVCMOS)**<br>These outputs can be configured to provide any frequency with a multiple of 8 kHz up to 100 MHz, locked to the incoming reference (packet or electrical). P0_CLK[1] is a multiple or division of P0_CLK[0]. |
| A15, A16 | P0_FP[0:1] | O | 2 | **Programmable Frequency Synthesizer 0 - Output Frame Pulses (LVCMOS)**<br>These outputs can be configured to provide any style of output frame pulse or low-frequency alignment signal associated with the P0 output clock.<br>For a ToP client, provided two-way server-client operation is enabled, these outputs may be aligned to the low frequency alignment signal at the server (e.g., 1 Hz or 8 kHz alignment signal). |
| A11, A12 | P1_CLK[0:1] | O | 2 | **Programmable Frequency Synthesizer 1 Output Clocks (LVCMOS)**<br>These outputs can be configured to provide any frequency with a multiple of 8 kHz up to 100 MHz, locked to the incoming reference (packet or electrical). P1_CLK[1] is a multiple or division of P1_CLK[0]. |

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| A5, A8 | ETH_CLK[0:1] | O | 2 | **Ethernet Interface Output Clocks (LVCMOS)**<br>These outputs can be configured to provide the Ethernet RMII / MII / GMII / TBI interface clocks as required. Capable of generating 25, 50, 62.5 and 125 MHz clocks. 62.5 MHz is the default start up frequency, care should be taken to re-configure after initialization and possibly re-initialize receiving device. |
| T1 | DPLL2_IN_REF | O | 1 | **Buffered Selected Output Reference (LVCMOS)**<br>This is a buffered copy of the selected input reference clock for DPLL2. Switching between input reference clocks at this output is not hitless. |

### 2.1.1 Output Clock Impedance Matching

To ensure signal integrity, ZL30310 output clock traces should be impedance matched with their drivers. The clock arrangements should be one driver to one receiver, with a source termination resistor placed as close as possible to the driver. Twenty to thirty-three ohms is a typical range of resistor value for matching impedance with a 50 ohm trace.

If clocks must be fanned out to multiple receivers, then an external clock buffer should generally be used. In some circumstances, it is acceptable to branch the clock at the source, provided that standard signal integrity practices are strictly followed to prevent degradations at the receiver.

## 2.2 Ethernet Output Clock Loop Filter

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| **APLL Loop Filter** | | | | |
| D1 | ETH_FILTER | Analog I/O | 1 | **External Analog PLL Loop Filter terminal (Analog)** |
| D2, D3 | FILTER_REF[0:1] | Analog I/O | 2 | **Analog PLL External Loop Filter References (Analog)** |

### 2.2.1 Ethernet Filter Components

The APLL in the ZL30310 uses external components to help optimize its loop bandwidth. For optimal jitter performance, the following component values are recommended (Note: Microsemi application support can provide alternative components values that would result in comparable jitter performance):

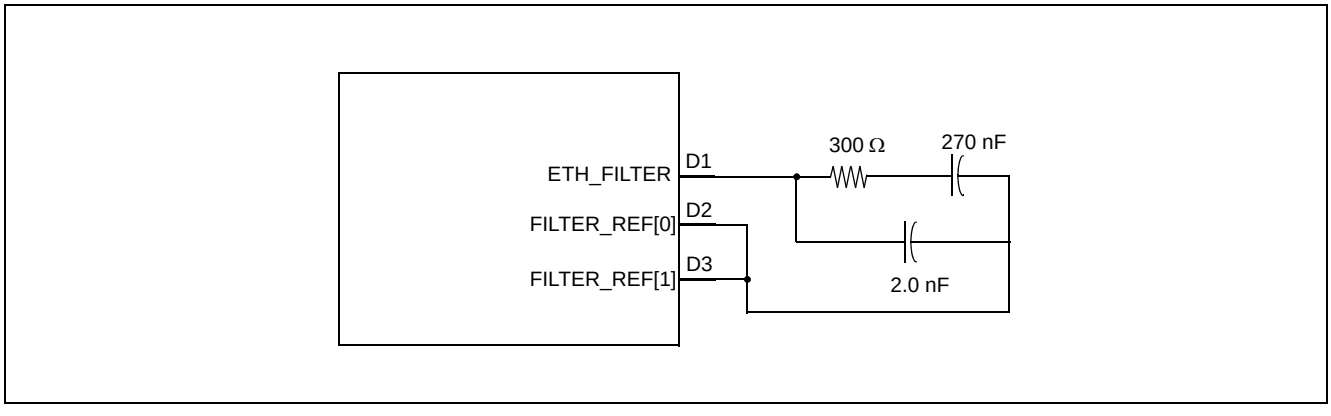



**Figure 3 - Ethernet Loop Filter Component Values**

The following guidelines for layout should be applied to minimize noise in the proximity of the loop filter.

- Group the loop filter discrete components tightly and as close to the ZL30310 body as possible, minimizing trace lengths.
- Keep all unrelated traces and components 100 mil away from the loop filter discrete components and traces (applies to all PCB layers).
- Cut away all planes in the cross section of PCB beneath the loop filter to prevent coupling with noise from power or ground planes.
- Keep components on same side as ZL30310 and do not use vias.

Figure 4 shows an example of the loop filter placement and routing:

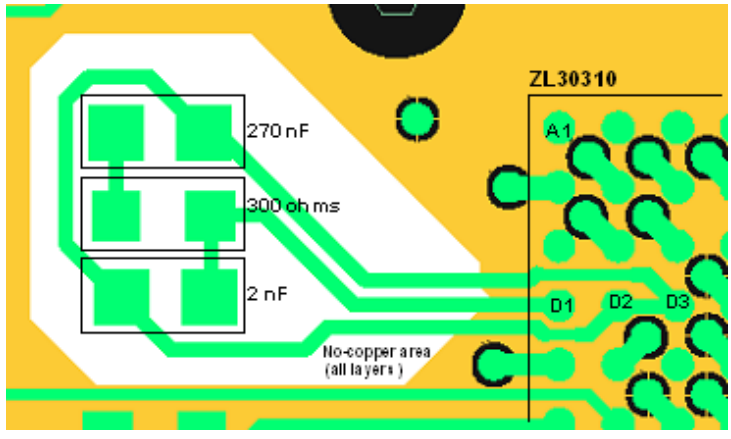



**Figure 4 - Ethernet Loop Filter Layout Example**

## 2.3 Local Oscillator

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| **Local Oscillator** | | | | |
| M16 | OSC_O | Analog O | 1 | **Oscillator Master Clock (Analog Output)**<br>For crystal operation, a 20 MHz crystal is connected from this pin to OSC_I. Not suitable for driving other devices.<br>For clock oscillator operation, this pin is left unconnected. |
| N16 | OSC_I | I | 1 | **Oscillator Master Clock (Input)**<br>For crystal operation, a 20 MHz crystal is connected from this pin to OSC_O.<br>For clock oscillator operation, this pin is connected to a clock source. |

### 2.3.1 Use of a Clock Oscillator

When using a clock oscillator as the master timing source, connect the oscillator's output clock to the OSC_I pin as shown in Figure 5. The connection to OSC_I should be direct and not AC coupled. The OSC_O pin must be left unconnected.

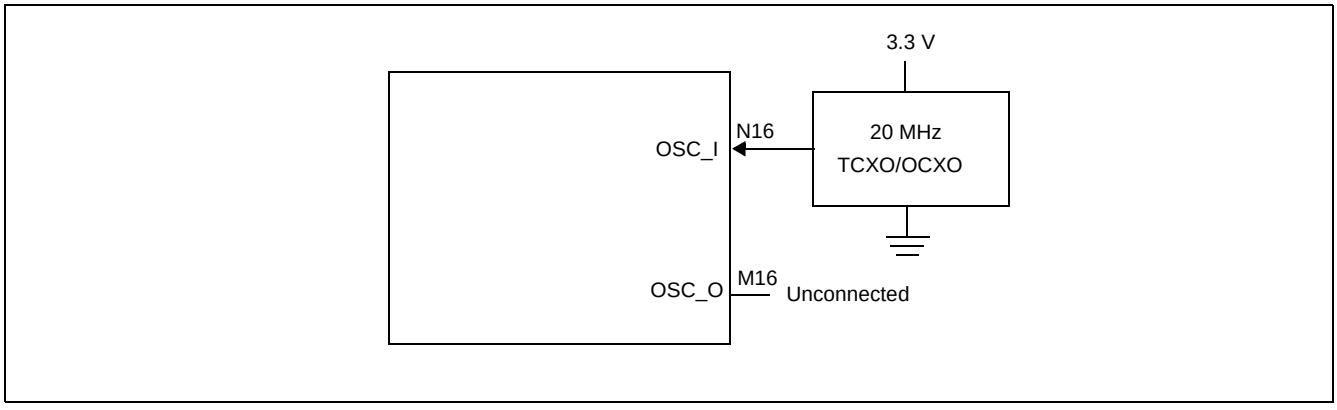



**Figure 5 - Clock Oscillator Circuit**

Measures should be taken to ensure that the oscillator output clock is optimal quality (low jitter, no cross talk, matched impedance). Specifically, these measures should include:

- Provision a source termination resistor at the output of the oscillator and assign to 0 ohms initially. If there is ringing at the receiver, then the value should be increased.
- Place oscillator within about 50 mm of the ZL30310 device, and keep the clock trace away from other traces.
- Ensure that the oscillator power noise is minimized. Include RLC or ferrite-C lowpass filters as required to achieve low noise.
- If an OCXO is provisioned, adequate bulk decoupling should be provisioned so that the device's oven activity does not cause "bumps" in the supply voltage.

## 2.4 Packet Interfaces

Port M1 = Processor Interface, Port M2 = Network Interface

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| **Packet Interface Ports** | | | | |
| J16, T6 | Mn_REFCLK | I U | 2 | **GMII/TBI - Reference Clock input at 125 MHz**<br>Can be used to lock receive circuitry (RX) to Mn_GTXCLK rather than recovering the RXCLK (or RBC0 and RBC1). Useful, for example, in the absence of valid serial data.<br>RMII - Reference Clock input at 50 MHz<br>Note: device DPLL cannot lock to this input clock |
| H16, T7 | Mn_RXCLK / Mn_RBC0 | I U | 2 | **GMII/MII - Mn_RXCLK**<br>Accepts the following frequencies:<br>25.0 MHz MII 100 Mbps<br>125.0 MHz GMII 1 Gbps<br>**TBI - Mn_RBC0**<br>Used as a clock when in TBI mode. Accepts 62.5 MHz and it is 180 degrees out of phase with Mn_RBC1. Receive data is clocked at each rising edge of Mn_RBC1 and Mn_RBC0, resulting in 125 MHz sample rate.<br>Note: device DPLL cannot lock to this input clock |

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| H15, P7 | Mn_COL / Mn_RBC1 | I U | 2 | **GMII/MII - Mn_COL**<br>Collision Detection. This signal is independent of Mn_TXCLK and Mn_RXCLK, and is asserted when a collision is detected on an attempted transmission. It is active high, and only specified for half-duplex operation. Microsemi highly recommends operating in full duplex mode only.<br>**TBI - Mn_RBC1**<br>Used as a clock when in TBI mode. Accepts 62.5 MHz and is 180 degrees out of phase with Mn_RBC1. Receive data is clocked at each rising edge of Mn_RBC1 and Mn_RBC0, resulting in 125 MHz sample rate. |
| K15, K14, K13, K12, L15, L14, L13, L12, T5, R5, P5, N6, T4, R4, P4, T3 | Mn_RXD[7:0] | I U | 16 | **Receive Data**<br>Only half the bus (bits [3:0]) are used in MII mode, and only bits [1:0] in RMII mode. Clocked on rising edge of Mn_RXCLK (GMII/MII), rising edge of Mn_REFCLK (RMII) or the rising edges of Mn_RBC0 and Mn_RBC1 (TBI). |
| K16, N7 | Mn_RXDV / Mn_CRS_DV / Mn_RXD[8] | I U | 2 | **GMII/MII - Mn_RXDV**<br>Receive Data Valid. Active high. This signal is clocked on the rising edge of Mn_RXCLK. It is asserted when valid data is on the Mn_RXD bus.<br>**RMII - Mn_CRS_DV**<br>Carrier Sense/Receive Data Valid. Active High. Asserted by the PHY when the receive medium is non-idle. Clocked on the rising edge of Mn_REFCLK.<br>**TBI - Mn_RXD[8]**<br>Receive Data. Clocked on the rising edges of Mn_RBC0 and Mn_RBC1. |
| J13, P6 | Mn_RXER / Mn_RXD[9] | I U | 2 | **GMII/MII/RMII - Mn_RXER**<br>Receive Error. Active high signal indicating an error has been detected. Normally valid when Mn_RXDV is asserted. Can be used in conjunction with Mn_RXD when Mn_RXDV signal is de-asserted to indicate a False Carrier.<br>**TBI - Mn_RXD[9]**<br>Receive Data. Clocked on the rising edges of Mn_RBC0 and Mn_RBC1. |
| J15, R6 | Mn_CRS / Mn_SIGNAL_DETECT | I/O U | 2 | **GMII/MII - Mn_CRS**<br>Carrier Sense. This asynchronous signal is asserted when either the transmission or reception device is non-idle. Active high. It is an input in MAC mode, and an output in PHY emulation mode.<br>**TBI - Mn_Signal Detect**<br>Similar function to Mn_CRS. |
| G16, T8 | Mn_TXCLK | I/O U | 2 | **MII only - Transmit Clock**<br>Accepts/generates 25 MHz for 100 Mbit/s operation.<br>It is an input in MAC mode, and an output in PHY emulation mode. |
| F15, F14, G12, G13, G15, H14, H13, H12, M9, N9, P9, R9, M8, N8, P8, R8 | Mn_TXD[7:0] | O | 16 | **Transmit Data**<br>Only half the bus (bits [3:0]) are used in MII mode, and only bits [1:0] in RMII mode. Clocked on rising edge of Mn_TXCLK (MII), rising edge of Mn_REFCLK (RMII) or the rising edge of Mn_GTXCLK (GMII/TBI). |
| F13, T10 | Mn_TXEN / Mn_TXD[8] | O | 2 | **GMII/MII/RMII - Mn_TXEN**<br>Transmit Enable. Asserted when the MAC has data to transmit, synchronously to Mn_TXCLK (Mn_REFCLK in RMII mode) with the first preamble of the packet to be sent. Remains asserted until the end of the packet transmission. Active high.<br>**TBI - Mn_TXD[8]**<br>Transmit Data. Clocked on rising edge of Mn_GTXCLK. |

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| E16, R10 | Mn_TXER / Mn_TXD[9] | O | 2 | **GMII/MII - Mn_TXER**<br>Transmit Error. Transmitted synchronously with respect to Mn_TXCLK, and active high. When asserted (with Mn_TXEN also asserted) the device will transmit a non-valid symbol, somewhere in the transmitted frame.<br>**TBI - Mn_TXD[9]**<br>Transmit Data. Clocked on rising edge of Mn_GTXCLK. |
| F16, T9 | Mn_GTX_CLK | O | 2 | **GMII/TBI only - Gigabit Transmit Clock**<br>Output of a clock for Gigabit operation at 125 MHz. |
| G1, F1 | Mn_MDC | I/O U | 2 | **MII management data clock**<br>It is an output pin when the port is operating in MAC mode, and an input pin when the port is operating in PHY emulation mode. Minimum period of 400 ns (maximum freq. 2.5 MHz), and is independent of the TXCLK and RXCLK. |
| G2, F2 | Mn_MDIO | I/O U | 2 | **MII management data I/O. Open Drain I/O requires external 10 kΩ pull-up resistor**<br>Common for all types of MII ports at up to 2.5 MHz. It is bi-directional between the device and the Ethernet station management entity. Data is passed synchronously with respect to Mn_MDC. |

### 2.4.1 GMII and MII Signal Routing

When routing xMII signals it is important to follow the guidelines as set out in the relevant IEEE 802.3 standards documents, such that the source-synchronous signal timing specifications are satisfied. To maintain signal integrity, the following practices are recommended:

- Apply signal spacings which will minimize cross talk between xMII clocks and all other signals, including trace switchbacks with same signal.
- Measures should also be taken to minimize cross talk between data signals of different xMII busses.
- Source termination resistors should be provisioned at all xMII clock drivers to support impedance matching. If excessive overshoot is a concern, then data signals should also have source terminations.

## 2.5 CPU and Control Interface

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| **Control and Status Interface** | | | | |
| E15 | RST_B | I U | 1 | **Reset (LVCMOS, Schmitt Trigger)**<br>A logic low at this input resets the device. To ensure proper operation, the device must be reset after power-up. The RST_B pin should be held low for a minimum of 300 ns. User can access device registers 5 msec after RST_B goes high, or pull register adr 0x00 bit 7 for reset ready. |
| G3 | PHY_RST_B | O | 1 | **PHY Reset Control**<br>Enables the PHY to be reset. |
| **CPU Control Interface** | | | | |
| N11 | INT_B[0] | O | 1 | **Host Interrupt Output - High Priority**<br>Flags a change of device status prompting the processor to read the enabled interrupt service registers (ISR).<br>This pin is an open drain, active low and requires an external pull up to $V_{DD}$. |
| P12 | INT_B[1] | O | 1 | **Host Interrupt Output - Low Priority**<br>Flags a change of device status prompting the processor to read the enabled interrupt service registers (ISR).<br>This pin is an open drain, active low and requires an external pull up to $V_{DD}$. |
| T11 | SCLK | I | 1 | **Clock for Serial Interface**<br>Provides the clock for serial microport interface. |
| T12 | SI | I | 1 | **Serial Interface Input Stream**<br>The serial input stream holds the access command, the address and the write data bits. |
| R12 | SO | O | 1 | **Serial Interface Output Stream**<br>The serial output stream holds the read data bits. |
| R11 | CS_B | I U | 1 | **Chip Select for Serial Interface**<br>This is an active low signal. This pin is internally pulled up to $V_{DD}$. |

### 2.5.1 Reset Circuit

To ensure proper operation, the device must be reset by holding the RST_B pin low for at least 300 ns after power-up. Following reset, the device will operate under specified default settings.

The reset pin can be controlled with on-board system reset circuitry or by using a stand-alone power-up reset circuit as shown in Figure 6. This circuit provides approximately 60 µs of reset low time. The RST_B input has schmitt trigger properties to prevent level bouncing.

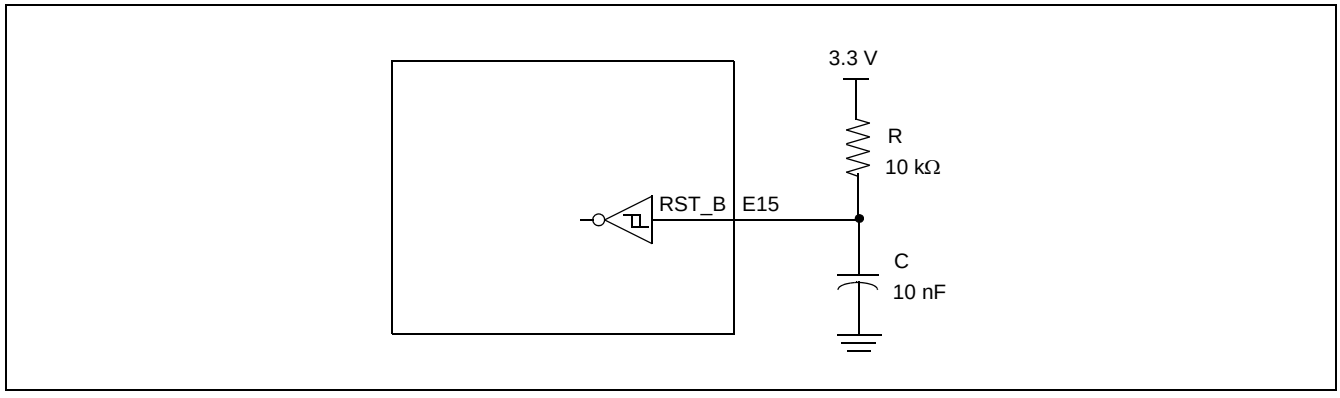



**Figure 6 - Typical Power-Up Reset Circuit**

## 2.6 JTAG and Test Interfaces

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| **JTAG and Test** | | | | |
| M14 | TDO | O | 1 | **Test Serial Data Out**<br>JTAG serial data is output on this pin on the falling edge of TCK.<br>This pin is held in high impedance state when JTAG scan is not enable. |
| M13 | TDI | I U | 1 | **Test Serial Data In**<br>JTAG serial test instructions and data are shifted in on this pin.<br>This pin is internally pulled up to $V_{DD}$. If this pin is not used then it should be left unconnected. |
| P16 | TRST_B | I U | 1 | **Test Reset**<br>Asynchronously initializes the JTAG TAP controller by putting it in the Test-Logic-Reset state. This pin should be pulsed low on power-up to ensure that the device is in the normal functional state.<br>This pin is internally pulled up to $V_{DD}$. During normal operation or if this pin is not used then it should be connected to $V_{SS}$. |
| P15 | TCK | I | 1 | **Test Clock**<br>Provides the clock to the JTAG test logic.<br>If this pin is not used then it should be pulled down to $V_{SS}$. |
| N14 | TMS | I U | 1 | **Test Mode Select**<br>JTAG signal that controls the state transitions of the TAP controller.<br>This pin is internally pulled up to $V_{DD}$. If this pin is not used then it should be left unconnected. |

## 2.7 Power and Ground Connections

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| **Power and Ground** | | | | |
| A9 B4 B9 B14 C6 C9 C14 C15 E1 E2 G14 J14 N3 N10 R7 | $V_{DD33}$ | | 15 | **I/O Positive Supply Voltage. +3.3 V DC nominal** |
| C5 D15 E13 H2 H5 J4 M4 R13 T2 | $V_{DD18}$ | | 9 | **Core Positive Supply Voltage. +1.8 V DC nominal** |
| B2 B3 | $AV_{DD33}$ | | 2 | **Analog I/O Positive Supply Voltage. +3.3 V DC nominal** |
| B1 C2 C3 T14 | $AV_{DD18}$ | | 4 | **Analog Core Positive Supply Voltage. +1.8 V DC nominal** Each of these pin requires specific power supply decoupling as shown in Figure 7 |
| B15 D6 D7 D8 D9 D10 D11 D12 D14 F3 F6 F7 F8 F9 F10 F11 G6 G7 G8 G9 G10 G11 H1 H6 H7 H8 H9 H10 H11 J6 J7 J8 J9 J10 J11 J12 K6 K7 K8 K9 K10 K11 L4 L6 L7 L8 L9 L10 L11 L16 P10 R3 T13 | $V_{SS}$ | | 53 | **Ground. 0 Volts. ($V_{SS}$ pads)** |
| D4 D5 E4 F4 G4 R14 | $AV_{SS}$ | | 6 | **Analog Ground. 0 Volts. ($V_{SS}$ pads)** |

### 2.7.1 Power Up/Down Sequence

The 3.3 V power rail should be powered before or simultaneously with the 1.8 V power rail to prevent the risk of latch-up. The power-down sequence is less critical, however it should be performed in the reverse order to reduce transient currents that consume power.

### 2.7.2 Power Supply Decoupling and Layout Practices

Jitter levels on the ZL30310 output clocks may increase if the device is exposed to excessive noise on its power pins. For optimal jitter performance, the ZL30310 device should be isolated from noise on power planes connected to its 3.3 V and 1.8 V supply pins, as shown in Figure 7. The following common layout practices are recommended for improved power rail noise rejection:

- "power islands” should be created for the device, for the 3.3 V and for the 1.8 V. A power island is a local copper area, separated from the main power plane by a series passive component. Its purpose is to provide improved isolation from noise on the board power planes. Ferrite beads provide additional suppression of digital switching noise generated by other integrated circuits connected to the main power planes. A recommended bead is LI0805H121R or similar. Note that beads have some DC resistance which increases the minimum required supply voltage for the device (by about 1% for the above bead).
- Each power island should be provisioned with bulk capacitors of a low-ESR 220 µF Tantalum and 10 µF Ceramic. For this configuration, broadband (20 Hz to 20 MHz) input power noise should not be greater than $20mV_{pk-pk}$ on the power supply side of the ferrite element.
- A 0.1 µF decoupling capacitor (ceramic X5R or X7R) must be allocated for each power pin and placed as close as possible to the via connected to the power pin. The smallest available package size should be used. Each decoupling capacitor should be connected directly to only one power pin, and should not share vias to power or ground with other capacitors. Device size should be EIA 0402 or smaller for best high-frequency response and to facilitate optimal placement.
- Priority should be given to placement of decoupling capacitors in nearest proximity to power pin groups $AV_{DD33}$, and $AV_{DD18}$.
- In addition, $AV_{DD18}$ pin sub-groups B1, C2 and T14 require specific RC filters as shown in Figure 7. The location requirements for 0.1 µF decoupling capacitors on these pins takes priority over the location for source terminators or any other components.
- There may be conflicts for “best” placement between capacitors and source termination resistors. It is important to use size 0402 or smaller capacitors to minimize the occurrence of such issues.
- The ball C3 ($AV_{DD18}$) can be connected to the $AV_{DD18}$ plane, however it is preferred for this ball to be connected to the $V_{DD18}$ plane in a star type connection as shown Figure 7.
- The balls B2 and B3 ($AV_{DD33}$) can be connected to the $AV_{DD33}$ plane, however it is preferred for these balls to be connected to the $V_{DD33}$ plane in a star type connection as shown Figure 7.

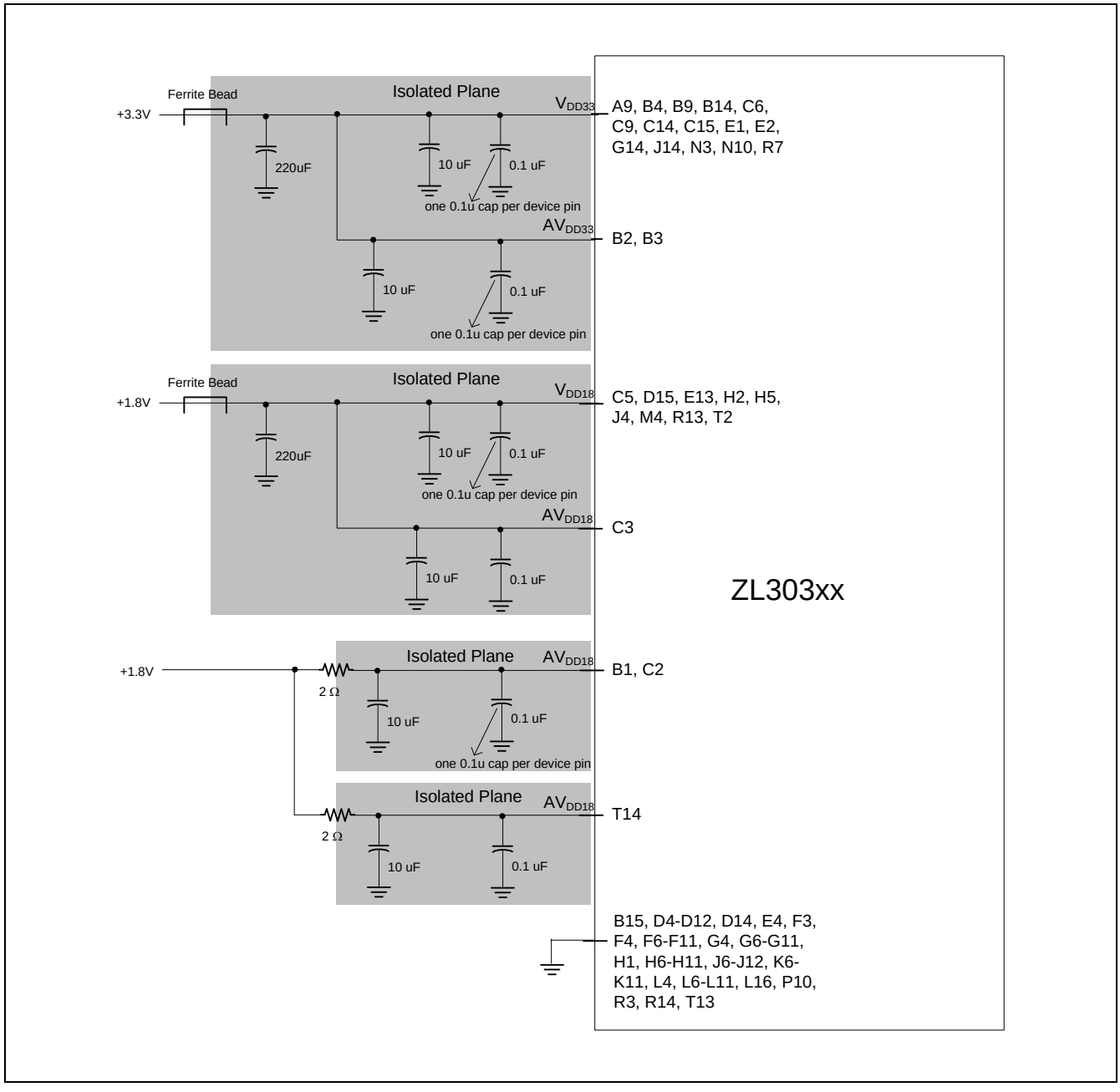



**Figure 7 - Power Supply Decoupling for the ZL30310**

## 2.8 Non-connects

| Ball # | Name | Type | No. | Description |
|---|---|---|---|---|
| **Internal and non-connects** | | | | |
| A6 A7 A10 C16 D16 J1 J2 M6 M7 M10 M11 N12 P11 | IC | | 12 | **Internally connected, leave open** |
| E14 P14 | IC (tie low) | | 2 | **Internally connected, tie to $V_{SS}$** |
| A1 A2 A3 A4 A13 B5 B6 B7 B8 B10 B11 B12 B13 C1 C4 C7 C8 C10 C11 C12 C13 D13 E3 E5 E6 E7 E8 E9 E10 E11 E12 F5 F12 G5 H3 H4 J3 J5 K4 K5 L5 M5 M12 M15 N2 N4 N5 N13 N15 P2 P3 P13 R1 R2 R15 R16 T15 T16 | NC | | 58 | **No connection, leave open** |

## 3.0 Modes of Operation

The ZL30310 can operate in four primary modes:

- as a conventional PLL
- as a timing server
- as a timing client
- as a boundary clock

It can also function in combinations of the different modes listed above, e.g. a combined PLL and time server, or a combined PLL and time client. Figure 8 shows an application diagram of the ZL30310 operating in server, client or boundary clock modes.

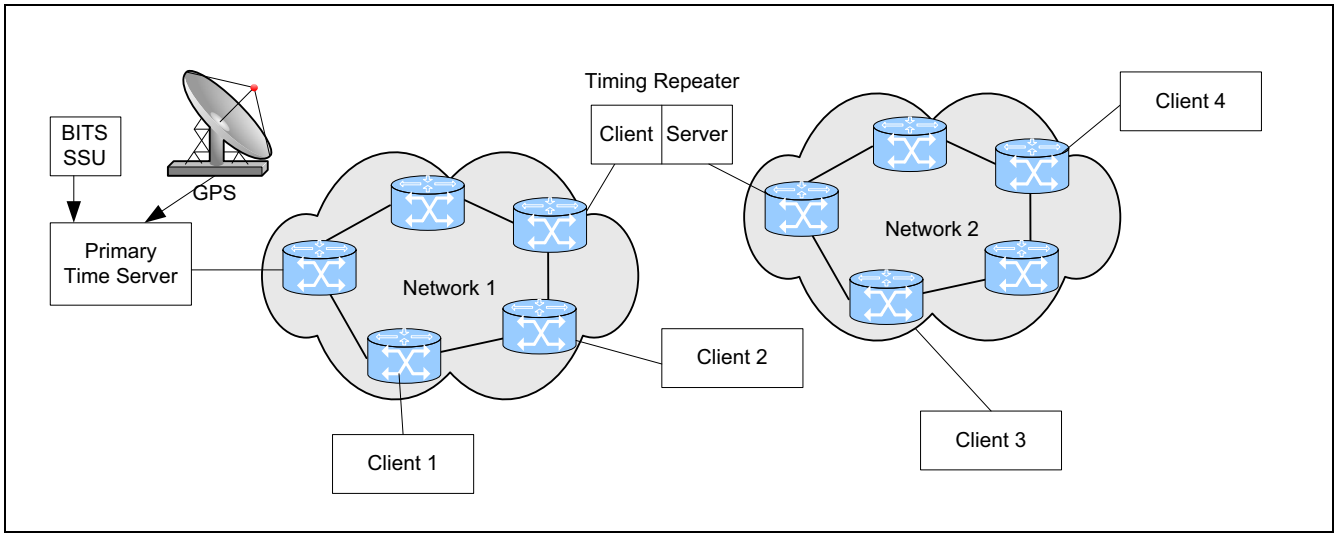



**Figure 8 - ZL30310 Operating Modes**

### 3.1 Time Server Operation

The ZL30310 is capable of transmitting network synchronization over Ethernet, IP and MPLS networks using both Synchronous Ethernet and Timing over Packet (ToP) techniques. It accepts a reference clock in the frequency range 8 kHz to 125 MHz, and generates a timestamp and Ethernet clock locked to this reference. It is also capable of accepting a low frequency alignment signal (e.g., a 1 pulse per second such as might be provided by a GPS reference clock, or an 8 kHz TDM framing pulse) and aligning the timestamp with this signal.

The ZL30310 generates the clock to the Ethernet PHY device at either 25, 50, 62.5 or 125 MHz depending on the requirements. This clock meets the wander requirements of synchronous Ethernet clock.

For ToP, the host microprocessor generates streams of packets in either the industry-standard IEEE1588™ “PTP” format (Precision Time Protocol), or the RTP format (Real-time Transport Protocol, RFC3550), which is compatible with Microsemi’s first generation Timing over Packet devices (ZL30301 and ZL30302). As these packets pass through the ZL30310 device, an accurate timestamp is inserted into the packet denoting the exact time of transmission of the packet into the network.

These timing packets are either broadcast to all devices in the network, multicast to a number of selected devices (i.e., those in the addressed multicast group), or unicast to a number of separate client devices. Typical packet rates are in the range 16 - 64 packets per second. The server may also receive timing messages from clients, requesting the server to respond with the time of arrival of the message. The ZL30310 device timestamps such messages on arrival, and forwards the timestamp to the host for it to generate the appropriate response.

Figure 9 shows an example of a Time Server connected to two different primary reference sources, a GPS and a BITS or SSU synchronization source. The PLL function in the device is used to acquire both references and select the primary source, switching between them in the event of a reference failure. The PLL generates the high frequency clock needed to drive the timestamp engine, locked to the input reference. The timestamp may be phase aligned to the framing signal, allowing the framing signal at the client to be aligned to that of the server. The device also generates the clock for the Ethernet PHY, and this may be used as a Synchronous Ethernet reference.

The GPS device produces two highly stable clock outputs, a 10 MHz sine wave and a 1 pulse per second digital signal accurately indicating the start of each new second. The ZL30310 is capable of taking both of these signals directly, and locking the timestamp engine to them. The GPS device also outputs a time-of-day message on an RS232 link, which can be connected to the host processor. This allows the host processor to align the timestamp to UTC (Universal Time Code), such that the time server can distribute not just frequency and phase, but also accurate time.

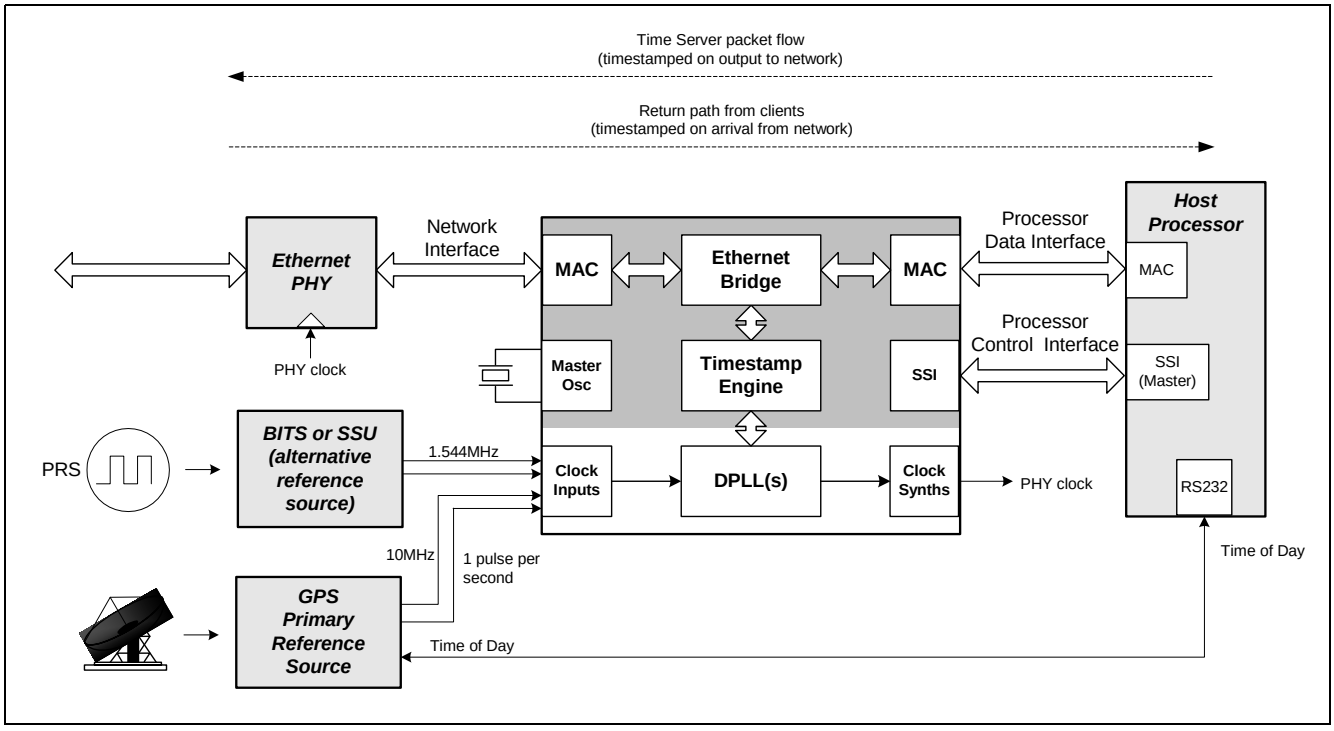



**Figure 9 - Example ZL30310 Time Server driven from a GPS or BITS/SSU reference**

## 3.2 Client Operation

In client mode, the ZL30310 can recover a clock from two independent ToP time servers or two Synchronous Ethernet references. In the event that the primary time server fails, the device can switch to the alternative server or to a conventional clock reference with no phase discontinuity. This may be utilized as part of a redundancy strategy, to minimize the effect of failure of the reference clock or its distribution path. It may use both Synchronous Ethernet and ToP technology simultaneously, by using Synchronous Ethernet to provide an extremely accurate frequency reference, and ToP to provide phase and time synchronization.

The quality of the recovered clock from a ToP server is targeted to meet ANSI standard T1.101 and ITU-T standards G.823 and G.824 for synchronization distribution over an appropriate network. It maintains a mean frequency accuracy of better than ±10 ppb and time alignment of better than ±1 μs when operated over a suitable network.

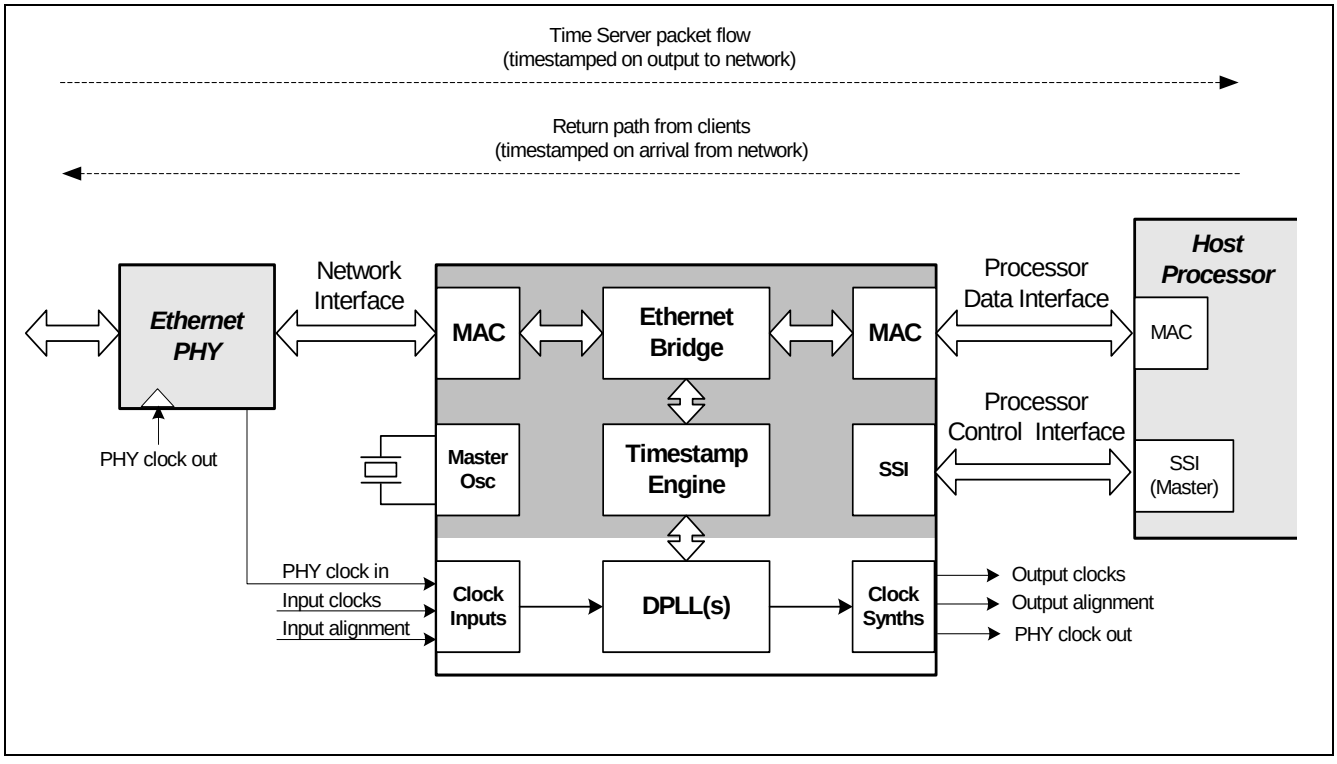



**Figure 10 - Example ZL30310 Timing Client**

## 3.3 Boundary Clock Mode of Operation

The ZL30310 can also function as a boundary clock. This feature is very useful if there is a need for the synchronization information to be transmitted over a large network, or where sections of the network are synchronous and sections asynchronous. For ToP systems, not only is the volume of timing packets reduced around the time server, reducing congestion, but the quality of the recovered clock is improved by breaking the trail through the packet network into two shorter segments.

For example, in Figure 8, at the boundary clock node the ZL30310 will work as a client node for the Network 1 and as a server node for the Network 2. The device simultaneously operates in both server and client modes to achieve the boundary clock function.

The ZL30310 is capable of implementing several different types of boundary clock configurations:

- Synchronous Ethernet to Synchronous Ethernet *(i.e., an intermediate node)*
- Synchronous Ethernet to IEEE1588 *(extending Synchronous Ethernet across an asynchronous network using IEEE1588)*
- Combined Synchronous Ethernet / IEEE1588 to combined Synchronous Ethernet / IEEE1588
- Combined Synchronous Ethernet / IEEE1588 to solely IEEE1588
- IEEE1588 to IEEE1588 *(simple 1588 boundary clock improves performance over large networks)*
- IEEE1588 to Synchronous Ethernet *(allows bridging of non-synchronous network sections)*
- IEEE1588 to Combined Synchronous Ethernet / IEEE1588 *(allows bridging of non-synchronous network sections, including preservation of phase/time information)*

The diagram in Figure 11 shows a boundary clock node which is capable of implementing all the above types of operation.

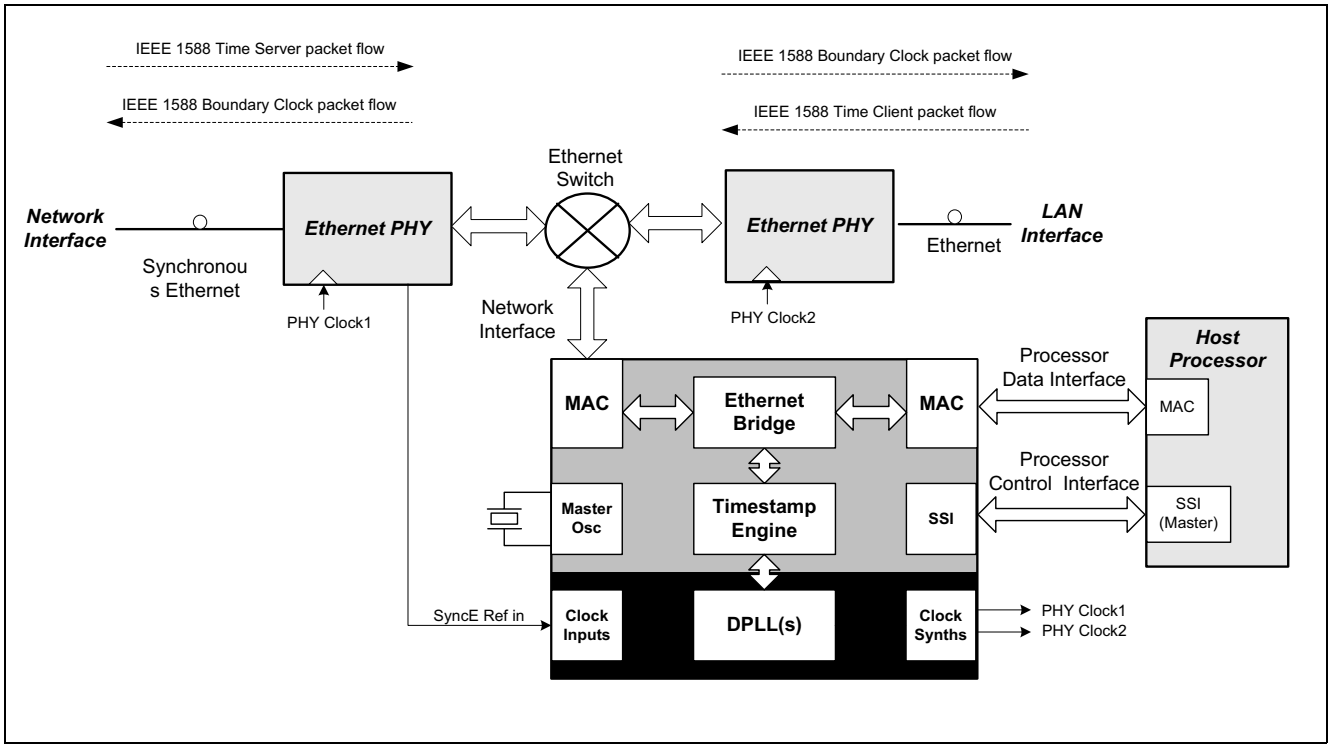



**Figure 11 - ZL30310 Boundary Clock**

## 3.4 Timing Redundancy and Holdover

The ZL30310 can recover a clock from two independent time servers. In the event that the primary time server fails, or that the packet network between the primary time server and the client, the recovered clock will go into a holdover condition. The device can then switch to the alternative server or to an electrical clock reference with no phase discontinuity. This may be utilized as part of a redundancy strategy, to minimize the effect of failure of the reference clock or its distribution path.

Various statistics on the status or the quality of the recovered clocks are available to base the choice of clock on.It is possible to switch the clock automatically to the best available server stream, or to switch manually once a failure has occurred.

## 4.0 Functional Description

The ZL30310 consists of the following main functional components, as shown in Figure 1:

- Ethernet Bridge and MAC Interfaces
- Timestamp Engine
- Digital Phase Locked Loop (DPLL)
- Synchronous Serial Interface
- Timing Software (running on the Host Microprocessor)

### 4.1 Ethernet Bridge and MAC Interfaces

Data packets (e.g., timing packets) are passed between the host microprocessor and the network via the Ethernet bridge in the Microsemi device. Timing packets are identified and the time that they exited or entered the network interface is recorded very accurately using the timestamp engine. This time is inserted into the packets, yielding the precise time of transmission or reception. The necessary UDP checksums or Ethernet FCS values are updated on-the-fly when the timestamp is inserted.

#### 4.1.1 Overview of Bridge Operation

The diagram in Figure 12 shows the bridge operation:

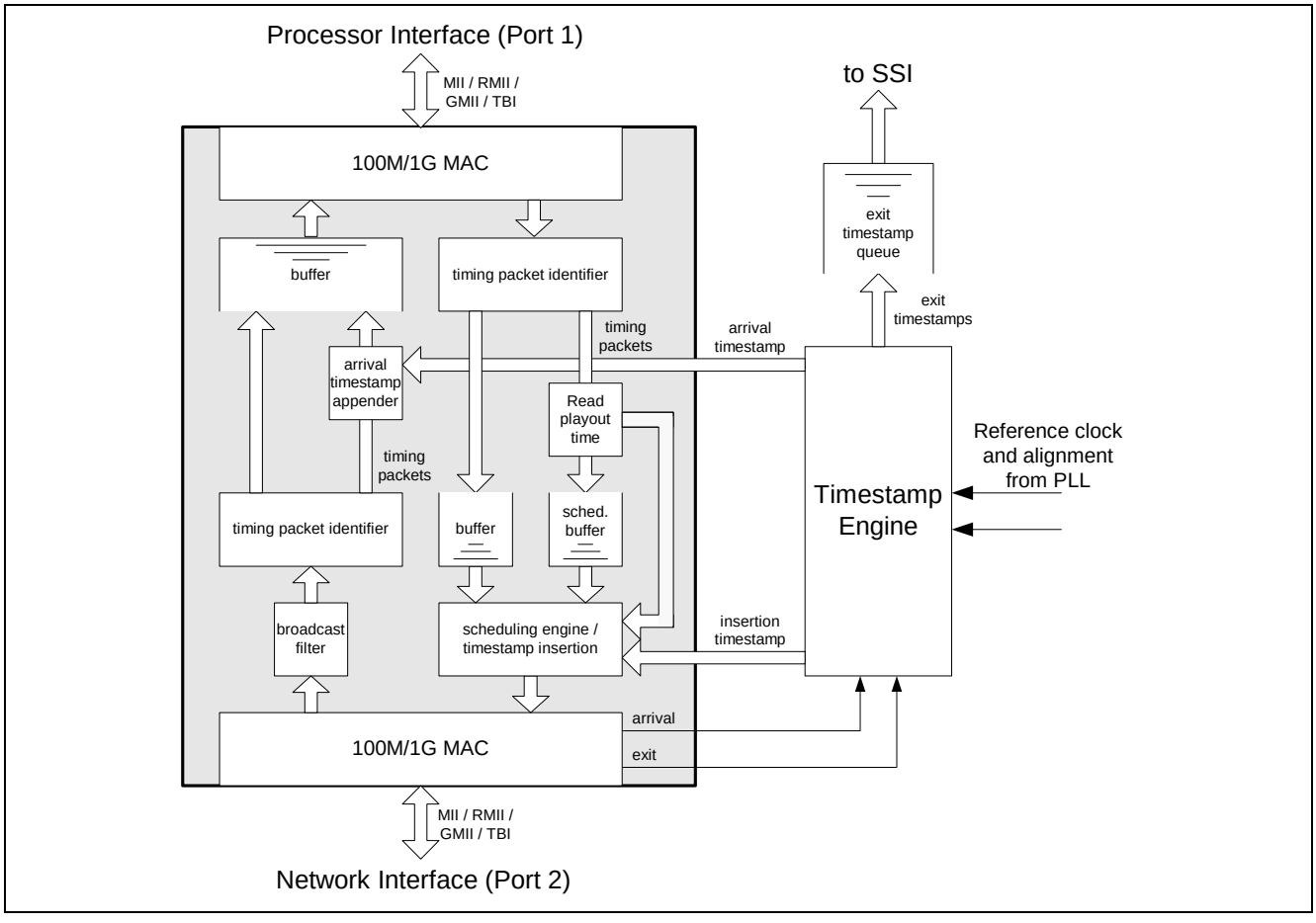



**Figure 12 - Ethernet Bridge Structure**

**Packets Arriving from the Network Interface**

Packets arriving from the network (bottom interface in Figure 12) come in through the network interface MAC (port 2), and are passed to a broadcast filter, which limits the admission rate of broadcast packets to protect against broadcast storms. They then pass to the timing packet identifier. Non-timing packets are forwarded straight to the opposite interface (processor interface, top of Figure 12), while timing packets have the precise time of arrival appended to the end of the packet, for use by the timing recovery algorithm. A small buffer collects both the timing and non-timing packets and queues them for exit via the processor interface MAC (port 1).

The precise arrival time is provided by the timestamp engine. A signal from the MAC indicates the exact time that each packet arrives at the interface, and this is used to generate the arrival timestamp for appending to the timing packet.

**Packets Arriving from the Processor Interface**

Packets arriving from the processor interface (port 1) are also split into timing and non-timing packets. Non-timing packets are forwarded straight to the network interface, while timing packets are forwarded into a separate buffer for scheduling. Timing packets may be either scheduled for playout, or played out immediately. If they are scheduled, the required playout time must be read from the packet and passed to the scheduling engine. The packet is then played out on the network interface (port 2) as close to the required time as possible. A precise timestamp is also inserted into the packet on playout indicating the actual time the packet is transmitted into the network.

Simultaneously with packet transmission, the exit time of the packet is forwarded into a small queue to be read by the processor over the SSI. This is required to enable follow-up messages in a timing server, and to provide the transmission time of delay request messages in a timing client.

### 4.1.2 Handling of Timing Packets

#### 4.1.2.1 Classification

The classification engine determines whether packets are timing packets or non-timing packets. It checks the packet header to examine the protocol stack. To allow for variations in the protocol stack, the classifier is a very simple “brute-force” mask and match comparator across the first 64 bytes of the packet. This is sufficient to allow for variations such as presence or absence of VLAN tags, use of IPv4 or IPv6, operation over MPLS, operation directly over Ethernet etc. The classifier can be configured to support one protocol stack at a time.

It is important to note that the comparator is simply a binary process: it only identifies whether a packet is a timing packet intended for a specific device or not. It does not attempt to identify flows or message types within that category.

Two classification rules are provided, to allow for differences in the protocol stack in each direction (e.g. the reversal of source and destination addresses, or the presence of VLAN tags in one direction only. These two rules are programmed using the API call ***zl303xx_LanWritePktClassifyRule***.

The mask register allows each bit of the header to be individually included or excluded from the comparison. Where a mask bit is set to 1, the corresponding bit of the match register is excluded. This allows the rule to be constructed such that any field or bit of the header can be included or excluded from the classification process.

#### 4.1.2.2 Timestamp Insertion

Once a packet has been identified as a timing packet, it may be processed in a number of possible ways.

**Timing Packets from Port 1 to Port 2**

For timing packets travelling from port 1 to port 2 (Processor Interface to Network Interface), a temporary field at a known location in the packet header indicates how to the handle the packet. There are three possible actions, which may be taken singly or in any combination:

1. Record the precise time of exit for use in a follow-up message
2. Insert the precise time of exit into a pre-determined timestamp location of the packet header
3. Schedule the packet for playout at a particular time, given by a timestamp in the packet header.

If the action is to insert a timestamp into the packet, this is inserted at a pre-determined location in the packet header. Typically this will overwrite the temporary action field, such that the resulting packet is a legal timing message. The device also updates the UDP checksum (if present) and the Ethernet Frame Check Sequence (FCS) as the packet is transmitted out of the network interface. The precision of the exit timestamp inserted into the packet is to within 52.5 ns (MII/RMII) and 20.5 ns (GMII/TBI).

The timestamp inserted into the packet can be in either PTP format (64 bits wide), NTP format (64 bits wide), or RTP format (32 bits wide). If RTP is used, the timestamp represents an unsigned 32-bit count of 10 MHz clock periods, starting from a random value. It should be noted that the initial software released with the device only operates with PTP (IEEE1588$^{TM}$-2007). An RTP version is planned for backwards compatibility with Microsemi's earlier Timing over Packet family (ZL30301 and ZL30302). It is not currently planned to produce an NTP version.

The locations of the temporary action field and where the timestamp is to be inserted are determined by the API call ***zl303xx_LanConfigTxTsControl***. The location of the UDP checksum is determined by the call ***zl303xx_LanConfigTxUDPChksum***.

If follow-up recording is enabled, the precise time of exit is recorded in a 64-entry FIFO queue, along with an index number uniquely identifying the packet. This can be read by the CPU for use in a follow-up message, or at a client for reading the time of transmission of a delay_request message.

**Timing Packets from Port 2 to Port 1**

For timing packets travelling from port 2 to port 1 (Network Interface to Processor Interface), the precise time of reception at port 2 is recorded. The precision of the reception timestamp inserted into the packet is to within 52.5 ns (MII/RMII), or 20.5 ns (GMII/TBI).

This time is inserted or appended to the packet at a pre-determined location. Normally it is not desired to overwrite any of the fields within the timing packet, therefore this is written into the Ethernet frame beyond the IP datagram. The software retrieves this information from the Ethernet frame when it processes the packet. The UDP checksum is not updated in this direction, since the timestamp is normally beyond the datagram and therefore doesn't contribute to the checksum. The Ethernet FCS is updated as the packet is transmitted out of port 1.

The location in the packet where the timestamp is to be inserted is determined by the API call ***zl303xx_LanConfigRxTsControl***.

#### 4.1.2.3 Timing Packet Scheduling

In some situations it is required to schedule a packet for transmission at a particular time. The device allows up to 64 packets to be queued for transmission. This is done by setting the scheduling bit in the temporary field, and inserting a transmission time into the timestamp field. The packet is then held for transmission until that scheduled time. The scheduler will transmit the packet within 1 µs of the scheduled time on a gigabit network. On transmission, the actual transmission time can be inserted into the packet, or a follow-up time can be recorded for later transmission.

This enables the software to control the rate and time of transmission much more precisely than is possible in software alone. The feature is useful in certain algorithms where the time of transmission of packets is important.

Packet scheduling is not controlled by the API. It is used by the clock recovery algorithms to improve the quality of the recovered clocks at the clients.

### 4.1.3 Handling of Non-Timing Packets

Non-timing packets are those that fail the classification check described in section 4.1.2.1. These packets are forwarded directly to the buffers awaiting transmission to the opposite port. Non-timing packets are not processed in any way, subject to rate control or filtered, with the exception of broadcast or ARP packets. The Microsemi device acts as a completely transparent pass-through bridge.

Non-timing packets may be dropped if the buffer memory overflows. Dropping may be avoided by the effective use of flow control (see section 4.1.4.2, “Flow Control“). If both ports operate at the same nominal rate, the buffers are sufficiently large to avoid packet dropping in most circumstances.

#### 4.1.3.1 Broadcast and ARP filtering

The device has the ability to drop broadcast and/or ARP packets above a given rate. This protects the CPU from the “broadcast storms” which sometimes occur, e.g., after a network re-configuration. This is programmed in terms of the maximum number of broadcast or ARP packets in a given timebase.

The timebase can be set in units of $2^n$ * 100 μs (where n is from 1 to 7), with the maximum number of packets being set between 0 and 255. Above that rate, all broadcast or ARP packets are dropped by the device.

### 4.1.4 Queuing System and Buffer Management

There are three principal queues in the device Ethernet Bridge:

- queue to port 1
- queue to port 2
- scheduled timing packet queue

These three queues use a shared memory architecture for efficiency and flexibility. This enables the memory to be allocated flexibly to each queue on demand. The total buffer memory available to the Ethernet bridge is 32 Kbytes. This is allocated in 128 byte segments, accommodating a maximum of 256 packets.

#### 4.1.4.1 Packet Dropping

In a shared memory architecture, one particular queue may become full due to congestion, and request all the free buffer space. This prevents the other queues from working normally, since there may be no more granules to request. In order to prevent this, each queue has a reserved number of memory granules allocated to it, which cannot be used by the other queues. The remainder of the granules are free for allocation on demand to whichever queue needs them.

When a particular queue reaches its reserved buffer space, it continues to request additional granules as necessary until the free pool is used up. At this point, there are no more granules available for request, and the queue will drop any further packets that arrive until the space is freed up by packets being transmitted out of the port. However, even though that queue is dropping packets, the other queues may still be able to accept packets normally, because they may be operating within their reserved buffer size.

#### 4.1.4.2 Flow Control

If flow control is enabled, it is activated when the buffers to the given port reach a programmable threshold size. Separate thresholds are available for the queues to port 1, port 2 and the scheduler.

The queues may back up for a number of reasons. For example, the receiving device on port 2 may send a flow control frame (also known as a pause frame) to pause the flow of packets from the Microsemi device. Since port 2 can no longer send packets, its queue will start to fill up. When it reaches the threshold value, the device will send a flow control frame out on port 1, to prevent any further packets from reaching the device before the queue fills up completely and packets start to be dropped.

When the congestion on port 2 clears and the flow control is released, the queues to port 2 will empty, and the flow control on port 1 will be released.

### 4.1.5 MAC Interfaces

The two Ethernet MAC interfaces can be independently configured to operate at either 100 Mbit/s or 1 Gbit/s, using the industry standard MII, RMII, GMII or TBI interfaces. These interfaces can be connected either to a PHY device, or directly to another MAC. Each interface can be configured in “PHY emulation mode” where the interface emulates a PHY device to the opposing MAC. The management interface can also be configured as either an STA (Station Management) or PHY device.

When operating as a PHY, the MII transmit clocks (M1_TXCLK or M2_TXCLK) become outputs, and the interface supplies the timing to the opposing MAC. This clock is used to time both the transmit and receive data. In this mode the receive clocks (M1_RXCLK and M2_RXCLK) are not used. PHY emulation mode may be configured separately for each interface.

The clock generation module in the Microsemi device can generate Ethernet clocks at 25, 50, 62.5 or 125 MHz in order to remove the requirement for a separate oscillator at the PHY device.

#### 4.1.5.1 GMII/TBI/MII/RMII Connectivity

**Signal Mapping and Internal Pull-Up/Down Configuration**

The device Gigabit Ethernet ports support the following interface options: GMII, TBI, MII & RMII. The table below summarizes the interface signals required for each interface, and how they relate back to the Pin Symbol name.

| Pin Symbol | GMII Mode | TBI Mode | MII Mode (MAC) | MII Mode (PHY) | RMII Mode |
|---|---|---|---|---|---|
| Mn_REFCLK | (optional)<br>125 MHz (I) | (optional)<br>125 MHz (I) | NC (U) | NC (U) | Mn_REFCLK (I) |
| Mn_RXCLK | Mn_RXCLK (I) | Mn_RBC0 (I) | Mn_RXCLK (I) | NC (U) | NC (U) |
| Mn_COL | Mn_COL (I) | Mn_RBC1 (I) | Mn_COL (I) | Mn_COL (O) | NC (U) |
| Mn_RXD0 | Mn_RXD0 (I) | Mn_RXD0 (I) | Mn_RXD0 (I) | Mn_RXD0 (I) | Mn_RXD0 (I) |
| Mn_RXD1 | Mn_RXD1 (I) | Mn_RXD1 (I) | Mn_RXD1 (I) | Mn_RXD1 (I) | Mn_RXD1 (I) |
| Mn_RXD2 | Mn_RXD2 (I) | Mn_RXD2 (I) | Mn_RXD2 (I) | Mn_RXD2 (I) | NC (U) |
| Mn_RXD3 | Mn_RXD3 (I) | Mn_RXD3 (I) | Mn_RXD3 (I) | Mn_RXD3 (I) | NC (U) |
| Mn_RXD4 | Mn_RXD4 (I) | Mn_RXD4 (I) | NC (U) | NC (U) | NC (U) |
| Mn_RXD5 | Mn_RXD5 (I) | Mn_RXD5 (I) | NC (U) | NC (U) | NC (U) |
| Mn_RXD6 | Mn_RXD6 (I) | Mn_RXD6 (I) | NC (U) | NC (U) | NC (U) |

**Table 2 - Gigabit Ethernet Ports Signal Mapping in Different Operation Mode**

| Pin Symbol | GMII Mode | TBI Mode | MII Mode (MAC) | MII Mode (PHY) | RMII Mode |
|---|---|---|---|---|---|
| Mn_RXD7 | Mn_RXD7 (I) | Mn_RXD7 (I) | NC (U) | NC (U) | NC (U) |
| Mn_RXDV | Mn_RXDV (I) | Mn_RXD8 (I) | Mn_RXDV (I) | Mn_RXDV (I) | Mn_CRS_DV (I) |
| Mn_RXER | Mn_RXER (I) | Mn_RXD9 (I) | Mn_RXER (I) | Mn_RXER (I) | NC (U) |
| Mn_CRS | Mn_CRS (I) | Mn_SIG_DET (I) | Mn_CRS (I) | Mn_CRS (O) | NC (U) |
| Mn_TXCLK | Mn_TXCLK (I) | NC (U) | Mn_TXCLK (I) | Mn_TXCLK (O) | NC (U) |
| Mn_TXD0 | Mn_TXD0 (O) | Mn_TXD0 (O) | Mn_TXD0 (O) | Mn_TXD0 (O) | Mn_TXD0 (O) |
| Mn_TXD1 | Mn_TXD1 (O) | Mn_TXD1 (O) | Mn_TXD1 (O) | Mn_TXD1 (O) | Mn_TXD1 (O) |
| Mn_TXD2 | Mn_TXD2 (O) | Mn_TXD2 (O) | Mn_TXD2 (O) | Mn_TXD2 (O) | NC (O) |
| Mn_TXD3 | Mn_TXD3 (O) | Mn_TXD3 (O) | Mn_TXD3 (O) | Mn_TXD3 (O) | NC (O) |
| Mn_TXD4 | Mn_TXD4 (O) | Mn_TXD4 (O) | NC (O) | NC (O) | NC (O) |
| Mn_TXD5 | Mn_TXD5 (O) | Mn_TXD5 (O) | NC (O) | NC (O) | NC (O) |
| Mn_TXD6 | Mn_TXD6 (O) | Mn_TXD6 (O) | NC (O) | NC (O) | NC (O) |
| Mn_TXD7 | Mn_TXD7 (O) | Mn_TXD7 (O) | NC (O) | NC (O) | NC (O) |
| Mn_TXEN | Mn_TXEN (O) | Mn_TXD8 (O) | Mn_TXEN (O) | Mn_TXEN (O) | Mn_TXEN (O) |
| Mn_TXER | Mn_TXER (O) | Mn_TXD9 (O) | Mn_TXER (O) | Mn_TXER (O) | NC (U) |
| Mn_GTXCLK | Mn_GTXCLK(O) | Mn_GTXCLK(O) | NC (O) | NC (O) | NC (O) |
| Mn_MDC | Mn_MDC (O) | Mn_MDC (O) | Mn_MDC (O) | Mn_MDC (I) | Mn_MDC (O) |
| Mn_MDIO | Mn_MDIO (IO) | Mn_MDIO (IO) | Mn_MDIO (IO) | Mn_MDIO (IO) | Mn_MDIO (IO) |

**Table 2 - Gigabit Ethernet Ports Signal Mapping in Different Operation Mode**

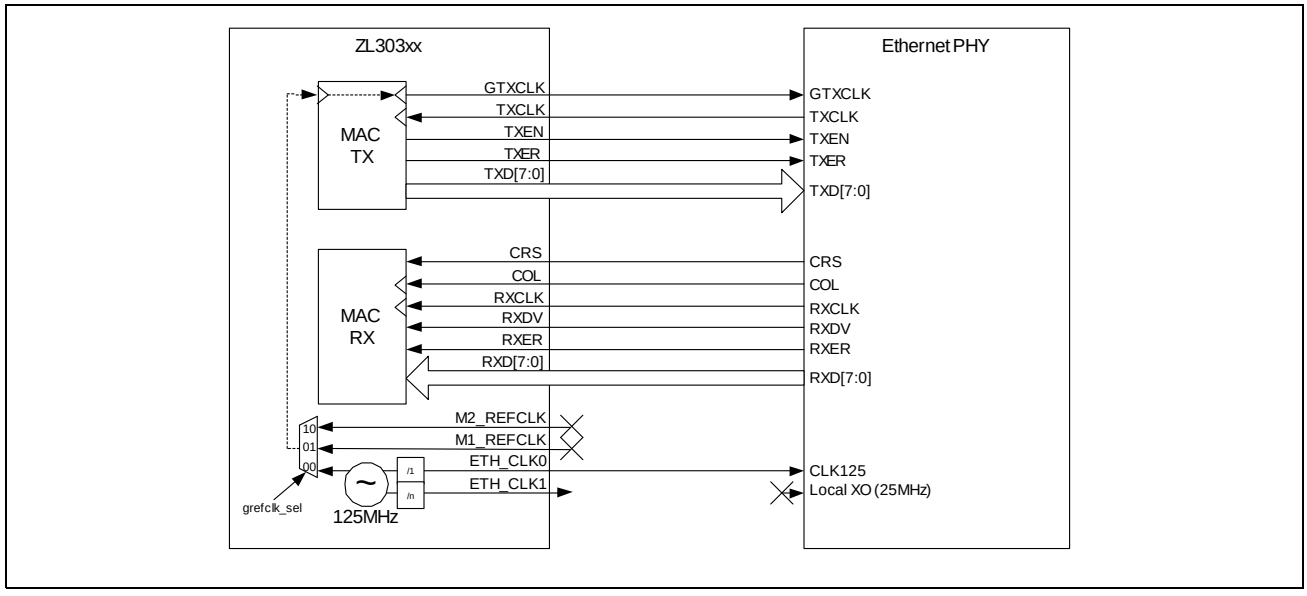



**Figure 13 - GMII Connection (MAC Mode)**

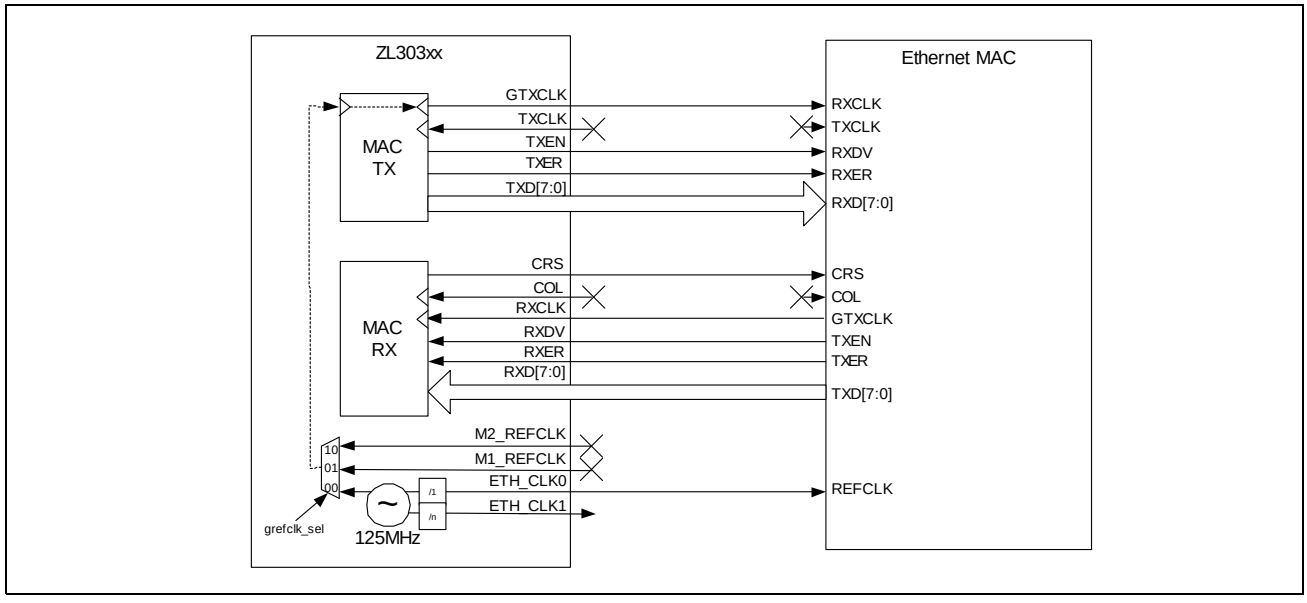



**Figure 14 - GMII Connection (PHY Mode)**

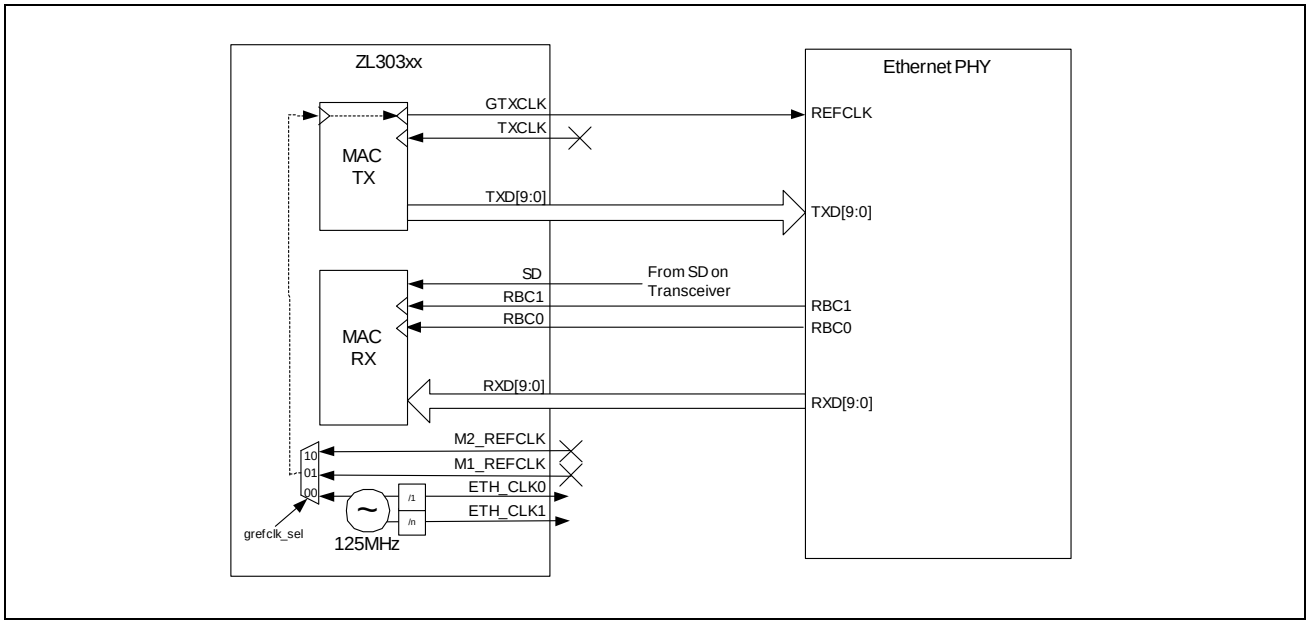



**Figure 15 - TBI Connection**

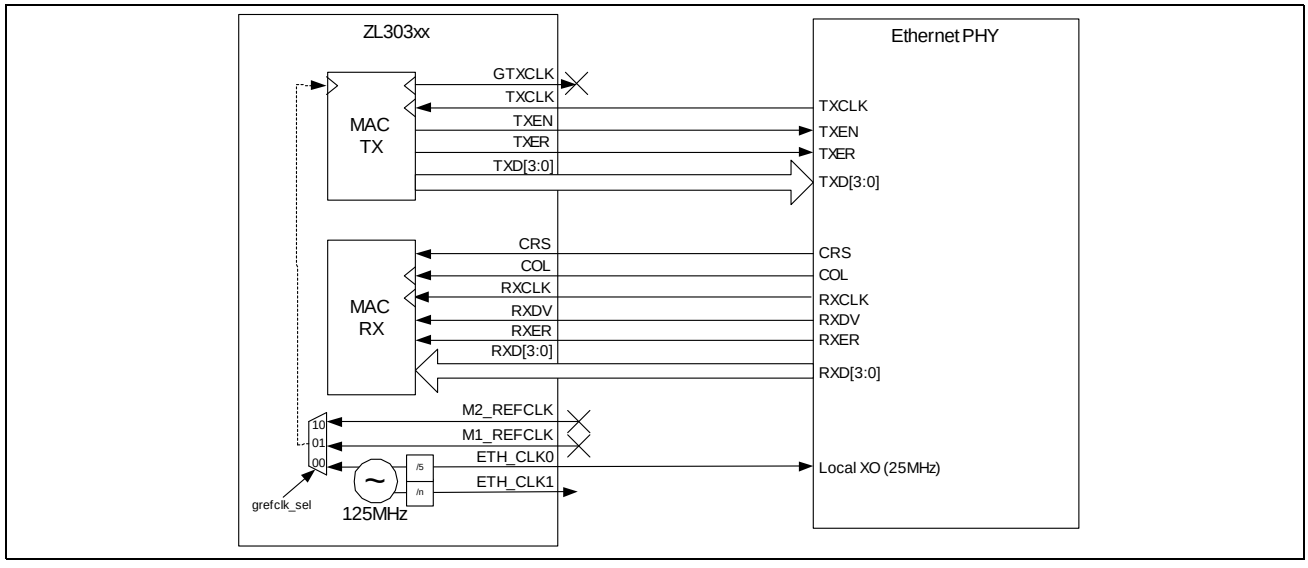



**Figure 16 - MII Connection (MAC Mode)**

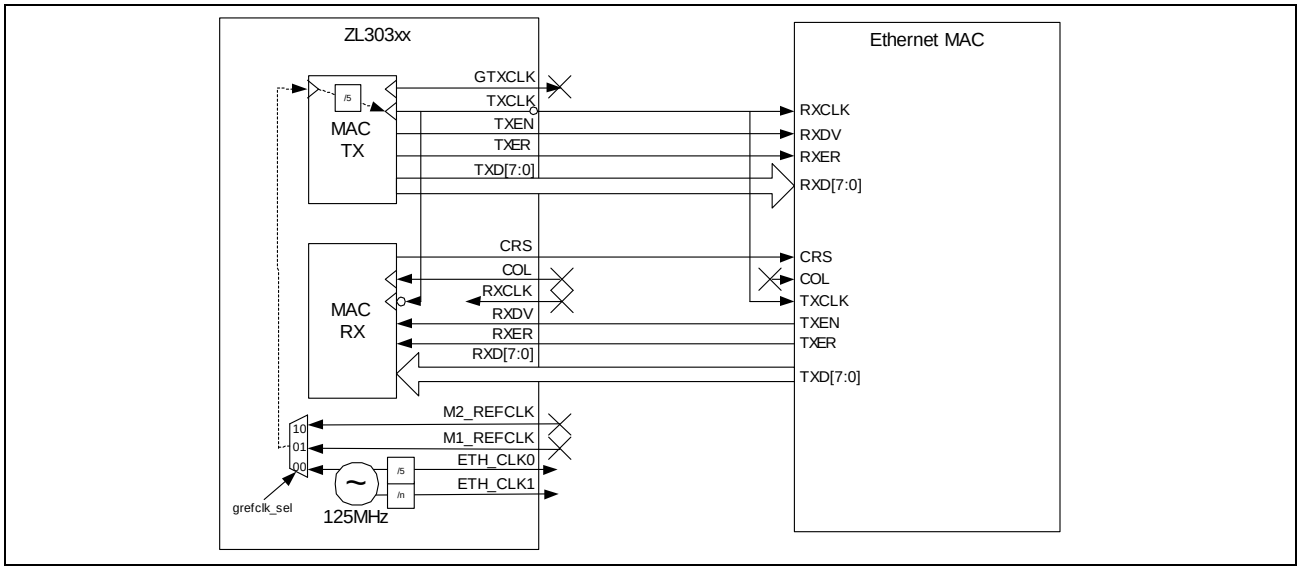



**Figure 17 - MII Connection (PHY Mode)**

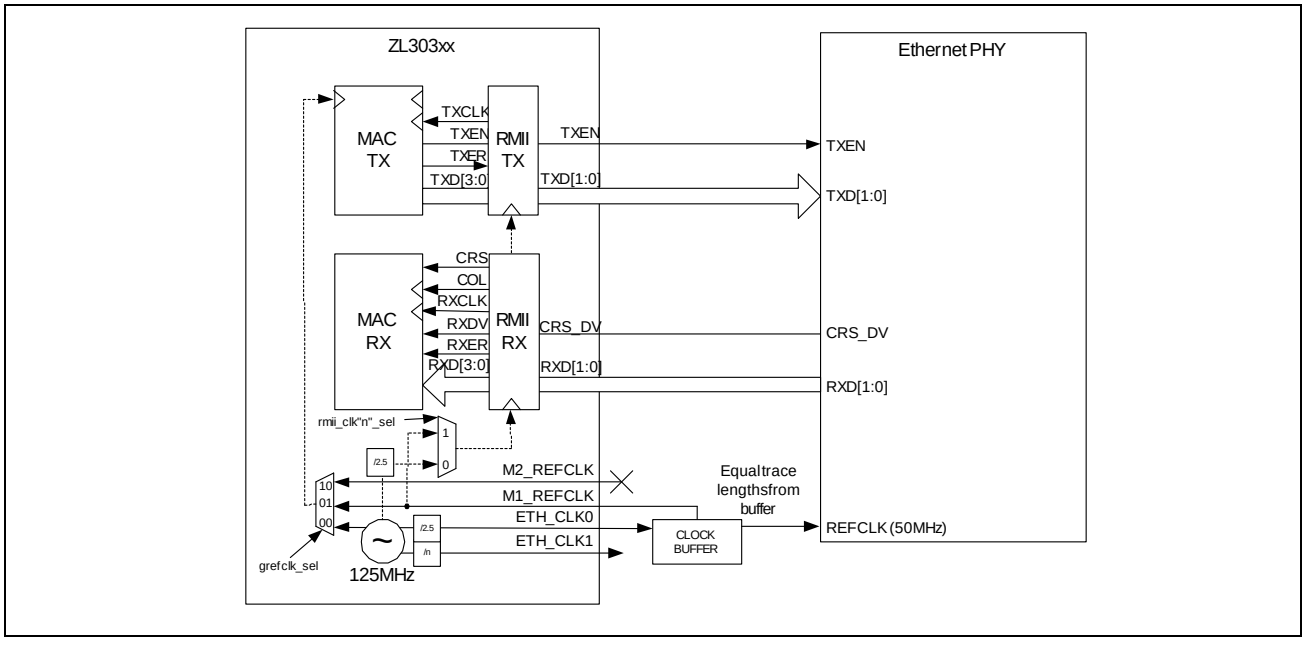



**Figure 18 - RMII Connection (MAC Mode)**

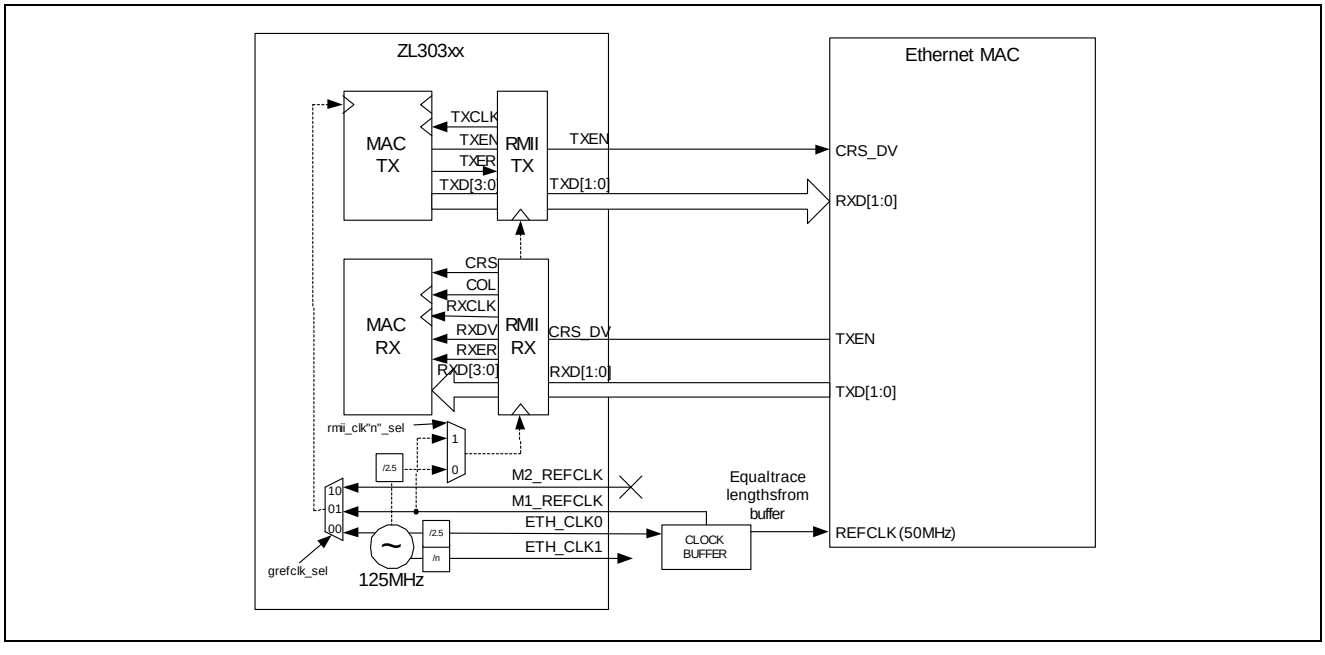



**Figure 19 - RMII Connection (PHY Mode)**

### MAC Module

The Gigabit Ethernet Media Access Control (GE MAC) module provides the necessary buffers and control interface between the Microsemi device and an external device. The GE MAC implements a GMII/MII interface, which offers a simple migration from 10/100M to 1 G. For GE fiber optics media, the device has an integrated Physical Code Sub-layer (PCS) module, which includes an 8B10B encoder and decoder, auto-negotiation, and a Ten Bit Interface (TBI). For reduced signal layout, the device has an integrated Reduced MII (RMII) module, which does a MII to RMII interface conversion.

The MAC of the Microsemi device meets the IEEE 802.3 specification. It is able to operate in 100 M in either Half or Full Duplex mode with a back pressure/flow control mechanism, or in 1000 M in Full duplex mode with flow control mechanism. It is highly recommended to always operate the MAC in full duplex mode.

The MAC ports are denoted as ports 1 and 2.

**Physical Coding Sub-layer (PCS) Module**

The 1000BASE-X PCS module is integrated in the Microsemi device and may be utilized in the absence of GMII. The PCS incorporates all the functions required by the GMII to include encoding (decoding) 8 B GMII data to (from) 8B/10B TBI format for PHY communication. The on-chip PCS may be disabled if a PCS block exists within the PHY.

The PCS comprises the PCS Transmit, Synchronization, PCS Receive and auto-negotiation processes for 1000BASE-X.

- The PCS Transmit process sends the TBI signals TXD[9:0] to the physical medium and generates the GMII Collision Detect (COL) signal based on whether a reception is occurring simultaneously with transmission. Additionally, the Transmit process generates an internal “transmitting” flag and monitors auto-negotiation to determine whether to transmit data or to reconfigure the link.
- The PCS Synchronization process determines whether or not the receive channel is operational.
- The PCS Receive process receives the TBI signals RXD[9:0] from the physical medium, and generates the GMII RXD[7:0] signals and the internal “receiving” flag for use by the Transmit processes.
- The PCS auto-negotiation process allows the Microsemi device to exchange configuration information between two devices that share a link segment and to automatically configure the link for the appropriate speed of operation for both devices.

The TBI interface is connected to the SERDES as shown in Figure 15.

**Reduced MII (RMII) Module**

The RMII module is integrated in the Microsemi device and may be utilized in the absence of MII. The module incorporates all the functions required to convert between the RMII and MII interfaces. In RMII mode the port will not support loop back mode, but port loop back in supported in MII or GMII modes.

### 4.1.5.2 MAC Configuration

The MAC interfaces are configured using the API calls ***zl303xx_LanConfigPort*** and ***zl303xx_LanSetLinkState***. These calls enable the line rate to be set, or auto-negotiation to be configured on the port.

Flow control can also be enabled on each port to be activated when the buffer to that port reaches a given threshold (see section 4.1.5, “MAC Interfaces“). It may be activated in both directions, or asymmetrically, which is where pause frames can be transmitted on the port, but are ignored on reception.

### 4.1.5.3 Link Up/Down Status

If the status of a link changes, an interrupt is generated. The CPU can then read the current status of the link in question, and take appropriate action. This action may include reporting the failure to the management layer, or re-configuration of the link. The API call ***zl303xx_LinkUpDownEventHandler*** is used to handle link status changes.

### 4.1.5.4 Management Link

When the port is configured as a MAC device (as opposed to a PHY), the management interface may be used to configure the PHY device connected to it. Messages on the management interface are set up through registers in the port.

When auto-negotiation is used and a port is in MAC mode, the addresses used to pll the PHY devices are fixed. The PHY device attached to port M1 is address 0x08, and the PHY device attached to port M2 is address 0x09. If user decides to use manual management polling any PHY address can be used.

When one of the two ports is configured in MAC mode, then the management link associated with that port is used to auto-negotiate with its PHY. But, when both ports are configured in MAC mode, then the management link of port M1 is used to auto-negotiate with both PHYs.

| M1<br>Processor Interface | M2<br>Network Interface | MDIO Control |
|---|---|---|
| MAC | PHY | M1 |
| MAC | MAC | M1 |
| PHY | MAC | M2 |
| PHY | PHY | N/A |

**Table 3 - MDIO Control**

#### 4.1.5.5 Statistics

The device maintains statistics counters, sufficient to enable the standard Ethernet MIBs such as RFC1757 to be supported. The statistics collected are:

- Total number of bytes sent
- Number of unicast frames sent
- Number of non-unicast frames sent
- Number of flow-control frames sent
- Number of frame send failures
- Number of bytes received (both good and bad)
- Number of frames received (both good and bad)
- Total number of bytes received
- Total number of frames received
- Number of flow-control frames received
- Number of multicast frames received
- Number of broadcast frames received
- Number of undersize frames received
- Number of frames received with length of 64 bytes
- Number of frames received with length 65 to 127 bytes
- Number of frames received with length 128 to 255 bytes
- Number of frames received with length 256 to 511 bytes
- Number of frames received with length 512 to 1023 bytes
- Number of frames received with length 1024 to maximum
- Number of jabber frames received
- Number of frame fragments received
- Number of frames received with alignment errors
- Number of frames received with FCS errors
- Number of short events
- Number of collisions
- Number of dropped frames

An interrupt is generated to the CPU on overflow of the respective counters, allowing the CPU to keep track of the various statistics for compiling into the MIB format. The state of each of the counters for each port can be read using the API call ***zl303xx_LanGetStats***. Formatting the raw data into a MIB is a function of the application software, and is not provided by the Microsemi solution software.

#### 4.1.5.6 Layout Guidelines

The MAC interfaces passes data to and from the ZL30310 with their related transmit and receive clocks. It is therefore recommended that the trace lengths for transmit related signals and their clock and the receive related signals and their clock are kept to the same length. By doing this the skew between individual signals and their related clock will be minimized.

It is recommended that the outputs are suitably terminated using a series termination through a resistor as close to the output pin as possible. The purpose of the series termination resistor is to reduce reflections on the line. The value of the series termination and the length of trace the output can drive will depend on the driver output impedance, the characteristic impedance of the PCB trace (recommend 50 ohm), the distributed trace capacitance and the load capacitance. As a general rule of thumb, if the trace length is less than 1/6th of the equivalent length of the rise and fall times, then a series termination may not be required.

the equivalent length of rise time = rise time (ps) / delay (ps/mm)

For example:

Typical FR4 board delay = 6.8 ps/mm
Typical rise/fall time for a ZL50110/11/12/14 output = 2.5 ns

critical track length = (1/6) x (2500/6.8) = 61 mm

Therefore tracks longer than 61 mm will require termination.

As a signal travels along a trace it creates a magnetic field, which induces noise voltages in adjacent traces causing crosstalk. If the crosstalk is of sufficiently strong amplitude, false data can be induced in the trace. The voltage that the external fields cause is proportional to the strength of the field and the length of the trace exposed to the field. Therefore to minimize the effect of crosstalk some basic guidelines should be followed.

First, increase separation of sensitive signals, a rough rule of thumb is that doubling the separation reduces the coupling by a factor of four. Alternatively, shield the victim traces from the aggressor by either routing on another layer separated by a power plane (in a correctly decoupled design the power planes have the same AC potential) or by placing guard traces between the signals (usually held ground potential).

Particular effort should be made to minimize crosstalk from ZL30310 outputs and ensuring fast rise time to these inputs.

In Summary:

- Place series termination resistors as close to the pins as possible
- Minimize output capacitance
- Keep common interface traces close to the same length to avoid skew
- Protect input clocks and signals from crosstalk

## 4.2 Timestamp Engine

The Timestamp Engine is integrated with DPLL1. It is driven directly from the main DCO of the DPLL, as shown in Figure 20. This DCO can either be controlled via the DPLL mechanism itself (known as “PLL Mode”), or it can be controlled directly by the CPU (known as “ToP Mode”). Whether operating in PLL or ToP modes, there is no difference to the way the output clocks are generated, or how the timestamps themselves are generated. This enables the device to be switched between a conventional, electrical reference clock (i.e., PLL Mode) to a packet clock (i.e., ToP Mode), or vice versa.

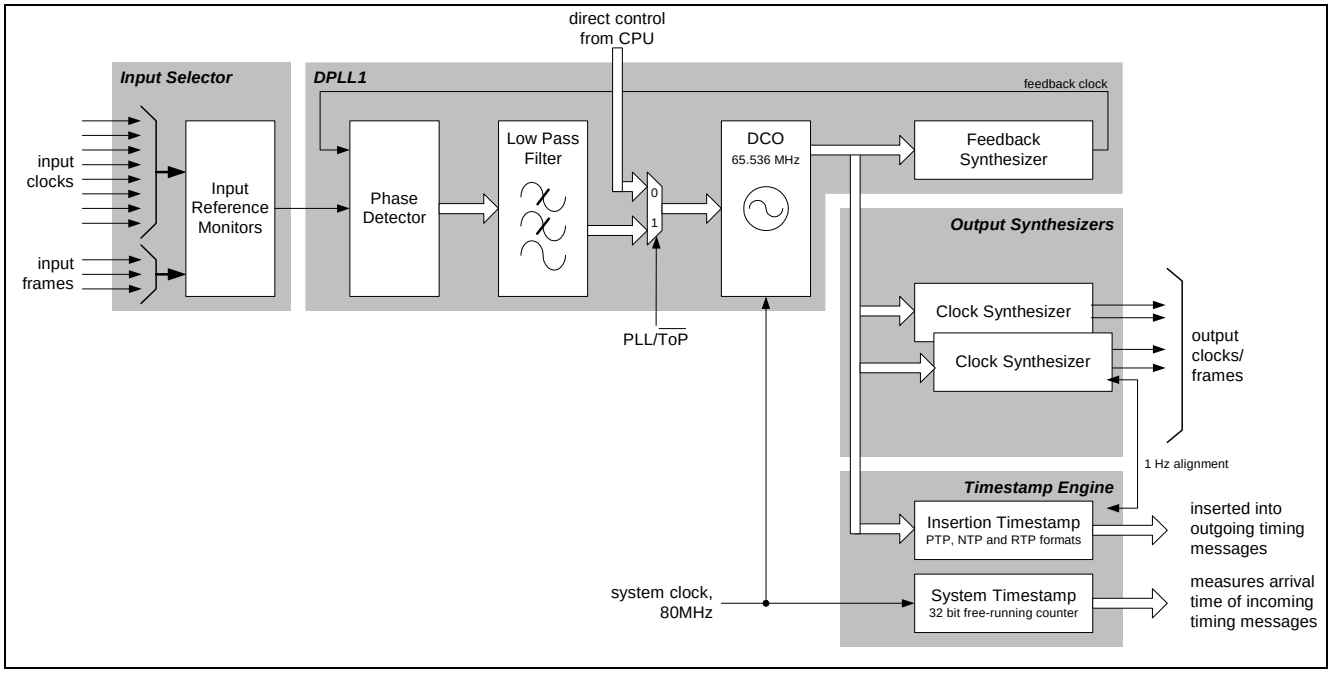



**Figure 20 - Relationship of Timestamp Engine to DPLL**

The timestamp engine contains two independent timestamp counters. The first, known as “system time” is a free-running, 32-bit timestamp driven from the system clock of the device. The second, known as “insertion time” is driven directly from the main DCO of the phase locked loop. This timestamp is in the format of the selected timing protocol, i.e., either PTP (IEEE1588), NTP or RTP. The two timestamps are sampled frequently at precisely the same instant so that the relationship between them is always accurately known.

Arrival time events are always captured in system time, so they are unaffected by any modulation of the PLL frequency. However, packets leaving the device are timestamped with timestamps related to insertion time. Insertion timestamps are formatted according to the IEEE 1588 timestamp format.

### 4.2.1 Server Operation

When used as a time server for ToP operation, the device functions as a conventional PLL, locking onto the input reference clock. The timestamps generated by the Insertion Timestamp counter are therefore locked to the input reference. These are inserted into PTP sync messages as they exit the device on the network interface.

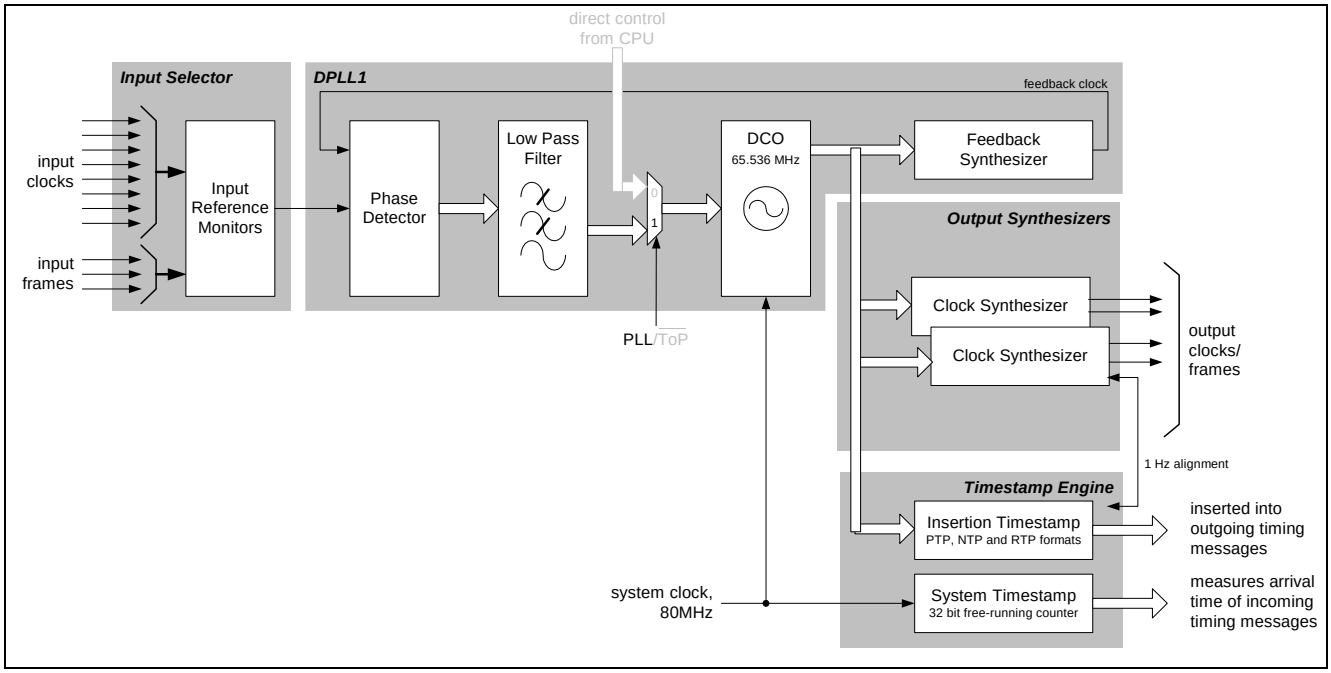



**Figure 21 - Operation as a Timing over Packet Server**

If it is required to lock the device to UTC time (or some other local time standard), a low-frequency alignment signal is required, for example a 1 pulse/second (1 pps or 1 Hz) output from a time source such as a GPS unit. The CPU reads an approximate time from a time server (e.g., NTP server, or RS232 output from a GPS unit), and programs the “whole seconds” portion of the timestamp. Then the “fractional seconds” portion is zeroed at the next 1 pps transition, such that the timestamp is now aligned to the 1 second transition. The result is that the timestamps to be inserted into the sync messages are now locked to UTC time.

The CPU should wait a second and then read back the insertion timestamp, just to make sure the operation completed successfully (e.g., that the programming of the “whole seconds” portion and zeroing of the “fractional seconds” portion took place in the same 1 s interval, since there is a danger that the timestamp ends up exactly one second out). However, once verified, the alignment process is normally a one-off alignment at initialization. The phase should be checked periodically against the 1 pps, but re-alignment shouldn’t be needed unless the reference clock loses lock against UTC.

Packets received from the client devices (e.g., PTP delay_request messages) are timestamped as they arrive at the network interface. This uses “system time”, the 32 bit free-running counter driven from the 80 MHz system clock (which is in turn locked to the 20 MHz local oscillator). For the construction of PTP delay_response messages that has to be translated back into insertion time. Since the precise relationship between the two is known, this process is a simple mathematical operation.

### 4.2.2 Client Operation

When used as a time client, the PLL loop is broken, and the centre frequency is controlled by the timing recovery software. In effect, a virtual PLL is formed in software, adjusting the DCO to lock it to the time source back at the server, as shown in Figure 22.

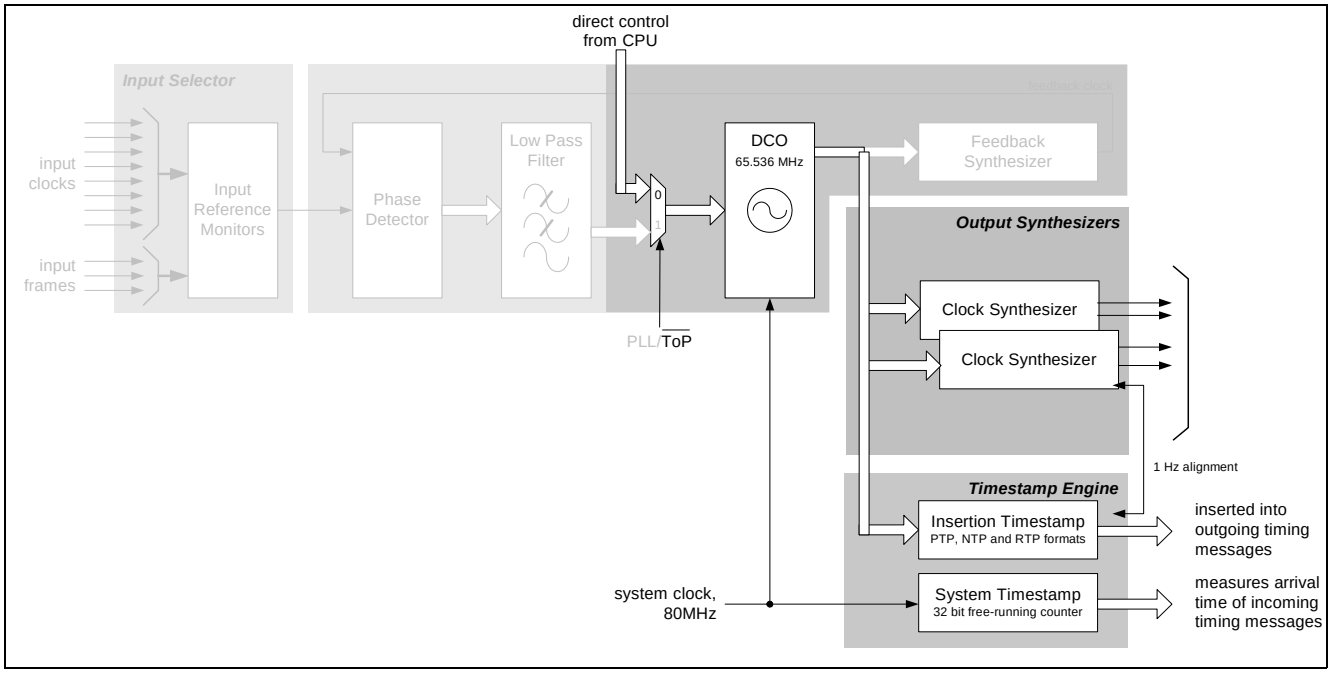



**Figure 22 - Operation as a Timing over Packet Client**

Sync messages arriving from the time server are timestamped as they arrive at the device. This is done using system time, so that the arrival time is recorded independently of any modulation of the recovered clock. This is enables the device to monitor several different packet time sources.

If two-way time transfer is used, the round-trip time may be calculated, and the insertion timestamp may be aligned with the insertion time at the time server. The 1 Hz frame pulse from the P0 synthesizer may then be aligned to the timestamp transition, ensuring that it is locked to the 1 Hz input at the server. Since insertion time is driven by the DCO, the timing recovery software is then able to keep it locked back to the server. As before, the alignment process should therefore be a one-off operation.

### 4.2.3 Boundary Clock Operation

For a boundary clock, the device acts like a client in the first instance, receiving packets from the server, and aligning its timestamp engine to the timestamps in those incoming packets. It then functions as a server by inserting timestamps from the same timestamp engine into new packets to be forwarded to the subsequent clients.

### 4.2.4 Combined Synchronous Ethernet and Timing over Packet Operation

In some circumstances, it is desired to lock to a physical clock (e.g., a synchronous ethernet reference) for frequency stability, and still use Timing over Packet to obtain a time reference. The Microsemi device is capable of this simultaneous operation. In this case, the device functions as a conventional PLL, locked to the physical reference. The insertion timestamp is driven by the DCO as normal, and hence count on at a rate determined by the local reference.

However, the packet timing messages (e.g., PTP sync, delay_request and delay_response messages) indicate the offset between the client and server timestamps. The CPU can adjust the absolute value of the timestamp, such that the client timestamp is aligned to the server. Provided the physical reference at the client is traceable to the same physical reference as the server, the timestamps should stay aligned.

## 4.3 DPLL Functional Description

The Microsemi device contains two independent digital phase locked loops (DPLL) which includes a System Synchronizer quality DPLL and a second smaller DPLL for additional rate conversion and clock generation use. This dual DPLL is a highly integrated component that provides the functionality required for synchronizing network equipment. It locks to one of several electrical input references and generates a wide variety of synchronized output clocks and frame pulses.

### 4.3.1 DPLL Features

| Feature | DPLL1 | DPLL2 |
|---|---|---|
| Modes of Operation | Free-run, Normal (locked), Holdover | Free-run, Normal (locked), Holdover |
| Loop Bandwidth (BW) | User selectable:<br>0.3 mHz, 1 mHz, 3 mHz,<br>0.1 Hz,<br>1.7 Hz, 3.5 Hz, fast lock (7 Hz),<br>14 Hz, 28 Hz[1],<br>or wideband[2] (890 Hz / 56 Hz / 14 Hz) | Fixed: 14 Hz |
| Lock Time | < 600 s for 1 mHz, 3 mHz BW<br>< 50 s for 0.1 Hz BW<br>< 50 s 1.7 Hz, 3.5 Hz BW<br>< 10 seconds for all other BW (PSL = 885 ns/s)<br>< 1 second for all other BW (PSL = 7.5 µs/s, 61 µs/s, or unlimited) | < 1 s (fixed 14 Hz BW) |
| Phase Slope Limiting | User selectable: 885 ns/s, 7.5 µs/s, 61 µs/s, or unlimited | User selectable: 61 µs/s, or unlimited |
| Pull-in Range | User selectable:<br>12 ppm, 52 ppm, 83 ppm, and 130 ppm | Fixed: 130 ppm |
| Holdover Parameters | Selectable Update Times: 26 ms, 1 s, 10 s, 60 s, and Selectable Holdover Post Filter BW: 18 mHz, 0.6 Hz, 10 Hz | Fixed Update Time: 26 ms<br>No Holdover Post Filtering |
| Holdover Frequency Accuracy | Better than 1 ppb initial frequency offset. Frequency drift depends on the 20 MHz external oscillator | Better than 50 ppb initial frequency offset. Frequency drift depends on the 20 MHz external oscillator. |
| Reference Inputs | Ref0 to Ref7 | Ref0 to Ref7 |
| Sync Inputs | Sync0, Sync1, Sync2 | Sync inputs are not supported. |
| Input Ref Frequencies | ref0, ref1: 2 kHz, N * 8 kHz up to 77.76 MHz, 25 MHz, 50 MHz, 62.5 MHz, 125 MHz, 155.52 MHz, | ref0, ref1: 2 kHz, N * 8 kHz up to 77.76 MHz, 25 MHz, 50 MHz, 62.5 MHz, 125 MHz, 155.52 MHz |
| | ref2 to ref7: 2 kHz, N * 8 kHz up to 77.76 MHz, 25 MHz, 50 MHz. | ref2 to ref8: 2 kHz, N * 8 kHz up to 77.76 MHz, 25 MHz, 50 MHz. |
| Supported Sync Input Frequencies | 1 Hz, 166.67 Hz, 400 Hz, 1 kHz, 2 kHz, 8 kHz, 64 kHz. | Sync inputs are not supported. |
| Input Reference Selection/Switching | Automatic (based on programmable priority and revertiveness), or manual | Automatic (based on programmable priority and revertiveness), or manual |

| Feature | DPLL1 | DPLL2 |
|---|---|---|
| Hitless Ref Switching | Can be enabled or disabled | Can be enabled or disabled |
| Input Reference Phase Transient Build-Out | Does not build out a phase transient that is < 1 us / 0.1 s<br>Builds out a phase transient that is > 3 us / 0.1 s, phase transients between 1 us / 0.1 s and 3 us / 0.1 s can be built-out through configuration registers | Does not phase build out on the active reference |

1. Limited to 14 Hz for 2 kHz references)
2. In the wideband mode, the loop bandwidth depends on the frequency of the reference input. For reference frequencies greater than 8 kHz, the loop bandwidth = 890 Hz. For reference frequencies equal to 8 kHz, the loop bandwidth = 56 Hz. The loop bandwidth is equal to 14 Hz for reference frequencies of 2 kHz.

### 4.3.2 DPLL Mode Control

The device DPLL independently support three modes of operation - free-run, normal, and holdover. The mode of operation can be manually set or controlled by an automatic state machine as shown in Figure 23.

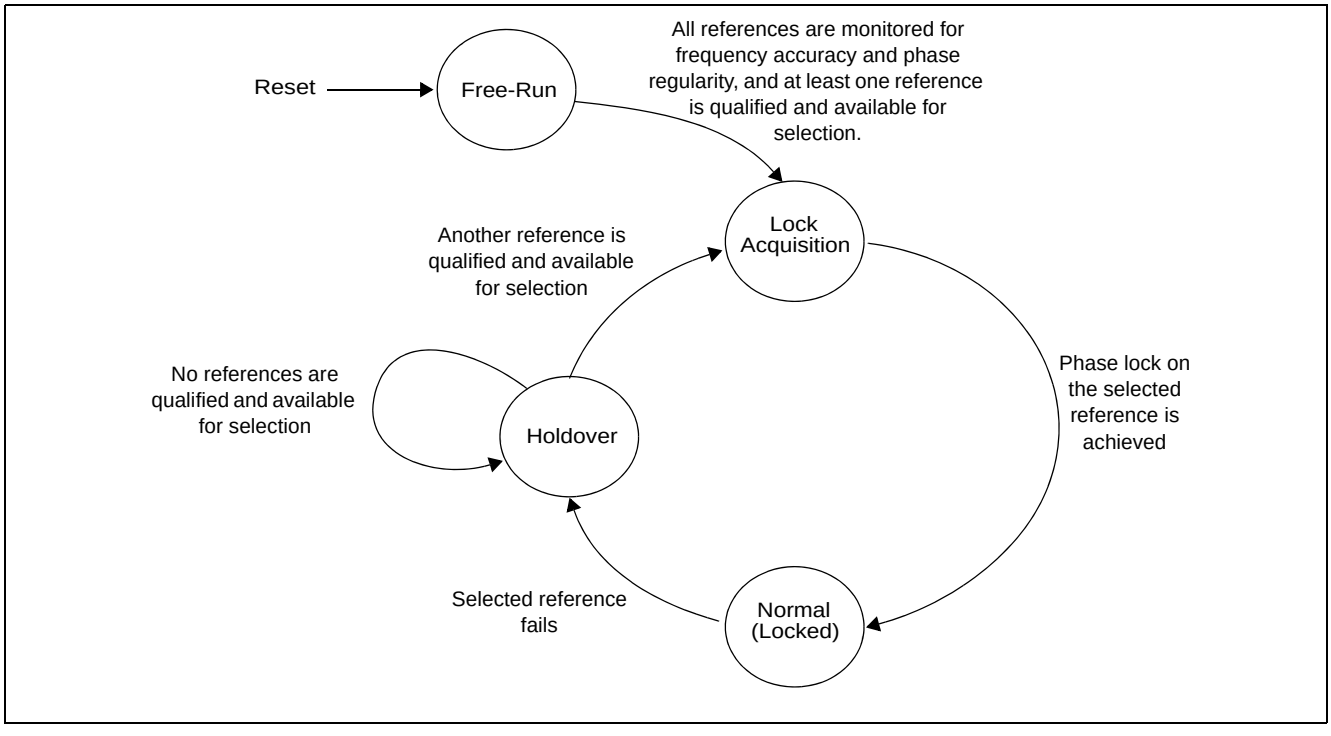



**Figure 23 - Automatic Mode State Machine**

#### Free-run

The free-run mode occurs immediately after a reset cycle or when the DPLL has never been synchronized to a reference input. In this mode, the frequency accuracy of the output clocks is equal to the frequency accuracy of the external master oscillator.

#### Lock Acquisition

The input references are continuously monitored for frequency accuracy and phase regularity. If at least one of the input references is qualified by the reference monitors, then the DPLL will begin lock acquisition on that input. Given a stable reference input, the device will enter in the Normal (locked) mode.

**Normal (locked)**

The usual mode of operation for the DPLL is the normal mode where the DPLL phase locks to a selected qualified reference input and generates output clocks and frame pulses with a frequency accuracy equal to the frequency accuracy of the reference input. While in the normal mode, the DPLL's clock and frame pulse outputs comply with the MTIE and TDEV wander generation specifications as described in Telcordia and ITU-T telecommunication standards.

**Holdover**

When the DPLL operating in the normal mode loses its reference input, and no other qualified references are available, it will enter the holdover mode and continue to generate output clocks based on historical frequency data collected while the DPLL was synchronized. The transition between normal and holdover modes is controlled by the DPLL so that its initial frequency offset is better than 1 ppb which meets the requirement of Stratum 3E. The frequency drift after this transition period is dependant on the frequency drift of the external master oscillator.

#### 4.3.2.1 DPLL1 Mode Of Operation

The DPLL supports three modes of operation: Free-run, Normal, and Holdover. These modes can be selected automatically using an internal state machine, or they can be selected manually. The mode of operation for DPLL1 is configurable using the *dpll1_modesel* register (0x1F). Configurable option are:

- **Automatic Normal Mode**. In this mode, the device DPLL uses an internal state machine to select the mode of operation as Free-run, Normal, or Holdover. Automatic reference switching is also enabled so that the highest priority qualified reference is selected. If that reference fails, an automatic reference switch-over to the next highest priority and qualified reference is initiated. If there are no suitable references for selection, DPLL1 will stay in free-run or enter the holdover state.
- **Manual Normal Mode**. In this mode, the device DPLL stays in Normal mode. Automatic reference switching is disabled and the selected reference is determined by the dpll1_refsel register (0x20). If the selected reference fails, the device automatically enters the Holdover mode.
- **Manual Holdover Mode**. In this mode, the device DPLL stays in Holdover mode which means that it is not locked to any reference input. Instead, it generates a frequency based on historical frequency data collected while the DPLL was locked to the last valid reference.
- **Manual Freerun Mode**. In this mode, the device DPLL stays in Free-run mode. The DPLL generates an output frequency that is based on the center frequency of its external reference oscillator.

The default mode of operation after reset the device is Automatic Normal Mode.

#### 4.3.2.2 DPLL2 Mode of Operation

DPLL2 has its own internal state machine for selecting the mode of operation and reference selection. This state machine also supports Automatic Normal, Manual Normal, Manual Holdover, and Manual Freerun modes. The mode is selectable using the *dpll2_modesel* register (0x2C). The default mode of operation after reset the device is Automatic Normal Mode.

### 4.3.3 Loop Bandwidth

The loop bandwidth determines the amount of wander and jitter filtering that is provided by the DPLL. The loop bandwidth for DPLL1 is programmable using the *dpll1_control_register_0* register (0x1D). The bandwidth should be set according to the application. Table 5 available bandwidth settings.

DPLL2's loop bandwidth is not programmable and is fixed at 14 Hz.

| BW (Hz) | Application |
|---|---|
| 0.1 | GR-253 SONET Stratum 3, SMC, G.813 option 2, G.8262 option 2 |
| 1.7 | GR-1244 Stratum 3, G.813 option 1 |
| 3.5 | G.813 option 1, G8262 option 1 |
| 14/56/890 | Wide Band Mode. BW depends on input frequency. |
| 7 | Fast Lock, reduced loop filter damping. |
| 0.3 mHz | Wireless Basestations |
| 1.0 mHz | GR-1244 Stratum 2 and 3E, G.812 types 2 and 3 |
| 3.0 mHz | G.812 type 1 |

**Table 5 - DPLL1 Loop Bandwidth Settings**

### 4.3.4 Pull-in/hold-in Range

The **pull-in range** defines the maximum input frequency range that the DPLL can lock to. The pull-in range for DPLL1 is programmable using the *dpll1_pull_in_range* register (0x29). The pull-in range should be set according to the application as shown in Table 6. The **hold-in range**, which defines the range of input frequencies that the PLL will continue to lock to, is equal to the pull-in range.

The pull-in/hold-in range for DPLL2 is fixed at +/-130 ppm.

| +/- ppm | Application |
|---|---|
| 12 | Stratum 3/3E, G.813 option 1, G.8262 option 1 and 2 |
| 52 | SONET Minimum Clock, G.813 option 2 |
| 83 | Stratum 4, G.824 |
| 130 | Stratum 4, G.823, and line card applications |

**Table 6 - DPLL1 Pull-in Range**

### 4.3.5 Phase slope Limiting

The device DPLL offers a phase slope limit feature which can be used to limit the rate of output phase movement of the output clocks and frame pulses during an input transient. This feature is used for meeting the phase slope requirements of Telcordia and ITU-T standards. The level of phase slope limiting depends on the application. Four levels of phase slope limiting is selectable for DPLL1 using the *dpll1_ctrl_0* register (0x1D). Table 7 shows available selections.

DPLL2 has two phase slope limits available (61 µs/s or Unrestricted) selectable using the *dpll2_ctrl_0* (0x2A) register.

| Phase Slope Limiting | Application |
|---|---|
| 885 ns/s | GR-1244 Stratum 2, 3E, 3 (objective), GR-253-CORE SMC and Stratum 3 for SONET, G.8262 option 2 |
| 7.5 µs/s | G.813 option 1, G.8262 option 1 |
| 61 µs/s | GR-1244 Stratum 3 |
| Unrestricted (default) | No phase slope limiting |

**Table 7 - DPLL Phase Slope Limiting**

### 4.3.6 Hitless Reference Switching

With hitless reference switching enabled, the phase difference between the originally selected reference and the newly selected reference is absorbed by the DPLL preventing a possible non-compliant phase transient at its output. The *dpll$_n$_ctrl_0* registers (0x1D, 0x2A) allows this feature to be enabled or disabled. When disabled, the DPLL will align its output to the new reference at a rate of alignment which is dependant on the set phase slope limit. Switching to a reference and sync pair is always non-hitless, regardless of the dplln_cltrl_0 register settings. The purpose of using the sync input beside the clock is to achieve phase alignment with to the input and not hitless switching.

### 4.3.7 Holdover

The device DPLL continuously collect phase data while synchronized to a valid reference. These data samples are accumulated and averaged to determine a stable holdover frequency in the event that all of the valid references are

lost. To prevent reference input jitter from corrupting the final holdover value, samples are taken on phase data filtered by the DPLL's loop bandwidth. DPLL1 offers an additional stage of filtering that can be enabled if the DPLL's loop bandwidth does not provide adequate filtering. This allows the DPLL to operate in a wide bandwidth mode and still provide an accurate holdover value. This is useful when the DPLL1 is used in a redundant mode. The holdover filter bandwidth is programmable using the *hold_filt_bw* field of the *dpll1_ctrl_1* register (0x1E).

The holdover performance of the output clocks will depend on two factors. One is the initial offset of the DPLL, and the other is the frequency drift (or stability) of the external oscillator. The initial offset of the DPLL meets the requirements for Stratum 3/G.813 opt 1, Stratum 3E/G.812 type 3, Stratum 2/G.812 type 2 and G.8262; the overall holdover performance dependant on the frequency drift of the external oscillator.Either an OCXO or TCXO can be used for Stratum 2/G.812 type 2 and Stratum 3E/6.812 type 3.

### 4.3.8 Phase Build-Out

One of the requirements for meeting Stratum 2 and 3E is phase build-out on incoming phase transients which may occur on the active reference. Phase build-out is the process of absorbing a phase transient to prevent it from passing through to the output. Telcordia GR-1244 requires that Stratum 2 and 3E clocks build-out a phase transient that is greater than 3.5 us in a 0.1 s window. Phase transients below 1 us in a 0.1 s window must not be built-out. A transient between the region of 1.0 us and 3.5 us in a 0.1 s window may or may not be built-out. This is described in Figure 24.

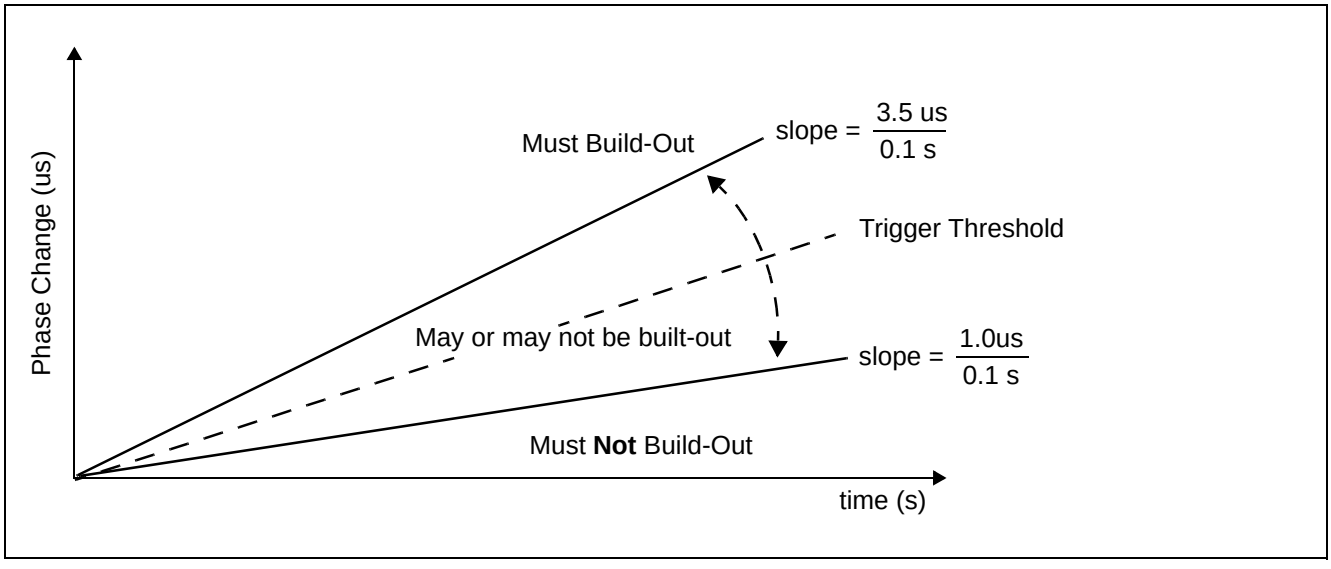



**Figure 24 - Telcordia GR-1244 Stratum 3E Phase Build-Out Requirements**

DPLL1 provides a programmable phase build-out trigger threshold that allows the user to select the point at which the phase build-out process occurs. A transient below the trigger threshold will not be built-out, and a transient above the trigger threshold will be built-out. The default trigger threshold is set to the minimum 1.0 us/0.1 s limit. The phase build-out trigger threshold is programmable using the *pbo_trig_threshold* register (08_0x74).

By default, phase build-out is disabled. It can be enabled using the *s3e_control* register (08_0x79).

### 4.3.9 Reference and Sync Inputs

There are eight reference clock inputs (**ref0** to **ref7**) available to both DPLL1 and DPLL2. Each of the DPLLs have independent reference selectors which can be controlled using a built-in state machine or set in a manual mode. The selected reference input is used to synchronize the output clocks.

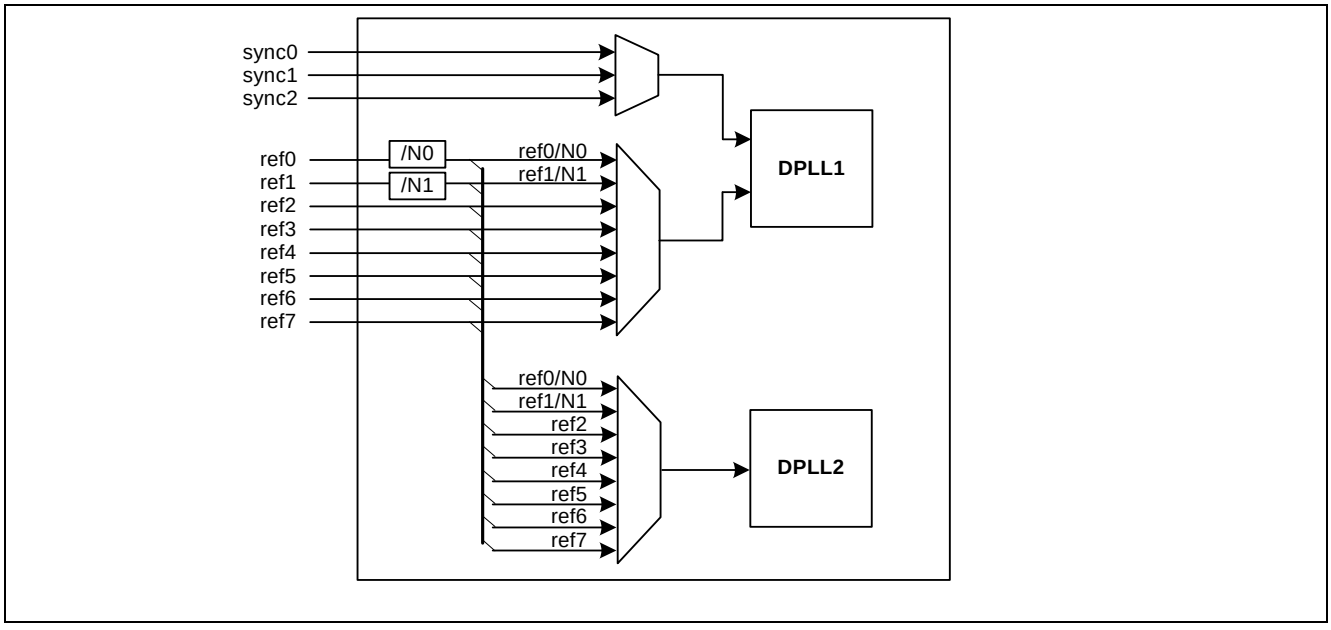



**Figure 25 - Reference and Sync Inputs**

Each of the **ref** inputs accept a single-ended LVCMOS clock with a frequency ranging from 2 kHz to 77.76 MHz. Built-in frequency detection circuitry automatically determines the frequency of the reference if its frequency is within the set of pre-defined frequencies as shown in Table 8. Once detected, the resulting frequency of the reference can be read from the ref_frq_detected registers (0x10 - 0x13).

| | | |
|---|---|---|
| 2 kHz | 2.048 MHz | 19.44 MHz |
| 8 kHz | 6.48 MHz | 38.88 MHz |
| 64 kHz | 8.192 MHz | 77.76 MHz |
| 1.544 MHz | 16.384 MHz | |

**Table 8 - Set of Pre-Defined Auto-Detect Clock Frequencies**

Two additional custom reference frequencies (Custom A and Custom B) are also programmable using the *custA_mult* and *custB_mult* registers (0x67, 0x68, 0x71, 0x72). These custom frequencies are programmable as 8 kHz * N up to 77.76 MHz (where N = 1 to 9720), or 2 kHz (when N = 0). The *ref_freq_mode_0* register (0x65) are used to configure each of the reference inputs as auto-detect, custom A, or custom B.

The first two reference inputs (**ref0** and **ref1**) have programmable pre-dividers (N0, N1) which allows them to lock to frequencies higher than 77.76 MHz or to non-standard frequencies. By default the pre-dividers divide by 1, but they can be programmed to divide by 1.5, 2, 2.5, 3, 4, 5, 6, 7, and 8 using the *predivider_control* register (0x7E). For example, an input frequency of 125 MHz can be divided down by 5 using the pre-dividers to create a 25 MHz input reference. The 25 MHz can then be programmed as a custom input frequency. Similarly, a 62.5 MHz input clock can be divided by 2.5 to create 25 MHz. Note: Division by non-integer numbers (e.g., 1.5, 2.5) uses both edges of the reference clock. As a result, higher jitter levels at the output clocks may occur if the reference clock is not 50% duty cycle.

In addition to the reference inputs, DPLL1 has three optional frame pulse synchronization inputs (**sync0** to **sync2**) used to align the output frame pulses. The $sync_n$ input is selected with its corresponding $ref_n$ input, where n = 0, 1, 2. Note that the sync input cannot be used to synchronize the DPLL, it only determines the alignment of the frame pulse outputs. An example of output frame pulse alignment is shown in Figure 26.

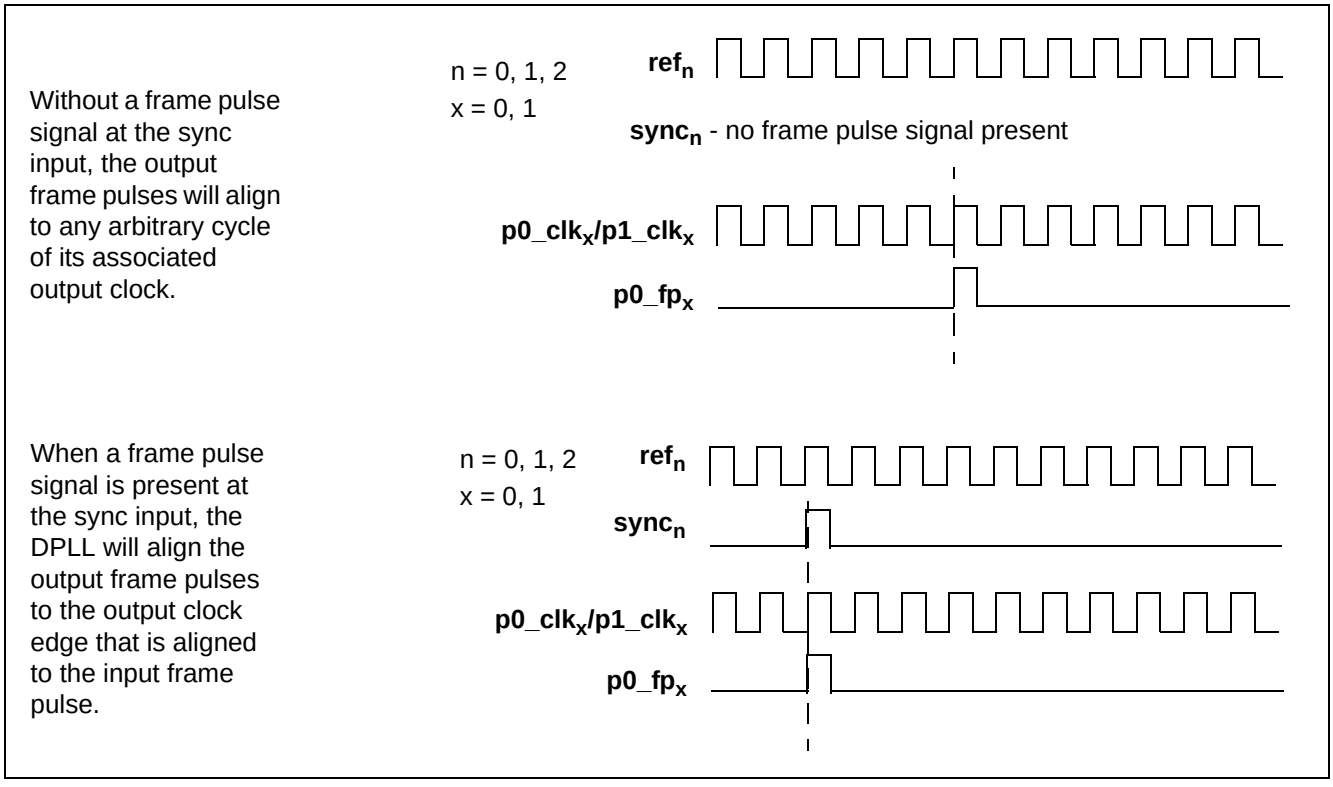



**Figure 26 - Output Frame Pulse Alignment**

Each of the **sync** inputs accept a single-ended LVCMOS frame pulse. Since alignment is determined from the rising edge of the frame pulse, there is no duty cycle restriction on this input, but there is a minimum pulse width requirement of 5 ns. Frequency detection for the sync inputs is automatic for the supported frame pulse frequencies shown in Table 9.

| |
|---|
| 1 Hz |
| 166.67 Hz<br>(48x 125 µs frames) |
| 400 Hz |
| 1 kHz |
| 2 kHz |
| 8 kHz |
| 64 kHz |

**Table 9 - Set of Pre-Defined Auto-Detect Sync Frequencies**

Each of the **sync** inputs can be enabled or disabled using the *sync_enable* register (08_0x68). By default all sync inputs are enabled so that DPLL1 generates frame aligned frame pulse outputs when a frame pulse is available at the selected sync input. It is also possible to invert the sync inputs.

### 4.3.9.1 Receiving Composite Clocks

The composite clock (CC) is a timing signal that provides both phase (bit/byte alignment) and frequency synchronization for SONET/SDH network elements. The CC signal is a 64 kHz, 5/8 duty cycle, return-to-zero, bipolar signal with bipolar violation (BPV) every 8th bit. The format of the CC signal is shown in Figure 27. Bit synchronization is achieved on the trailing edge of the 64 kHz CC signal, byte synchronization is indicated on the occurrence of the 8 kHz BPV. Specifications for the CC signal are covered in Telcordia GR-378 and ITU-T G.703.

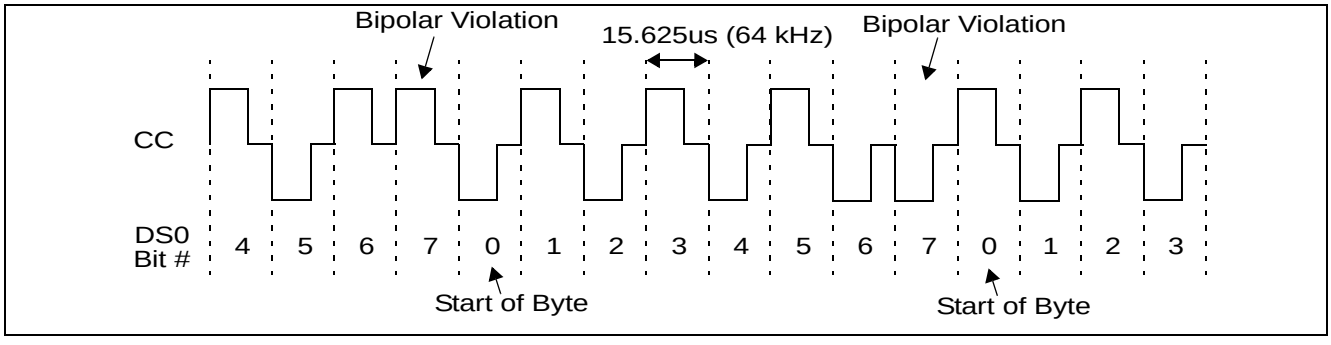



**Figure 27 - GR-378/G.703 Composite Clock Format**

Internal decoding logic extracts the 64 kHz reference clock and the 8 kHz frame pulse from the CC signal. Support for two CC references are possible using existing ref/sync input pins. One CC signal is received on the ref0/sync0 pins, and the other on the ref1/sync1 pins. Before interfacing the bipolar CC signal to the ref/sync pair, it must be separated into two unipolar signals using a simple external circuit as shown in Figure 28.

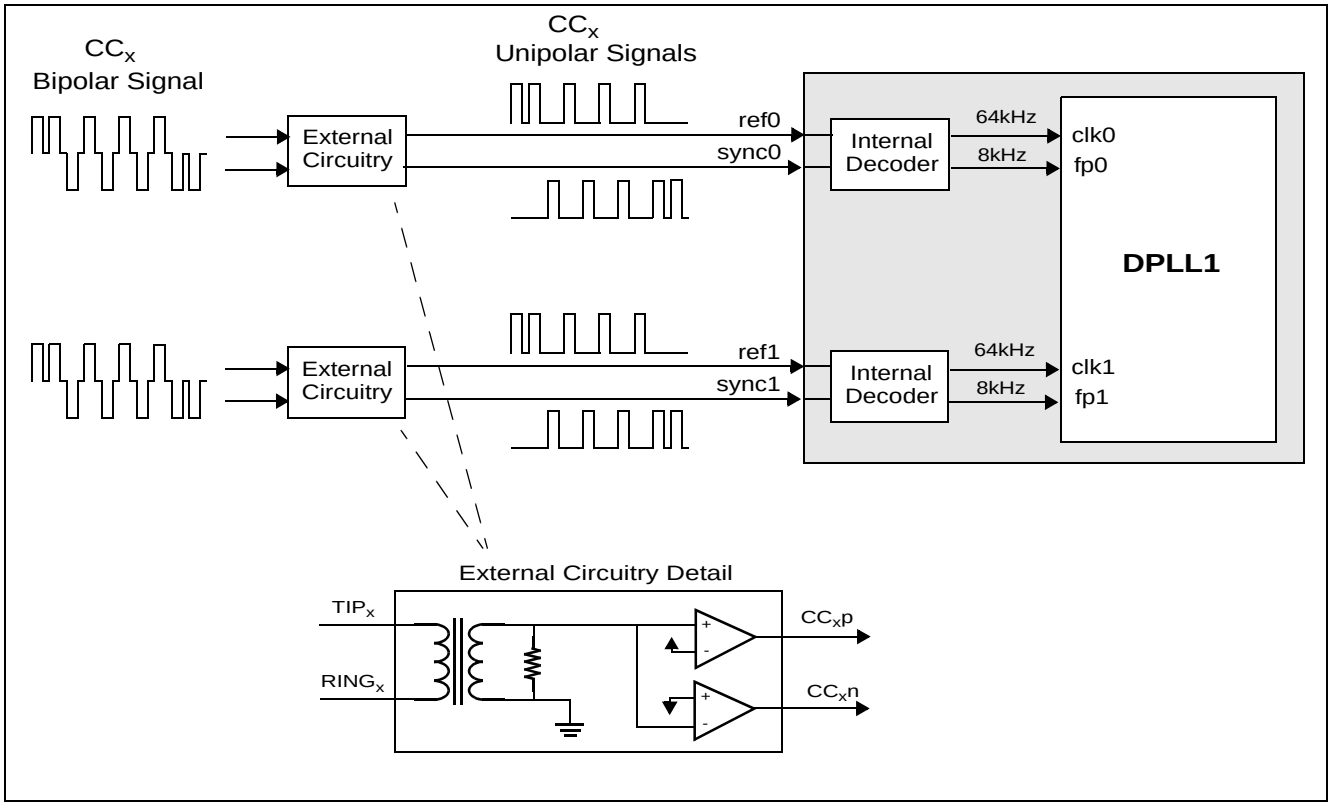



**Figure 28 - External Circuit for Converting a Bipolar CC Signal into Two Unipolar Signals**

By default a ref/sync pair accepts a normal clock and optional frame pulse signal. Enabling a ref/sync pair to accept a unipolar CC signal requires modification of the *cc_ref0_ctrl (0x7B) and/or the cc_ref1_ctrl (0x7C)* registers. When the composite clock mode is selected, the output clocks locked to DPLL1 will be synchronous with the 64 kHz clock rate of the selected CC signal. The frame pulse output can be made to align with the 8 kHz BPV by configuring *the cc_ref0_ctrl (0x7B) and/or cc_ref1_ctrl (0x7C)* registers. By default the frame pulse outputs have arbitrary alignment to the 8 kHz BPV.

Another form of the CC signal (mainly used in Japan) is shown in Figure 29. Its format is similar to the CC shown in Figure 27 with the exception that the byte alignment is marked during the occurrence of the 8 kHz BPV instead of the bit after the BPV. It also provides for optional multi-frame synchronization by violating the 8 kHz bipolar violation at a 400 Hz rate. The specification for this CC signal is covered in ITU-T G.703 Appendix II.

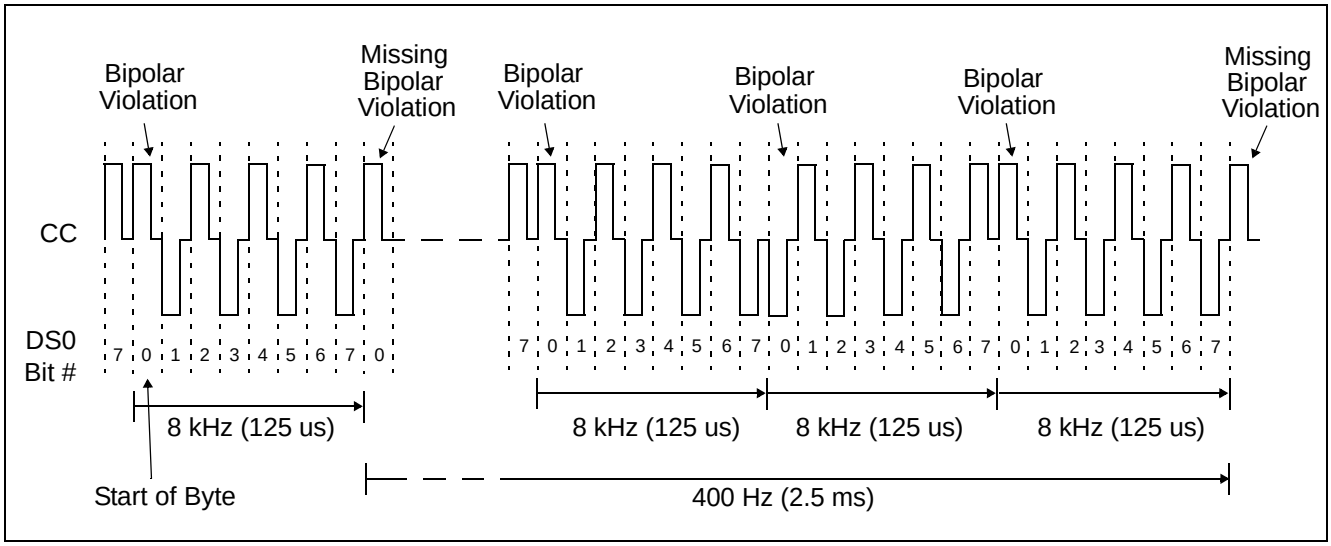



**Figure 29 - G.703 Japan Composite Clock (Appendix II)**

The main difference between the CC signal in Figure 27 and in Figure 29 is in the definition of byte alignment with respect to the *8 kHz BPV.* Start of byte alignment in the CC signal of Figure 27 occurs after the 8 kHz BPV, whereas start of byte alignment in Figure 29 occurs during the 8 kHz BPV. The frame pulse outputs can be made to align with either of these CC signal formats by configuring the *cc_ref0_ctrl* (0x7B) and/or *cc_ref1_ctrl (0x7C) registers.*

The CC format in Figure 29 also has an additional 400 Hz frame indicator. It is triggered by a missing BPV every 2.5 ms. The frame pulse outputs can be made to align to the missing BPV by configuring the *cc_ref0_ctrl* (0x7B) and *cc_ref1_ctrl (0x7C) registers.*

Composite clocks are monitored for bipolar violation (BPV) errors. The *cc_isr* register (0F_0x69) is asserted whenever two PBV occur within two consecutive eight bit periods.

An interrupt can be triggered in the event of a BPV error. Mask bits for the BPV error are controlled in the cc_isr_mask register (0F_0x6A). By default these errors are masked from triggering an interrupt.

### 4.3.10 Reference Input Selection

Reference selection can be automatic or manual depending on the *dpll$_n$_modesel* registers (0x1F, 0x2C). For automatic reference selection, the mode selection register must be set to the “Automatic Normal Mode” setting. For manual reference selection, set the mode selection registers to the “Manual Normal Mode”.

In the case of automatic reference selection, the selection criteria is based on reference qualification, input priority, and the revertive setting. Only references that are valid can be selected by the automatic state machine. If there are no valid references available, then the DPLL will automatically enter the holdover mode. Each of the references has an assignable priority definable in the *dpll1_ref_pri_ctrl* registers (0x24 to 0x27), and the input priority for DPLL2 is

defined in the *dpll2_ref_pri_ctrl* registers (0x30 to 0x34). Any of the references can be prevented from being selected by setting their priority to “1111”.

The *dpll$_n$_control_register_1* registers (0x1E, 0x2B) controls the revertive switching option for the DPLLs. With revertive switching enabled, the highest priority reference input with a valid reference is always selected. If a reference with a higher priority becomes valid, then a reference switchover to that reference will be initiated. With non-revertive switching, the active reference will always remain selected while it is valid. If this reference becomes invalid, a reference switchover to a valid reference with the highest priority will be initiated. Note that if two or more references have been assigned the same priority, then priority will be given to the lowest reference number (e.g., if ref4 and ref7 have the same assigned priority, then ref4 will have higher priority over ref7).

The revertive feature can also be applied to individual references using the *dpll$_n$_reference_revertive_control* registers (0x23, 0x30).

When the *dpll_modesel* register is set to the “Manual Normal Mode”, the active reference is selected using the *dpll1_refsel* or the *dpll2_refsel* registers (0x20, 0x2D). If the defined reference is not valid, then the DPLL will automatically enter the holdover mode.

### 4.3.11 Reference Monitoring

Input references **ref0** to **ref7** are monitored for frequency accuracy and phase regularity. New references are qualified before they can be selected as a synchronization source and qualified references are continuously monitored to ensure that they are suitable for synchronization.

The process of qualifying a reference depends on four levels of monitoring.

#### Single Cycle Monitor (SCM)

The SCM block measures the period of each reference clock cycle to detect phase irregularities or a missing clock edge. In general, if the measured period deviates by more than 50% from the nominal period, then an SCM failure (scm_fail) is declared.

#### Coarse Frequency Monitor (CFM)

The CFM block monitors the reference frequency over a measurement period of 30 µs so that it can quickly detect large changes in frequency. A CFM failure (cfm_fail) is triggered when the frequency has changed by more than 3% or approximately 30000 ppm.

#### Precise Frequency Monitor (PFM)

The PFM block measures the frequency accuracy of the reference over a 10 second interval. To ensure an accurate frequency measurement, the PFM measurement interval is re-initiated if phase or frequency irregularities are detected by the SCM or CFM. The PFM provides a level of hysteresis between the acceptance range and the rejection range to prevent a failure indication from toggling between valid and invalid for references that are on the edge of the acceptance range.

When determining the frequency accuracy of the reference input, the PFM uses the external oscillator’s output frequency ($f_{ocsi}$) as its point of reference.

#### Guard Soak Timer (GST)

The GST block mimics the operation of an analog integrator by accumulating failure events from the CFM and the SCM blocks and applying a selectable rate of decay when no failures are detected.

As shown in Figure 30, a GST failure (gst_fail) is triggered when the accumulated failures have reached the upper threshold during the disqualification observation window. When there are no CFM or SCM failures, the accumulator decrements until it reaches its lower threshold during the qualification window.

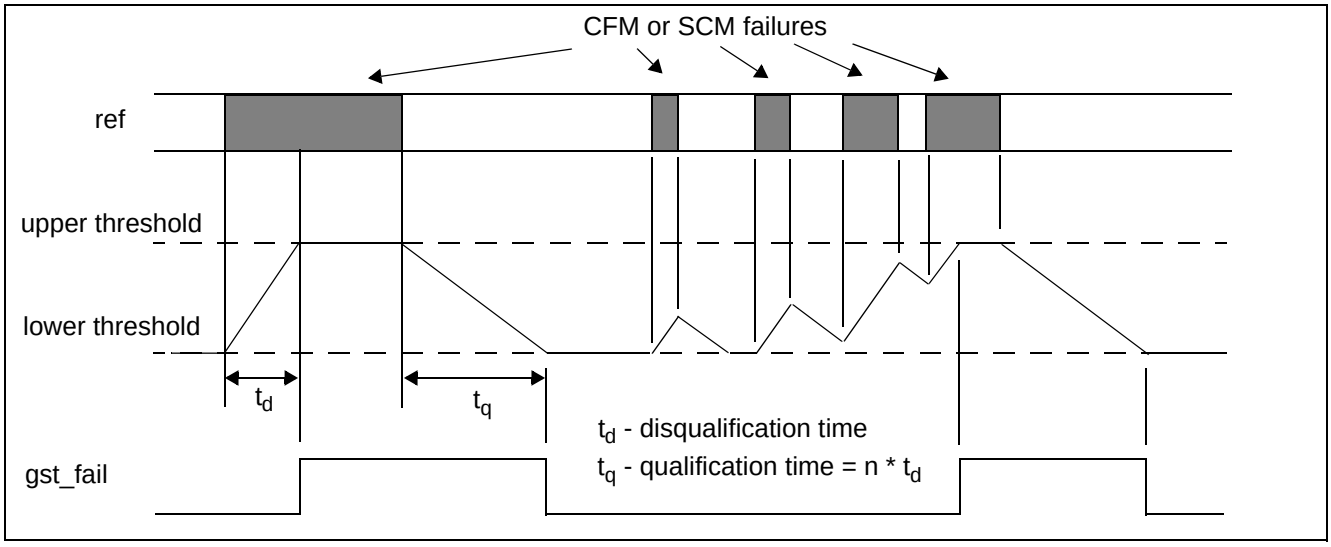



**Figure 30 - Behaviour of the Guard Soak Timer during CFM or SCM Failures**

### Precise Frequency Monitor (PFM)

The PFM is used to keep track of the frequency of the reference clock. It measures its frequency over a 10 second period and indicates a failure when the measured frequency exceeds the out-of-range (OOR) limits configured in the *oor_ctrl[0:3]* registers (0x16 to 0x19). The OOR should be set according to the application as shown in Table 10.

| Acceptance Range | Rejection Range | Typical Application |
|---|---|---|
| +/- 9.2 ppm | +/- 12 ppm | Stratum 3/3E, G.813 option 1 |
| +/- 13.8 ppm | +/- 18 ppm | |
| +/- 24.6 ppm | +/- 32 ppm | |
| +/- 36.6 ppm | +/- 47.5 ppm | |
| +/- 40 ppm | +/- 52 ppm | SONET Minimum Clock, G.813 option 2 |
| +/- 52 ppm | +/- 67.5 ppm | |
| +/- 64 ppm | +/- 83 ppm | Stratum 4, G.824 |
| +/- 100 ppm | +/- 130 ppm | G.823 |

**Table 10 - Frequency Out of Range Limits**

To ensure an accurate frequency measurement, the PFM measurement interval is re-initiated if phase or frequency irregularities are detected by the SCM or CFM. The PFM provides a level of hysteresis between the acceptance range and the rejection range to prevent a failure indication from toggling between valid and invalid for a reference that is on the edge of the acceptance range.

When determining the frequency accuracy of the reference input, the PFM uses the external oscillator's output frequency ($f_{ocsi}$) as its point of reference. As a result, the actual acceptance and rejection frequencies can be offset with respect to the external oscillator's output frequency. This is accounted for in the acceptance and rejection requirements as described in Telcordia GR-1244 section 3.4.1. An example of the acceptance and rejection ranges for Stratum 3/3E application (acceptance in the range of +/- 9.2 ppm, rejection at +/- 12 ppm) given a +/- 4.6 ppm free-run frequency accuracy of a Stratum 3/3E reference oscillator is shown in Figure 31.

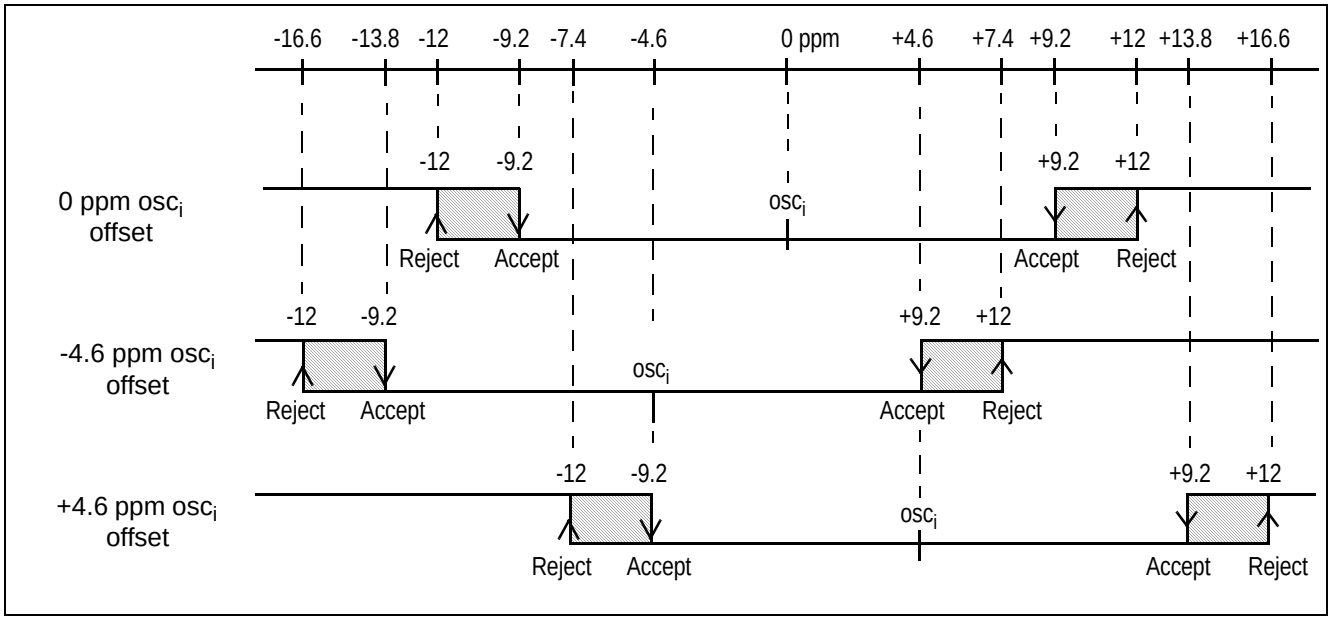



**Figure 31 - Frequency Acceptance and Rejection Ranges**

SCM, CFM, PFM, and GST failures are indicated in the ref_mon_fail registers (0x05 to 0x08). As shown in Figure 32, the SCM, CFM, PFM, and GST indicators are logically ORed together to form a reference failure indicator. An interrupt is triggered when the failure indicator is triggered. The status of the failure indicators can be read in the *ref_fail_isr* interrupt service register (0x02). A change in the bit status of this register will cause the interrupt pin (**int_b**) to go low. It is possible to mask this interrupt with the *ref_fail_isr_mask* register (0x09) which is represented as “mask_isr$_n$”.

It is possible to mask an individual reference monitor from triggering a reference failure by setting the *ref_mon_fail_mask_3:0* registers (0x0C - 0x0F). These are represented by mask_scm$_n$, mask_cfm$_n$, mask_gst$_n$, and mask_pfm$_n$ in Figure 32. In addition, the CFM and SCM reference monitor indicators can be masked from indicating failures to the GST reference monitor using the gst_mask1:0 registers (0x1A - 0x1B). These are represented as mask_cfm_gst$_n$ and mask_scm_gst$_n$.

Setting mask bit to logic 0 will disable the individual monitor ability from triggering a reference fail. While setting it to logic 1 will allow that monitor to trigger a reference fail.

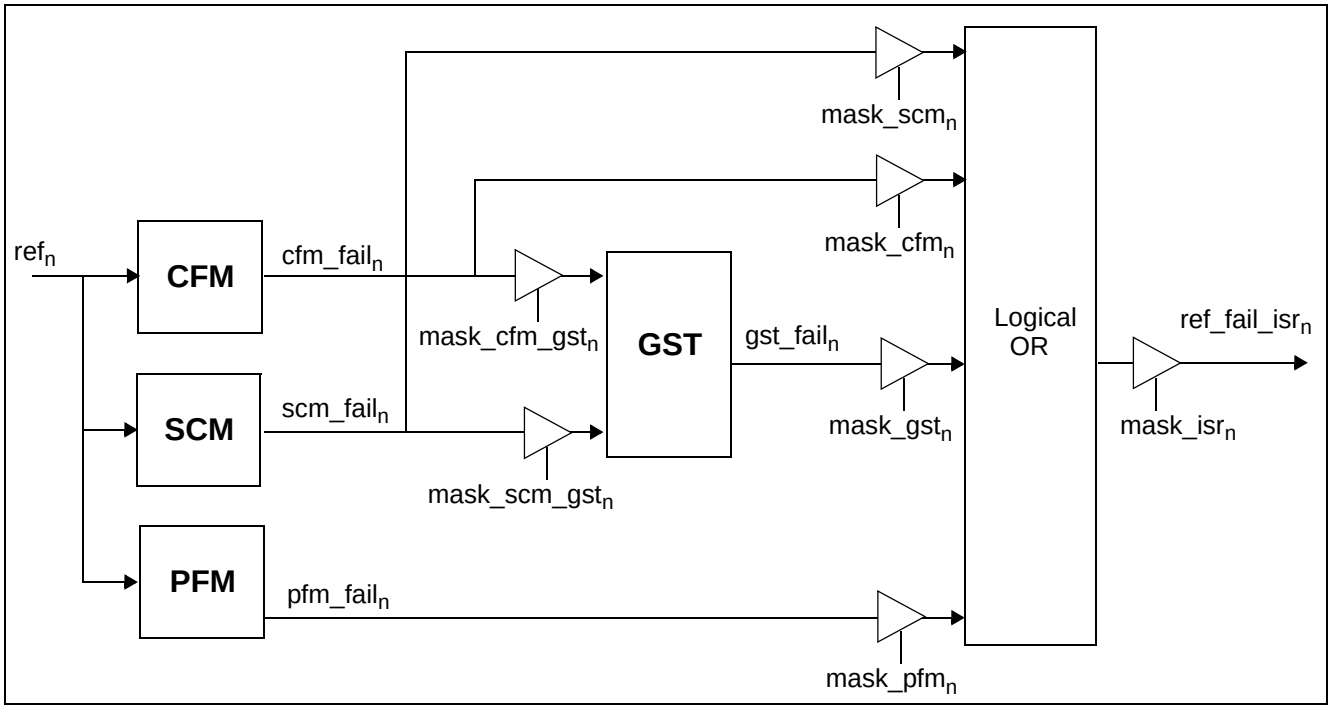



**Figure 32 - Reference Monitoring Block Diagram**

### 4.3.12 Sync Monitoring

Sync inputs (**sync0 to sync2**) are continuously monitored by the Sync Ratio Monitor (SRM). The SRM ensures that the sync inputs are valid by verifying that there is a correct number of reference cycles within the sync period. The status of this monitor is reported in the *detected_sync* registers (0x14, 0x15).

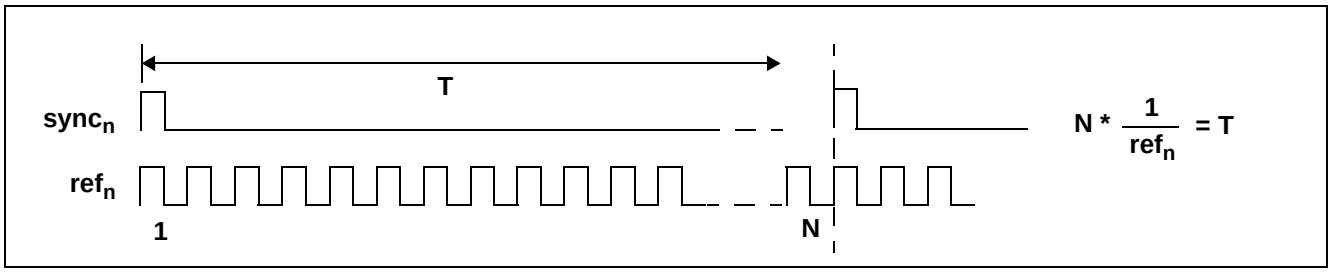



**Figure 33 - Sync Monitoring**

### 4.3.13 Reference Monitoring for Custom Configurations

Two additional custom reference input frequencies (Custom A, Custom B) are definable allowing a reference input to accept any multiple of 8 kHz up to 77.76 MHz.

Each of the custom configurations also have definable SCM and CFM limits. The SCM limits are programmable using the *custA_scm_low_lim, custA_scm_high_lim, custB_scm_low_lim, custB_scm_high_lim* registers (0x69, 0x6A, 0x73, 0x74). The SCM low and high limits determine the acceptance window for the clock period as shown in Figure 34. Any clock edge that does not fall into the acceptance window will trigger an SCM failure. High and low limits are programmed as multiples of a 300 MHz cycle (3.33 ns).

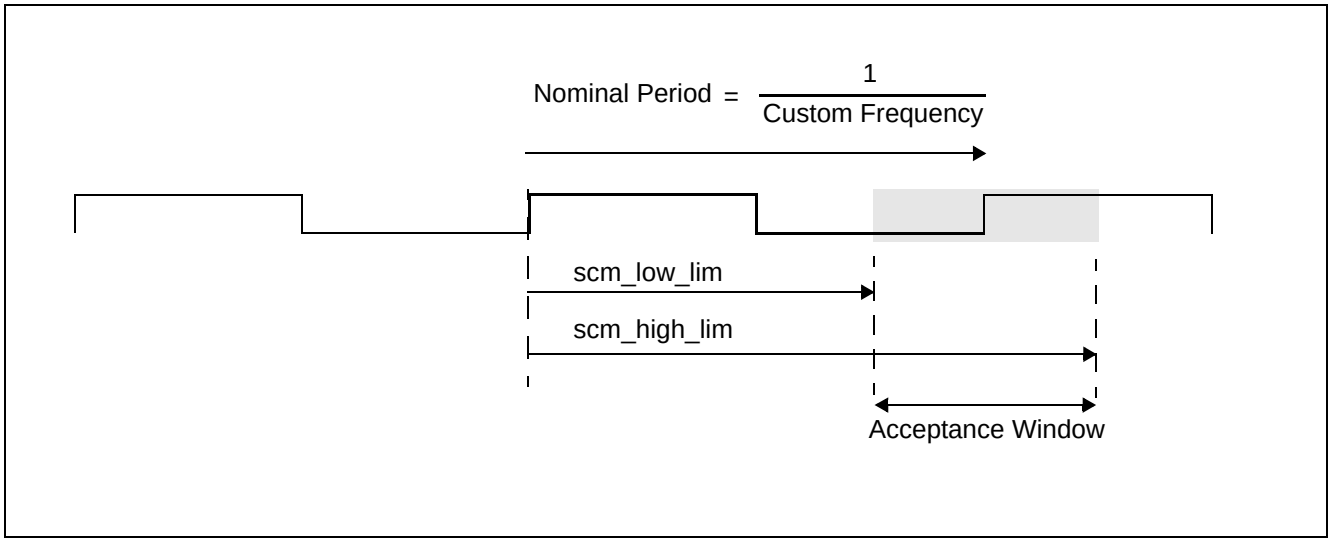


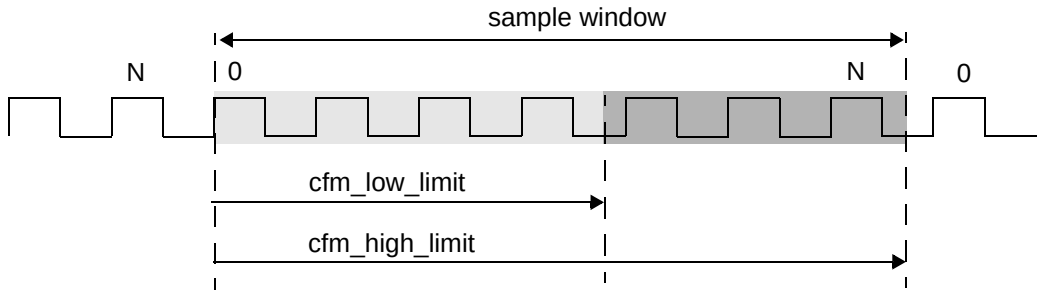

**Figure 34 - Defining SCM Limits for Custom Configurations**

Since the SCM is used to identify a missing clock edge, the acceptance window should be set to approximately +/-50% of the nominal period. Using a smaller window may trigger unwanted SCM failures.

For example, if the Custom A frequency was defined as 25 MHz (using registers 0x67, 0x68), its nominal period is 40 ns. To fail the input reference when its period falls below 20 ns (-50% of the nominal period), the *custA_scm_low* register is programmed to 0x06 (6 x 1/300 MHz = 20 ns). To fail the input reference if its period exceeds 60 ns (+50% of the nominal period), the *custA_scm_high* register is programmed with 0x12 (18 x 1/300 MHz = 60 ns).

For low speed input references less than 1.8 MHz, the SCM counter does not provide enough range to reliably perform its function. Therefore for custom inputs of less than 1.8 MHz the device should set the scm_low_lim and scm_high_lim to 0 and the CFM should be used as the single cycle monitor.

The CFM quickly determines large changes in frequency by verifying that there are N amount of input reference clock cycles within a programmable sample window. The value of N is programmable in the *custA_cfm_cycle* and the *custB_cfm_cycle* registers (0x6F, 0x79). The size of the sample window is defined in terms of high and low limits and are programmed as multiples of 80 MHz cycles. These are defined using the *custA_cfm_low_0*, *custA_cfm_low_1*, *custA_cfm_high_0*, *custA_cfm_high_1*, *custB_cfm_low_0*, *custB_cfm_low_1*, *custB_cfm_high_0*, *custB_cfm_high_1* registers (0x6B-0x6E, 0x75-0x78). A divide-by-4 circuit can be enabled to increase the resolution of the sample window. This is recommended when the input reference frequency exceeds 19.44 MHz. The divide-by-4 is enabled using the custA_div and custB_div registers (0x70, 0x7A). Equations for calculating the high and low limits are shown in Figure 35.




$$\textbf{cfm_low_limit} = \frac{\textbf{D}}{\text{cust_freq} + 3\%} \times \textbf{N} \times 80\ \text{MHz} \qquad \textbf{cfm_high_limit} = \frac{\textbf{D}}{\text{cust_freq} - 3\%} \times \textbf{N} \times 80\ \text{MHz}$$

where **N** and **D** are dependant on the setting of the custom frequency. Recommended values are shown in the following table:

| Input Frequency Range | D (Divider) | N (Number of cycles) |
|---|---|---|
| 38.88 MHz < freq ≤ 77.76 MHz | 4 | 256 |
| 19.44 MHz < freq ≤ 38.88 MHz | 4 | 128 |
| 8.192 MHz < freq ≤ 19.44 MHz | 1 | 256 |
| 2.048 MHz < freq ≤ 8.192 MHz | 1 | 128 |
| 1.8 MHz < freq ≤ 2.048 MHz | 1 | 32 |
| 2 kHz < freq ≤ 1.8 MHz (CFM limits should be set to +/-50%) | 1 | 1 |

**Example:** Custom configuration A is set for 25 MHz (*custA_mult13_8* = 0x0C, *custA_mult7_0* = 0x35)
($0C35_{hex} = 3125_{dec}$, 3125 x 8 kHz = 25 MHz)

The values for D and N are determined using the table above with respect to a 25 MHz input reference.
D = 4 (*custA_div* = 0x01)
N = 128 (*custA_cfm_cycle* = 0x7F)

The CFM low and high values are calculated using the equations above:

$$\text{cfm_low_limit} = \frac{4}{25.75\ \text{MHz}} \times 128 \times 80\ \text{MHz} = 1591_{dec} = 0637_{hex}$$
(*custA_cfm_low15_8* = 0x06)
(*custA_cfm_low7_0* = 0x37)

$$\text{cfm_high_limit} = \frac{4}{24.25\ \text{MHz}} \times 128 \times 80\ \text{MHz} = 1689_{dec} = 0699_{hex}$$
(*custA_cfm_high15_8* = 0x06)
(*custA_cfm_high7_0* = 0x99)

**Figure 35 - Custom CFM Configuration for 25 MHz**

### 4.3.14 Output Clocks and Frame Pulses

The device offers two programmable Ethernet LVCMOS (**eth_clk0, eth_clk1**), and four programmable LVCMOS (**p0_clk0, p0_clk1, p1_clk0, p1_clk1**) output clocks. In addition to the clock outputs, two LVCMOS programmable frame pulses (**p0_fp0, p0_fp1**) are also available. The supported frequencies for the output clocks and frame pulses are shown in Table 11.

| Clock or Frame Pulse Output | Supported Output Frequency |
|---|---|
| **eth_clk0**, **eth_clk1** | 12.5 MHz, 25 MHz, 50 MHz, 62.5 MHz(default), 125 MHz |
| **p0_clk0**, **p1_clk0** | 2 kHz, N * 8 kHz (up to 100 MHz), where N ranges from 0 to 9270 (N=0 selects 2 kHz) |
| **p0_clk1**, **p1_clk1** | p[0..1]_clk1 = p[0..1]_clk0 / $2^M$, where M = -128 to 127, defined as an 8-bit two's complement value (+ve values divide, -ve values multiply) |
| **p0_fp0**, **p0_fp1** | 166.67 Hz, 400 Hz, 1 kHz, 2 kHz, 4 kHz, 8 kHz, 32 kHz, 64 kHz, 1 Hz, 1 PPS |

**Table 11 - Supported Output Frequencies**

As shown in Figure 36, the output clocks and frame pulses derived from the APLL are always synchronous with DPLL1, and the clocks and frame pulses generated from the programmable synthesizers can be synchronized to either DPLL1 or DPLL2. This allows the device to have two independent timing paths. This is programmable by setting the *p0_enable* register (0x36), and the *p1_enable* register (0x48).

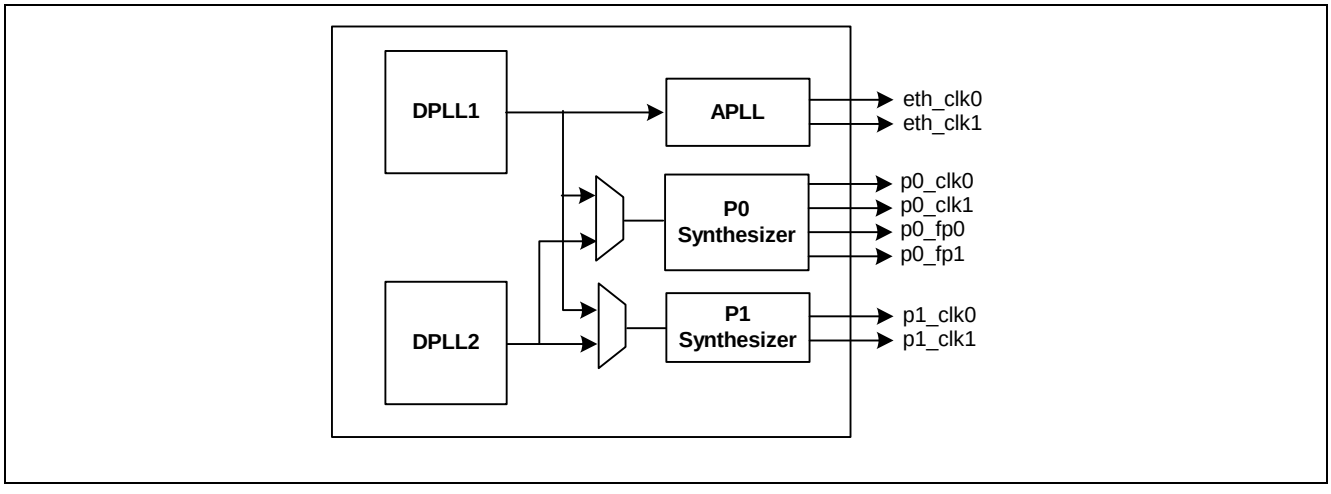



**Figure 36 - Output Clock Configuration**

The single ended Ethernet output clock (**eth_clk0**, **eth_clk1**) frequencies are programmable using the *apll_clk$_n$_freq* field of the *eth_clk_freq* register (0x52). The APLL can only generate Ethernet frequencies. This is programmable using the *eth_clk$_n$_run* bit of the apll_run_register (0x51). Valid frequencies are listed in Table 12.

| **apll_clk$_n$_freq bit settings** | **eth_clk$_n$ Output Frequency** |
|---|---|
| | eth_clk$_n$_run = 1 |
| 0001 | 125 MHz |
| 0010 | 62.5 MHz |
| 0011 | Reserved |
| 0100 | Reserved |
| 0101 | 50 MHz |
| 0110 | 25 MHz |
| 0111 | 12.5 MHz |
| 1xxx | Reserved |

**Table 12 - APLL LVCMOS Output Clock Frequencies**

The frequency of the **p0_clk0** output is programmable from 2 kHz up to 100 MHz where,

$$f_{p0_clk0} = N \times 8 \text{ kHz}$$

The value of N is a 16-bit word which is programmable using the *p0_freq_0* and *p0_freq_1* registers (0x38, 0x39). For an output frequency of 2 kHz, let N = 0.

The **p0_clk1** output frequency is programmed as a multiple of the p0_clk0 output frequency where

$$f_{p0_clk1} = \frac{f_{p0_clk0}}{2^M}$$

The value of M is defined in the *p0_clk1_div* register (0x3B). The minimum and maximum frequency limits of 2 kHz to 100 MHz are also applicable to p0_clk1.

The frequency of the **p1_clk_0** and **p1_clk1** output clocks are programmable in the same way as the p0_clk_0 and p0_clk1 output clocks where N is defined using the *p1_freq_0* and *p1_freq_1* registers (0x4A, 0x4B), and M is defined in the *p1_clk1_div* register (0x4D).

The frequency of the frame pulses generated from the p0 synthesizer (**p0_fp0**, **p0_fp1**) is programmable using the p0_fp0_freq register and the p0_fp1_freq registers (0x3E, 0x43). Valid frequencies are listed in Table 13.

| **p0_fp$_n$ Frequency** |
|---|
| 166.6667 Hz (48x 125 μs frames) |
| 400 Hz |
| 1 kHz |
| 2 kHz |
| 4 kHz |
| 8 kHz |
| 32 kHz |
| 64 kHz |
| 1 Hz (Synchronous to p0_clkn, with either 1UI or 4 msec pulse width) |
| 1 PPS (Asynchronous with 200 nsec pulse width) |

**Table 13 - P0 Frame Pulse Frequencies**

The pulse width of the frame pulse is programmable using the *p0_fp$_n$_type* register (0x3F, 0x44). Valid pulse widths are shown in Table 14.

| **p0_fp$_n$ Pulse Width** | **Comment** |
|---|---|
| One period of a 4.096 MHz clock | These are pre-defined pulse widths that are usable when p0_clk$_n$ is set to a frequency that is a multiple of the E1 rate (2.048 MHz). When p0_clk$_n$ is not an E1 multiple, the p0_fp$_n$_type must be set to “111” |
| One period of a 8.192 MHz clock | |
| One period of a 16.384 MHz clock | |
| One period of a 32.768 MHz clock | |
| One period of a 65.536 MHz clock | |
| One period of p0_clk$_n$ | The frame pulse width is equal to one period of the p0_clk$_n$. This setting must be used when the p0_clk$_n$ is not an E1 multiple. |

**Table 14 - P0 Frame Pulse Widths**

The style (frame pulse or 50% duty cycle clock), alignment (rising or falling edge of its associated clock), and its polarity (positive or negative) is programmable using the *p0_fp$_n$_type* register (0x3F, 0x44).

#### 4.3.14.1 Output Clock and Frame Pulse Squelching

A clock squelching feature is available which allows forcing an output clock to a specific logic level. The *apll_run*_register (0x51) controls the single ended eth_clk$_n$ outputs, and the p$_n$_run register (0x37, 0x49) controls the programmable outputs. The frame pulse outputs can be forced to a logic level using the p0_run register (0x37).

### 4.3.14.2 Disabling Output Clocks and Frame Pulses

Unused outputs can be set to a high impedance state to reduce power consumption. The single ended eth_clk$_n$ outputs can be disabled using the *apll_enable* register (0x50). The programmable clocks can be individually disabled using the *p$_n$_enable* register (0x36, 0x48). When not in use, the frame pulse outputs can be disabled using the p0_enable register (0x36).

### 4.3.14.3 Disabling Output Synthesizers

In applications where none of the APLL clocks are used, the entire APLL can be disabled to conserve power using the apll_enable register (0x50). Both of the programmable synthesizers can also be disabled by using the *p$_n$_enable* register (0x36, 0x48).

## 4.3.15 Configurable Input-to-Output and Output-to-Output Delays

The Microsemi device allows programmable static delay compensation for controlling input-to-output and output-to-output delays of its clocks and frame pulses.

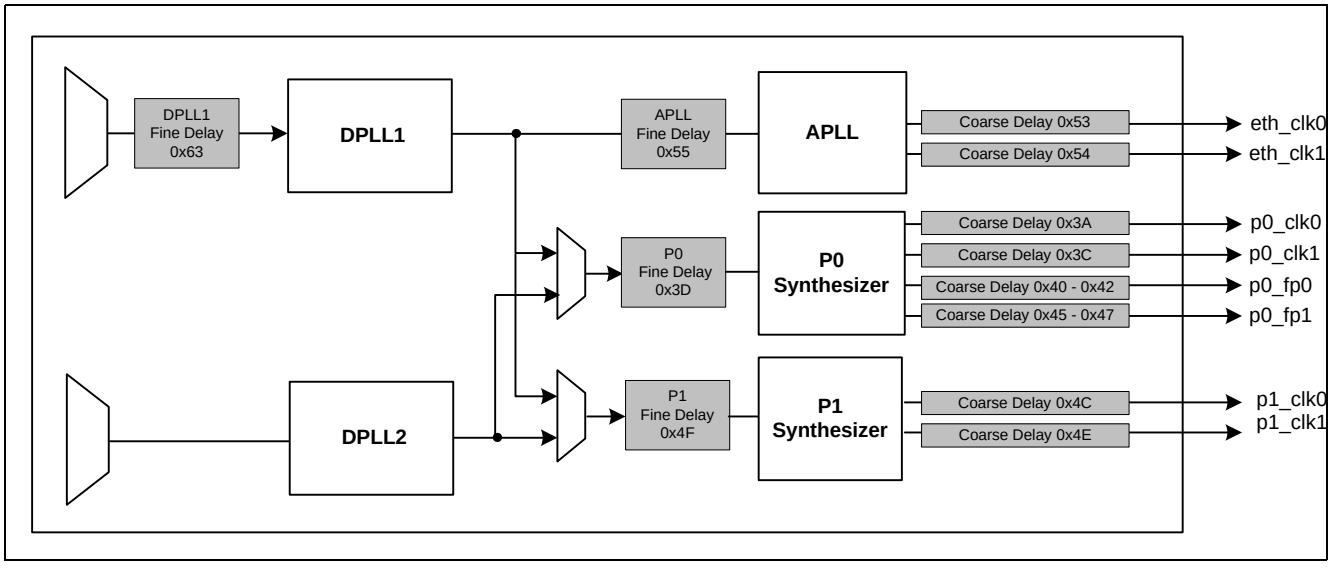



**Figure 37 - Phase Delay Adjustments**

All of the output synthesizers locked to DPLL1 can be configured to lead or lag the selected input reference clock. Register 0x63 allows delay adjustments in steps of 119.2 ps definable as an 8-bit two's complement value in the range of -128 to +127. Negative values delay the output clock, positive values advance the output clock. This gives a total delay adjustment in the range of -15.26 ns to +15.14 ns. Synthesizers that are locked to DPLL2 are unaffected by this delay adjustment.

In addition to the fine delay introduced in the DPLL1 path, the APLL, P0, and P1 synthesizers have the ability to add their own fine delay adjustments by programming registers 0x55, 0x3D, and 0x4F. These registers are also programmed as 8-bit two's complement values representing delays defined in steps of 119.2 ps with a range of -15.26 ns to +15.14 ns.

The output clocks of the APLL, P0, and P1 synthesizers can be independently offset by 90, 180, and 270 degrees using the coarse delay registers (0x53, 0x54, 0x3A, 0x3C, 0x4C, 0x4E).

The output frame pulses can be independently offset with respect to each other using the frame pulse delay registers (0x40 - 0x42, 0x45 - 0x47). Frame pulses generated from the p0 synthesizer (**p0_fp0**, **p0_fp1**) associated with p0 clocks (**p0_clk0**, **p0_clk1**) that are multiples of 2.048 MHz (E1) can be delayed in steps of 1/262.144 MHz (or approx. 3.81 ns). The delay value is programmed as a 16-bit value defined in registers 0x40/0x41 for p0_fp0 and 0x45/0x46 for p0_fp1. The maximum amount of delay is 125 μs (= 32767 * 1/262.14 MHz). In addition, the frame pulses can be delayed in steps of 125 μs (up to 2^6 * 125 μs = 8 ms) using the 0x42 register for p0_fp0 and 0x47 for p0_fp1.

### 4.3.16 Active/Redundant Hardware Configuration

In systems that provide redundant timing sources, it is desirable to minimize the output skew between the master and the slave's output clocks. This can be achieved by synchronizing the slave to one of the master's output clocks instead of synchronizing the slave to an external reference. If frame pulse alignment between the timing sources is required, then the crossover link should consist of a clk/fp pair.

One method of connecting two Microsemi devices in a master/slave configuration is shown in Figure 38 where there is a dedicated crossover link between timing cards. Any of the master's unused outputs and the slave's unused inputs can be used as a crossover link.

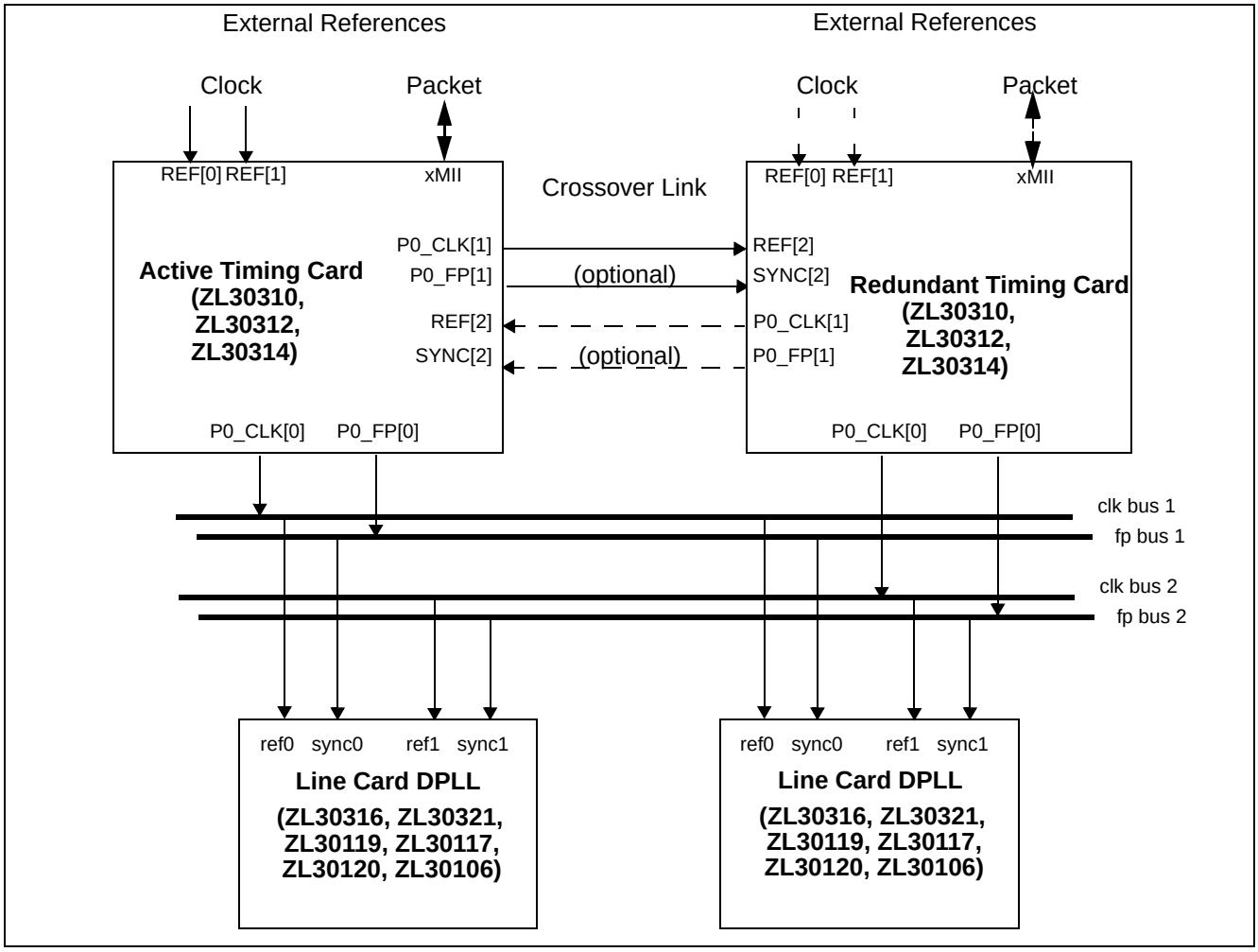



**Figure 38 - Typical Master/Slave Configuration**

### 4.4 Synchronous Serial Interface

The Synchronous Serial Interface is used to perform register read/write operations. This is an extremely simple, low-bit rate interface, and is used for non-time critical configuration tasks. The interface is compatible with a Motorola SPI (Serial Peripheral Interface).

The interface allows for multiple of Microsemi devices to be controlled by a single CPU, by having an individual CS_B for each of Microsemi devices and sharing the data and clock signals.

#### 4.4.1 Transmission Modes

This interface supports two modes of access: Most Significant Bit (MSB) first transmission or Least Significant Bit (LSB) first transmission. The mode is automatically selected based on the state of SCK pin when the CS_B pin is active. If the SCK pin is low during CS_B activation, then MSB first timing is selected. If the SCK pin is high during CS_B activation, then LSB first timing is assumed.

The serial peripheral interface supports half-duplex processor mode which means that during a write cycle to the Microsemi device, output data from the SO pin must be ignored. Similarly, the input data on the SI pin is ignored by the device during a read cycle from the device.

Functional waveforms for the LSB first mode are shown in Figure 39, while Figure 40 describe the MSB first mode.

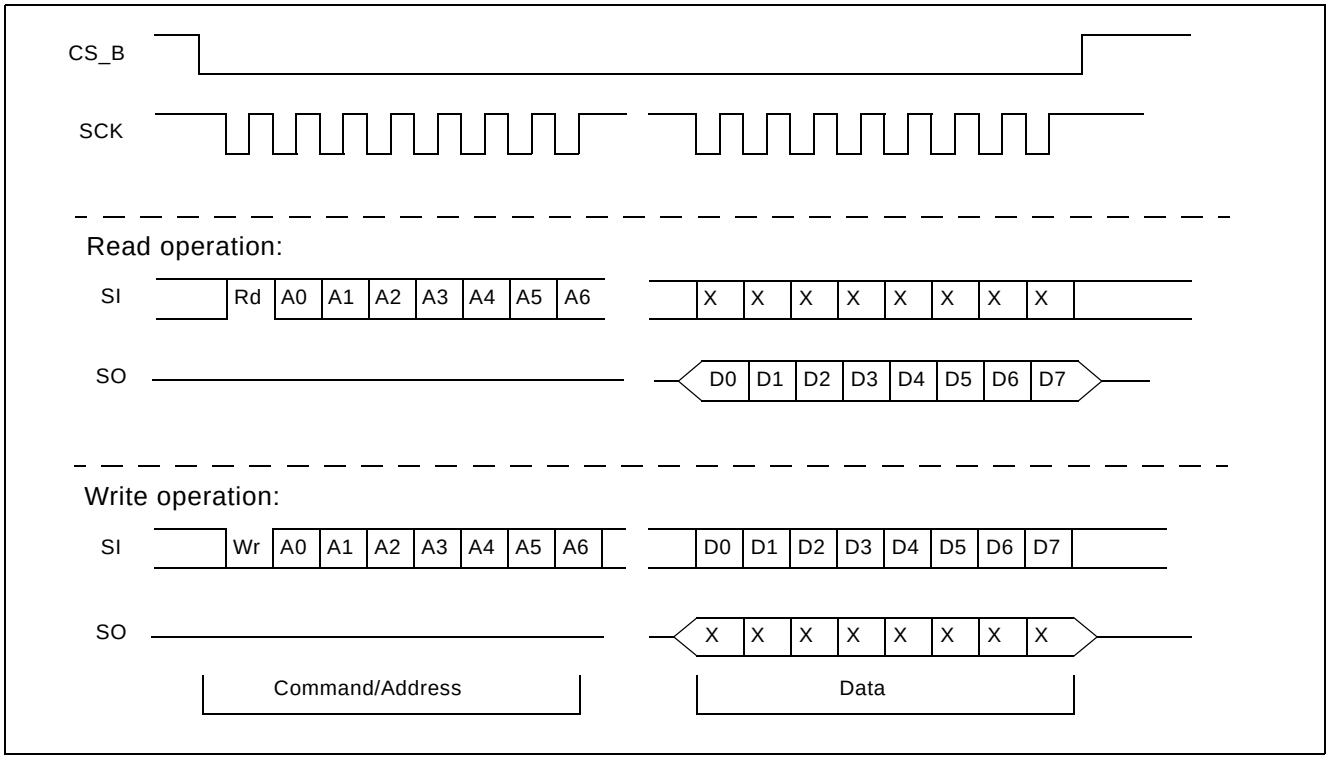



**Figure 39 - Serial Peripheral Interface Functional Waveforms - LSB First Mode**

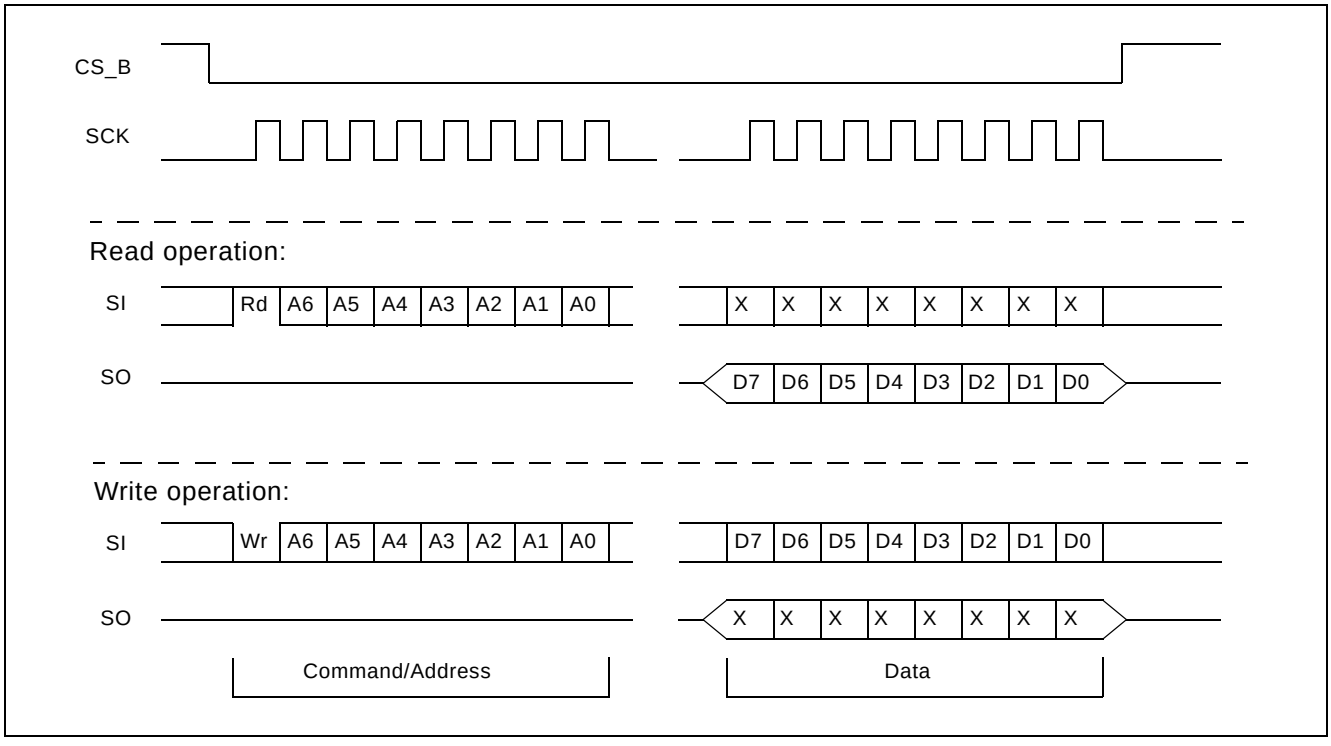



**Figure 40 - Serial Peripheral Interface Functional Waveforms - MSB First Mode**

### 4.4.2 Page Addressing

The device supports a paged addressing scheme, which means that for many registers, the appropriate page must be established first before an access can be made. The addressing scheme is shown in Figure 41. Registers at addresses 0x00 to 0x64 are always accessible directly, with no need to write a new page. Therefore an access to these registers only consumes 16 bits of bandwidth on the SPI (8 bits command/address, 8 bits data).

For registers at addresses 0x65 to 0x7F, there are 16 pages of registers (although only 13 are actually used). To access these registers, the page pointer register at address 0x64 must be written to first, selecting the correct page. The access can then continue as normal. Therefore for registers at these addresses, 32 bits of bandwidth are consumed on the SPI (8 bits command/address, 8 bits page pointer, followed by 8 bits command/address, 8 bits data).

### 4.4.3 Accessing Multi-byte Registers

Multi-byte registers must be accessed as a series of single byte registers, e.g., address/data/address/data. It is not necessary to re-program the page pointer between each access.

To write to a a multi-byte register, the least significant byte (i.e., the byte with the lowest address) should always be written first. The least significant bytes are stored in an intermediate buffer and when the most significant byte arrives from the serial microport, the contents of the buffer register and the last byte are transferred at once to the destination register.

A similar behavior applies to reading dynamic multi-byte registers that are automatically updated by the device. For any of those registers, the user must start by reading the least significant byte which has the lowest address. When the read command for the least significant byte is given, the content of the multi-byte register is copied to an intermediate buffer at once and the bytes are read from that buffer with the most significant byte last.

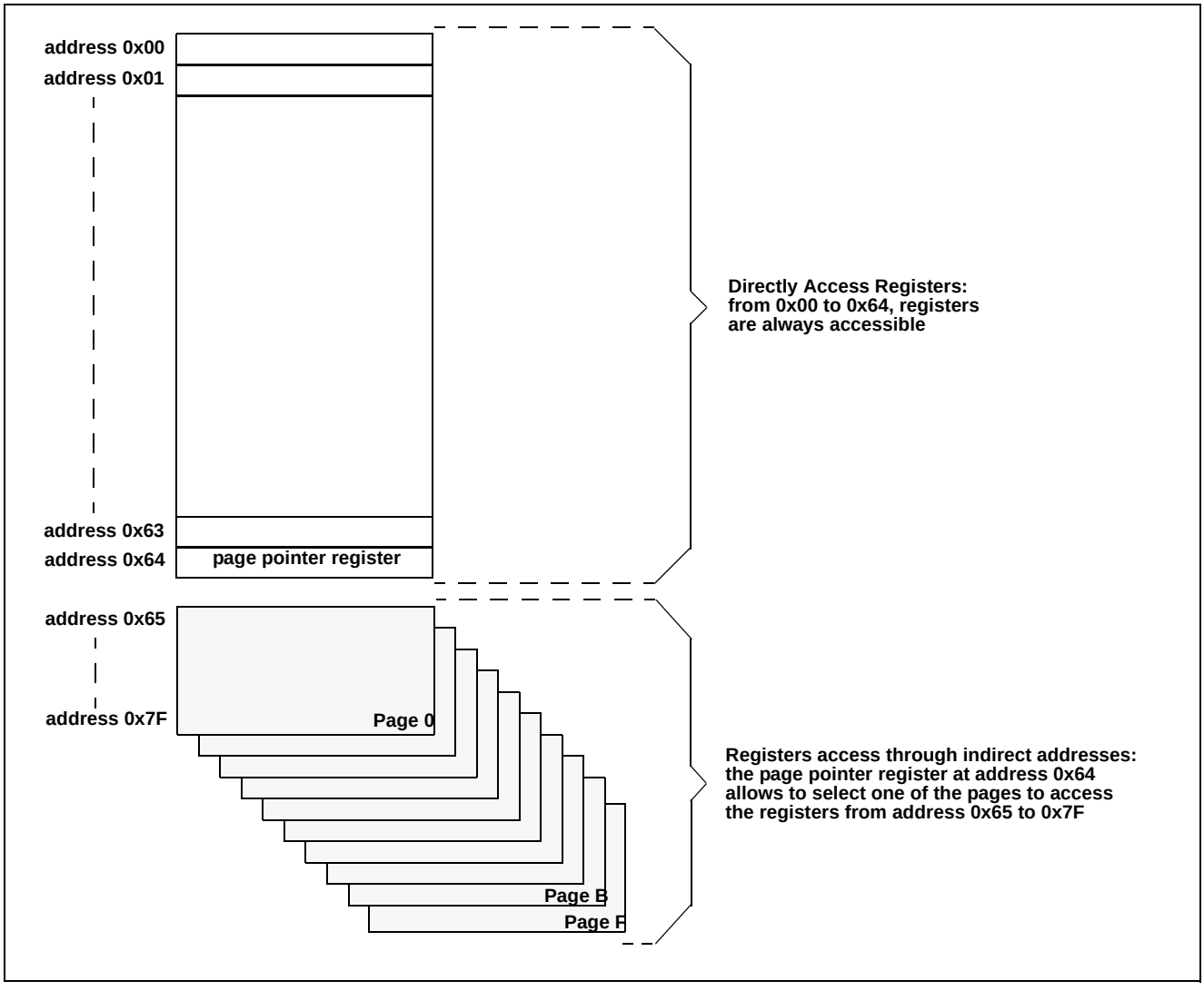



**Figure 41 - Page Addressing Scheme**

## 4.5 Timing Software

The device works in conjunction with a microprocessor to perform the packet processing tasks and the timing recovery algorithms. The software for these tasks is provided as an Application Programmers' Interface (API). The software architecture is shown in Figure 42. The software system consists of Microsemi provided components together with customer provided components. These are discussed in the following sections.

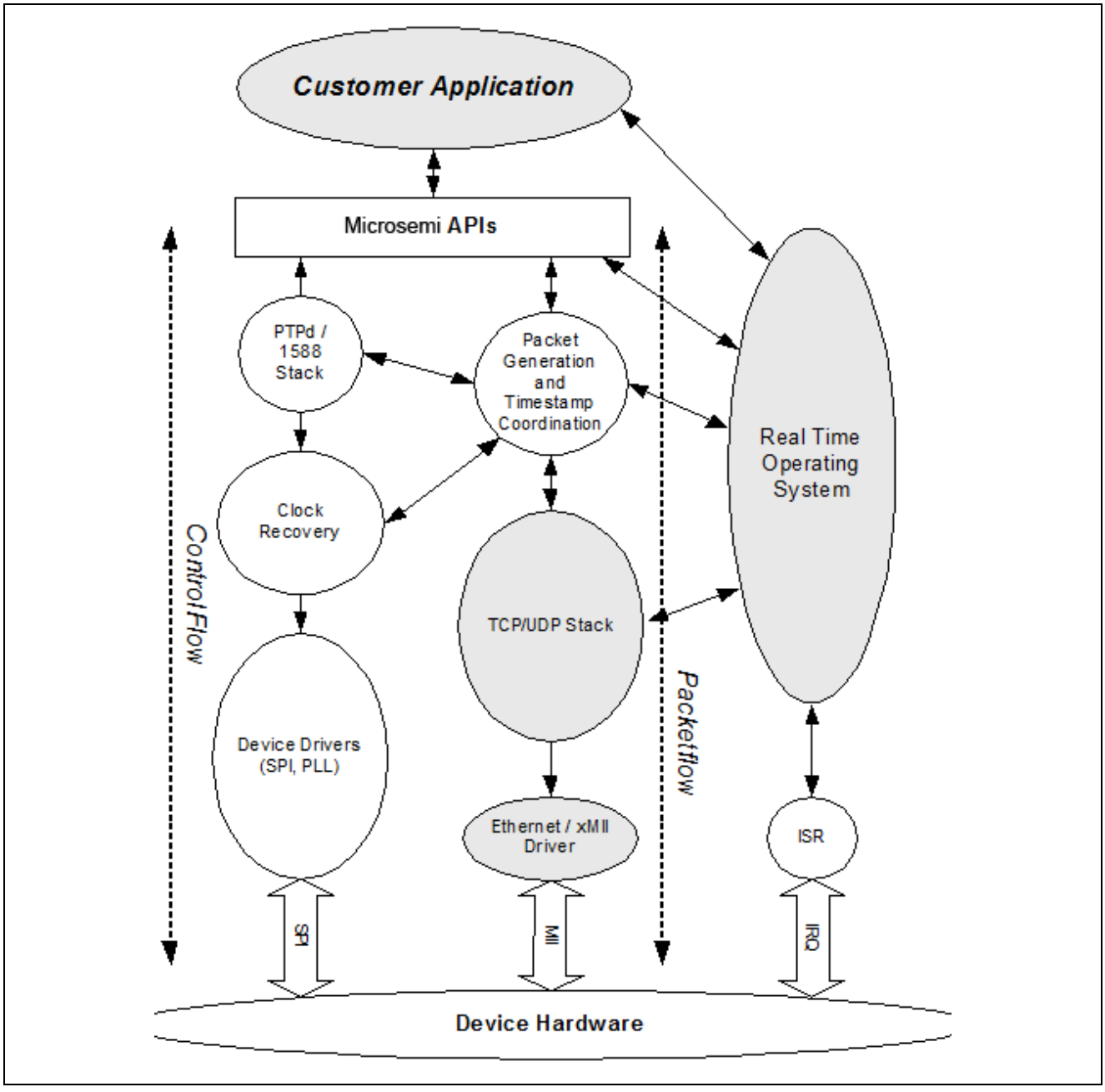



**Figure 42 - High Level Software Architecture**

### 4.5.1 Hardware Requirements

The microprocessor to run the software needs requires at least the following facilities:

- Synchronous Serial Interface compatible with Motorola™ SPI
- Ethernet interface
- At least one external interrupt request pin. Two pins are preferred and used in the sample code provided.
- Timer to provide system tick to the OS
- Optional (but preferred) second timer to provide higher resolution internal timing

### 4.5.2 User Application

The user application is responsible for the following operations:

- initialising the CPU hardware
- configuring initial parameters
- initialising the Microsemi hardware
- binding socket connections to the Microsemi software
- starting up all the Microsemi software tasks/threads

Once started the Microsemi software requires little intervention as it runs freely and will autonomously handle the generation and reception of timing packets and the control of the local oscillator to maintain frequency lock.

Occasional operations, usually in response to user initiated events, will be performed by the application. These are controlled by the Configuration/Control interface. Examples include setting time of day, hardware re-configuration and addition or removal of recovered frequency streams.

The application may choose to provide periodic monitoring of statistics and status information to ensure the system is all running smoothly. This is performed through the Statistics Collection interface.

The Microsemi software uses specific sockets for transporting timing packets but the application may also use the socket interface for sending and receiving non-timing packets. These can be physically routed via the Ethernet driver and through the device which behaves transparently.

#### 4.5.2.1 User-Provided Software Components

In addition to the application code, the user needs to provide the following software elements:

- Real time operating system
- IP/UDP Stack
- Ethernet Port Driver
- RX Transport Function
- Processor Specific SSI Driver
- Interrupt Interface

Further details of these components are provided in the solution API User Manual.

### 4.5.3 Microsemi-Provided Software Components

#### 4.5.3.1 Solution API

The solution API is a proven software interface which can be used by end users as a solid foundation on which to base their application code development. It is coded in ANSI C for maximum portability, and the API includes tracing facilities to allow users to gain visibility of what is going in inside its functions. Microsemi Support may ask a customer to turn on tracing to troubleshoot a problem.

The solution API has been developed around the VxWorks Real Time Operating System, however it has been designed to minimize the dependency on the RTOS, and make porting to a new RTOS as easy as possible. Therefore all operating system calls have been mapped via macros and isolated to one header file. This file contains the macros and detailed comments about their usage.

#### 4.5.3.2 IEEE 1588 Stack

The IEEE1588 protocol stack is provided by Microsemi, and implements relevant parts of the IEEE 1588 Precision Timing Protocol version 2. It generates messages in the required format, extracts information from received packets and responds to received messages in an appropriate manner.

The stack provides the following features:

- Operation as an IEEE 1588 Ordinary Clock. i.e. as the server or client of an end-to-end connection, with or without intervening boundary or transparent clocks
- IEEE 1588 v2 packet message formats required for the operation as an Ordinary Clock
- IEEE 1588 v2 message responses and protocol state machine
- Pre-configured server-client hierarchy. Supports IEEE 1588 best master clock (BMC) algorithm, but also provides customization to configure a packet stream as server only or client only. This can aid in controlling the configuration of the timing network where it may be preferable to dictate the hierarchy of which clocks will be clients to which servers. The BMC potentially allows any clock to become server which may result in unintentional timing paths.
- IEEE 1588 v2 increased sync message rates up to 128 packets per second (to provide improved timing resolution).
- IEEE 1588 v2 stack in source code
- Frequency recovery
- Time of day

#### 4.5.3.3 Other Software

The other blocks of Microsemi-provided code provide the means to drive the device in an appropriate manner. These blocks are internal to the software system and would not generally need to be called by the customer application although some initial parameter configuration of these blocks may be required as shown in the example code.

- RTOS abstraction: This layer allows porting to different OS environments.
- Socket abstraction: This layer allows porting to different socket and transport mechanism libraries
- Timestamp Coordination. This interface allows the IEEE1588 stack to interface to the device hardware for the purposes of timestamping transmitted and received packets.
- Packet launch: This allows the IEEE1588 stack to launch packets at particular times. Part of this module is provided as object code.
- PTP Interface: A flexible interface to the PTP stack

- Clock Recovery: This uses the information in the timing packets to recover a clock at a particular frequency. Part of this module is provided as object code only.
- Device Control. Provides a software interface for controlling the device.
- PLL/DCO Control. For a client device this controls the recovered frequency output.
- Generic SSI driver. This block provides a generic interface for an SSI device.

### 4.5.4 Software Performance

The primary driver of software performance is the protocol stack itself, rather than the clock recovery function or device control, therefore the CPU utilization is almost directly proportional to the number of timing packets to be processed and mode timing packet are sent or received, unicast vs. multicast packets. This is true at both the client and server functions.

#### 4.5.4.1 Use of Multicast vs. Unicast Packets

The Microsemi device can use either multicast or unicast packets for packet timing streams, however, users should be aware of the various trade-offs that are involved with each.

**Use of Multicast Packets**

The use of multicast packets has two distinct advantages:

1. It reduces the overall volume of timing packets (e.g., PTP *sync* messages) in the network, and in particular the concentration of packets leaving the server
2. It increases the number of clients a server can address, since it doesn't have to generate a unique timing stream for each client (note it still has to service PTP *delay_request* messages from each individual client).

However, there are also some disadvantages. Multicast packets are not forwarded as efficiently in the network. Whenever a multicast packet reaches a switch or router, it has to be replicated many times in order to be sent out of each exit port of the device. This replication process adds both delay and delay variation to the packet. Therefore the delay variation accumulated by the time a timing packet stream reaches a client is much greater when using multicast than with unicast. Further, some routers are configured to block or limit the rate of multicast packets, making those streams more unreliable.

**Use of Unicast Packets**

For the reasons outlined above, unicast packets are expected to be the main choice in telecommunications applications. They propagate through the packet network better, and are much more efficient at the client, which is usually the cost-sensitive element where the CPU power needs to be minimized.

The main disadvantage of unicast packets is that the server is much less efficient. This is because the server has to create a separate *sync* message stream to each individual client. Where the sync message rate is high, and the number of clients is also high, this consumes a large amount of CPU time. For this reason, the CPU at a server device should normally be dedicated to the task of running the server, and shouldn't be expected to cope with other tasks.

# 5.0 DC Characteristics

## 5.1 Absolute Maximum Ratings*

| | Parameter | Symbol | Min. | Max. | Units |
|---|---|---|---|---|---|
| 1 | I/O Supply Voltage | $V_{DD33}$, $AV_{DD33}$ | -0.5 | 4.6 | V |
| 2 | Core Supply Voltage | $V_{DD18}$, $AV_{DD18}$ | -0.5 | 2.5 | V |
| 3 | Voltage on any digital pin | $V_{PIN}$ | -0.5 | 5.5 | V |
| 4 | Voltage on OSC_I and OSC_O pins | $V_{OSC}$ | -0.3 | 3.6 | V |
| 5 | Package power dissipation | PD | - | 1.5 | W |
| 6 | Storage Temperature | TS | -55 | +125 | °C |

* Exceeding these figures may cause permanent damage. Functional operation under these conditions is not guaranteed. Voltage measurements are with respect to ground ($V_{SS}$) unless otherwise stated.

* The core supply voltages must never be allowed to exceed the I/O supply voltage by more than 0.5 V during power-up. Failure to observe this rule could lead to a high-current latch-up state, possibly leading to chip failure, if sufficient cross-supply current is available. To be safe ensure the I/O supply voltages supply always rise earlier than the core supply voltages.

## 5.2 Recommended Operating Conditions

| | Characteristics | Symbol | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| 1 | Supply Voltage | $V_{DD33}$, $A_{VDD33}$ | 3.1 | 3.3 | 3.5 | V |
| 2 | Core Supply Voltage | $V_{DD18}$, $AV_{DD18}$ | 1.7 | 1.8 | 1.9 | V |
| 3 | Operating Temperature | $T_A$ | -40 | 25 | +85 | °C |

* Voltage measurements are with respect to ground ($V_{SS}$) unless otherwise stated.

## 5.3 DC Electrical Characteristics

Typical characteristics are at 1.8 V core, 3.3 V I/O, 25°C and typical processing. The min. and max. values are defined over all process conditions, from -40 to 125°C junction temperature, core voltage 1.7 to 1.9 V and I/O voltage 3.1 to 3.5 V unless otherwise stated.

| | Characteristics | Symb. | Min. | Typ. | Max. | Units. | Notes |
|---|---|---|---|---|---|---|---|
| 1 | 1.8 V core supply current | $I_{DD18}$ | | 227 | 310 | mA | $f_{OSC_I}$ = 20 MHz<br>Both Ethernet ports operating at 1 Gbit/s<br>All outputs at maximum frequency. |
| 2 | 3.3 V I/O supply current | $I_{DD33}$ | | 159 | 220 | mA | |
| 3 | Total Power Dissipation | $P_{TD}$ | | 0.94 | 1.4 | W | |
| 3 | Input Leakage current | $I_{IL}$ | -15 | | 15 | µA | $V_I = V_{DD33}$ or 0 V |
| 4 | Input Leakage current for pull-up pads | $I_{IL_PU}$ | -150 | | -30 | µA | $V_I$ = 0 V |
| 5 | Input Leakage current for pull-down pads | $I_{IL_PD}$ | 30 | | 150 | µA | $V_I = V_{DD33}$ |

**Input Levels**

| | Characteristics | Symb. | Min. | Typ. | Max. | Units | Test Condition |
|---|---|---|---|---|---|---|---|
| 6 | Input Low Voltage | $V_{IL}$ | | | 0.8 | V | |
| 7 | Input High Voltage | $V_{IH}$ | 2.0 | | | V | |
| 8 | Positive Schmitt Threshold | $V_{T+}$ | 1.35 | | 1.85 | V | |
| 9 | Negative Schmitt Threshold | $V_{T-}$ | 0.8 | | 1.15 | V | |

**Output Levels**

| | Characteristics | Symb. | Min. | Typ. | Max. | Units | Test Condition |
|---|---|---|---|---|---|---|---|
| 10 | Output Low Voltage | $V_{OL}$ | | | 0.4 | V | $I_{OL}$ = 12 mA for packet interface clocks<br>$I_{OL}$ = 8 mA for packet interface data pins and PLL output clocks<br>$I_{OL}$ = 4 mA for other outputs |
| 11 | Output High Voltage | $V_{OH}$ | 2.4 | | | V | $I_{OH}$ = 12 mA for packet interface clocks<br>$I_{OH}$ = 8 mA for packet interface data pins and PLL output clocks<br>$I_{OH}$ = 4 mA for other outputs |

# 6.0 AC Characteristics

## 6.1 Clock Interface Timing

### 6.1.1 Input Timing For Sync References*

| | Characteristics | Symbol | Min. | Max. | Units | Notes |
|---|---|---|---|---|---|---|
| 1 | SYNC[0,1,2] lead time | $t_{SYNC_LD}$ | | 0 | ns | |
| 2 | SYNC[0,1,2] lag time | $t_{SYNC_LG}$ | 0 | $t_{REFP}$ - 5 | ns | $t_{REFP}$ = minimum period of ref0/1/2 clock |
| 3 | SYNC[0,1,2] pulse width high or low | $t_{SYNC_W}$ | 5 | | ns | |

* Supply voltage and operating temperature are as per Recommended Operating Conditions.

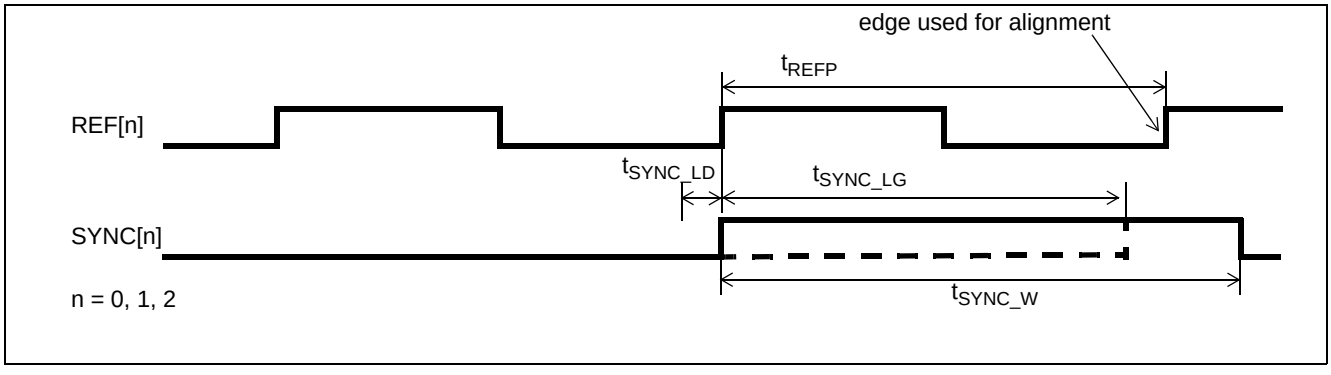



**Figure 41 - Sync Input Timing**

### 6.1.2 Input To Output Timing For Ref<n> References*

| | Characteristics | Symbol | Min. | Max. | Units |
|---|---|---|---|---|---|
| 1 | LVCMOS Clock Outputs (P0_CLK[0,1], P1_CLK[0,1]) | $t_D$ | -2 | 4.2 | ns |
| 2 | LVCMOS Ethernet Clock Outputs (ETH_CLK[0,1]) | $t_D$ | -2 | 4 | ns |

* Input to output timing is measured over the specified operating voltage and temperature ranges using the same input and output spot frequencies of 2 kHz, 8 kHz, 1.544 MHz, 2.048 MHz, 6.48 MHz, 8.192 MHz, 16.384 MHz, 19.44 MHz, 38.88 MHz and 77.76 MHz.

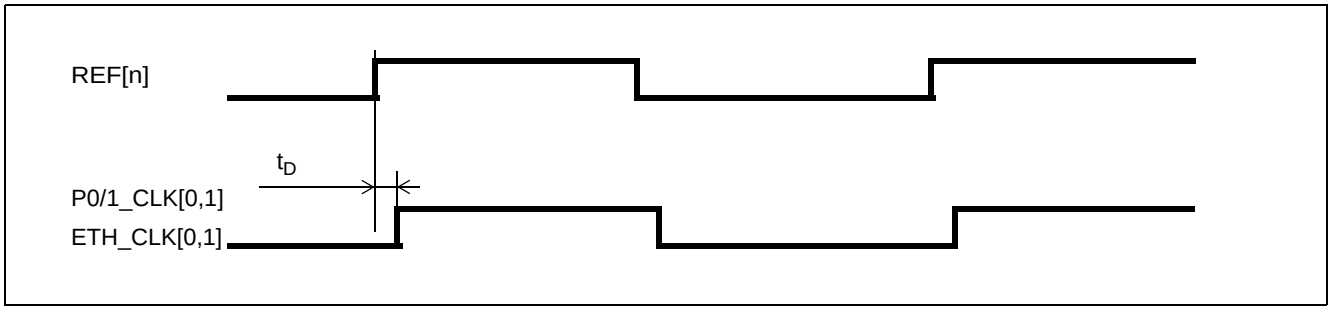



**Figure 42 - Input To Output Timing**

### 6.1.3 Output Clock Duty Cycle[1]

| | Characteristics | Symbol | Min | Max | Units | Notes |
|---|---|---|---|---|---|---|
| 1 | LVCMOS Output Duty Cycle[2] | $t_{SYM}$ | 45 | 55 | % | 2 kHz < $f_{clk}$ ≤ 125 MHz |
| 2 | | | 40 | 60 | % | 50 MHz on eth_clk0/1 |

1. Duty cycle is measured over the specified operating voltage and temperature ranges at specified spot frequencies.

2. Measured on spot frequencies of 1.544 MHz, 2.048 MHz, 3.088 MHz, 4.096 MHz, 6.312 MHz, 8.192 MHz, 8.448 MHz, 16.384 MHz, 32.768 MHz, 34.368 MHz, 44.736 MHz, 65.536 MHz, 125 MHz.

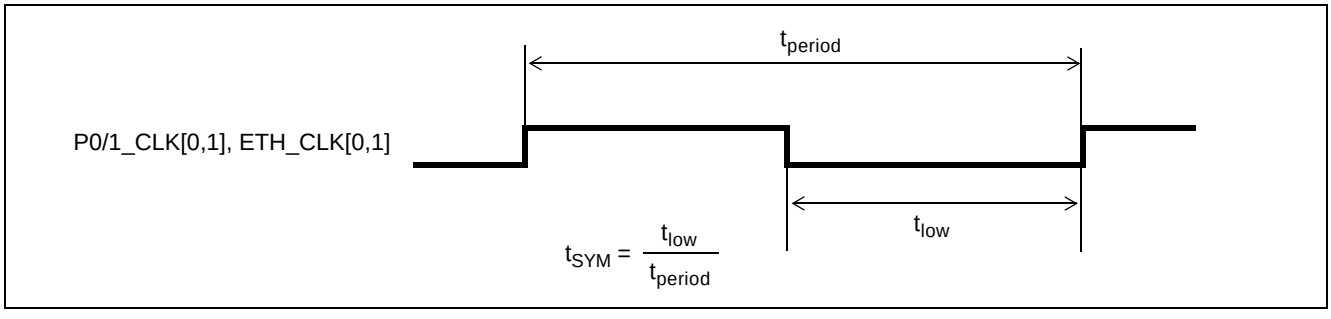



**Figure 43 - Output Duty Cycle**

### 6.1.4 Output Clock and Frame Pulse Fall and Rise Times[1]

| | Characteristics | Symbol | Min | Max | Units | $C_{LOAD}$ |
|---|---|---|---|---|---|---|
| 1 | Output Rise Time | $t_{rise}$ | 2.0 | 4.0 | ns | 25 pF |
| 2 | Output Rise Time | $t_{rise}$ | 1.3 | 2.7 | ns | 15 pF |
| 3 | Output Fall Time | $t_{fall}$ | 2.0 | 4.2 | ns | 25 pF |
| 4 | Output Fall Time | $t_{fall}$ | 1.3 | 2.9 | ns | 15 pF |

1. Output fall and rise times are specified over the operating voltage and temperature ranges at 10 MHz.

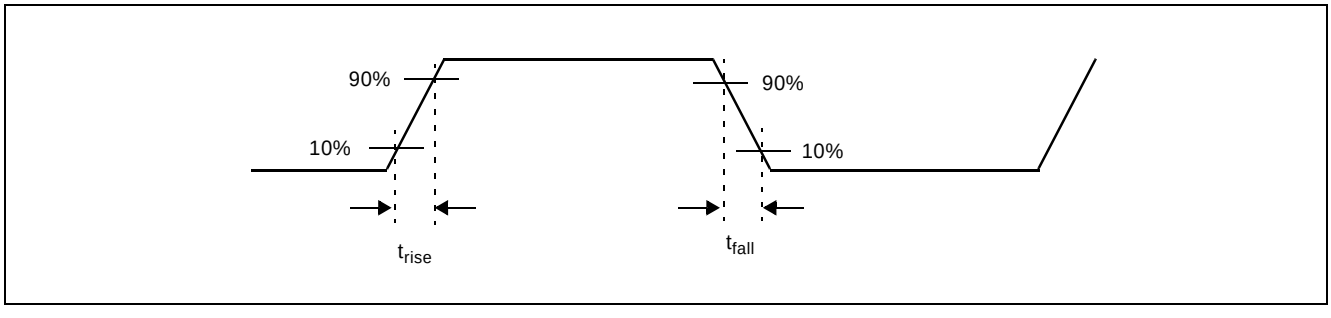
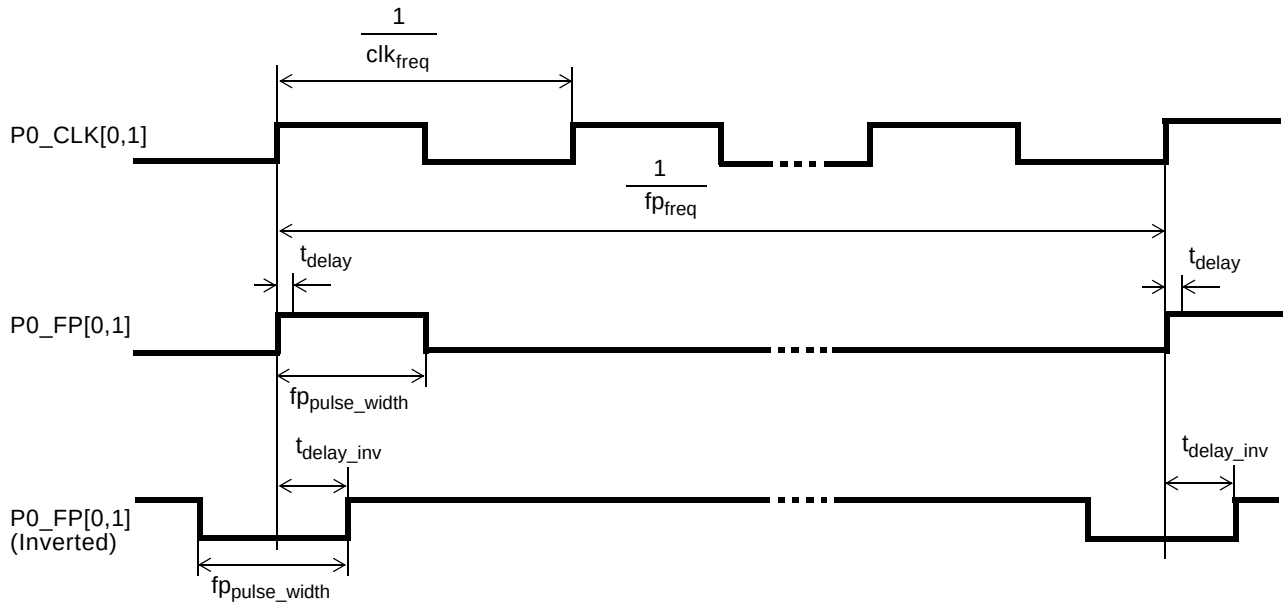



**Figure 44 - Output Clock Fall and Rise Times**

### 6.1.5 E1 Output Frame Pulse Timing*

| | Pulse Width Setting | $fp_{pulse_width}$ | | $t_{delay}$ | | $t_{delay_inv}$ | | Units |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min. | Max. | Units |
| 1 | One period of a 4.096 MHz clock | 243 | 245 | -2 | 2 | 120 | 124 | ns |
| 2 | One period of a 8.192 MHz clock | 121 | 123 | -2 | 2 | 59 | 63 | ns |
| 3 | One period of a 16.384 MHz clock | 60 | 62 | -2 | 2 | 28 | 33 | ns |
| 4 | One period of a 32.768 MHz clock | 30 | 31 | -2 | 2 | 13 | 17.5 | ns |
| 5 | One period of a 65.536 MHz clock | 14 | 16 | -2 | 2 | 5.6 | 9.6 | ns |

* All measurements taken over the specified operating voltage and temperature range




Notes: 1. $clk_{freq}$ and $fp_{freq}$ are configurable using register settings.
2. $clk_{freq}$ = 2.048 MHz, 4.096 MHz, 8.192 MHz, 16.384 MHz, 32.768 MHz, or 65.536 MHz.
3. $fp_{freq}$ = 64 kHz, 32 kHz, 8 kHz, 4 kHz, 2 kHz, 1 kHz, 400 Hz, or 166.6667 Hz.

**Figure 45 - E1 Output Frame Pulse Timing**

### 6.1.6 Measured Output Jitter On Ethernet Clock CMOS Outputs with Active MII Interface (ETH_CLK[0,1])

| Output Frequency | Jitter Measurement Filter | Jitter Generation | | |
|---|---|---|---|---|
| | | Typ[1] | Max[2] | Units |
| 25 MHz | 12 kHz to 10 MHz | 5 | 12 | $ps_{RMS}$ |
| | | 50 | 135 | $ps_{P-P}$ |
| 125 MHz | 12 kHz to 20 MHz | 4 | 7 | $ps_{RMS}$ |
| | | 35 | 50 | $ps_{P-P}$ |

[1] Typical jitter specifications are measured when operating at nominal voltages of 1.8 V and 3.3 V and at an ambient temperature of 25°C.

[2] Maximum jitter specifications takes into account process variations and is measured over the entire operating temperature range and voltage range with all outputs enabled.

### 6.1.7 Measured Output Jitter On Ethernet Clock CMOS Outputs with Active GMII Interface (ETH_CLK[0,1])

| Output Frequency | Jitter Measurement Filter | Jitter Generation | | |
|---|---|---|---|---|
| | | Typ[1] | Max[2] | Units |
| 25 MHz | 12 kHz to 10MHz | 14 | 30 | $ps_{RMS}$ |
| | | 160 | 350 | $ps_{P-P}$ |
| 125 MHz | 12 kHz to 20 MHz | 13 | 30 | $ps_{RMS}$ |
| | | 150 | 350 | $ps_{P-P}$ |

[1] Typical jitter specifications are measured when operating at nominal voltages of 1.8 V and 3.3 V and at an ambient temperature of 25°C.

[2] Maximum jitter specifications takes into account process variations and is measured over the entire operating temperature range and voltage range with all outputs enabled.

### 6.1.8 Measured Output Jitter On Programmable CMOS Outputs (P0/1_CLK[0,1])

| Output Frequency | Jitter Measurement Filter | Jitter Generation | | |
|---|---|---|---|---|
| | | Typ[1] | Max[2] | Units |
| 2 kHz to 77.76 MHz | unfiltered | 22 | 30 | $ps_{RMS}$ |

[1] Typical jitter specifications are measured when operating at nominal voltages of 1.8 V and 3.3 V and at an ambient temperature of 25°C.

[2] Maximum jitter specifications takes into account process variations and is measured over the entire operating temperature range and voltage range with all outputs enabled.

[3] Note that locking DPLL2 to a reference input of 2 kHz or to any value of N*8 kHz that results in an exact integer division of 80 MHz may cause output jitter as high as 13 ns pk-pk. This includes 2 kHz, 8 kHz, 16 kHz, 32 kHz, 40 kHz, 64 kHz, 80 kHz, 128 kHz, 160 kHz, 200 kHz, 320 kHz, 400 kHz, 640 kHz, 800 kHz, 1.000 MHz, 1.600 MHz, 2.000 MHz, 3.200 MHz, 4.000 MHz, 5.000 MHz, 8.000 MHz, 10.000 MHz, 16.000 MHz, 20.000 MHz and 40.000 MHz.

## 6.2 Packet Interface Timing

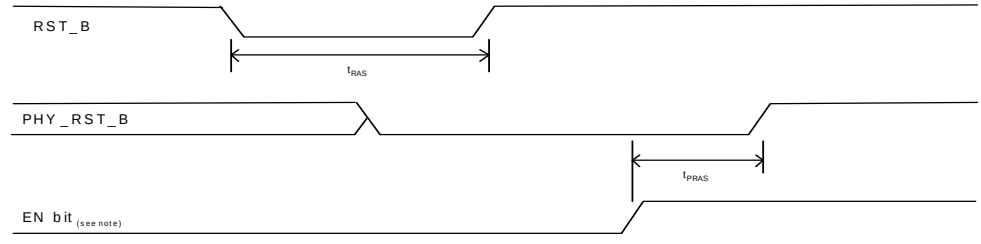
Data for the MII/GMII/TBI packet interface is based on IEEE Standard 802.3 - 2005.
Data for the RMII packet interface is based on the RMII™ Specification from the RMII Consortium™, March 1998.

For MII/GMII/TBI Setup and Hold trigger levels in reference to the clock are based on IEEE Standard 802.3 - 2005.

### 6.2.1 Typical Reset Timing Diagram

| Parameter | Symbol | Min. | Typ. | Max. | Units | Notes |
|---|---|---|---|---|---|---|
| RST_B assert time | $t_{RAS}$ | 300 | | | ns | |
| PHY_RST_B assert time | $t_{PRAS}$ | | 1 | | ms | |

**Table 15 - Reset Timing**




NOTE: EN bit is set via bridge enable register

**Figure 46 - Typical Reset Timing Diagram**

### 6.2.2 MII Transmit Timing: MAC to PHY Connections

For MAC to PHY connections, Mn_TXCLK is an input.

| Parameter | Symbol | 100 Mbps | | | Units | Notes |
|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | | |
| Mn_TXCLK period | $t_{TCC}$ | - | 40 | - | ns | |
| Mn_TXCLK high time | $t_{TCH}$ | 14 | - | 26 | ns | |
| Mn_TXCLK low time | $t_{TCL}$ | 14 | - | 26 | ns | |
| Mn_TXCLK rise time | $t_{TCR}$ | - | - | 5 | ns | |
| Mn_TXCLK fall time | $t_{TCF}$ | - | - | 5 | ns | |
| Mn_TXCLK rise to Mn_TXD[3:0] active delay | $t_{DV}$ | 2 | - | 25 | ns | Load = 25 pF |
| Mn_TXCLK rise to Mn_TXEN active delay | $t_{EV}$ | 2 | - | 25 | ns | Load = 25 pF |
| Mn_TXCLK rise to Mn_TXER active delay | $t_{ER}$ | 2 | - | 25 | ns | Load = 25 pF |

**Table 16 - MII Transmit Timing - 100 Mbps**

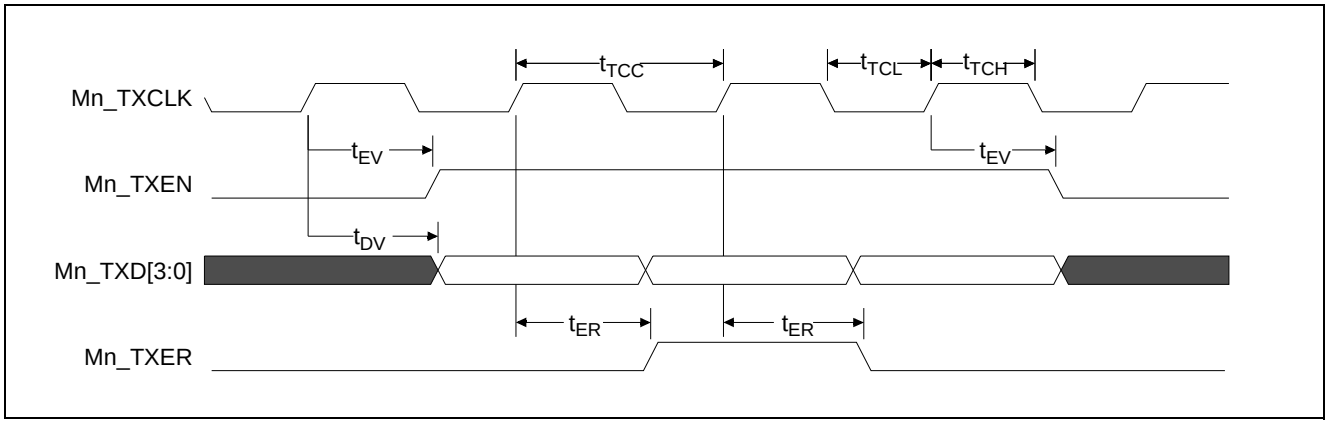



**Figure 47 - MII Transmit Timing Diagram**

### 6.2.3 MII Receive Timing: MAC to PHY Connections

| Parameter | Symbol | 100 Mbps | | | Units | Notes |
|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | | |
| Mn_RXCLK period | $t_{RCC}$ | - | 40 | - | ns | |
| Mn_RXCLK high wide time | $t_{RCH}$ | 14 | | 26 | ns | |
| Mn_RXCLK low wide time | $t_{RCL}$ | 14 | | 26 | ns | |
| Mn_RXCLK rise time | $t_{RCR}$ | - | - | 5 | ns | |
| Mn_RXCLK fall time | $t_{RCF}$ | - | - | 5 | ns | |
| Mn_RXD[3:0] setup time (Mn_RXCLK rising edge) | $t_{DS}$ | 8 | - | - | ns | |
| Mn_RXD[3:0] hold time (Mn_RXCLK rising edge) | $t_{DH}$ | 2 | - | - | ns | |
| Mn_RXDV input setup time (Mn_RXCLK rising edge) | $t_{DVS}$ | 8 | - | - | ns | |
| Mn_RXDV input hold time (Mn_RXCLK rising edge) | $t_{DVH}$ | 2 | - | - | ns | |
| Mn_RXER input setup time (Mn_RXCLK rising edge) | $t_{ERS}$ | 8 | - | - | ns | |
| Mn_RXER input hold time (Mn_RXCLK rising edge) | $t_{ERH}$ | 2 | - | - | ns | |

**Table 17 - MII Receive Timing - 100 Mbps**

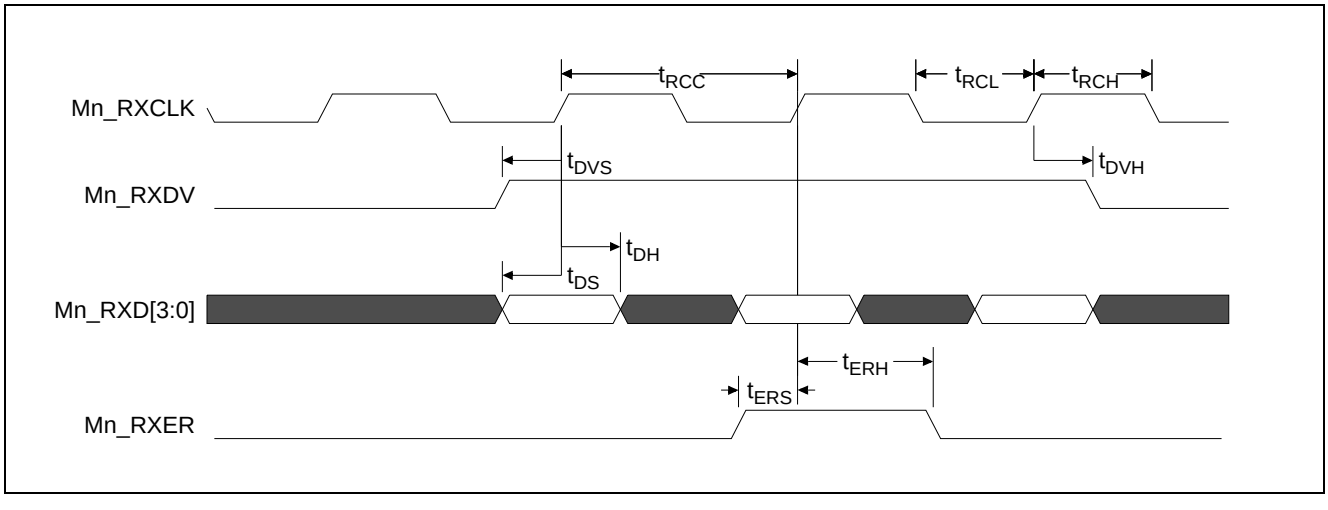



**Figure 48 - MII Receive Timing Diagram**

### 6.2.4 MII Transmit Timing: PHY Emulation Mode (MAC to MAC connections)

For MAC to MAC connections, the ZL30310 can emulate a PHY, with Mn_TXCLK as an output. Data signals (Mn_TXD, Mn_TXEN and Mn_TXER) are driven from the falling edge of Mn_TXCLK to provide sufficient hold time at the receiving device.

| Parameter | Symbol | 100 Mbps | | | Units | Notes |
|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | | |
| Mn_TXCLK period | $t_{TCC}$ | - | 40 | - | ns | |
| Mn_TXCLK high time | $t_{TCH}$ | 16 | - | 24 | ns | |
| Mn_TXCLK low time | $t_{TCL}$ | 16 | - | 24 | ns | |
| Mn_TXCLK rise time | $t_{TCR}$ | - | - | 1 | ns | |
| Mn_TXCLK fall time | $t_{TCF}$ | - | - | 1 | ns | |
| Mn_TXCLK rise to Mn_TXD[3:0] active delay | $t_{DV}$ | 2 | - | 25.5 | ns | Load = 25 pF |
| Mn_TXCLK rise to Mn_TXEN active delay | $t_{EV}$ | 2 | - | 25.5 | ns | Load = 25 pF |
| Mn_TXCLK rise to Mn_TXER active delay | $t_{ER}$ | 2 | - | 25.5 | ns | Load = 25 pF |

**Table 18 - MII Transmit Timing - 100 Mbps - PHY Emulation Mode**

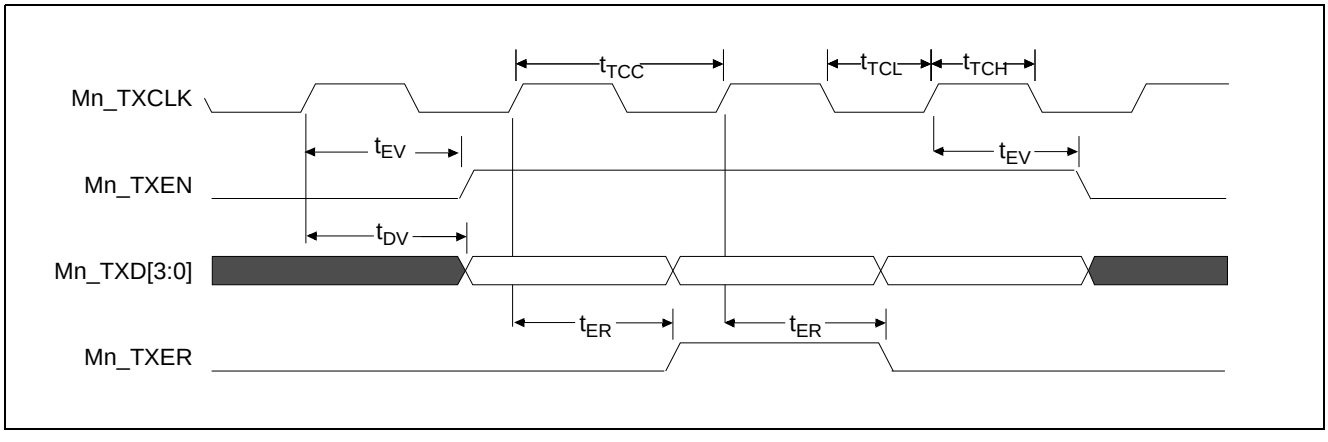



**Figure 49 - MII Transmit Timing Diagram - PHY Emulation Mode**

### 6.2.5 MII Receive Timing: PHY Emulation Mode (MAC to MAC connections)

For MAC to MAC connections, the ZL30310 can emulate a PHY. In this mode Mn_RXCLK is not connected, and the data is timed relative to Mn_TXCLK which is an output.

| Parameter | Symbol | 100 Mbps | | | Units | Notes |
|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | | |
| Mn_RXD[3:0] setup time (Mn_TXCLK rising edge) | $t_{DS}$ | 15 | - | - | ns | |
| Mn_RXD[3:0] hold time (Mn_TXCLK rising edge) | $t_{DH}$ | 0 | - | - | ns | |
| Mn_RXDV input setup time (Mn_TXCLK rising edge) | $t_{DVS}$ | 15 | - | - | ns | |
| Mn_RXDV input hold time (Mn_TXCLK rising edge) | $t_{DVH}$ | 0 | - | - | ns | |
| Mn_RXER input setup time (Mn_TXCLK rising edge) | $t_{ERS}$ | 15 | - | - | ns | |
| Mn_RXER input hold time (Mn_TXCLK rising edge) | $t_{ERH}$ | 0 | - | - | ns | |

**Table 19 - MII Receive Timing - 100 Mbps**

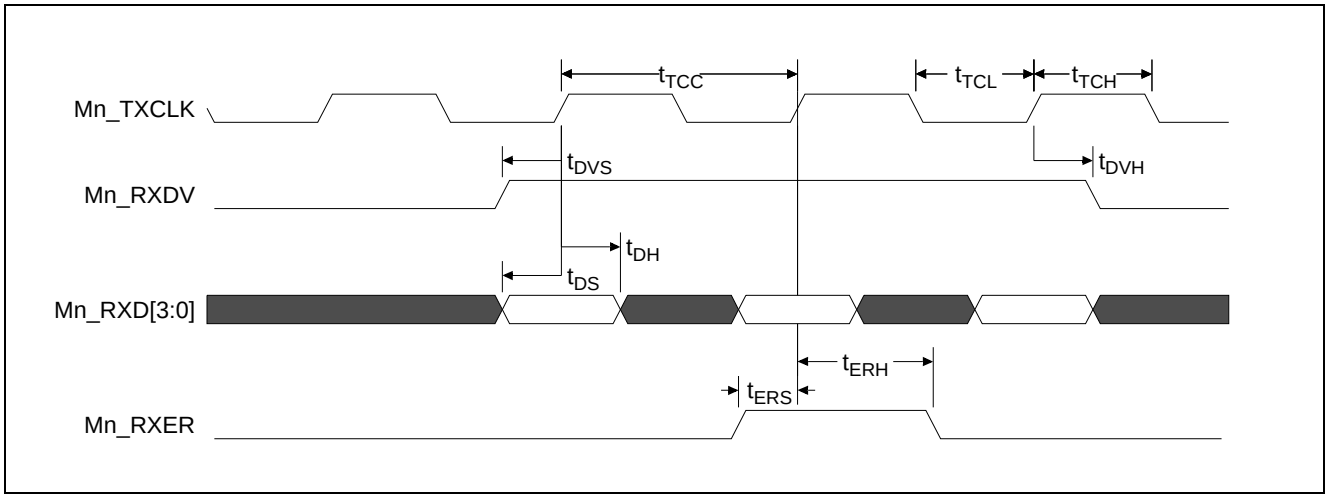



**Figure 50 - MII Receive Timing Diagram**

### 6.2.6 RMII Transmit Timing

| Parameter | Symbol | 100 Mbps | | | Units | Notes |
|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | | |
| Mn_REFCLK period | $t_{RRC}$ | - | 20 | - | ns | ±50ppm |
| Mn_REFCLK high time | $t_{RCH}$ | 7 | - | 13 | ns | |
| Mn_REFCLK low time | $t_{RCL}$ | 7 | - | 13 | ns | |
| Mn_REFCLK rise time | $t_{RCR}$ | 1 | - | 5 | ns | |
| Mn_REFCLK fall time | $t_{RCF}$ | 1 | - | 5 | ns | |
| Mn_REFCLK rise to Mn_TXD[1:0] active delay | $t_{DV}$ | 2 | - | 15 | ns | Load = 25 pF |
| Mn_REFCLK rise to Mn_TXEN active delay | $t_{EV}$ | 2 | - | 15 | ns | Load = 25 pF |

**Table 20 - RMII Transmit Timing - 100 Mbps**

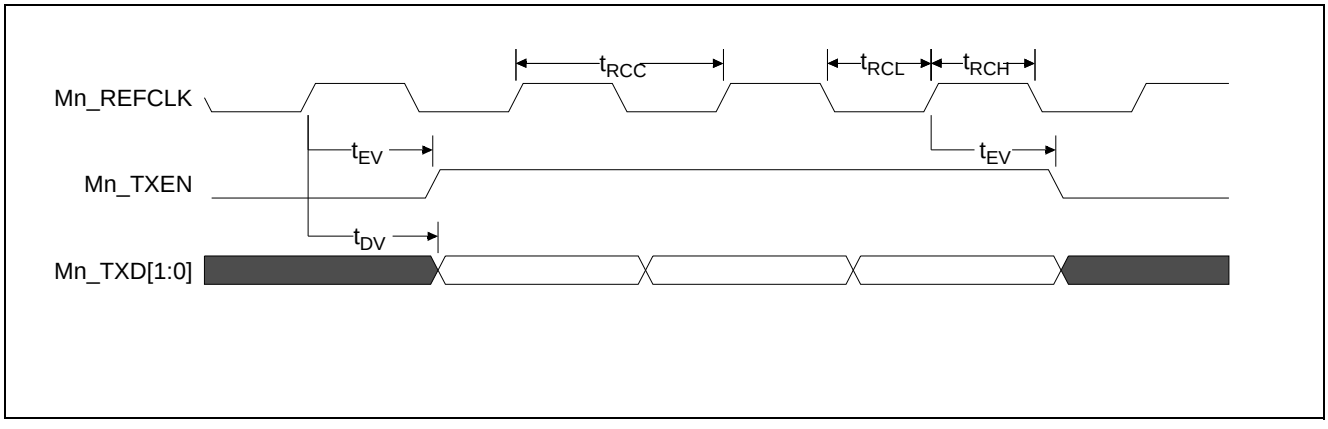



**Figure 51 - RMII Transmit Timing Diagram**

### 6.2.7 RMII Receive Timing

| Parameter | Symbol | 100 Mbps | | | Units | Notes |
|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | | |
| Mn_RXD[1:0] setup time (Mn_REFCLK rising edge) | $t_{DS}$ | 4 | - | - | ns | |
| Mn_RXD[1:0] hold time (Mn_REFCLK rising edge) | $t_{DH}$ | 2 | - | - | ns | |
| Mn_RXDV input setup time (Mn_REFCLK rising edge) | $t_{DVS}$ | 4 | - | - | ns | Equivalent to CRS_DV signal in RMII spec. |
| Mn_RXDV input hold time (Mn_REFCLK rising edge) | $t_{DVH}$ | 2 | - | - | ns | |
| Mn_RXER input setup time (Mn_REFCLK edge) | $t_{ERS}$ | 4 | - | - | ns | |
| Mn_RXER input hold time (Mn_REFCLK rising edge) | $t_{ERH}$ | 2 | - | - | ns | |

**Table 21 - RMII Receive Timing - 100 Mbps**

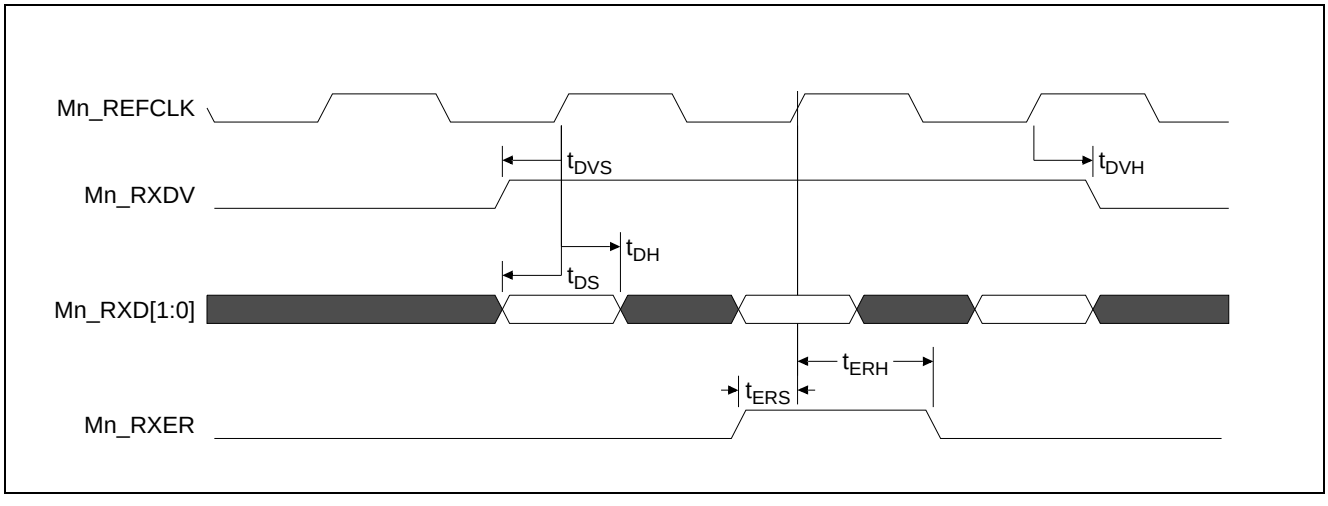



**Figure 52 - RMII Receive Timing Diagram**

### 6.2.8 GMII Transmit Timing

| Parameter | Symbol | 1000 Mbps | | | Units | Notes |
|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | | |
| Mn_GTXCLK period | $t_{GCC}$ | 7.5 | 8 | - | ns | |
| Mn_GTXCLK high time | $t_{GCH}$ | 2.5 | - | - | ns | |
| Mn_GTXCLK low time | $t_{GCL}$ | 2.5 | - | - | ns | |
| Mn_GTXCLK rise time | $t_{GCR}$ | - | - | 1 | ns | |
| Mn_GTXCLK fall time | $t_{GCF}$ | - | - | 1 | ns | |
| Mn_GTXCLK rise to Mn_TXD[7:0] active delay | $t_{DV}$ | 1 | - | 5 | ns | Load = 25 pF |
| Mn_GTXCLK rise to Mn_TXEN active delay | $t_{EV}$ | 1 | - | 5 | ns | Load = 25 pF |
| Mn_GTXCLK rise to Mn_TXER active delay | $t_{ER}$ | 1 | - | 5 | ns | Load = 25 pF |

**Table 22 - GMII Transmit Timing - 1000 Mbps**

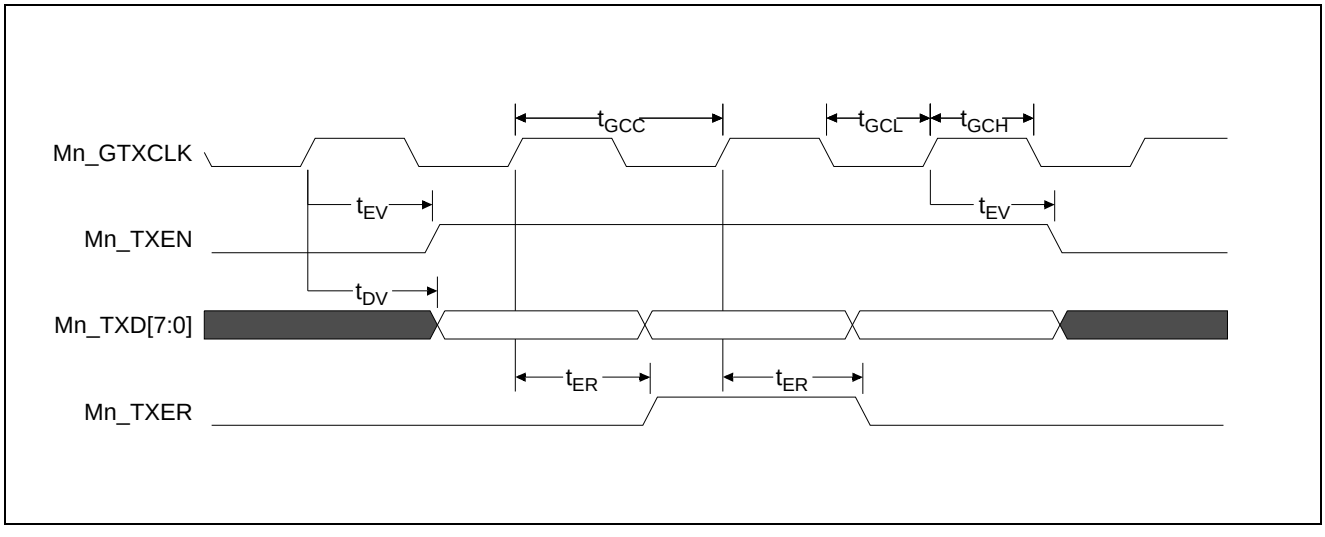



**Figure 53 - GMII Transmit Timing Diagram**

### 6.2.9 GMII Receive Timing

| Parameter | Symbol | 1000 Mbps | | | Units | Notes |
|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | | |
| Mn_RXCLK period | $t_{RCC}$ | 7.5 | 8 | - | ns | |
| Mn_RXCLK high wide time | $t_{RCH}$ | 2.5 | - | - | ns | |
| Mn_RXCLK low wide time | $t_{RCL}$ | 2.5 | - | - | ns | |
| Mn_RXCLK rise time | $t_{RCR}$ | - | - | 1 | ns | |
| Mn_RXCLK fall time | $t_{RCF}$ | - | - | 1 | ns | |
| Mn_RXD[7:0] setup time (Mn_RXCLK rising edge) | $t_{DS}$ | 2 | - | - | ns | |
| Mn_RXD[7:0] hold time (Mn_RXCLK rising edge) | $t_{DH}$ | 0 | - | - | ns | |
| Mn_RXDV setup time (Mn_RXCLK rising edge) | $t_{DVS}$ | 2 | - | - | ns | |
| Mn_RXDV hold time (Mn_RXCLK rising edge) | $t_{DVH}$ | 0 | - | - | ns | |
| Mn_RXER setup time (Mn_RXCLK rising edge) | $t_{ERS}$ | 2 | - | - | ns | |
| Mn_RXER hold time (Mn_RXCLK rising edge) | $t_{ERH}$ | 0 | - | - | ns | |

**Table 23 - GMII Receive Timing - 1000 Mbps**

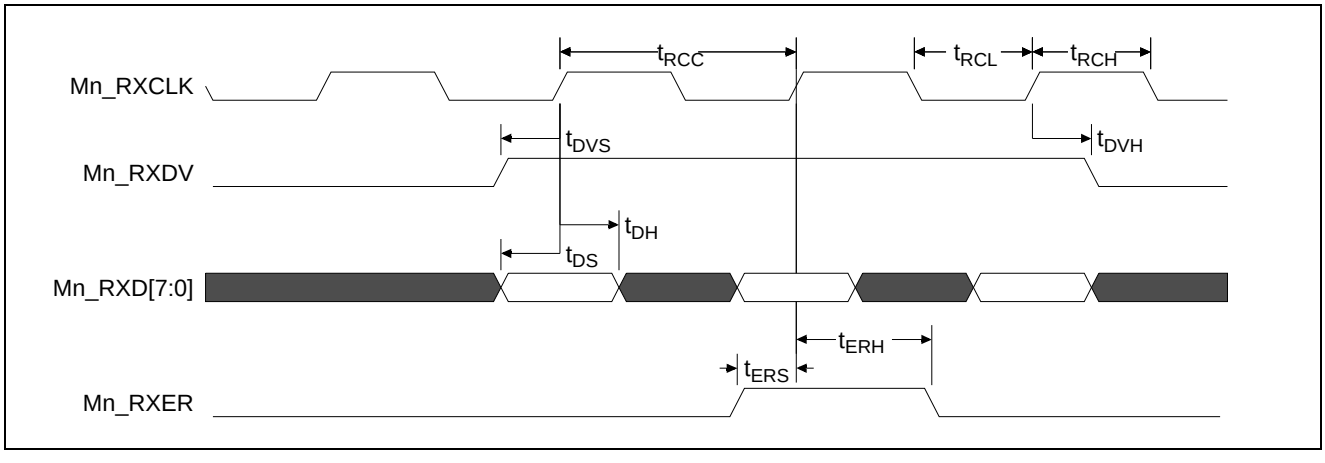



**Figure 54 - GMII Receive Timing Diagram**

### 6.2.10 TBI Interface Timing

| Parameter | Symbol | 1000 Mbps | | | Units | Notes |
|---|---|---|---|---|---|---|
| | | Min. | Typ. | Max. | | |
| Mn_GTXCLK period | $t_{GCC}$ | 7.5 | 8 | - | ns | |
| Mn_GTXCLK duty cycle | | 40 | - | 60 | % | |
| Mn_GTXCLK rise time | $t_{GCR}$ | - | - | 1 | ns | |
| Mn_GTXCLK fall time | $t_{GCF}$ | - | - | 1 | ns | |
| Mn_TXD[9:0] Output Delay (Mn_GTXCLK rising edge) | $t_{DV}$ | 1 | - | 5.5 | ns | |
| Mn_RBC0/Mn_RBC1 period | $t_{RC}$ | - | 16 | - | ns | |
| Mn_RBC0/Mn_RBC1 duty cycle | | 40 | - | 60 | % | |
| Mn_RBC0/Mn_RBC1 rise time | $t_{RR}$ | - | - | 2 | ns | |
| Mn_RBC0/Mn_RBC1 fall time | $t_{RF}$ | - | - | 2 | ns | |
| Mn_RXD[9:0] setup time (Mn_RBC0 rising edge) | $t_{DS}$ | 2 | - | - | ns | |
| Mn_RXD[9:0] hold time (Mn_RBC0 rising edge) | $t_{DH}$ | 1 | - | - | ns | |
| Mn_REFCLK period | $t_{FC}$ | - | 8 | - | ns | |
| Mn_REFCLK duty cycle | | 40 | - | 60 | % | |

**Table 24 - TBI Timing - 1000 Mbps**

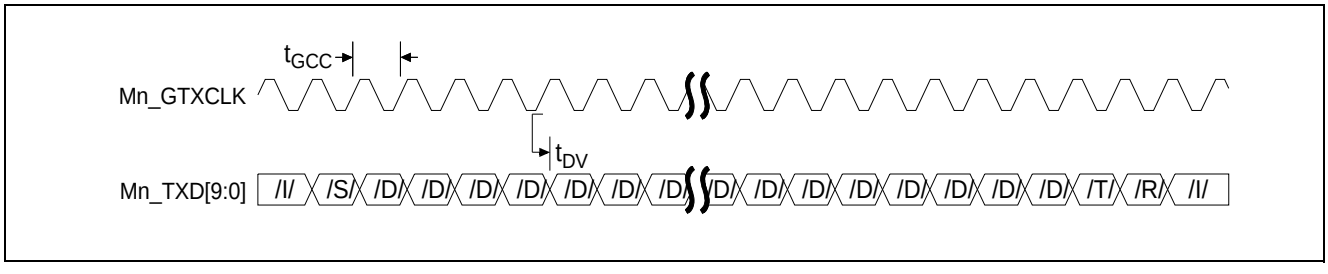



**Figure 55 - TBI Transmit Timing Diagram**

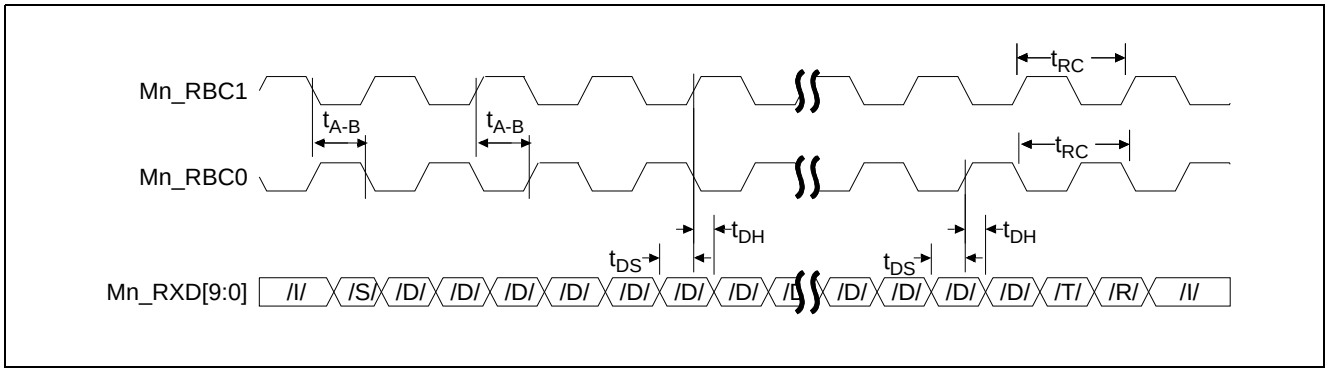



**Figure 56 - TBI Receive Timing Diagram**

### 6.2.11 Management Interface Timing

The management interface is common for all inputs and consists of a serial data I/O line (Mn_MDIO) and a clock line (Mn_MDC). When the ZL30310 acts as the Station Management entity (STA), Mn_MDC is an output. When the ZL30310 acts as a PHY, Mn_MDC is an input.

| Parameter | Symbol | Min. | Typ. | Max. | Units | Notes |
|---|---|---|---|---|---|---|
| Mn_MDC Clock Output period | $t_{MP}$ | 1900 | 2000 | - | ns | Note 1 |
| Mn_MDC high | $t_{MHI}$ | 900 | 1000 | - | ns | |
| Mn_MDC low | $t_{MLO}$ | 900 | 1000 | - | ns | |
| Mn_MDC rise time | $t_{MR}$ | - | - | 10 | ns | |
| Mn_MDC fall time | $t_{MF}$ | - | - | 10 | ns | |
| Mn_MDIO setup time (rising edge Mn_MDC output) | $t_{MS}$ | 10 | - | - | ns | |
| Mn_MDIO hold time (rising edge Mn_MDC output) | $t_{MH}$ | 0 | - | - | ns | |
| Mn_MDIO Output Delay (falling edge Mn_MDC output) | $t_{MD}$ | 10 | - | 400 | ns | |
| Mn_MDIO setup time (rising edge Mn_MDC input) | $t_{MS}$ | 10 | - | - | ns | Note 2 |
| Mn_MDIO hold time (rising edge Mn_MDC input) | $t_{MH}$ | 10 | - | - | ns | Note 2 |
| Mn_MDIO Output Delay (falling edge Mn_MDC input) | $t_{MD}$ | 0 | - | 300 | ns | Note 2 |

**Table 25 - MAC Management Timing Specification**

Note 1: Refer to Clause 22 in IEEE802.3 (2005) Standard for input/output signal timing characteristics.

Note 2: Refer to Clause 22.3.4 in IEEE802.3 (2005) Standard for output load description of MDIO.

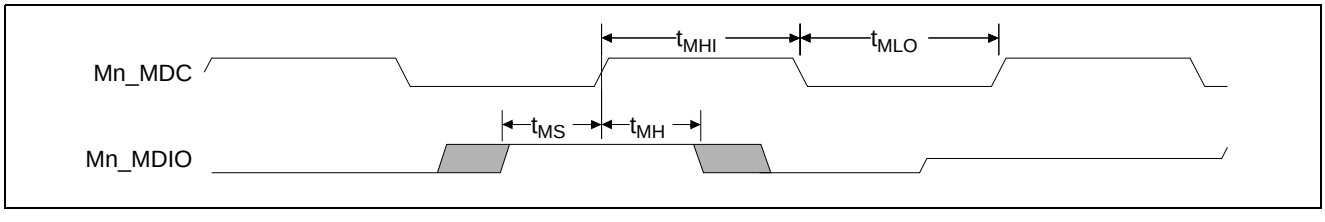



**Figure 57 - Management Interface Timing for Ethernet Port - Read**

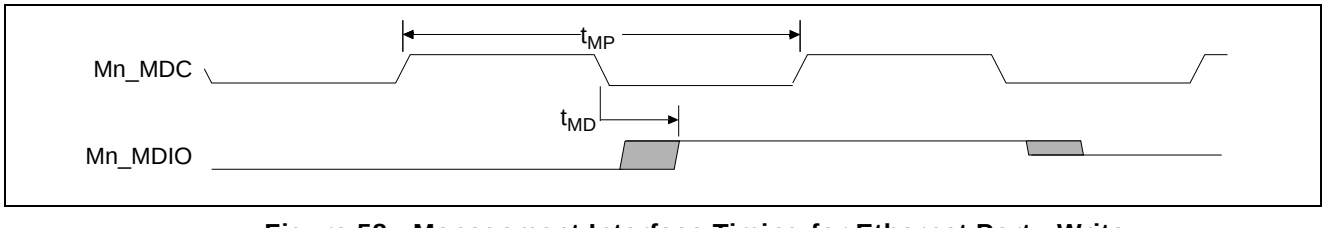



**Figure 58 - Management Interface Timing for Ethernet Port - Write**

### 6.3 Synchronous Serial Interface Timing

| Parameter | Symbol | Min. | Max. | Units |
|---|---|---|---|---|
| SCK period | $t_{SCC}$ | 124 | | ns |
| SCK high time | $t_{SCH}$ | 62 | | ns |
| SCK low time | $t_{SCL}$ | 62 | | ns |
| SI setup time to SCK rising edge | $t_{SDS}$ | 10 | | ns |
| SI hold time from SCK rising edge | $t_{SDH}$ | 10 | | ns |
| SO delay from SCK falling edge | $t_{SD}$ | | 25 | ns |
| CS_B setup time to SCK rising edge (LSB first) or falling edge (MSB first) | $t_{SCS}$ | 20 | | ns |
| CS_B hold time from SCK rising edge (LSB first) or falling edge (MSB first) | $t_{SCH}$ | 10 | | ns |
| CS_B to output high impedance | $t_{OHZ}$ | | 60 | ns |

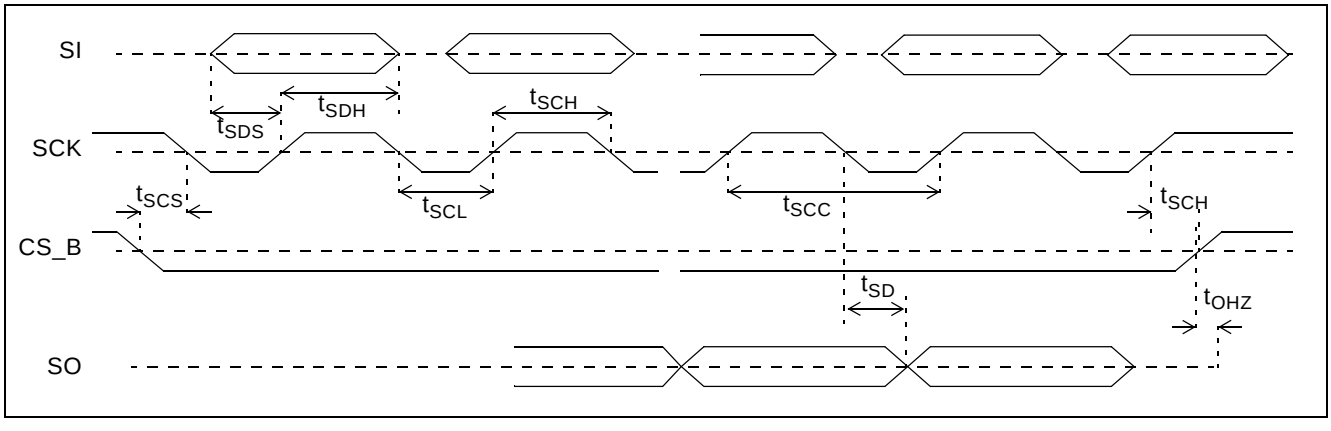



**Figure 59 - Serial Peripheral Interface Timing - LSB First Mode**

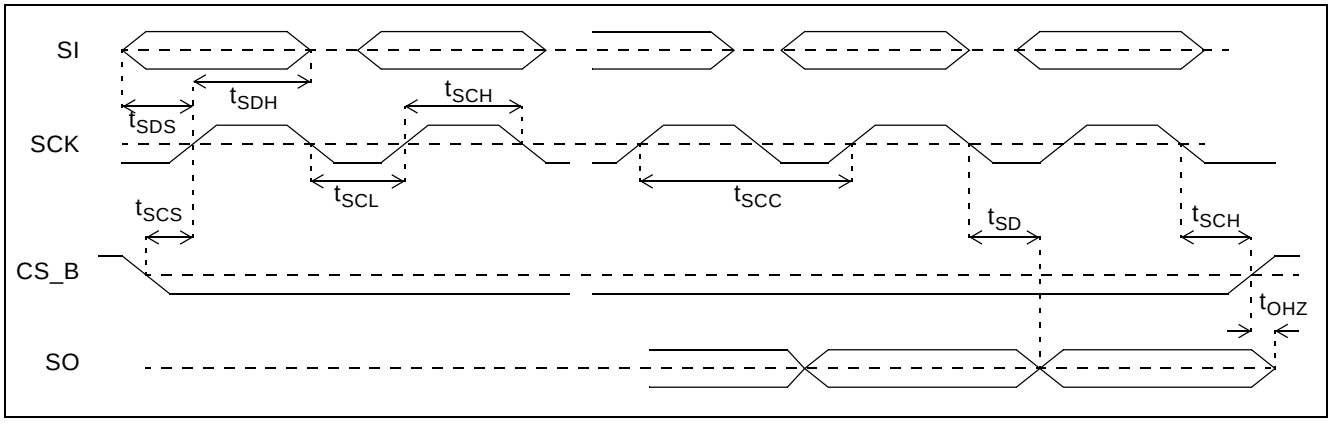



**Figure 60 - Serial Peripheral Interface Timing - MSB First Mode**

## 6.4 JTAG Interface Timing

| Parameter | Symbol | Min. | Typ. | Max. | Units | Notes |
|---|---|---|---|---|---|---|
| TCK period | $t_{JCP}$ | 40 | 100 | | ns | |
| TCK clock pulse width | $t_{JL}$, $t_{JH}$ | 20 | - | - | ns | |
| TCK rise and fall time | $t_{JRF}$ | 0 | - | 3 | ns | |
| $\overline{\text{TRST}}$ setup time to TCK falling edge | $t_{RS}$ | 10 | - | - | ns | Note 1 |
| $\overline{\text{TRST}}$ assert time | $t_{RAS}$ | 10 | - | - | ns | |
| TMS, TDI setup time to TCK rising edge | $t_{TS}$ | 5 | - | - | ns | |
| TMS, TDI hold time to TCK rising edge | $t_{TH}$ | 15 | - | - | ns | |
| TDO delay from TCK falling edge | $t_{TDV}$ | 0 | - | 15 | ns | |
| TDO high impedance from TCK falling edge | $t_{TDZ}$ | 0 | - | 15 | ns | |

**Table 26 - JTAG Interface Timing**

Note 1: $\overline{\text{TRST}}$ is an asynchronous signal. The setup time is for test purposes only.

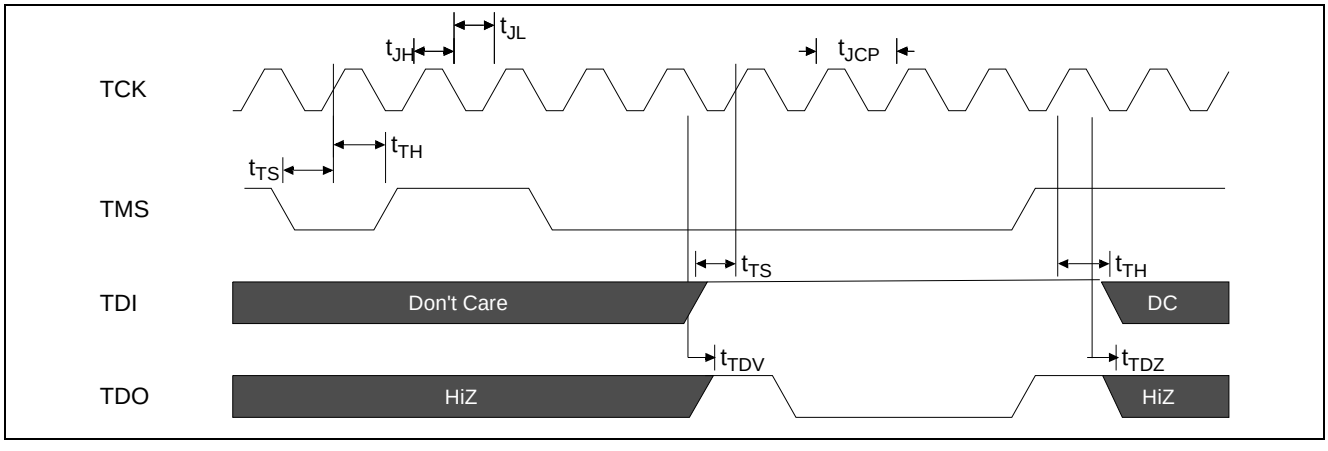



**Figure 61 - JTAG Signal Timing**

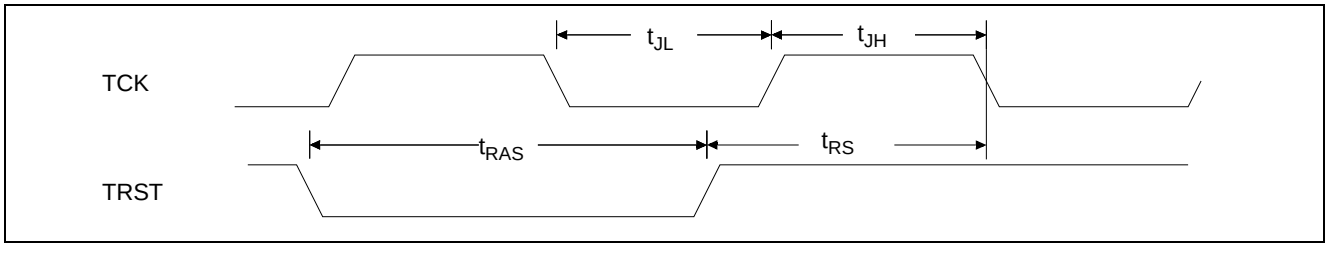
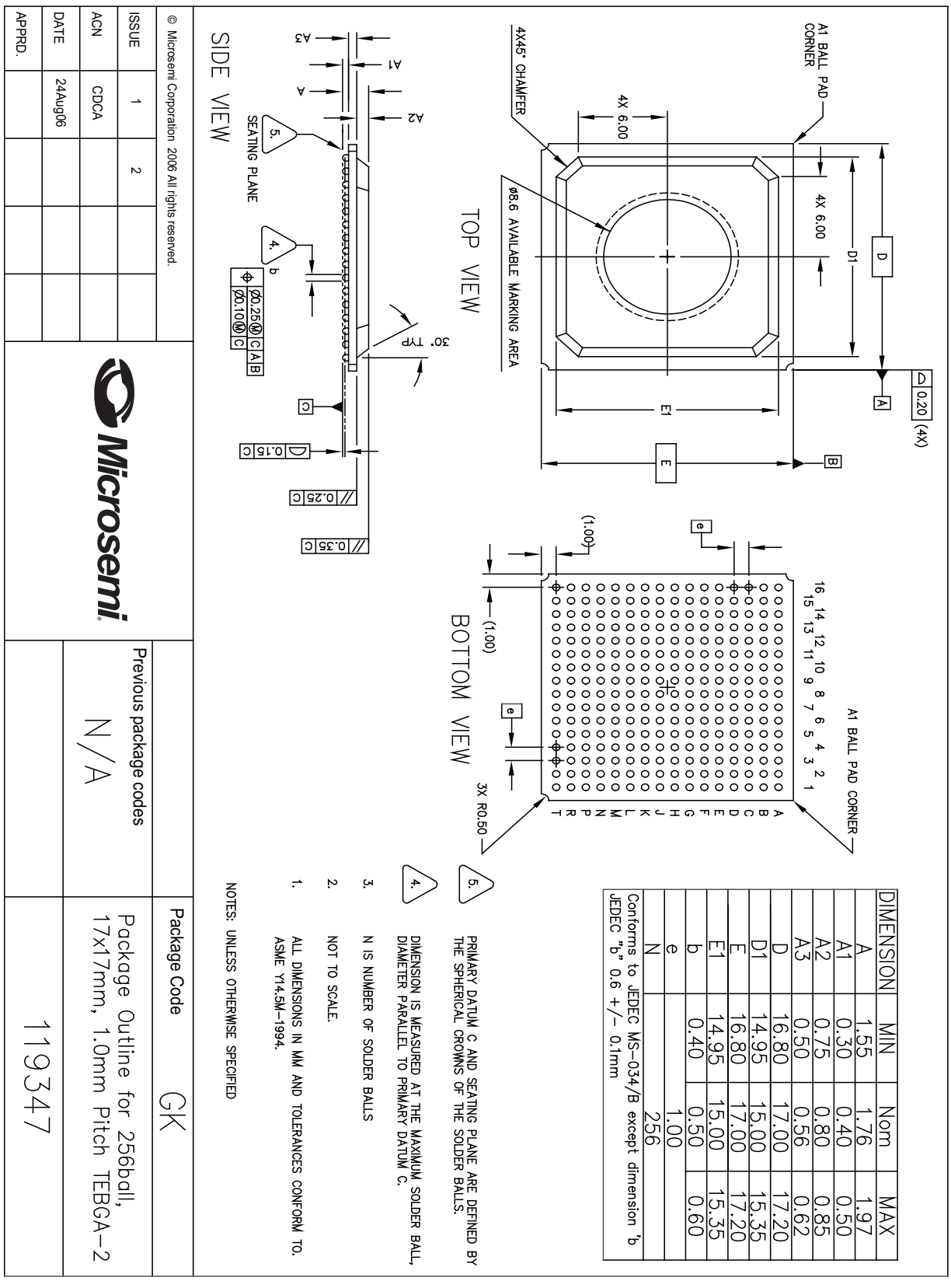



**Figure 62 - JTAG Clock and Reset Timing**

## 7.0 Package Dimensions




| DIMENSION | MIN | Nom | MAX |
|---|---|---|---|
| A | 1.55 | 1.76 | 1.97 |
| A1 | 0.30 | 0.40 | 0.50 |
| A2 | 0.75 | 0.80 | 0.85 |
| A3 | 0.50 | 0.56 | 0.62 |
| D | 16.80 | 17.00 | 17.20 |
| D1 | 14.95 | 15.00 | 15.35 |
| E | 16.80 | 17.00 | 17.20 |
| E1 | 14.95 | 15.00 | 15.35 |
| b | 0.40 | 0.50 | 0.60 |
| e | | 1.00 | |
| N | | 256 | |
| Conforms to JEDEC MS-034/B except dimension "b" JEDEC "b" 0.6 +/- 0.1mm | | | |

5. PRIMARY DATUM C AND SEATING PLANE ARE DEFINED BY THE SPHERICAL CROWNS OF THE SOLDER BALLS.

4. DIMENSION IS MEASURED AT THE MAXIMUM SOLDER BALL DIAMETER PARALLEL TO PRIMARY DATUM C.

3. N IS NUMBER OF SOLDER BALLS

2. NOT TO SCALE.

1. ALL DIMENSIONS IN MM AND TOLERANCES CONFORM TO ASME Y14.5M-1994.

NOTES: UNLESS OTHERWISE SPECIFIED

© Microsemi Corporation 2006 All rights reserved.

| ISSUE | 1 | 2 | | |
|---|---|---|---|---|
| ACN | CDCA | | | |
| DATE | 24Aug06 | | | |
| APPRD. | | | | |

Microsemi

| Previous package codes | Package Code | GK |
|---|---|---|
| N/A | Package Outline for 256ball, 17x17mm, 1.0mm Pitch TEBGA-2 | |
| | 119347 | |

Information relating to products and services furnished herein by Microsemi Corporation or its subsidiaries (collectively “Microsemi”) is believed to be reliable. However, Microsemi assumes no liability for errors that may appear in this publication, or for liability otherwise arising from the application or use of any such information, product or service or for any infringement of patents or other intellectual property rights owned by third parties which may result from such application or use. Neither the supply of such information or purchase of product or service conveys any license, either express or implied, under patents or other intellectual property rights owned by Microsemi or licensed from third parties by Microsemi, whatsoever. Purchasers of products are also hereby notified that the use of product in certain ways or in combination with Microsemi, or non-Microsemi furnished goods or services may infringe patents or other intellectual property rights owned by Microsemi.

This publication is issued to provide information only and (unless agreed by Microsemi in writing) may not be used, applied or reproduced for any purpose nor form part of any order or contract nor to be regarded as a representation relating to the products or services concerned. The products, their specifications, services and other information appearing in this publication are subject to change by Microsemi without notice. No warranty or guarantee express or implied is made regarding the capability, performance or suitability of any product or service. Information concerning possible methods of use is provided as a guide only and does not constitute any guarantee that such methods of use will be satisfactory in a specific piece of equipment. It is the user’s responsibility to fully determine the performance and suitability of any equipment using such information and to ensure that any publication or data used is up to date and has not been superseded. Manufacturing does not necessarily include testing of all functions or parameters. These products are not suitable for use in any medical and other products whose failure to perform may result in significant injury or death to the user. All products and materials are sold and services provided subject to Microsemi’s conditions of sale which are available on request.

**For more information about all Microsemi products visit our website at www.microsemi.com**

TECHNICAL DOCUMENTATION – NOT FOR RESALE




**Microsemi Corporate Headquarters**
One Enterprise, Aliso Viejo CA 92656 USA
Within the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

Microsemi Corporation (NASDAQ: MSCC) offers a comprehensive portfolio of semiconductor solutions for: aerospace, defense and security; enterprise and communications; and industrial and alternative energy markets. Products include high-performance, high-reliability analog and RF devices, mixed signal and RF integrated circuits, customizable SoCs, FPGAs, and complete subsystems. Microsemi is headquartered in Aliso Viejo, Calif. Learn more at **www.microsemi.com**.

© 2012 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.