Data Sheet, DS1, June 2002

GREEN

24+2G Ethernet Switch on Chip

PLB 2224 Version 1.3

Wired
Communications

Infineon technologies

Never stop thinking.

Data Sheet

Revision History:	**2002-06-03**	DS1
Previous Version:	01.01	

Page	**Subjects (major changes since last revision)**
Page 23 **Page 202** **Page 141**	Updated Package & Ball Pin Assignment, Added AC characteristics, Added Register Definitions
Page 106 **Page 218**	Updated IIC interface, Package details
Page 202 **Page 196**	Updated timing information, Register default values

For questions on technology, delivery and prices please contact the Infineon Technologies Offices in Germany or the Infineon Technologies Companies and Representatives worldwide: see our webpage at http://www.infineon.com.

Edition 2002-06-03

Published by Infineon Technologies AG,
St.-Martin-Strasse 53,
D-81541 München, Germany

Attention please!

The information herein is given to describe certain components and shall not be considered as warranted characteristics.

Terms of delivery and rights to technical change reserved.

We hereby disclaim any and all warranties, including but not limited to warranties of non-infringement, regarding circuits, descriptions and charts stated herein.

Infineon Technologies is an approved CECC manufacturer.

Information

For further information on technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies Office in Germany or our Infineon Technologies Representatives worldwide.

Warnings

Due to technical requirements components may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies Office.

Infineon Technologies Components may only be used in life-support devices or systems with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system, or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body, or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

Table of Contents Page

Table of Contents Page

Table of Contents Page

Table of Contents Page

Table of Contents **Page**

List of Figures Page

List of Tables

List of Tables

List of Tables Page

1 Overview

1.1 Introduction

PLB 2224 is a single chip Multiport Ethernet switch, with embedded Frame Buffer & Address Table memory. PLB 2224 is a single-chip solution for building high-port count workgroups and wiring closet Fast Ethernet switches. The architecture allows the device to implement standalone and stackable Fast Ethernet switches, which are low-cost, low part-count and have low power implementations.

PLB 2224 provides a high level of integration, It provides 24 Fast Ethernet and 2 Gigabit Ethernet ports. It is capable of supporting wire-speed, full-duplex packet traffic on all ports under worst-case traffic conditions. PLB 2224 Provides 32-bit PCI or generic CPU interface as well as a 2-wire master/slave serial interface for connecting an external Micro-controller for Managed Switch Implementation.

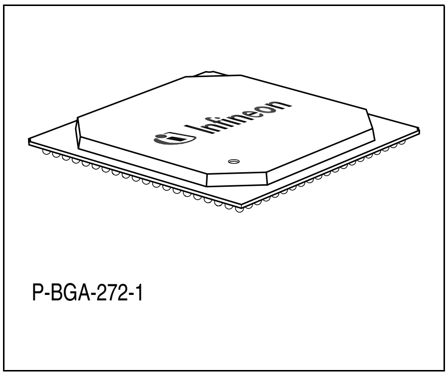
The device is based on Infineon 0.18 µ Embedded DRAM process & integrates all the necessary Memory required to implement Full duplex wire speed switch for 24 Fast ethernet ports with 2 Gigabit uplink ports. Hence the device can be targeted for both high performance ethernet core switches or high density ethernet edge switches. The on chip Frame Buffer is organized as 2 banks of 256-bit wide embedded DRAM, operating at 100 MHz, providing up to 16 Gbit/s memory bandwidth. PLB 2224 can achieve packet forwarding/switching speed of up to 8 Million packets per second. It provides 2 priority queues per port to enable time sensitive data have access to the network with minimal delay. SMII support on 10/100 Mbit/s interfaces & GMII/MII/TBI on Gigabit interface is provided. Matrix & serial interface for port status LEDs is provided. Embedded Packet buffer & address table allows low pin count packaging in a P-BGA-272 pin package. This “system on chip” solution enables system designers to develop cost effective unmanaged/managed/intelligent switches. The core logic operates at 1.8 V & I/Os operate at 3.3 V.

24+2G Ethernet Switch on Chip GREEN

PLB 2224

Version 1.3

1.2 Features


P-BGA-272-1

1.2.1 Ports & Network Interfaces

- High performance Gigabit Fast Ethernet switch controller, wire-speed operation on every port.
- Single chip with twentyfour x 10/100 Mbit/s and two x Gbit/s Ports (10/100/1000 Mbit/s ports)
- SMII Interface on 10/100 Mbit/s ports – supports auto-negotiation for speed and duplex setting.
- GMII and PMA Interface on Gigabit ports (full-duplex only) and optional MII interface support on Gigabit ports, Half duplex is supported in 10/100 Mbit/s mode only.
- Full-duplex IEEE 802.3x flow control and collision-based congestion control in half-duplex
- Provide packet switching functions between 1000/100/10 Mbit/s & fast Ethernet ports
- Support 802.3ad based port trunking for high-bandwidth inter-switch links.
- 2/4/8 10/100 Ports or 2 Gigabit ports can be combined to form trunks – multiple trunks per device

1.2.2 Switch Engine

- Support wire-speed switching with low latency using store-and-forward switching. Fast latency time for both unicast and broadcast. Best broadcast throughput performance
- Address learning and resolution – up to 8 K entries, auto-learning and auto-aging.
- Two priority levels per port for CoS support – WFQ scheduling.
- Supports both port-based and tagged 802.1Q VLAN scheme for up to 1 K Active VLANs
- Port monitoring and forwarding of packets to the CPU port

Type	Package
PLB 2224	P-BGA-272

1.2.3 Management

- Allow network administrators to create Layer 2 MAC-address based filters to constrain users to particular destinations, implement basic Layer 2 security, and simulate a multicast environment using static filters.
- Support extensive traffic/network management through SNMP, four critical groups of RMON (statistics, history, alarms, events) and MIBs.
- Supports IEEE 802.1D Spanning Tree for network loop detection and disabling (particularly useful in larger networks) and for fault-tolerant connectivity supports Secure Mode Traffic filtering on a per port basis
- PCI compliant 32-bit, 33 MHz CPU interface. Can also be used as a generic interface with multiplexed or separate address/data. Glueless Interface to many 32-bit processors.
- 2-wire serial interface for configuration I^2C E2PROM or CPU connection for unmanaged, minimally managed or stackable switches.
- Supports local console and Telnet interface for operation, administration, and maintenance. Simple to use and troubleshoot, with minimum reconfigurations required. New software releases can be downloaded either locally or over the network
- Glue less LED interfaces – four status LEDs per port for displaying buffer utilization at each port (Optional Serial interface for LEDs for customer specific glue logic)

1.2.4 Benefits

- Embedded Packet Buffer & Address Table enables low cost system design.Support for up to 1 K Port/Tag based VLAN (IEEE 802.1Q) with up to 3 MIBs per VLAN, enables application in MDU/MTU markets.

1.3 Logic Symbol

- The vaious interfaces that are available on PLB 2224 is shown in the figure below.

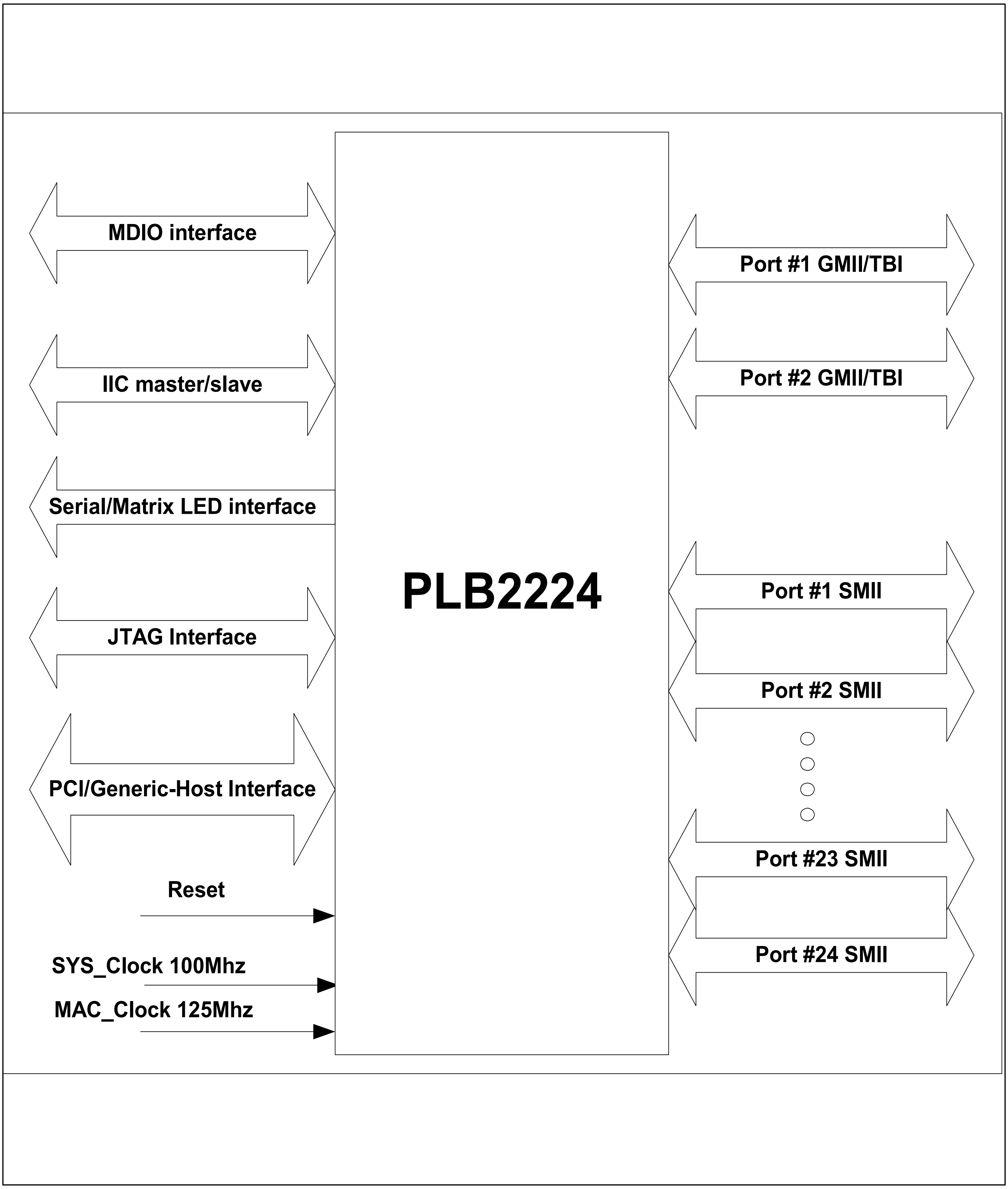


Figure 1 Logic Symbol

1.4 Typical Applications

Figure 2 to Figure 4 gives a general overview of system integration of the PLB 2224

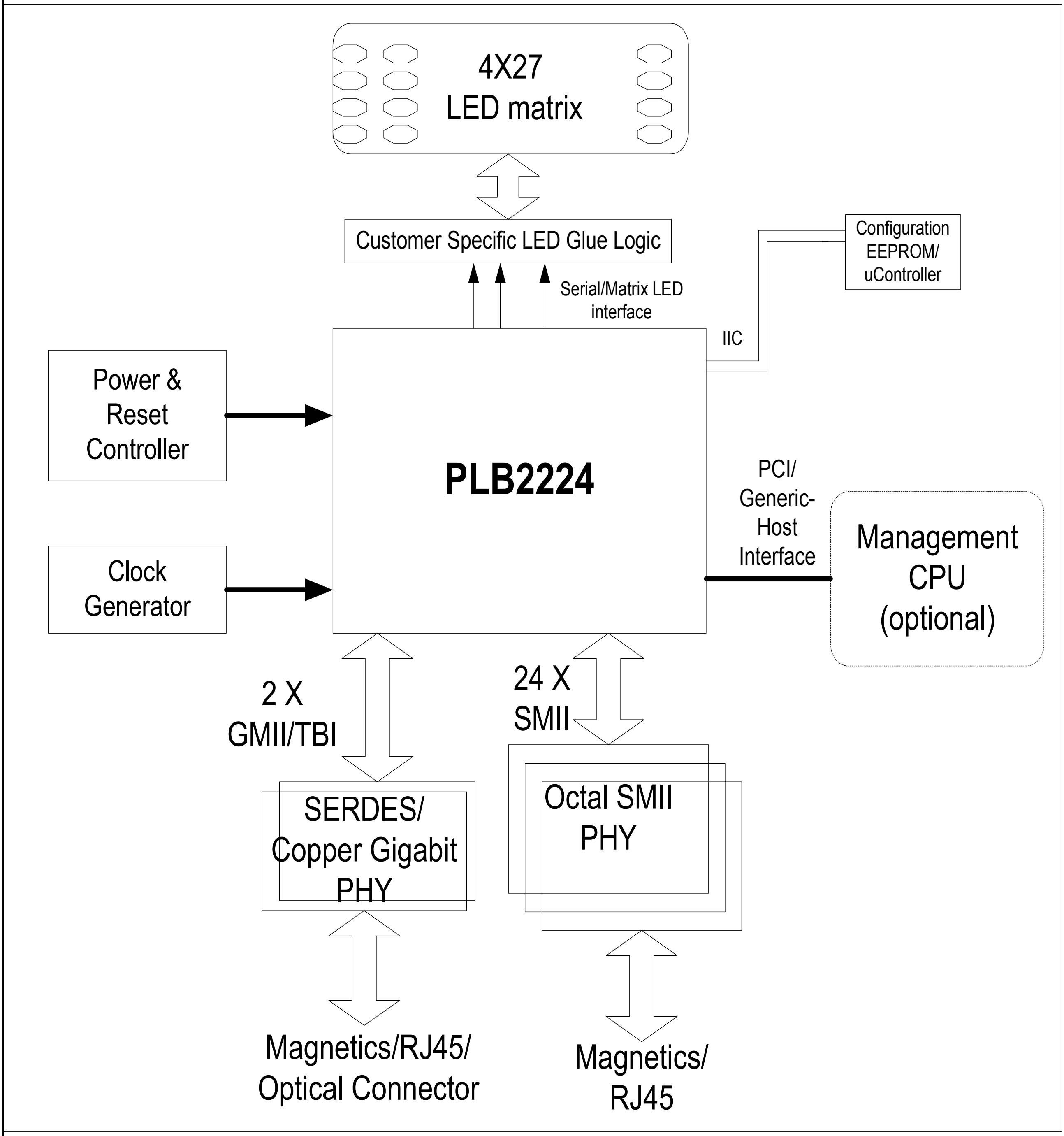

4X27
LED matrix
Customer Specific LED Glue Logic
Serial/Matrix LED interface
Configuration EEPROM/ uController
IIC
Power & Reset Controller
PLB2224
PCI/ Generic-Host Interface
Management CPU (optional)
Clock Generator
2 X GMII/TBI
24 X SMII
SERDES/ Copper Gigabit PHY
Octal SMII PHY
Magnetics/RJ45/ Optical Connector
Magnetics/ RJ45

Figure 2 General System Integration

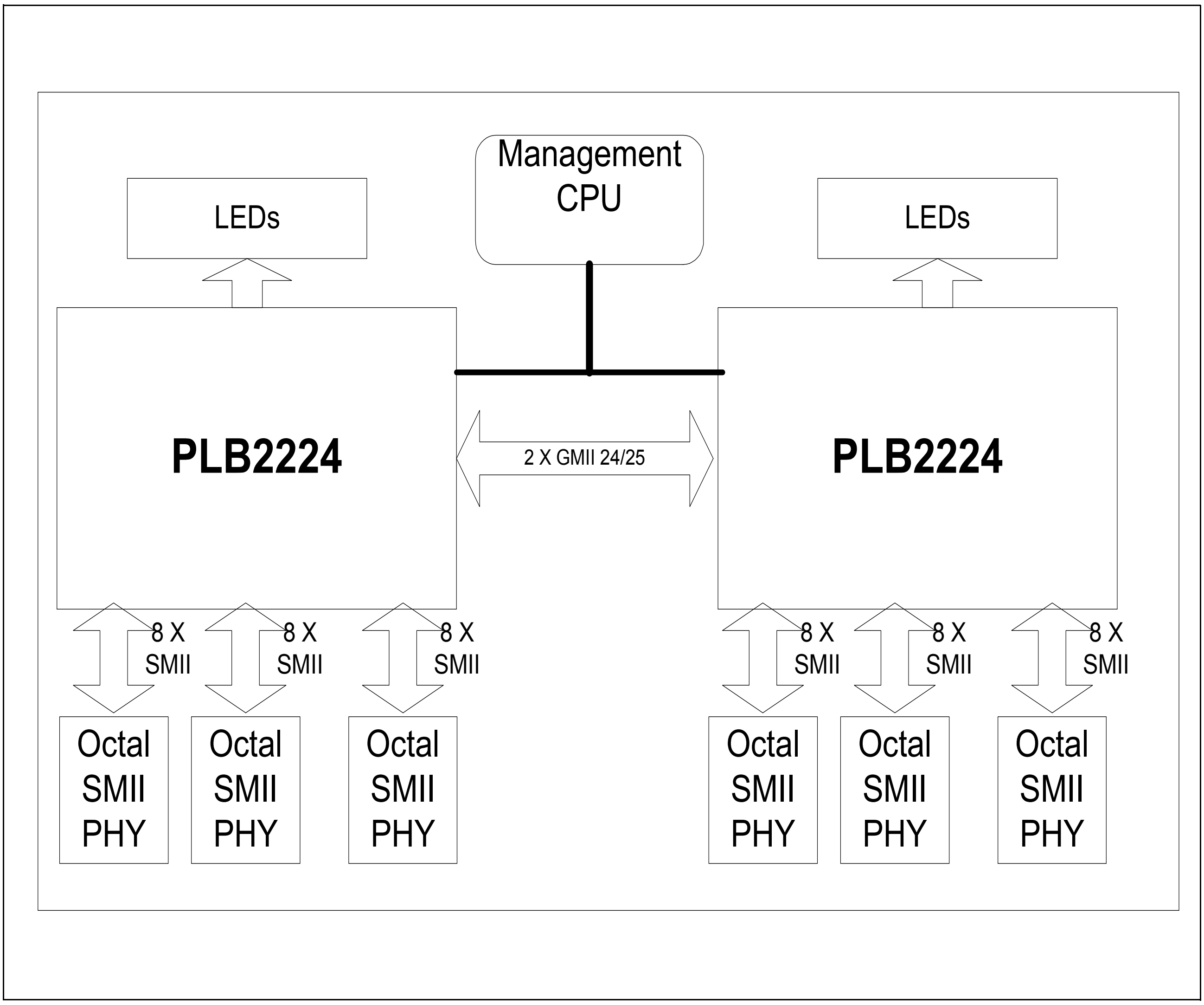


Figure 3 48 x 10/100 Mbit/s Port Managed Solution

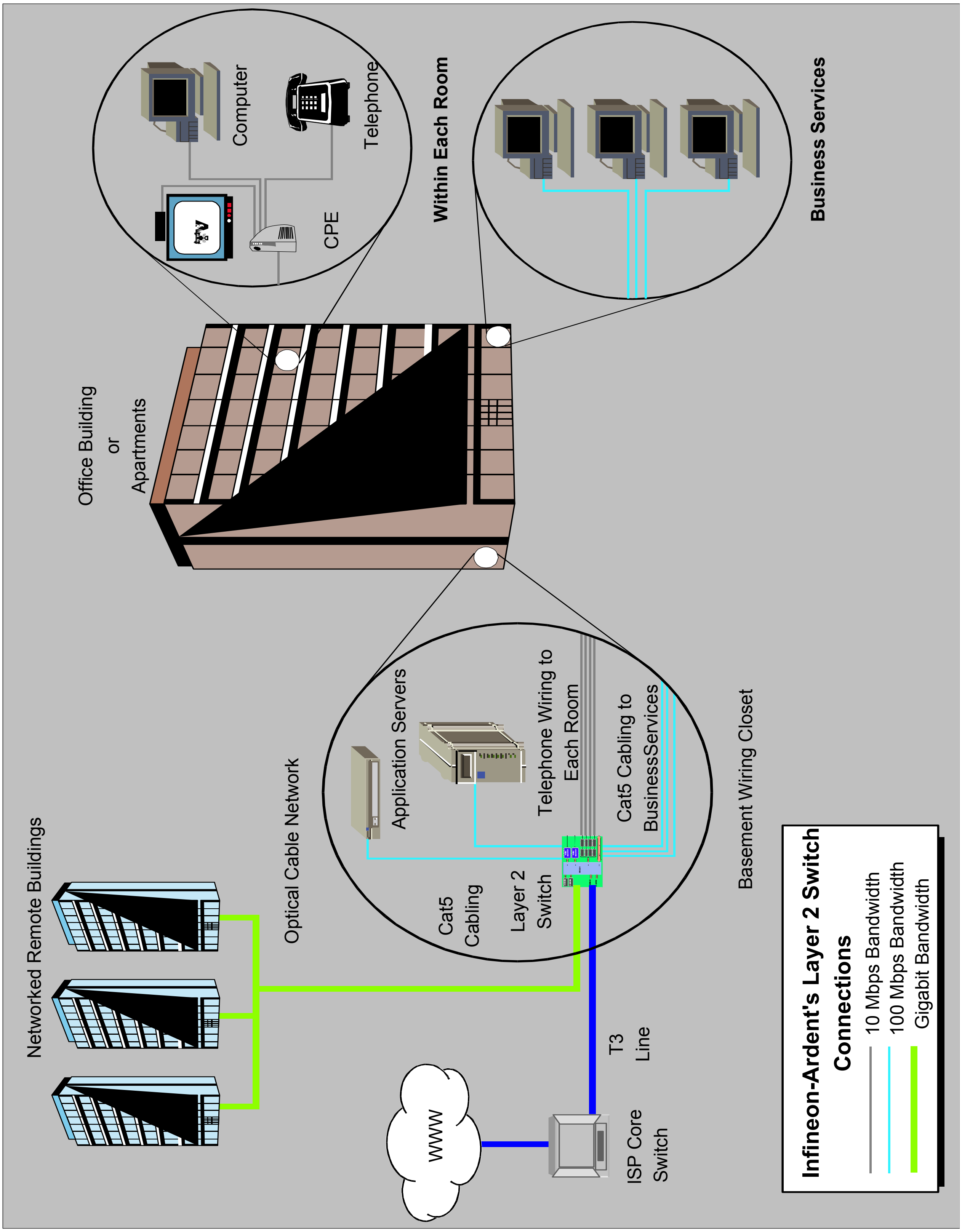


Figure 4 MDU/MTU Application

1.5 Typical Configurations

Switches in a variety of different configurations can be designed using PLB 2224. The table below shows a few configurations that can be built and the major components that are needed to build them.

Table 1 Typical Configurations Using PLB 2224

	PLB 2224	10/100 Octal Phy	GBit Phy	CPU Subsys	Comments
24 10/100 + 2GE fully managed	1	3	2	32-bit Generic or PCI	
24 10/100 + 2GE minimally managed	1	3	2	8-bit with 2 wire Serial Interface	Spanning Tree Protocol only
24 10/100 + 2 GE SOHO with WAN Router	2	3	2	32-bit Generic or PCI	
48 10/100 standalone, managed	2	6	0	32-bit Generic or PCI	
48 10/100 Low-cost unmanaged	2	6	0	None	Uses EEPROM for initial configuration
24 x 10/100 Stackable managed	1	3	2	32-bit Generic with 2-wire Serial interface	GE Ports used to connect stack elements, 2-wire Serial interface is used to communicate between stack elements
32x 10/100 Stackable, Managed	2	6	0	32-bit Generic with 2-wire Serial Interface	8 10/100 ports from each PLB 2224 are used to form two trunks that are used for connecting the stack. 2 wire Serial interface is used to communicate between stack elements

2 Pin Descriptions

The pin diagram for PLB 2224 is shown in **Figure 5**. The package used is P-BGA-272

Pin Diagram

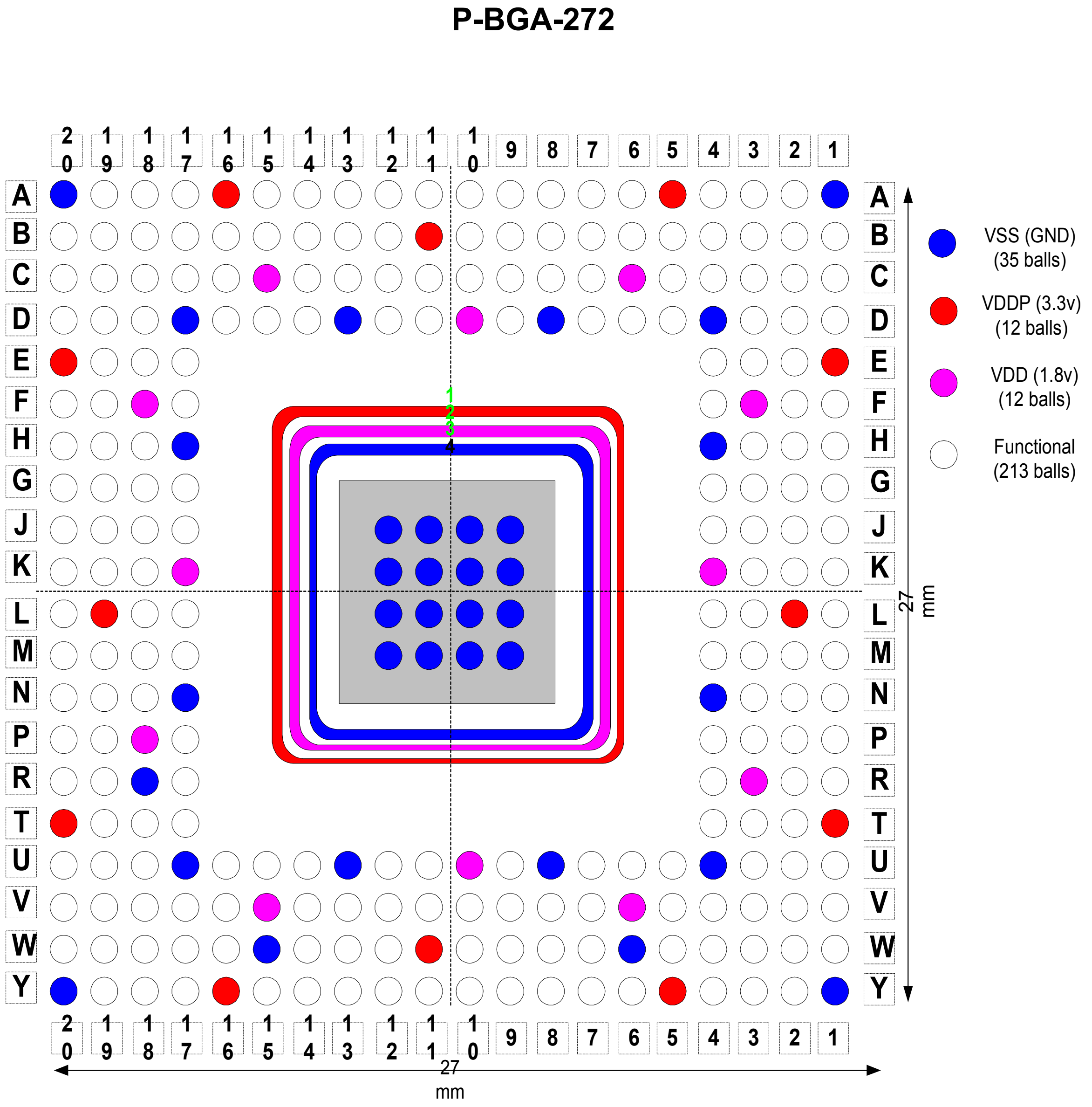


Figure 5 Pin Configuration (Bottom View: Power & Ground Pins)

	1	2	3	4	5	6	7	8	9	10
A	GND	TRST	BIST_ENABLE	FRAME_N	VDDP	PAR	PCI_CLK	CBE_N0	CBE_N3	AD3
B	SCL	TMS	EEPROM_SPEED	MDC	DEVSEL_N	PERR_N	TRDY_N	CBE_N1	AD0	AD4
C	T_RSV1	TDI	RESET_N	MDIO	STOP_N	VDD	IDSEL	CBE_N2	AD1	AD5
D	LED_ROW_N1	T_RSV2	TCK	GND	INTA_N	SERR_N	IRDY_N	GND	AD2	VDD
E	VDDP	NORMAL_MODE	SEL_IIC	T_RSV						
F	LED_COL24	LED_ROW_N0	VDD	SDA						
G	LED_COL23	LED_COL25	LED_ROW_N3	TDO						
H	LED_COL20	LED_COL22	LED_COL26	GND						
J	LED_COL18	LED_COL19	LED_COL21	LED_ROW_N2					GND	GND
K	LED_COL15	LED_COL17	CLK100	VDD					GND	GND

Figure 6 Pin Configuration (Top View: Functional Part 1)

11	12	13	14	15	16	17	18	19	20	
AD6	AD7	AD10	AD14	AD18	VDDP	AD23	AD27	AD31	GND	A
VDDP	AD9	AD13	AD16	AD20	AD22	AD26	AD30	CRS0	R_0_1	B
AD8	AD11	AD15	AD19	VDD	AD25	AD28	TEST_ENABLE	R_0_0	R_0_5	C
AD12	AD17	GND	AD21	AD24	AD29	GND	COLL0	R_0_4	R_0_8	D
						EWRAP	R_0_3	R_0_6	VDDP	E
						R_0_2	VDD	RCLKN0	RCLK0	F
						R_0_7	R_0_9	T_0_0	T_0_2	G
						GND	T_0_1	T_0_3	T_0_6	H
GND	GND					T_0_4	T_0_5	T_0_7	T_0_8	J
GND	GND					VDD	T_0_9	MII_TXCLK0	GTX_CLK0	K

Figure 7 Pin Configuration (Top View: Functional Part 2)

L	LED_COL13	VDDP	LED_COL14	LED_COL16					GND	GND
M	LED_COL12	LED_COL11	LED_COL10	LED_COL9					GND	GND
N	LED_COL8	LED_COL7	LED_COL6	GND						
P	LED_COL5	LED_COL4	LED_COL2	SMII_CLK_7_0						TOP
R	LED_COL3	LED_COL1	VDD	SMII_TX01						
T	VDDP	LED_COL0	SMII_RX01	SMII_RX02						
U	SMII_SYNC_7_0	SMII_RX00	SMII_TX02	GND	SMII_TX04	SMII_TX06	SMII_RX08	GND	SMII_RSYNC_15_8	VDD
V	SMII_TX00	SMII_RCLK_7_0	SMII_RX03	SMII_RX05	SMII_RX07	VDD	SMII_RX09	SMII_TX10	SMII_RCLK_15_8	SMII_RX13
W	SMII_RSYNC_7_0	SMII_TX03	SMII_TX05	SMII_TX07	SMII_CLK_15_8	GND	SMII_TX09	SMII_RX11	SMII_RX12	SMII_TX13
Y	GND	SMII_RX04	SMII_RX06	SMII_SYNC_15_8	VDDP	SMII_TX08	SMII_RX10	SMII_TX11	SMII_TX12	SMII_RX14
	1	2	3	4	5	6	7	8	9	10

Figure 8 Pin Configuration (Top View: Functional Part 3)

GND	GND					R_1_4	R_1_0	VDDP	SDET0	L
GND	GND					CLK125	R_1_5	R_1_2	R_1_1	M
						GND	R_1_8	R_1_6	R_1_3	N
VIEW						T_1_0	VDD	R_1_9	R_1_7	P
						T_1_5	GND	RCLK1	RCLKN1	R
						T_1_8	SDET1	MII_TXCLK1	VDDP	T
SMII_SYNC_23_16	SMII_CLK_23_16	GND	SMII_TX19	SMII_TX20	SMII_TX22	GND	T_1_7	T_1_3	T_1_1	U
SMII_TX14	SMII_RX16	SMII_RX18	SMII_RSYNC_23_16	VDD	SMII_TX21	SMII_TX23	GTX_CLK1	T_1_6	T_1_2	V
VDDP	SMII_TX16	SMII_TX17	SMII_RX19	GND	SMII_RX20	SMII_RX22	CRS1	T_1_9	T_1_4	W
SMII_RX15	SMII_TX15	SMII_RX17	SMII_TX18	SMII_RCLK_23_16	VDDP	SMII_RX21	SMII_RX23	COLL1	GND	Y
11	12	13	14	15	16	17	18	19	20	

Figure 9 Pin Configuration (Top View: Functional Part 3)

2.1 Pin Definitions and Functions

2.1.1 Signal List

Table 2 Pin Descriptions

Symbol	Type/ (default internal state of the pins)	Function	Width
System Clock, Reset & JTAG Control Pins			
CLK100	I	System clock, 100 MHz	1
CLK125	I	Clock input for all MACs (125 MHz)	1
RESET_N	I	Power on reset. Must be asserted longer than 2 cycles of clock to cause reset. (being defined in PCI spec.)	1
TRSTN, TCK, TMS, TDI,	I / (pull down) I / (pull up) I / (pull up) I / (pull up)	These pins are for test purposes including Full-scan, boundary-scan, bist-mode, etc. They function only when 'normal_mode' pin is 1. Otherwise, the pins are only for debugging purposes. The functions of the pins in 'normal_mode' mode are as follows: TRST: Boundary Scan's ENABLE or!RESET. TCK: Boundary Scan's CLOCK. TMS: Boundary Scan's MODE SELECT. TDI: Boundary Scan's DATA IN.	5
TDO	O	This pin is used as Boundary Scan's TDO	1
Bootstrap Pins (Must be initialized for proper operation)			
NORMAL_MODE	I / (pull up)	This bit needs to be 1 for normal operation. It is pulled down only for debugging purposes.	1

Table 2 Pin Descriptions (cont'd)

Symbol	**Type/ (default internal state of the pins)**	**Function**	**Width**
SEL_IIC	I	PCI Interface: *SEL_IIC = 0 / SEL_PCI =1* Generic Interface: *SEL_IIC = 0 / SEL_PCI = 0* Serial IIC: *SEL_IIC = 1 / SEL_PCI = 0*	1
T_RSV (NC)	I	Reserved (= 0)	1
T_RSV1 (SEL_PCI)	I	PCI Interface: SEL_PCI=1 Generic Interface: SEL_PCI = 0	1
T_RSV2 (One_Wait_Cycle / Muxed AD)	I	PCI Interface: Number of TRDY wait cycles Generic Interface: Multiplexed Address/ Data bus.	1
BIST_ENABLE	I	Enable for BIST Mode of operation: To be tied to GND for normal operation.	1
TEST_ENABLE	I	For scan mode: To be tied to '0' for normal operation.	1
EPROM_SPEED	I	1: Data Rate 3.3 MHz 0: Data rate = 100 kHz (This is valid only when led_row_n[2] is pin strapped to 1)	1
SMII: Fast Ethernet Interface			
SMII_SYNC_7_0	O, (15 mA)	SMII sync output for Ports 0 to 7	1
SMII_SYNC_15_8	O, (15 mA)	SMII sync output for Ports 8 to 15	1
SMII_SYNC_23_16	O, (15 mA)	SMII sync output for Ports 16 to 23	1
SMII_RSYNC_7_0	I	SMII sync input for Ports 0 to 7	1
SMII_RSYNC_15_8	I	SMII sync input for Ports 8 to 15	1
SMII_RSYNC_23_16	I	SMII sync output for Ports 16 to 23	1
SMII_CLK_7_0	O, (15 mA)	SMII clk output for Ports 0 to 7	1
SMII_CLK_15_8	O, (15 mA)	SMII clk output for Ports 8 to 15	1
SMII_CLK_23_16	O, (15 mA)	SMII clk output for Ports 16 to 23	1

Table 2 Pin Descriptions (cont'd)

Symbol	Type/ (default internal state of the pins)	Function	Width
SMII_RCLK_7_0	I	SMII clk input for Ports 7 to 0	1
SMII_RCLK_15_8	I	SMII clk input for Ports 15 to 8	1
SMII_RCLK_23_16	I	SMII clk input for Ports 16 to 23	1
SMII_RX [23:0]	I (pull down)	Fast Ethernet receive data (If some of the ports are not used they can be left open, pulldown value 15 K)	24
SMII_TX [23:0]	O, (15 mA)	Fast Ethernet transmit data	24
TBI/GMII: Gigabit Ethernet Interface			
EWRAP	O, (15 mA)	Gb PHY to loopback internally if 1.	1
RCLK[1:0], RCLKN[1:0]	I (pull up)	Gb receive clocks	4
MII_TXCLK[1:0]	I (pull up)	mii_tx clock inputs when MII Interface is used	2
T_(1-0)_[9:0]	O, (15 mA)	Gb Transmit character for the 2 Gport	20
GTX_CLK[1:0]	O, (15 mA)	Gb PHY'S 125 MHz Gb's tx_clk	2
SDET[1:0]	I (pull down)	Gb light detection. Internal pulldown value is 15 K	2
R_(1-0)_[9:0]	I (pull down)	Gb Receive character. Internal pulldown value is 15 K	20
CRS[1:0]	I	Carrier Sense, used in MII mode for Half-Duplex operation	2
COLL[1:0]	I	Collision Detect, used in MII mode for Half-Duplex operation	2
PHY Management and LED Interface			
MDC	O, (15 mA)	PHY Management Clock	1
MDIO	B, (15 mA)	PHY Management data	1
LED_ROW_N[3:0]	B, (5 mA)	LED Row Output. Active low. (Internal pullup of 20 K and a value of 1 K can be used for pulling down this line for pinstrap inputs)	4

Table 2 **Pin Descriptions** (cont'd)

Symbol	**Type/ (default internal state of the pins)**	**Function**	**Width**
LED_COL[26:0]	B, (5 mA)	LED Column Output. Active High (Internal pullup of 20 K and a value of 1 K can be used for pulling down this line for pinstrap inputs)	27
CPU Interface			
PCI_CLK	I	PCI CLK	1
AD<31:0>	B, (6 mA)	*PCI Interface*: Address Data Bus *Generic Interface*: AD Bus	32
CBE_N<3>	B, (6 mA)	*PCI Interface*: Command Byte Enable *Generic Interface*: ALE or address Strobe	1
CBE_N<2>	B, (6 mA)	*PCI Interface*: Command Byte Enable *Generic Interface*: Data Strobe	1
CBE_N<1>	B, (6 mA)	*PCI Interface*: Command Byte Enable *Generic Interface*: Data Strobe or Read	1
CBE_N<0>	B, (6 mA)	*PCI Interface*: Command Byte Enable *Generic Interface*: Read/Write	1
FRAME_N	B, (6 mA)	*PCI Interface*: Framing Signal *Generic Interface*: Chip Select *Serial Interface*: Master Mode	1
TRDY_N	O, (6 mA)	*PCI Interface*: Target Ready *Generic Interface*: Ready	1
IDSEL	I	*PCI Interface*: Initialization Device Select *Generic Interface*: Address Bit 8	1
IRDY_N	I	*PCI Interface*: Initiator ready signal *Generic Interface*: Address Bit 2 *Serial*: Device ID bit 0	1
PAR	B, (6 mA)	*PCI Interface*: Parity Check bit *Generic Interface*: Address Bit 3 *Serial*: Device ID bit 0	1

Table 2 Pin Descriptions (cont'd)

Symbol	**Type/ (default internal state of the pins)**	**Function**	**Width**
PERR_N	B, (6 mA)	*PCI Interface*: Parity Error Signal *Generic Interface*: Address Bit 4 *Serial*: Device ID bit 2	1
SERR_N	B, (6 mA)	*PCI Interface*: System Error Signal *Generic Interface*: Address Bit 5 *Serial*: Device ID bit 3	1
DEVSEL_N	B, (6 mA)	*PCI Interface*: Device Select Signal *Generic Interface*: Address Bit 6 *Serial*: Device ID bit 4	1
STOP_N	B, (6 mA)	*PCI Interface*: Bus Stop Signal *Generic Interface*: Address Bit 7 *Serial*: Device ID bit 5	1
INTA_N	O, (6 mA)	*PCI Interface*: Interrupt Signal *Generic Interface*: Interrupt Signal *Serial*: Interrupt Signal	1
SDA	B, (10 mA, IIC pad)	Serial Data	1
SCL	I	Serial Clock	1
Power and Ground Pins			
VSS	Power	Ground	35
VDD	Power	Core Supply (1.8 V)	12
VDDP	Power	I/O Supply (3.3 V)	12

3 Functional Description

The functional block diagram of PLB 2224 is shown in **Figure 10**.

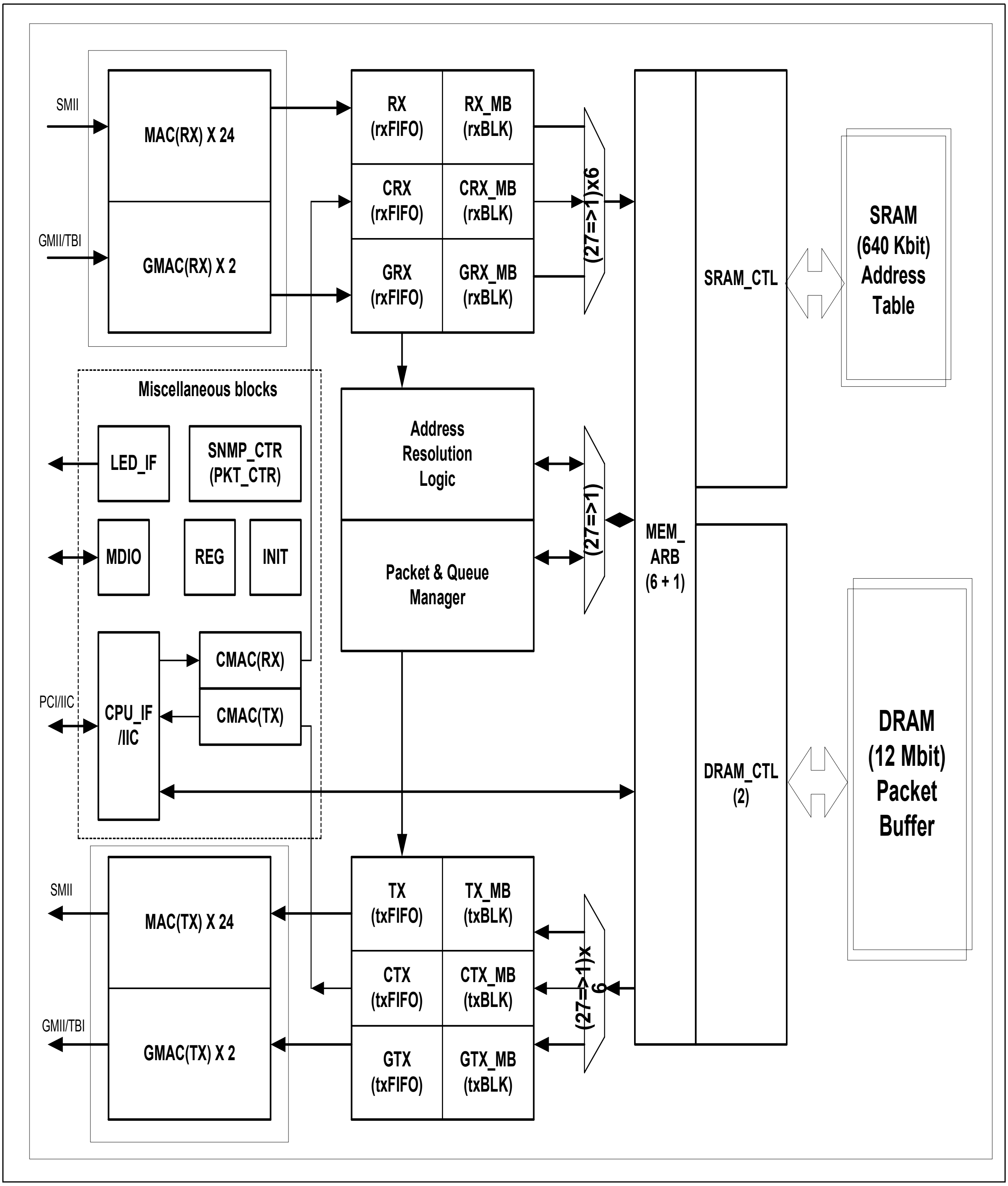


Figure 10 Functional Block Diagram

3.1 Introduction

Figure 10 shows the functional block diagram of PLB 2224. Each of the 24 Serial Media Independent Interface ports consists of a configurable MAC core, which supports back pressure, MII management, programmable backoff & full duplex. Each port has 256-byte RX & 224 byte TX FIFOs & 802.3x flow control.

Two 10/100/1000 interfaces consists ofa configurable GMAC core. The Fast Ethernet interface with the MAC's are via SMII ports only. The 10/100/1000 Gigabit port can interface to external transceivers via the Gigabit Media Independent Interface (GMII) or Ten Bit Interface (TBI). The Gigabit interfaces include a Physical Coding Sublayer (PCS) block, which can be used when interfacing to fiber via TBI. The PCS block can be bypassed when the GMII is used with copper transceivers.

The Address Resolution Logic & Queue manager support both 802.1p & 802.1Q for traffic class prioritization of multimedia or real time applications & also for increased security, simplified moves/changes & fault localization. Support for Port based & TAG based VLAN is provided. PLB 2224 can support up to 1022 Tag/port based VLAN, making it ideal for emerging MDU/MTU market.

PLB 2224 supports Two levels of priority per output port. This allows for time sensitive data to have access to the network with minimal latency. The Packet & Queue manager implements a Hardware based Flow control. The Watermarks for the Flow control can be programmed via external CPU. The onchip embedded DRAM is used entirely for Packet buffering.

The Packet buffer is organized as buckets of 1536 bytes. The packet manager controls the utilization & allocation of the packet buffer in a dynamic fashion.

The packet buffer is physically implemented with 2 banks of 256-bit wide DRAM, running at 100 MHz, giving a raw memory bandwidth of up to 16 Gigabit per second.

The IEEE 802.1Q Port & VLAN based tagging is done via the Address Resolution Logic & the TX blocks. They can handle up to 1024 VLANs for broadcast traffic isolation purposes (including the membership corresponding to entries of "0" & "1023").

For Network management purposes RMON groups 1,2,3 & 9 are fully supported using 32-bit wide counters. Additional debugging features are offered by Port Mirroring support. Port Based Traffic can be copied to any designate Port or CPU.

IP multicast is supported via Software for sending Data streams to multiple nodes. The INTERNET GROUP MANAGEMENT PROTOCOL (IGMP) is supported to further reduce the IP multicast streams, by forwarding packets to only those nodes which are requesting them.

Spanning Tree software is supported for eliminating redundant links as well as loops in the network. IP BOOTP & DHCP software is supported for automatic assignment of IP addresses in intelligent system configuration

Server Load balancing & Failover mechanisms are supported by allowing flexible link aggregation based on MAC DA, SA. Stack management is through IIC interface.

With the help of an External CPU, managed & intelligent switch configurations can be implemented with low level software drivers running over industry standard RTOS. An integrated System Development Platform (Wind River Tornado for Managed Switches (TMS)) is provided to enable Network equipment vendor to explore the chip architecture & begin immediate software development.

3.2 Functional Block Description

As shown in the Functional Block diagram in **Figure 10**, the PLB 2224 consists of seven major blocks.

- 10 / 100 / 1000 Mbit/s Media Access Control MAC/GMAC,
- Receive FIFO / Rx. control (RX),
- Address Resolution Logic (ARL),
- Packet & Queue manager (PQC),
- TX (Transmit FIFO),
- CPU (cpu subsystem),
- Memory block (arbitration and storage).

3.2.1 10/100/1000 Mbit/s Media Access Control (GMAC/MAC)

The MAC on each port interfaces to the external PHY through the GMII/TBI in case of Gigabit port & SMII in case of 10/100 Mbit/s ports. It receives packet data from PHY & transmits them to PHY based on IEEE 802.3 specification. The Port stays disabled until auto-negotiation & memory initialisation is done. Link, Speed & Duplex modes are polled via the MDIO interface. The MAC supports full duplex & half duplex modes in 10/100 Mbit/s modes & supports only full duplex mode in Gigabit mode. IEEE 802.3x based flow control is supported via pause frame generation & back pressure.

3.2.1.1 10/100 Ethernet Ports

The 10/100 ports (Fast Ethernet) support 10 or 100 Mbit/s speed, in either full or half duplex transmission mode. The ports interface to the PHY devices through the SMII . SMII is an industry standard serial interface that operates at 125 MHz. Using the MDIO interface, each port is capable of carrying out auto-negotiation with the PHY device for link speed and duplex mode. The speed and duplex mode of the port can also be set using the CPU. The CPU can alter the speed and duplex settings by overwriting configuration registers in the PHY devices using the MDIO bus.

The PLB 2224 provides three sync and three clock signals, allowing easy interface with octal SMII PHY devices. Eight SMII ports share a sync signal (SMII_SYNC) and clock signal (SMII_CLK) and also support Source Synchronous SMII where the interface can also take in a clock (SMII_RCLK).

3.2.1.2 Gigabit Ethernet Ports

There are three possible modes of operation for the two gigabit ports on the PLB 2224.

- *Connecting to external gigabit PHY devices*

The two gigabit Ethernet ports connect to PHY devices using either the GMII or TBI . The GMII is used with PHY devices that provide the 8B/10B physical sublayer (PCS) encoding – for example Broadcom’s BCM5400 and Level One’s LXT1000, while the TBI is used with the PHY devices (called serializer/deserializer or SERDES devices) that do not contain the PCS functionality – for example Vitesse's VSC7123 and HP's HDMP1636. For each gigabit port, the corresponding configuration bit sel_Gb is set to 1 to choose 1 Gbit/s speed, and bit *TBI / GMII* bit is set to 1 for TBI and the GMII bit is set to select between GMII mode of operation or MII mode of operation. See Configuration Register. The gigabit ports are full duplex only for 1000Mbit/s and can be used for Half Duplex in 10/100 Mbit/s mode.

- *Connecting to external 100 Mbit/s PHY devices*

The gigabit Ethernet ports can also connect to 100 Mbit/s PHY devices supporting the MII . For example, the two gigabit ports can be used for 100FX uplinks. For each gigabit port, the corresponding configuration bit sel1000 bit is set to 0 to choose 100 Mbit/s speed with MII in the **Table 109 "G_Mode Register" on Page 193**. The gigabit ports when used as 100 Mbit/s ports can operate in full duplex mode or half duplex.

- *Connecting PLB 2224s back-to-back*

The GMII on the PLB 2224 can also be used to connect two PLB 2224 devices on the same board, effectively acting as a high capacity trunk between them. Connecting two PLB 2224s back-to-back in this fashion makes it possible to design a fixed configuration or stackable 48-port 10/100 switch that is rack mountable. *Note that this configuration does not require gigabit PHY devices*.

3.2.1.3 MDIO Interface

- The MDIO Serial Interface defined in the MII specification is used for communicating with the PHY device(s). Using this interface, PLB 2224 can access the internal registers of the PHY devices.
- Auto-Negotiation
- PLB 2224 supports auto-negotiation on the 10/100 ports. Autonegotaiation on the Gports is supported when the GMII mode of operation is selected. Using the MDIO interface the ports can negotiate the link parameters - speed, duplex mode and pause_enable (i.e. full-duplex control) and pause_augment (for gigbit speed only). This allows the switch to neogotiate the status and capabilities of the PHY device and configure itself appropriately.
- For SMII, the link information can be obtained by reading the status information continuously coming over the SMII or by accessing the PHY registers using the MDIO interface. See *use_mdio_mode* bit in **"Chip Configuration Register" on Page 141** section. For the gigabit ports, the auto-negotiation is performed using the MDIO interface when the GMII or MII option is selected.
- Using the MDIO interface, the CPU can access the PHY device registers and override the auto-negotiated settings if desired. e_hw_mode bit in **"Switch Configuration Register" on Page 145** must be cleared to 0 in order to disable auto-negotiation.

For the MDIO interface, the Ethernet (PHY) port addresses are mapped into MDIO address space as follows:

- MDIO addresses #4 and #5 Gigabit Ethernet Ports #0 and #1
- MDIO addresses #8 through #31 <-> 10/100 Ethernet Ports #0 through #23

3.2.2 Receive Module (RX)

The RX Interfaces to the GMAC/MAC, handles all signals in both Full Duplex (FD) and Half Duplex (HD) modes of operation. It also determines whether packet is SNAP encapsulated/VLAN tagged. If VLAN tagged, VLAN ID and priority is communicated to ARL. The RX block also forms Packet Header (PBH) information. PBH is written to the eDRAM along with the packet data. Data is written to the eDRAM in 32-byte bursts. Each 1536-byte segment can be referenced using a Packet Numbers (pbnum), which are obtained from Packet & Queue Manager (PQC). It Discards packets less than 64 byte in length. It also consist of MIB counters to support RMON Ethernet statistics group. Interfacing with ARL/eDRAM is implemented via synchronous handshaking.

It makes decision on dropping frames with errors (CRC error/FIFO overrun/Collision Fragment etc.), PAUSE frames and non-tagged frames when so programmed. It also supports port based mirroring. Support for up to 1022 VLANs /Multicast groups (802.1Q/ 802.1D) is provided.

MAC Rx and Port Rx

The MAC Rx block implements the media access functionality for the 10/100 ports as per the IEEE 802.3 standard. It strips the preamble of the incoming Ethernet packet, deserializes the data, checks for legal packet length and correct CRC. Any packets longer than the maximum allowed length of 1536 byte are truncated. If a CRC error is found, the crc_err bit in the Packet Buffer Header (PBH) is set.

Several configuration bits control the behavior of the MAC Rx block. These configuration options are described in the section Port Configuration, Status and Event Registers.

When a port is in full duplex mode with flow control enabled, the MAC Rx block processes the PAUSE frames it receives and communicates the flow control status to the MAC Tx block. The MAC Rx block also requests the MAC Tx block to send PAUSE frames if necessary. A PAUSE frame received at MAC Rx is not forwarded to the destination address unless *rx_pause_frame* bit Switch Configuration Register is set to 1.

The Port Rx block receives packet data from the MAC Rx and segments it into an 8-byte chunk followed by a number of 32-byte chunks. The 8-byte chunk is stored in the SRAM while the remaining packet data (i.e., all the 32-byte chunks including the very last chunk, which could be 32-byte or less) is stored in the EDRAM. In addition, the Port Rx blocks extracts the 6-byte SA and 6-byte DA from the packet and sends them to the ARL block in the Switch Controller. The SA is looked up for learning and update. The DA is looked up for determining the destination port(s) for the packet.

The Port Rx block creates a packet header (PBH) to record packet related information such as the packet length, CRC status, source port id (src_pid) and the SDRAM number in which the first 32-byte chunk of packet data is stored.

The format of the PBH (Packet Buffer Header), is described below.

Table 3 PBH Header Format

32 21	20	15			10
filter_match[9:0]	src_pid[5:0]	start_m em[2:0]	crc_gen	crc_err	pkt_len[10:0]

Table 4 Packet Buffer Header

Bit Fields	Name	Description
31:22	filter_match[9:0]	Matching results for user defined filters.
21:16	src_pid	Port id of the source port.
15:13	start_mem[2:0]	EDRAM number used as the starting logical EDRAM location for packet storage.
12	crc_gen	Indicates that the packet is from the CPU port and MAC Tx needs to generate and append the CRC before transmitting the packet. For packets received from non-CPU ports, this bit should be 0.
11	crc_err	Indicates that packet was received with a CRC error.
10:0	pkt_len[11:0]	Packet length. Maximum allowed length without truncation is 1536 byte.

The PLB 2224 supports the spanning tree protocol. When enabled, the address learning and packet forwarding functions are determined by the bridge state for the port as described in the section Spanning Tree Protocol Support.

The PLB 2224 also implements flow control. Depending on the configuration setup, the packet forwarding treatment is modified as described in the section Flow Control.

The Port Rx blocks for 10/100 ports and gigabit ports are essentially identical. The interface to the packet buffer memory are different to account for the difference in line speed on 10/100 and gigabit ports.

3.2.3 Port Configuration, Status and Event Registers

Several registers are used to keep information, either on a global or per port basis, about portconfiguration, priority, monitoring, status and critical events. These registers are summarized below anddescribed in detail in the section **“Operational Description” on Page 71**

- “Chip Configuration Register” on Page 141: - global parameters such as flow control and packet dropping when Tx queues are full.
- “Switch Status and Mask Register” on Page 152: Switch Status and Interrupt Mask Register – global status parameters such as activation of congestion control.
- “Port Status Register” on Page 154: – active/inactive status and detecting late collisions, CRC errors and packet drops.
- “Port Event Register” on Page 155 – detecting Tx queue full and other critical events such as station moves, oversize packet reception.
- “Port Underrun/Overrun Register” on Page 156 - detecting underrun and overrun condition for internal Tx and Rx FIFOs respectively. For internal use only.
- “Port MII Register” on Page 157 - speed, flow control, duplex and link configuration parameters.
- “Port Monitor Register” on Page 158 – monitoring status to indicate whether packets received and transmitted from this port are to be monitored.
- “Port Priority Register” on Page 158 – priority status, for implementing source port based priority.
- “Port Trunk Register” on Page 159 – trunk membership and configuration for implementing trunking support.
- “Port Bridge State Register” on Page 160 - bridging status for implementation of the Spanning Tree protocol.
- “Port Index Register_20_0” on Page 161 & “Port Index Register 26_21” on Page 184 – VLAN index for implementation of the port-based VLAN functionality.

3.3 Switch Controller

The Switch Controller performs a variety of functions, including packet data buffering, address learning and resolution, managing packet buffer memory, packet queuing and scheduling for transmission.

3.3.1 Packet Data Buffering

The PLB 2224 uses a combination of SRAM and EDRAM for packet storage. Every incoming packet requires 12 byte of SRAM – 4 byte for PBH and 8 byte for the first chunk of packet data. A total of 12 Kbyte of the SSRAM is reserved for this purpose. The PLB 2224 allocates fixed, 1.5 Kbyte (1,536 byte) cells for storing packet data in the EDRAM. The PLB 2224 architecture supports a maximum of K (1,024) packets.

The Switch Controller uses a link list mechanism to keep track of where the packets are stored in the packet buffer memory. The packet link list buffer (PBLL) is maintained on-chip and can store a total of 1K entries.

The PLB 2224 maintains read and write pointers to manage a total of 57 queues that are on chip. The queues are:

- 26x2 = 52 variable length, unicast traffic queues for the 26 Ethernet ports, each with two priorities – TxHQ and TxLQ.
- 2x2 = 4 variable length, unicast traffic queues for the two CPU transmit ports, each with two priorities. CTxHQ and CTxLQ
- One variable length queue to keep track of all free or unassigned pointers (FreeQ).

In addition, there are 28, 256-entry broadcast tables (BcastT), one per port, to keep track of multiple destination packets (i.e., multicast, broadcast and unknown DA packets that are destined for multiple ports).

These pointers point to the entries in the PBLL. As a packet arrives into the switch the next available free entry (i.e., packet buffer number) in the PBLL gets assigned to that packet and the read pointer for the FreeQ is updated. For single destination packets, the write pointer of either the HIGH or LOW priority Tx queue of the destination port gets updated. As packets are transmitted on to the link (or possibly dropped) from the destination port queue, the corresponding read pointer for the Tx queue is updated; and the packet buffer is returned back to the free packet buffer pool, and the write pointer for the FreeQ is updated.

3.3.2 Address Resolution Logic (ARL)

The PLB 2224 supports auto-learning and auto-aging as configurable options. See bit *in_ma_en* in **"Chip Configuration Register" on Page 141** and *ma_freeze* and *ma_new_freeze* bits in **"Chip Configuration Register" on Page 141**. The ARL block performs these functions.

A MAC Address table (MA Table) is used to manage address learning. An 8-byte entry is created and maintained for each known unicast and multicast MAC address. The SRAM is used for storing the MA Table. The PLB 2224 uses hashing mechanism for fast address lookup and supports up to a total of 2K hash values. There are four entries per hash value, for total maximum of 8-K entries in the MA table. In addition, the PLB 2224 uses a small, 16-entry MA Table on chip to store overflow entries in an unlikely event that five or more MAC addresses generate the same hash vector. This minimizes the probability of MA Table thrashing (i.e., constant swapping of a valid entry with a new entry).

The ARL searches Mac table entry based on Destination address of the received packet. It determines VLAN tag/priority for all untagged frames. Support is provided for port based/Tag based VLAN and priority assignment. It Implements the MAC address Lookup/Learn/Aging functions. up to 8192 MAC addresses can be resolved via 2048 buckets with 4 entries each. ARL Supports Link Aggregation (802.1ad) when deciding on destination port list. It forwards queue headers for each packet to the Queue Manager (PQC) once packet has been successfully stored in eDRAM.

Table 5 MA Entry Formats for Unicast and Multicast Addresses

Bit Fields	Name	Description
Unicast		
63	Unicast	Denotes whether the entry is Unicast or Multicast
62:57	rsv	
56:52	port_id[4:0]	Index to VLAN/Floodmap for port list lookup
51:48	ma_state[3:0]	Denotes age; plus criticality, priority and lock attributes
47:0	MA[47:0]	MAC address
Multicast		
62	Multicast	Denotes whether the entry is Unicast or Multicast
58:52	flood_ix	Index to VLAN/Floodmap for port list lookup (Upto 256 Multicast entries)
51:48	ma_state[3:0]	Denotes age; plus criticality, priority and lock attributes
47:0	MA[47:0]	MAC address

Each MA Table entry is 8B. Entries with the ma_state[3:0] field value between 4’b0000 and 4’b0011 (inclusive) are considered special – locked entries, and high priority and/or critical entries. These entries cannot be aged or bumped from the MA Table to make room for new entries. During chip initialization all MA Table entries are set to invalid. **“MA**

Table Entry Types” on Page 38, below show the different entry types based on the ma_state[3:0] value.

Table 6 MA Table Entry Types

ma_state[3:0]	Locked	Critical	High Priority	Description
0000	Yes	No	No	Locked entry. Cannot be aged or bumped.
0001	Yes	Yes	No	Locked and Critical entry. Cannot be aged orbumped.
0010	Yes	No	Yes	Locked and (high) Priority entry. Cannot be aged or bumped.
0011	Yes	Yes	Yes	Locked, Critical and (high) Priority entry. Cannot be aged or bumped.
0100-1110	No	-	-	Normal entry that can be aged or bumped.
1111	No	-	-	Invalid entry.

3.3.3 Address Learning and Updating

When a new, valid unicast packet is received at an Ethernet port, the Source MAC Address (SA), along with the source port number (src_pid) is pushed on to the Source Address Learning Queue (SAQ). The MA Table is searched for the existence of an entry corresponding to the SA and one of the following actions is performed, depending on certain configuration options.

- If an entry corresponding to the SA is in the table and the actual source port number (src_pid) matches with the value stored in the MA Table entry AND the entry is not marked as a locked entry, the ma_state[3:0] field value is reset to 4’b0100. If, on the other hand the entry is marked as locked, the ma_state[3:0] field value is left unchanged. This requires the PLB 2224 to be configured for auto-learning.
- If an entry corresponding to the SA is in the table but the actual source port number (src_pid) does not match with the value stored in the MA Table entry, AND the entry is not marked as a locked entry, the src_pid in the entry is updated to the new source port id and the ma_state[3:0] field value is reset to 4’b0100. If, on the other hand the entry is marked as locked, the ma_state[3:0] field value is left unchanged. This requires the PLB 2224 to be configured for auto-learning.
- If an entry corresponding to the SA is not in the MA Table or has become invalid as a result of aging, a new entry is created in the MA Table corresponding to the SA and src_pid.The status field is set to 4’b0100. The new entry in the Table is created in one

of the following possible ways, in the specified sequence. Note that the creation of a new entry requires the PLB 2224 to be configured for auto-learning matching with

1. An invalid entry in the main MA Table corresponding to the hash value of the SA is replaced.
2. If entries in the main MA Table corresponding to the hash value are valid, then the four entries and all the enties in the 16-entry on-chip MA Table are searched and the oldest entry (i.e., entry with the highest ma_state[3:0] value in the range 4'b0100 to 4'b1111) is replaced.

3.3.3.1 PLB 2224 is not Configured for Auto Learning

- If the PLB 2224 is NOT configured for auto learning and either an entry corresponding to the SA does not exist or the src_pid associated with the entry is different from the actual src_pid, this event is considered as *intrusion* and the appropriate programmed action - either updating the MA Table entry if the bit ma_freeze_new is set to 1, or dropping the packet, or sending it to a pre-configured port list .

3.3.3.2 Global Updating

- As shown in **"Chip Configuration Register" on Page 141**, the PLB 2224 also supports two global configurations options that apply to the entire MA Table. The ma_freeze bit when set treats all MA Table entries as being locked while ma_new_freeze bit treats all MA Table entries as locked but allows SA to src_pid associations to be changed.

The CPU must explicitly create all multicast entries in the MA Table. See the section Configuring MA Table Using CPU, below for details on how the MA Table entries are created using the CPU.

3.3.3.3 Address Resolution

The address resolution process involves searching the MA Table for an entry corresponding to the destination address (DA) of the incoming packet and extracting information regarding the associated (destination) port id (for unicast DA), destination port_ix (for multicast DA), priority and criticality. This information is used to create appropriate entry(s) in the transmit queue(s) of the egress ports.

The address resolution process (i.e. computation of the destination port list for the DA) is quite involved and depends on the type of packet (unicast or multicast or broadcast), whether the DA is known or unknown, and whether the PLB 2224 is configured for VLAN support (source port security and/or multicast/broadcast containment). **Figure 11**, below shows the process in the absence of VLANs, and is described below.

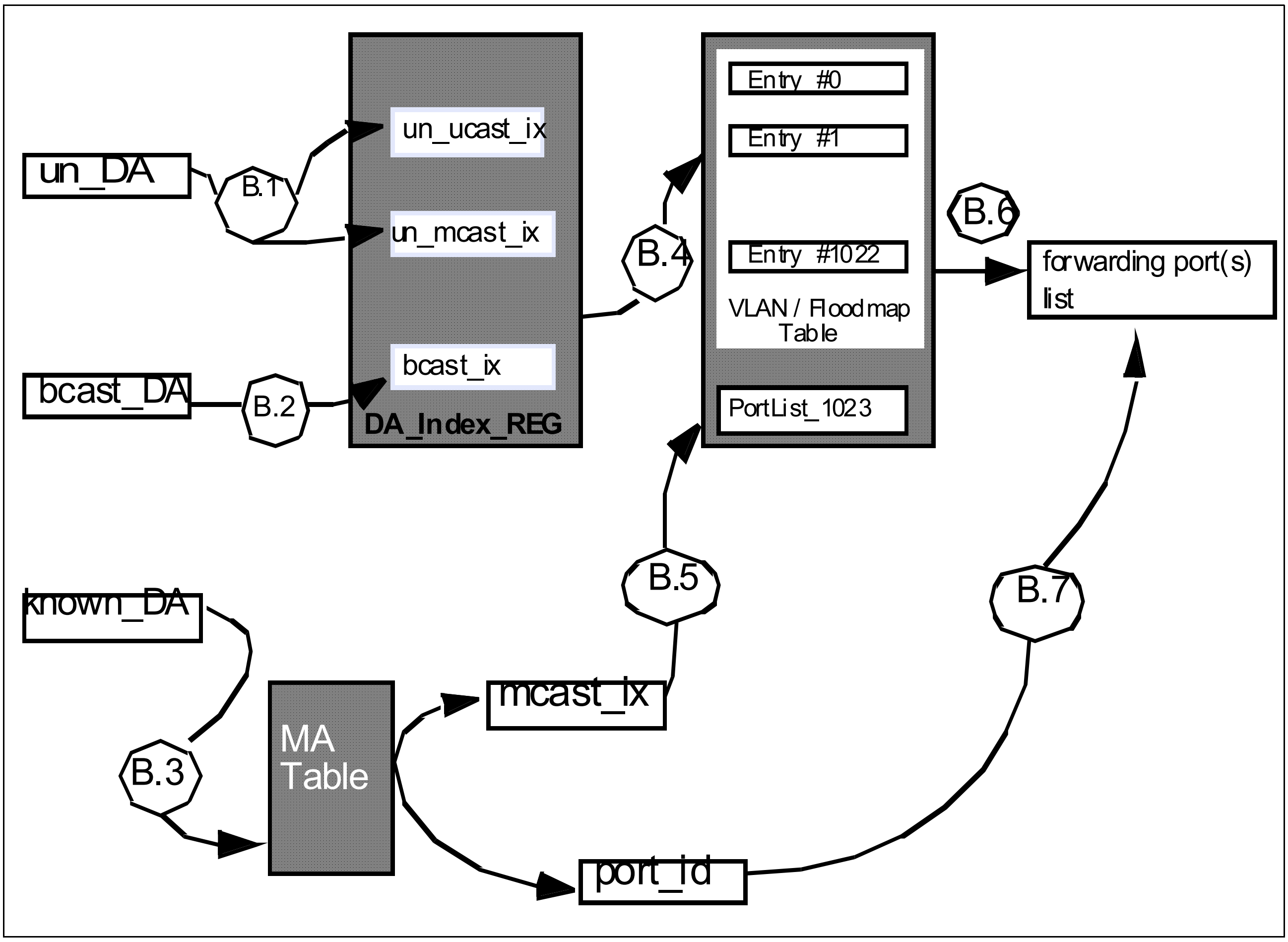


Figure 11 Destination Port List Creation (VLAN's Disabled)

- **Known Unicast DA** : B.3 -> B.7 : The MA Table DA look up directly returns destination port id (dst_pid).
- **Known Multicast DA** : B.3 -> B.5 -> B.6 : The MA Table lookup returns an index (mcast_ix) which is used to look up the VLAN/Floodmap Table or port_list1023 register if mcast_ix = 10'b1111111111, which returns the destination port list.
- **Broadcast DA** : B.2 -> B.4 -> B.6: The DA_index_reg lookup returns an index which (bcast_ix) is used to lookup the VLAN/Floodmap Table or port_list1023 register if mcast_ix = 10'b1111111111, which returns the destination port list.
- **Unknown Unicast DA** : B.1 -> B.4 -> B.6: The DA_index_reg lookup returns an index (un_ucast_ix) which is used to lookup the VLAN/Floodmap Table or port_list register if mcast_ix = 10'b1111111111, which returns the destination port list.
- **Unknown Multicast DA** : B.1 -> B.4 -> B.6: The DA_index_reg lookup returns an index (un_mcast_ix) which is used to lookup the VLAN/Floodmap Table or port_list1023 register if mcast_ix = 10'b1111111111, which returns the destination port list.

3.3.3.4 Address Aging

The address aging function is used for automatically aging out address entries in the MA Table. Periodically, all entries in the MA Table are examined sequentially and the ma_state[3:0] for the entries that are not locked (i.e., entries with *ma_state[3:0]* > 4'b0100) value is incremented by one. As described in **"Switch Configuration Register" on Page 145**, the timer tick period for the aging function is determined by the field *timer_tick_sel[2:0]*. When the *ma_state[3:0]* equals 4'b1111, the entry is considered aged (invalid) and cannot be used for DA lookup. The locked entries are unaffected by the aging process.

3.3.3.5 Configuring MA Table Using CPU

In a CPU-based system, the CPU can create entries in the MA Table by sending special packets over the CPU port. The process involves two steps. In the first step, the ARL Register described **"ARL Register" on Page 170** , is set as appropriate by writing to it. This is followed by sending a packet whose SA is the address that is to be learnt. The MA Table entry is constructed as follows:

- **Unicast Entry** – MA = SA, src_pid = ma_ports bits 4 through 0, *ma_state[3:0]* = 4'b0100, assumes ma_pri, ma_critical, ma_locked bits (see **"Chip Configuration Register" on Page 141**) are set to 0.
- **Multicast Entry** – MA = SA, portlist_ix = ma_ports bits 10 through 0, ma_state[3:0] = 4'b0100, assumes ma_pri, ma_critical, ma_locked bits are set to 0.
- **Special Unicast Entry** - MA = SA, *src_pid* = ma_ports bits 4 through 0, *ma_state[3:0]* = encoded value of ma_pri, ma_critical, ma_locked bits.
- **Special Multicast Entry** - MA = SA, portlist_ix = ma_ports bits 10 through 0, ma_state[3:0] = encoded value of ma_pri, ma_critical, ma_locked bits.

The CPU can also delete (invalidate) entries in the MA Table by first setting the ma_delete bits in the ARL Register to '1', followed by sending a special packet with SA that matches the MAC address in the entry to be deleted.

The creation or deletion of MA Table entries requires that *e_ma_learn* bit in **"ARL Register" on Page 170** be set to 1.

The MA Table is accessible for CPU reads and writes. Although not advised, the CPU can also create or invalidate entries by creating MA Table entries in the software, calculating the hash value for the MAC address and writing to the appropriate location in the MA table.

3.3.4 Port Tx and MAC Tx

The port transmit function continuously monitors the transmit queues to see if there is a packet that needs to be transmitted from that port. When a packet is available for transmission, it reads the packet data from the packet buffer – first 8-byte of the packet data are read from the SRAM while the remaining bytes from the EDRAM. The information about packet length and starting EDRAM number is obtained from the packet header (PBH). The packet data is assembled and forwarded to the MAC Tx block for the actual transmission.

In full duplex mode, the MAC Tx block works by altering the transmission behavior based on the reception of any PAUSE frames on MAC Rx. The MAC Tx block also sends out PAUSE frames to activate flow control if requested by MAC Rx. Refer to the section on Flow Control (**Page 52**) for more details. For packets generated by the CPU, Port Tx calculates the CRC before forwarding the data to the MAC Tx.

MAC Tx block generates the preamble, serializes the data and sends the packet to the PHY device.

3.3.4.1 Packet Queuing and Port Queues

The PLB 2224 supports two priorities per port. The priority for the packet is determined either by the ingress port priority or the results of the MA Table lookup. See the section on Packet Prioritization (**Page 52**) for more details. Each port maintains two queues for packet transmission – HIGH and LOW priority queue. The switch controller modifies the packet buffer link list to append the packet to the appropriate queue of all the destination ports for the packet.

The fixed-size, 256 entry Broadcast Table, (BcastT), allows each port to have a maximum of 256 multiple destination packets in the Tx queue at any one time. A new multiple destination packet that is to be transmitted from the port that has reached this limit is dropped. When the priority feature is enabled, the BcastT is split in half, allowing the HIGH and LOW priority queues to have a maximum of 128 multiple destination packets. The maximum number of total (i.e., unicast and multiple destination) packets in the HIGH and LOW priority queues is determined by the watermarks, as described in the section on Flow Control (**Page 52**).

3.3.4.2 Packet Scheduling

The PLB 2224 uses Weighted Fair Queuing (WFQ) for transmitting packets that are in the port queues. Setting the bandwidth_ratio[2:0] bits appropriately configures the relative weights. The TxHQ to TxLQ bandwidth ratio can vary from 1:0 (i.e., TxLQ can send a packet only when the TxHQ is empty), to 1:1 (i.e., about equal access for the two queues). Refer to **"Chip Configuration Register" on Page 141** for more details.

3.3.5 Packet & Queue Manager (PQC)

The Packet manager maintains free packet buffers (pb) It allocates packet buffer to Rx for incoming packets. It stores service count(a packet may be forwarded to more than 1 destination ports) in case of Broadcasts/Multicasts. It de-allocates the Packet Buffer if it decides to drop the packet. Packet drop decisions are based on error,control information from ARL& Switch Configuration Registers.It de-allocates Packet buffer or decrement service count after a packet is transmitted by Tx. When requested it gives TX the next pbnum of a packet from its corresponding egress queue.

Based on append requests from ARL, it establishes egress queue for each port. It generates service count of a ingress packet. It sends congestion status to all the port MAC for flow/congestion control. It schedules one packet from each port's all available queues to the corresponding port's Tx.

3.3.6 Transmit Module (TX)

·The TX block generates requests to PQC when its transmit queue is ready to receive new data. It Interfaces to GMAC & MAC, handling all signals in both Full Duplex (FD) and Half Duplex (HD) modes of operation. Collisions are handled in 10/100 Mbit/s half duplex mode of operation. it is capable of Inserting/deleting or modifying the VLAN tag/ priority to/from all frames based on Packet Header information and ARL indication in the Queue header. It Interfaces with PQC to obtain first pbnum/tagging information/packet length/Source PID for new frame to be transmitted. It Instructs PQC to release frames, which have been transmitted.

µController Subsystem (CPU):

The CPU subsystem consist of a PCI v2.1 compliant interface, a Master-slave IIC interface, A Motorola/Intel Generic Host interface (shared with PCI), A generic Packet filter & configuration registers.The PCI port interfaces to the most popular 32-bit processors with minimal or no glue logic. The generic interface supports 32-bit CPUs with either multiplexed or de-multiplexed address and data.

The 32-bit CPU interface is a slave or target-only interface. In addition, a 2-wire master/ slave serial interface is provided. In the master mode, a serial enhanced expanded programmable read-only memory (EEPROM) containing power-on configuration data can be connected to this interface. In the slave mode, a CPU can be connected to provide initialization and management functions. Only one interface (out of 32-bit generic, 32-bit PCI, 2-wire serial master and 2-wire serial slave) can be active and must be selected at reset via pinstrapping.

3.3.6.1 Receiving and Sending Packets from the CPU Port

The CPU port on the PLB 2224 is a logical port that can be accessed by one of the three physical CPU interfaces – 32-bit PCI, 32-bit generic or 2-wire serial. Only one physical interface is allowed in the system. This section describes the operation of the logical interface, while the physical interfaces are described in the section titled CPU Interface (**"CPU Interface" on Page 106**).

The logical CPU port has one receive and two transmit ports. The receive port is identified as Port 26, and the transmit ports are identified as Ports 26 and 27. In keeping with the terminology for the user ports, the CPU sends packets to the PLB 2224 Ethernet ports over the Rx port while the data from the Ethernet ports to the CPU is sent from the Tx ports.

3.3.6.2 CPU Port Rx

A small block called CMAC Rx is used to emulate the functionality of the MAC Rx block on the Ethernet ports. This block collects data written by the CPU, computes the length of the packet and presents it to the CPort Rx block.

Typically, the packets received from the CPU are special packets and may carry with them other optionssuch as:

- whether the packet is to be forwarded to an Ethernet port without looking up the DA in the MATable
- whether the SA is to be learnt and entry needs to be created/modified in the MA table
- if the SA is to be learnt, then whether the SA is to be associated with a port id other than the ingress port (i.e., the CPU Port) when the MA Table entry is created
- whether the SA needs to be associated with special attributes such as locked, critical and high priority in the MA Table.

The CPU sends packet data using the registers described in Table 36: CMAC Data Register and **"CMAC RX Register" on Page 167**. The Rx FIFO is 16-byte deep, so the data is written 16-byte at a time. The following protocol

is used:

- Check the crx_cpu_pkt_rdy bit in the CMAC RX Register to see if the PLB 2224 is ready to accept the packet.
- Check the crx_cpu_fifo_rdy bit in the CMAC RX Register to see if the Rx FIFO is available
- Set the crx_sof bit in the CMAC RX register to indicate start of packet data.
- Write 16-byte (or less in the case of last chunk) of data, 4-byte at a time, in the CMAC Data Register using any of the four base addresses and then wait for the crx_cpu_fifo_rdy bit to set. If the CPU writes to this register at a rate of 100 Mbit/s or less, then the crx_cpu_fifo_rdy bit need not be checked.
- Before writing the very last 4-byte or less of data, set the *crx_bytecnt[1:0]*, *crx_eof*, *crx_crc_err*, *crx_crc_gen* and *e_cpu_pkt_padding* in the CMAC RX Register.
- Write the last 4-byte or less of data.
- Following table lists the consequence of setting *crc_gen* and *e_cpu_pkt_padding* bits

Table 7 PBH Packet Length for Different crc_gen & pkt_padding Bits

crc_gen	0	1	0	1
e_cpu_pkt_padding	0	0	1	1
pbh pkt_len	actual packet length	actual packet Lenght + 4	Packet Length = 64 & crc_gen set to ’1’ internally	Packet Length = 64

3.3.6.3 CPU Port Tx

The CPU reads the packets being transmitted to the CPU Ports 26 and 27 using the registers **“CMAC Data Register” on Page 167**, **“CMAC TX Register” on Page 169** and **“CMAC_TX1 Register” on Page 170**. The following protocol is used:

- Poll CMAC Tx0 Register, bits ctx_cpu_pkt_rdy[27:26] to see if a packet available in either of the CPU Tx ports.
- Poll CMAC Tx0 Register , bits ctx_cpu_fifo_rdy[27:26] to see if data is available in either of the Tx FIFOs.
- Set either the cpu_txq_rd_req[1] or cpu_txq_rd_req[0] bit to 1 in the CMAC Tx0 Register indicating that the CPU wants to read the packet.
- Read the packet header from the appropriate CMAC Tx register to retrieve information such as packet length, and CRC status.
- Read the packet data, 16-byte at a time from the CMAC Data Register using any of the four base addresses and then wait for the ctx_cpu_fifo_rdy bit to be set to 1. If the CPU reads from this register at a rate of 100 Mbit/s or less, then the ctx_cpu_fifo_rdy bit need not be checked.

CPU can flush the CMAC Data Register for the Tx port it is currently reading by setting the ctx_flush bit in the CMAC Tx0 Register.

3.3.7 MDIO Interface

The MDIO serial interface is used for communicating with the PHY device(s). Using this interface, PLB 2224 can access the internal registers of the PHY devices.

3.3.7.1 Auto-Negotiation

PLB 2224 supports auto-negotiation on the 10/100 Mbit/s ports. Auto-negotiation on the gigabit ports is supported when the GMII is selected. Using the MDIO interface PLB 2224 can negotiate the link parameters – speed, duplex mode and pause_enable (i.e., full-duplex flow control), and pause_augment (for gigabit speed only) by understanding the status and capabilities of the PHY device and configure itself appropriately.

For SMII, the link information can be obtained by reading the status information continuously coming over the SMII or by accessing the PHY registers using the MDIO interface. See *use_mdio_mode* bit in **"Chip Configuration Register" on Page 141** . For the gigabit ports, the auto-negotiation is performed using the MDIO interface when the GMII or MII option is selected. Using the MDIO interface, the CPU can access the PHY device registers and override the auto-negotiated settings if desired. *e_hw_mode* bit in **"Switch Configuration Register" on Page 145** must be cleared to 0 in order to disable auto-negotiation.

With e_hw_mode = '0', PLB 2224 will be in software mode of operation and for speed/ duplex mode/pause_enable settings the **"Port MII Register" on Page 157** comes into picture. Through this PLB 2224 can read the phy registers and set the corresponding parameters speed/duplex mode/pause_enable accordingly.

3.3.8 LED Interface

A flexible LED interface is supported. The interface can be either used in the matrix format or serial format. Four LEDs per port can be used to display link, speed, duplex, activity status and warning conditions.

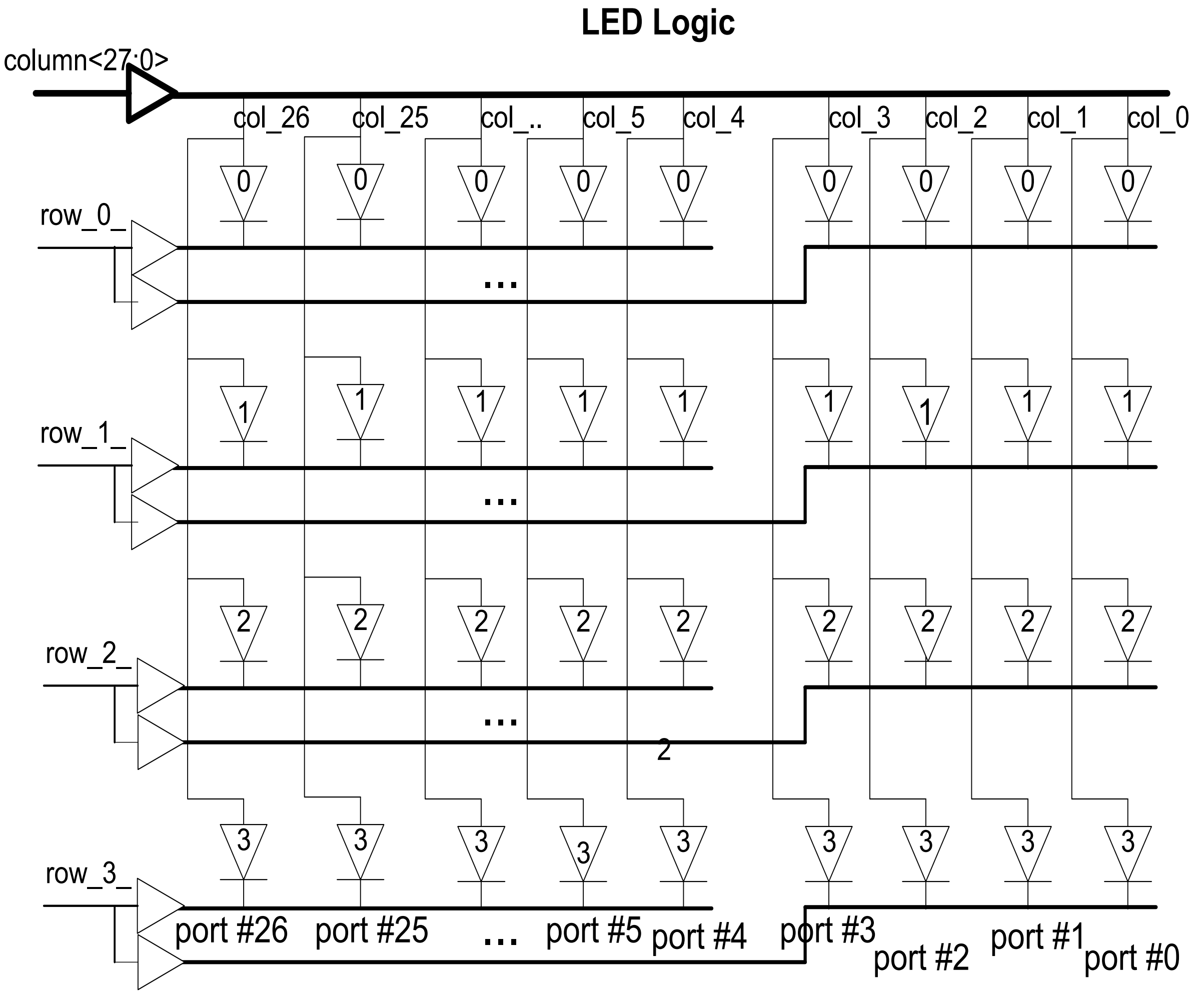


Figure 12 Matrix Mode LED Connection

For each port on PLB 2224, there are four LED's. LED has two display modes. In mode 1, LED0 indicates speed of either 10/100 Mbit/s, LED1 indicates port link status and activities, LED2 indicates full/half duplex, and LED3 displays the status warning/error.

Table 8 summarises status of LEDs in various modes.

Table 8 **Matrix LED Modes (per port)**

MODE	LED #	LED Activity	Description
MODE 0	LED #0	On/Off	10 Mbit/s Link OK/Not OK
		Flashing	10 Mbit/s Recieving/Transmitting
	LED #1	On/Off	100 Mbit/s Link OK/Not OK
		Flashing	100 Mbit/s Receiving/ Transmitting
	LED #2	Off	Half Duplex
		On	Full Duplex & Link OK
	LED #3	Off	Normal
		On	Port Disabled
		Flashing	Error (CRC Error | Pkt Overflow | Packet Drop
MODE 1	LED #0	On	100 Mbit/s & Link OK
		Off	10 Mbit/s
	LED #1	On/Off	Link without activity/Link Down
		Falshing	Port receiving/transmitting
	LED #2	Off	Half Duplex
		On	Full Duplex & Link OK
	LED #3	Off	Normal
		On	Port disabled
		Flashing	same as Mode 0

To Display status/activity of 26 Ethernet ports plus CPU's port per chip, each chip will need to drive 108 LEDs. It is ensured by design that, there is only one row active at a given time, i.e. logic 0. Each row alternately has 5 ms of active time. LED assignment is defined such that Port 0 is assigned to column 0, Port 1 to column 1, and Port 26 is to column 26. Each column has four LED's from row 0 to 3. LED's for Port 26 can only be controlled by CPU. LEDs for Port 0 to 25 directly refelct the status of Ethernet ports if soft_led is 0, and they are written by CPU if soft_led is 1. If soft_led is 1, led_d[3:0] is matched to port#0, led_d[7:4] is to Port 1, and led_d[103:100] is to port 25. CPU led, i.e. led_d[107:104], are always for Port 26.

At RESET, all the LED’s are turned on for 0.5 s and then turned off.

The status LED lights up whenever there is a change in the following

- A packet received with CRC error

- Oversize packet
- A dropped packet
- ma change
- collision (in conjunction with mask_coll_led bit in switch configuration register)
- late collision

Serial Mode LED

The above information is serialy shifted out, when the LED interface is configured in serial mode

LED_row_n[0]	: led_sync
LED_row_n[1]	: led_load
LED_row_n[2]	: led_data
LED_row_n[3]	: led_clk

3.3.9 Reset and Initialisation

Setting reset_n low (for a minimum of 3 clocks or 30 ns) activates the reset and initialization block. When in the reset state, all the internal logic is reset and all configuration registers are set to their default values.

When the RESET_N line goes from low to high, the logic levels on the strapping pins are registered and stored in the appropriate configuration registers and the PLB 2224 enters a secondary reset state. During this state, the PLB 2224 initializes the data structures such as MA Table, reset sequence for the EDRAM and Packet Buffer Link List (PBLL). If the PLB 2224 is configured as a non-CPU system, some of the configuration registers are read from the EPROM. Data integrity check on the EPROM data is carried out. If it fails, the status LED (i.e., LED #3) for port #7 continues to blink indicating an EPROM read error. Tthe port logic is disabled until the PLB 2224 successfully completes power on sequence, is configured correctly and determines that the 10/100 and gigabit links are OK.

3.3.9.1 Power Strapping

The LED interface pins led_row_n[3:0] and led_col[26:0] are also used as power strapping pins to set certain operating parameters for the PLB 2224. On the rising edge of the reset_n pulse, the logic level of led_row_n[3:0] and led_col[23:0] are sensed and latched into the Chip Configuration Register as shown in **"Chip Configuration Register" on Page 141**. All the LED signals have internal pullups. The power strapping pins can be pulled-up through ~10 K resistor or pulled-down through a ~1 K ohm resistor to set the required value.

The following table shows the power strapping information

Table 9 Power Strapping Table

Signal	Power Starp Pin	Register Bit
Intelligent	led_row_n[3]	Chip Configuration Register Bit 31
EEPROM present	led_row_n[2]	
disable_pause	led_row_n[1]	Chip Configuration Register Bit 28
GMII/SERDES_25	led_row_n[0]	GPort1 Gmode Register Bit 0
GMII/MII_25	led_col[25]	GPort1 Gmode Register Bit 5
GMII/SERDES_24	led_col[26]	GPort0 Gmode Register Bit 0
GMII/MII_24	led_col[24]	GPort0 Gmode Register Bit 5
delay_sel5[1:0]	led_col[23:22]	Chip Configuration Register Bits 23/22
delay_sel4[1:0]	led_col[21:20]	Chip Configuration Register Bits 21/20
delay_sel3[1:0]	led_col[19:18]	Chip Configuration Register Bits 19/18
delay_sel2[1:0]	led_col[17:16]	Chip Configuration Register Bits 17/16
delay_sel1[1:0]	led_col[15:14]	Chip Configuration Register Bits 15/14
delay_sel0[1:0]	led_col[13:12]	Chip Configuration Register Bits 13/12
sim_mode[1:0]	led_col[11:10]	Chip Configuration Register Bits 11/10
in_ma_en	led_col[9]	Chip Configuration Register Bit 9
use_mdio_mode	led_col[8]	Chip Configuration Register Bit 8
LED Parallel-Serial Mode	led_col[7]	
e_hd_cg_ctl	led_col[6]	Chip Configuration Register Bit 6
mask_coll_led	led_col[5]	Chip Configuration Register Bit 5
en_tx_led	led_col[4]	Chip Configuration Register Bit 4
led_mode	led_col[3]	Chip Configuration Register Bit 3
pcb_config	led_col[2:0]	Chip Configuration Register Bit 2:0

3.3.9.2 Delay Select for SMII_RCLK

The signals delay_sel in the above table are used to control the delay the SMII_RCLK internally. This is mainly used in applications where source synchronous SMII is not used. The details of the delay selection is shown in the figure below

For non-Source synchronous application, SMII_CLK is connected back to SMII_RCLK (like smii_clk_7_0 is connected to smii_rclk_7_0)

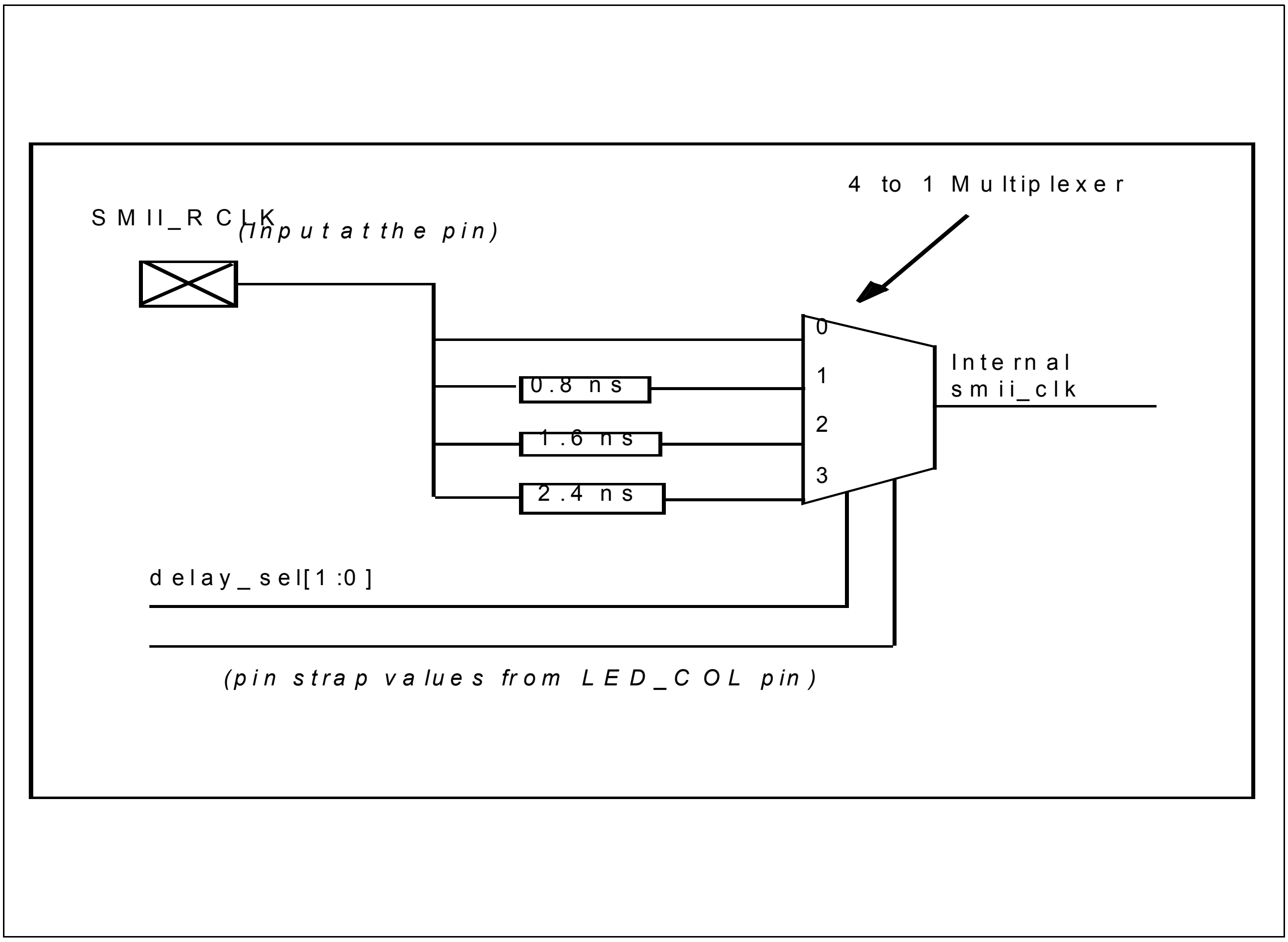


Figure 13 Delay Control for SMII Clock Internally

3.4 Features for a Managed Switch

3.4.1 Flow Control

The PLB 2224 supports flow control in both half and full duplex modes. Flow control in the half duplex mode is implemented by creating collisions on the link. The flow control in full duplex mode uses PAUSE frames as per the 802.3x specification.

3.4.2 Spanning Tree Protocol Support

The PLB 2224 supports the Spanning Tree Protocol as per the 802.1D specification. As specified in **"Port Bridge State Register" on Page 160**, each port has two configuration bits that define the bridging state for that port. For the Ehthernet (i.e., non-CPU) ports, the following actions are performed, depending on the state of the port. Upon reset, each port is in the default forwarding state (2'b11). Depending on the state, the non-transmitted packets are automatically purged from the Tx queue.

Table 10 Spanning Tree States and Actions

Port State[1:0]	Receive Packets	Learn SA	Lookup DA	Transmit Packets
00 - Diabled	No	No	No	No
01 - Listening	Yes	No	Yes	Only those with the Critical Packets
10 - Learning	Yes	Yes	Yes	Only those with the Critical packets
11 - Forwarding	Yes	Yes	Yes	All packets

3.4.3 Packet Monitoring

Packets can be monitored based on the ingress port or egress port, if the monitor bit as described in **"Port Monitor Register" on Page 158** is set to 1. The id of the monitoring port (i.e., port to which all the monitored packets are sent) is specified in **"Switch Configuration Register" on Page 145**. The PLB 2224 can also be configured to monitor a) packets with CRC errors and b) packets whose source port number does not match with the port id value the MA Table entry corresponding to the packet SA. Setting the appropriate configuration bits in **"Switch Configuration Register" on Page 145** activates packet monitoring.

3.4.4 Packet Prioritization

The PLB 2224 architecture supports two priorities – HIGH and LOW per port. The priority can be based on the ingress port and/or DA. Two configuration bits, *pri_at_src* and *pri_at_da*, specified in **"Switch Configuration Register" on Page 145** are used to determine how packet priority is set. When pri_at_src is set, the configuration register in

“Port Priority Register” on Page 158 is looked up to set the priority. When pri_at_da bit is set, the MA Table is used to set the priority. When both *pri_at_src* and *pri_at_da* bits are set, the priority is set to HIGH if either lookup results in HIGH priority (i.e., the result is an OR function).

The following rules are used in determining DA based priority:

- For broadcast packets, priority is always set to LOW,
- For unknown packets, priority is always set to LOW,
- When MA Table entry corresponding to the DA is not a locked entry, the priority is always set to LOW.

3.5 Registers in PLB 2224

Following table describes the Chip Registers & their address map

Table 11 Register Table

Byte Offset	Register Name	Access Mode	Description
0x00	chip_config	R/W	Chip Configuration Register
0x04	chip_type	R	Revision No. & ID
0x08	switch_config	R/W	Switchi Engine & VLAN configuration
0x0C	switch_status	R	Address table & Interrupt Status
0x10-0x1C	port_status	R	Depicts status of each port
0x20-0x24	port_event	R	Error & queue status for each port
0x28-0x2C	port_run	R/W	Depicts overrun/underrun status of each port
0x30-0x3C	port_mii	R/W	Port enable/disable & configuration register
0x40	port_monitor	R/W	Port monitoring control register
0x44	port_priority	R/W	Port based priority allocation register
0x48-0x4C	port_trunk	R/W	Port trunking/aggregation grouping register
0x50-0x54	port_bridge_state	R/W	Depicts Bidge state (STP)
0x58	portlist_1023	R/W	Floodmap bypass register
0x60-0x78	port_vid_ix20-0	R/W	Port Index definition register

0x7C	da_index	R/W	Index setting for unknown unicast, flood & broadcast
0x80	mem_u_addr	R/W	Upper Memory base address
0x84	mem_access	R/W	Indirect Memory access register
0x90-0x9C	cmac_data	R/W	Data to/from CPU port
0xA0	cmac_rx	R/W	CPU RX control register
0xA4 – 0xA8	cmac_tx0/1	R/W	CPU TX control register
0xAC	Arl	R/W	Address lookup control register for CPU ports
0xB0	freeq_ptr	R	Read/Write pointer of the Free queue (only for diagnostic)
0xB4	freeq_cnt	R	current available packet buffer count in Free queue
0xB8-BC	cpu_txq26/27	R	Current TX queue count
0xC0-D0	water_mark_count	W	Water mark register for TX/RX
0xD4	Tag_pri	R/W	Tag Priority mapping control
0xD8	E_pri	R/W	Egress priority demapping control
0xDC	Tag	R/W	Tag set control
0xE0	Vlan_aware	R/W	Set ingress rule for each port and overall switch VLAN awareness
0xEC	Vlan_Ingress Filtering	R/W	Enable Ingress filtering
0xE4	Port_index23-21	R/W	The index number is defined for each of the ports. The number based on source port will be used to lookup FLOODMAP for the destination port list when a non-unicast packet is received
0xE8	Port_index26-24	R/W	Same as above
0xF0-FC	led_data	W	32-bits of LED display data if CPU has the control of LED

0x100-0x11C	pattern_mask	R/W	The register with address offset of 0x100 is for pattern #0, the next one with address offset of 0x104 is for pattern #1, …and the last one with address offset of 0x11C is for pattern #7. There are eight patterns totally in this chip
0x120-0x124	offset03/47	R/W	Offset for each half word
0x128-0x12C	op_table0/1	R/W	opcode table
0x130-0x16C	action_table0-15	R/W	pattern match action table
0x170	e_filter	R/W	Filter enable control
0x1C0 or 0x1E0	g_rxtx	R/W	GPORT control register
0x1CC or 0x1EC	g_mode	R/W	GPORT mode select
0x1D0-8 or 0x1F0-8	g_pcs0-2	R/W	GPORT autonegotiation & link status control

4 Data Structure

There are three data areas in PLB 2224. The first one is EDRAM for packet data only. The second one is SSRAM for PBH (Packet Buffer Header), the first 8-bytes of packet data, MA table. The third one includes all the FIFO's, PBL (PB Link List), BCASTQ (Broadcast Queue), FLOODMAP (Flooded port list) and PKT_CTR. The detailed format of each entry is also described in this section.

CPU can only access all the memories indirectly through mem_access register.

Table 12 EDRAM Details

Quantity	Entry Unit Size (Bits)	No. of Packets	Total Size (Bytes)	Comments
2 Banks	256	1 K	1.5 MB	6 MB in each Memory Bank

Table 13 SRAM Details

Type	Entry Unit Size (Bits)	No. of Entries	Total Size (KBytes)
MAC Address	8	4 x 2 K	64
Packet Data	8	1 K	8
Packet Buffer Header	4	1 K	4
Total			76

4.1 Internal Memory

All memory blocks are separated

Table 14 **Internal Memories**

Description	Entry Unit Size (Byte)	Entry #	Total Size (Byte)	Comment
RX FIFO	192 byte	28 ports	6 Kbyte	Separate (Built using registers)
TX FIFO	384 byte	28 ports	12 Kbyte	Dual port memory used here
PBL	2 byte	1 Kbyte	2 Kbyte	SRAM
BCASTQ	2 byte	256 x 32	16 Kbyte	SRAM
FLOODMAP	4 byte	1024	4096 byte	Destination port list for each entry. This can be vlan_port_list in case of a VLAN aware switch or dst_port list in the case of non-VLAN aware switch.

4.1.1 VLAN Memories

Table 15 **VLAN Memories**

Description	Entry Size	Entry #	Total Size	Comment
VLAN Table	40	1K	40 Kbits	Inputs vid_ix[9:0] Outputs {vid[11:0], port_egress_tagged[27:0]}

VID_IX Table	10	4K	40 Kbits	Inputs: { vid[11:0] }
				Outputs : { vid_ix[9:0] }
				VLAN ID index will be referenced to one entry within the VLAN Table. Multiple VLAN's i.e. multiple vid[], may be assigned to one vid_ix[]

Floodmap memory stores the destination port list for each MA's or VLAN membership. For a non-single destination packet the flood_ix[] index from DA lookup result will be used to reference Floodmap for a destination port list. Lower 28-bits of the floodmap stands for the respective port for the destination. In case of a VLAN aware switch, Floodmap table indicates the ports belonging to a particular VLAN. In this case the input lookup is vid_ix. This is used to filter non-member ports at receiving and transmitting side.

where

vid[11:0] = indicates the global VLAN id for the entry. vid[11:0] is the VLAN id to the tagged packets. When receiving, each packet will be referenced to one of the VLAN Table to get the vid_ix[].

port_egress_tagged = list of ports to determine whether the transmitted packet to each destination port should be tagged. For those ports with corresponding bits equal to ‘1’, the packet needs to be tagged.

4.1.2 PBL (Packet Buffer Link List) – 1 K x 2 Bytes

PBL is to store all the PB’s link list. There are 26 links (TXQ) for each of 26 forwarding ports and two links for two CPU ports, and there is one free link for all the PB’s which are yet to receive packet data.

Note: All the entries are assigned to the free link after reset.

Each Ethernet port may occupy two queues instead of one if high priority is enabled.

4.1.3 BCASTQ (BroadCAST Queue) – 256 x 2 Bytes x 32 = 16 KBytes

Each port has an independent 256 entries of queue from BCASTQ to store all the nxt_pbnum[] when the packet is not a single_dst. The nxt_pbnum[] indicates the next PB number to be forwarded right after the current PB number’s transmission.

SRAM

All the packet data are stored in EDRAM (packet buffer). For each packet, the first 8-byte data and Packet Buffer Header (PBH) are stored in SSRAM, the rest of packet data are in ERAM. The maximum of packet data size is (1536) bytes. PBH stores the packet information such as pkt_len[10:0], crc_err, crc_gen, src_pid[] and filter_match[].

4.1.4 MA (MAC Address) - 64 KByte for up to 8 K Addresses

There are 8 K different hash values. Each of them indexes to 2 entries of MA.

Entry Format for PBL, PBH, MA table, etc.

Packet Buffer Link List (**PBL**)

Table 16 PBL Format

single_dst = 0	monitor	nxt_pbnum[12:0]	
single_dst = 1		rsv	ser_cnt[4:0]

Where

single_dst = the PB is for single destination if this bit is set, otherwise, it is for multiple destination ports.

nxt_pbnum[] = if current PB is not ‘flood’ and is appended to one of the TXQ, nxt_pbnum[] indicates the next PB number to be forwarded right after this PB’s transmission. If the current PB belongs to FREEQ instead of any TXQ, nxt_pbnum[] points to the next FREEQ entry. After chip reset, all nxt_pbnum[] entries should be linked.

ser_cnt[] = if ‘single_dst’ is not set, ser_cnt[] indicates how many more ports still need to transmit this packet, and it is initialized when TXQ is appended and should be in sync with the number of ports to be forwarded. ser_cnt[] decrements for each port’s forwarding until it is zero. This field is valid only when the current PB is for non-single_dst packet.

monitor = PB needs to be forwarded to the monitoring port. When ‘monitor’ is set, PB cannot be released until both monitor equal to 0 and ser_cnt[] equal to 0.

Broadcast Queue (BCASTQ)

Table 17 Broadcast Queue Entry Format

	rsv	nxt_pbnum[12:0]

Where

nxt_pbnum[] = indicates the next PB number to be forwarded right after this PB’s transmission.

Note: Snooping may be needed if the entry is read before nxt_pbnum[] is written for the following packet.

Packet Buffer Header (PBH)

Table 18 Packet Buffer Header

31:22	21:16		
filter_match[9:0]	src_pid[5:0]		
15:13	12	11	10:0
start_mem[2:0]	crc_gen	crc_err	pkt_len[10:0]

Where

filter_match[] = The matching results for all the filtering patterns.

start_mem[] = indicating which EDRAM bank stores the first burst data of the packet. Each packet is divided onto 32 B data cell, and the cell is stored onto, in turn, one of EDRAM's until EOF. 'start_mem[]' indicates which memory is the first EDRAM to store the first data cell.

src_pid[] = indicating which port has received this packet from. This information is used to record the number of PB each port has used. The most significant bit of this field is the parity bit for this field. The sum of one's in this field should be odd number. If parity error is detected, it will set the status bit src_pid_parity_err.

src_pid[] is also encoded for crc_gen bit if source port is from CPU, i.e. either from #26 or #27. For most received packets from CPU, crc_gen is asserted. CPU's src_pid[] can be changed to #28 or #29 to indicate crc_gen not asserted.

crc_err = Indicating the packet data with CRC error.

pkt_len[] = packet data length. Due to the PB size, it can only go up to 1536. If the received packet data size is more than 1536, MAC will hold the FIFO write to 1536 and set 'crc_err' for the packet.

crc_gen = indicating the packet is from CPU and MAC requires to transmit CRC bytes after the packet data. For normal packets being received from ports, it should be 0. This bit should be passed to MAC to append CRC bytes.

Note: This bit is encoded within the src_pid[].

MA Entries (MA)

There are 8 K entries of which index, i.e. ma_entry_ix[12:0], is the hashed value of {ma[47:0]]}.

Table 19 MA Format for Unicast Entry

Bit 63	Bits 62:57	Bits 56:52	Bits 51:48	Bits 47:0
Unicast	reserved	port_id	ma_state	MA

Table 20 MA Format for Multicast Entry

Bit 63	Bits 61:52	Bits 51:48	Bits 51:48	Bits 47:0
Multicast	flood_ix[9:0]	ma_state	ma_state	MA

Where

Unicast/Multicast = the MA is for single destination if this bit is 0, otherwise, it is for multiple destination ports. This bit is always 0 for learned SA from any of Ethernet ports. It can be 0 or 1 if the entry is set by CPU.

ma_state[] = Aging state. After initialization, all the ma_state[] fields should be ‘f’. When any SA is learned, it is cleared to ‘4’. Field ma_state[] of each entry will increment once every 28.5 seconds until it becomes ‘f’. If it is ‘0-3’, MA is locked not to be aged and the associated port_id[] should not be changed. If ‘1’ or ‘3’, MA is locked with ‘critical’ attribute. If ‘0’ or ‘2’, MA is locked without ‘critical’.

Note: MA such as BPDU should be ‘critical’.

port_id[] = the MA associated port number if single_dst.

flood_ix[] = the index number for a list of destination ports from FLOODMAP if MA is a ‘flood’ entry. If flood_ix[] is equal to 1023, the port list is from portlist_1023 register without checking FLOODMAP data base.

Note: FLOODMAP is merged into VLAN Attribute Table which is also used for VLAN attributes. Both will share the VLAN Attribute Table.

MA[] = 48-bits of MAC address. The hashed value is created based on these 48 bits.

4.2 Indirect Access to the Memories

A brief description on how to access the different memories in PLB 2224 is given below.

Two registers (memory upper address register and memory access register) are defined for the purpose of cpu indirect access of internal memories.

These two registers are defined below:

Memory Upper Address Register

Name: mem_u_addr

Offset: 0x80

Access: Read/Write

Description

This register is used to store the upper address of the memory. It should be combined with the lower bits in mem_access register to have the complete address offset. The final memory address for each access is concatenated from mem_addr[20:11] from mem_u_addr register and mem_addr[10:1] from mem_access register.

Table 21 Memory Upper Address Register

Bits **Field**	**Name**	**Block** **(Access)** **Initial Value**	**Description**
31:25	rsv	0	Reserved
24:21	blk_sel_msb [3:0]	MEM (R/W) x	The value of this filed indicates which memory will be accessed for CPU to READ or WRITE mem_access register. The definition of memory access is described below: **0 - 1: EDRAM block #0 to #1**, **2 - 7: reserved**. **8: MIB counters**. **9: ARL**. **10: PQC**. **11:15: reserved**.
20:11	mem_addr [20:11]	MEM_IF (R/W) x	Upper address of memory access. The definition of each access is described in *blk_sel_msb[]* field above.
10:0	rsv	0	Reserved

MEMORY ACCESS Register

Name: mem_access

Offset: 0x84

Access: Read/Write

Description

After CPU writes this register, a memory access state machine, either Read or Write process, will be activated. This register is designed for all the device access including all the internal memories, and external MII's PHY register. Each access can only be 16 bits. For 32-bits memory data, it is in little endian format. So, the first two bytes are from bit[15:0] and the second two bytes of data with mem_addr[1] equal to 1 are from bit[31:16].

Table 22 Memory Access Register

Bits Field	Name	Access Initial Value	Description
31	access_rdy	(R/W) 0	When this bit is set on a read operation, it indicates that the mem_data[15:0] field is valid. For a write operation this bit gets set when the write operation is complete. This bit should always be written to 0 by CPU, and it is set when the read operation is complete or the data is written to the destination. Access_rdy is asserted when the command read or write is completed..
30	wr	(R/W) 0	This bit is set for Memory Write command.
29	rd	(R/W) 0	This bit is set for Memory Read command.
28:27	mem_sel	(R/W) 00	00 to select MII's PHY. 01 – reserved. 10 – reserved. 11 for memory access. Upper address bits are needed. Refer to mem_u_addr register for them.
26	reserved	0	Reserved

25:16	mem_addr [10:1]	(R/W) X	Memory access lower address bits. For each memory Write access, it is always 2 bytes that will be temporarily stored in a part of 16-byte register with the offset given by mem_addr[3:1]. The real memory operation won’t start until mem_addr[3:1] stands for the last access, i.e. equal to ‘e’. access_rdy won’t be set until receiving memory Write ack. For each 16-bytes of memory Write, there are eight Writes to this register, with bit [3:1] equal to 0 for the very first Write. After receiving the eighth Write with mem_addr[3:0] equal to ‘e’, the memory operation starts. For a 16-bytes of Read, PLB 2224 issues a command to request for 16-bytes of data from memory after receiving a Read command with mem_addr[3:1] equal to 0. CPU then polls this register until access_rdy is asserted, after which mem_data[] is valid. After detecting access_rdy, CPU reads the first two bytes. It continues to issue the write of Read command with mem_addr[3:1] issued as a continuous offset, and then polls for mem_data[]. At the second and subsquent polling, access_rdy should be always ready, because all the 16-bytes of data are acquired at the first command with mem_addr[3:1] equal to 0.
15:0	mem_data [15:0]	(R/W) x	Write data for Write command, and Read data for Read command.

4.2.1 Access to EDRAM

This access is for diagnostic purpose only.

256 bit (16 x16 bits) should be read or write per indirect access.

Unused address bits should be set to 0.

For a write, the sequence is as follows:

- Write memory upper address register first.
- Write 16 times memory access register with “wr” bit set. The address for the first write must be at 256-bit boundary. Then every write, address should be incremented by 1 word (2 bytes).
- Then read memory access register to check “access_bit” bit until “access_bit” is ‘1’. Next indirect access can be issued.

For Read, the Sequence is

- Write memory upper address register first.
- Write memory access register with “rd” bit set. The address for the write must be at 256-bit boundary.
- Then read memory access register to check “access_bit” bit until “access_bit” is ‘1’.
- Store the first 16-bytes.
- Then read another 16 times access register with “rd” bit set. Then every read address should be incremented by 1 word (2-bytes).
- Next indirect access can be issued.

4.2.2 Access to MIB Counters

32 bit (2 times 16 bits) should be read or written per indirect access. Unused address bits should be set to 0.

There are two parts in this area: MIB counters for non VLAN aware switch and VLAN MIB for VLAN aware switch. The selection is controlled by mem_addr[14].

Mem_addr[14]:

0: MIB

1: VLAN MIB

For Normal MIB Counter

Each counter has a unique address defined in mem_addr[12:2]. Mem_addr[12:2] consists of {cnt_sel,’0’, port_id[4:0], item_num[3:0]}.

cnt_sel = 1 for receiving counter, and 0 for transmitting counter.

Port_id[] is the logical port number for each Ethernet port or CPU port.

And item_num[] is defined below under the headingNetwork Management.

For VLAN MIB

Each VLAN has a 3 mib counters which are used to represent the Number of Unicast packets received, number of Multicast packets received and the number of packets dropped. Incase of Tx VLAN mibs this represents the number of packets transmitted for each category.

Each counter has a unique address defined in mem_addr[13:2]. *Mem_addr[13:2]* consists of *{cnt_sel, vlan_id[9:0]}*.

cnt_sel = 00 for ucast packet counter

01 for mcast packet counter

10 for number of packets that are dropped

- For a write, the sequence is as follows
- Write memory upper address register first.

- Write 2 times memory access register with “wr” bit set. The address for the first write must be at 32-bit boundary. Then every write, address should be incremented by 1 word (2 bytes).
- Then read memory access register to check “access_bit” bit until “access_bit” is ‘1’.
- Next indirect acccess can be issued.

For Read, the Sequence is

- Write memory upper address register first.
- Write memory access register with “rd” bit set. The address for the write must be at 32-bit boundary.
- Then read memory access register to check “access_bit” bit until “access_bit” is ‘1’.
- Store the first 16 bytes
- Then read another access register with “rd” bit set. Then every read address should be incremented by 1 word (2 bytes).
- Next indirect acccess can be issued.

Access to ARL Memory Area

128 bit (8 times 16 bits) should be read or write per indirect access.

Unused address bits should be set to 0.

There are four different memories that can be accessed in this area.

Mem_addr[19:18] is used to select one of the memory

00: to select SSRAM 20 k x 32

01: to select internal small ARL cache

The above two areas are for diagnostic purpose only.

10: to select vlan_ix table 4 k x 10

11:to select vlan_table 1 k x 40

For Write, the Sequence is as Follows

- Write memory upper address register first.
- Write 8 times, the memory access register with “wr” bit set. The address for the first write must be at 128 boundary. Then every write, address should be incremented by 1 word (2 bytes).
- Then read memory access register to check “access_bit” bit until “access_bit” is ‘1’.
- Next indirect acccess can be issued.

For Read, the Sequence is as Follows

- Write memory upper address register first.
- Write memory access register with “rd” bit set. The address for the write must be at 128 boundary.

- Then read memory access register to check "access_bit" bit until "access_bit" is '1'.
- Store the first 16 bytes
- Then read another 7 times access register with "rd" bit set. Then every read address should be incremented by 1 word (2 bytes).
- Next indirect acccess can be issued.

Since internal bus size is 32 bit. When accessing the vlan_ix table. Lower 10-bits per even number of reads or writes is valid. During odd number of reads or writes, the lower 10-bits are invalid.

For vlan_table which is 40-bit wide, every even number of read or write is valid andonly lower 8-bits in the case off odd number of read or write is valid.

Access to PQC Memory Area

32 bits (16-bits read twice) should be read or written per indirect access. Unused address bits should be set to 0.

There are seven parts in this area: PBL, Floodmap, BCASTQ0 to BCASTQ3

Mem_addr[17:15] is used to select one of the memory inside PQC.

000: to select PBL memory

001: to select internal FloodMap

010 – 011: reserved

100 – 111: to select BCASTQ0 to BCASTQ3 respectively

These areas except FloodMap are for diagnostic purpose only.

The read and write sequnce is same as access to MIB.

The Internal bus is also 32-bit wide. When access PBL. Lower 12-bit per even number of rd or write is valid. Odd number read or wrote is invalid. When access floodmap. even number and lower 12-bit of odd number of rd or write is valid.

When accessing BCAST. Lower 10 bit per even number of rd or write is valid. Odd number read or wrote is invalid.

4.2.2.1 VLAN Memories

Table 23 VLAN Memories

Description	Entry Size	Entry #	Total Size	Comment
FLOODMAP	4 B	1024	4096 B	Destination port list for each entry. This can be vlan_port_list in case of a VLAN aware switch and dst_port list in the case of non-VLAN aware switch.
VLAN Table	40	1 K	40 Kbits	Inputs vid_ix[9:0] Outputs {vid[11:0], port_egress_tagged[27:0]}

VID_IX Table	10	4 K	40 Kbits	Inputs: { vid[11:0] }
				Outputs : { vid_ix[9:0] }
				
				VLAN ID index will be referenced to one entry within the VLAN Table. Multiple VLAN's i.e. multiple vid[], may be assigned to one vid_ix[]

4.3 Network Management

There are statistic counters for each port in PLB 2224. Each counter is 32-bit wide. The on-chip SRAM to store the counter is also 32-bit wide. And it is reset to zero at each chip reset. The counters being supported in PLB 2224 are shown below. The offset of each counter is equal to (4 bytes x item_number). The base address of this register is defined in *Memory_Upper_Address* Register in the register list.

Receiving Counters

0. Receive good packet

1. Receive good octet
2. Receive good broadcast packet
3. Receive good multicast packet
4. Receive crcerr packet with 64 bytes or more, including alignment error.
5. Receive fragment (less than 512 bits) with good or bad CRC.
6. Receive oversized packet with good or bad CRC (1519 to 1536 bytes).
7. Receive jumbo packet greater than 1536 bytes.
8. Receive dropped packet due to lack of resource.
9. Receive packet with 64-bytes with good or bad CRC.
10.Receive packet with 65 to 127-bytes with good or bad CRC.
11.Receive packet with 128 to 255-bytes with good or bad CRC.
12.Receive packet with 256 to 511-bytes with good or bad CRC.
13.Receive packet with 512 to 1023-bytes with good or bad CRC.
14.Receive packet with 1024 to 1536-bytes with good or bad CRC.
15.Receive good pause packet

For the VLAN aware switch there are 3 MIB counters per VLAN, which indicate the number of unicast packets recevied, number of multicast packets received and the number of packets dropped

Transmitting Counters

0. Transmit packet

1. Transmit octet
2. Transmit broadcast packet
3. Transmit multicast packet
4. Transmit oversized packet (1519 to 1536 bytes)
5. Tranmit packet with 64 bytes
6. Tranmit packet with 65 to 127 bytes
7. Transmit packet with 128 to 255 bytes
8. Transmit packet with 256 to 511 bytes
9. Transmit packet with 512 to 1023 bytes
10.Transmit packet with 1024 to 1536 bytes

11.Late collision.
12.Collision
13.Transmit pause packet
14.Filter matched packet counter
15.Bridge filter counter: Packet received and being filtered without forwarding to any port

For VLAN aware switch there are 3 transmit MIB counters per VLAN, which indicate the number of unicast packets received, number of multicast packets received and number of packets dropped.

5 Operational Description

The PLB 2224 is a network-on-a-chip switch device. All the functions such as address learning, packet filtering, aging, and port monitoring are done by the PLB 2224 chip itself.

The CPU in the PLB 2224 switch performs the following operation, administration, and maintenance (OAM) functions:

1. Configure PLB 2224 chip and PLB 2224 switch through the console,
2. Access and manage MAC address tables,
3. Access MIB and RMON registers,
4. Provide an SNMP interface for NMS,
5. Provide a console through RS232 or telnet sessions,
6. Provide TFTP for software download and upgrade,
7. Process 802.1D Spanning Tree protocol BPDUs,
8. Process other packets delivered to the CPU port.Data FlowData Flow

5.1 Data Flow

A MAC bridge relays individual MAC user data frames between the separate MAC entities of the bridged LANs connected to its Ports. The order of frames is preserved.

Before leaving PLB 2224, frames coming into PLB 2224 will typical go through:

1. Frame Reception
2. Ingress Filtering
3. Frame Forwarding
4. Egress Filtering
5. Frame Transmission

A terminating frame will be forwarded to one of the CPU ports at the Frame Forwarding phase. A frame originated from the PLB 2224 will go through all the phases through an emulated MAC port (CMAC).

5.1.1 Frame Reception

Frame Reception performs the following functions:

1. Discard on receiving a frame in error,
2. Discard a frame if the non user_data_frame and request_with_no_response mac_action parameters are met,
3. Regenerate user priority, if required.

5.1.2 Ingress Filtering

Ingress Filtering performs the following functions:

1. Screen VLAN-tagged frames if necessary,
2. Filter frames based on ingress filtering rules.

5.1.3 Frame Forwarding

Frame Forwarding performs the following functions:

1. Enforcing (bridge) topology restrictions and legal frame sizes,
2. Filtering frames based on the destination MAC address and VLAN ID in the filtering database,
3. Default group filtering for the transmission port,
4. Queuing the frame based on traffic class,
5. Scheduling frames for transmission based on certain criteria,
6. Mapping priorities between user priority and outbound access priority,
7. Regenerating FCS in the case of frame payload change.

5.1.4 Egress Filtering

Egress Filtering filters out frames destined to members of a different VLAN.

5.1.5 Frame Transmission

Frame Transmission transmits frames.

5.1.6 Terminating or Originating Frames

A frame typically does not go to the CPU, unless one of the followings happened:

1. Receiving BPDUs or GVRP PDUs (in case of 802.1Q),
2. For Address Learning,
3. For other monitoring purpose.

In the case of a frame being forwarded to the CPU, different process will pick up and process the frame.

5.1.7 CPU Transmission

The CPU will originate frames for the following reasons:

1. Generating BPDUs or GVRP PDUs (in case of 802.1Q).
2. Programming the MAC address table.

The packet will follow the same data path as a regular frame through an emulated MAC port (CMAC).

5.2 Flow Control

The PLB 2224 supports flow control in both half and full duplex modes. Flow control in the half duplex mode is implemented by creating collisions on the link. The flow control in full duplex mode uses PAUSE frames as per the 802.3x specification.

Each port maintains three counters. The rx_cnt keeps track of the total received packets on a port that are in the switch (i.e., have still to be transmitted from at least one port). The txHQ_cnt and txLQ_cnt maintain the count of the number of packets in the HIGH and LOW priority Tx queues for the port. In addition, there are four Rx watermarks and two Tx watermarks per port. The default values of the watermarks may be changed by the CPU. See **"CPU Water Mark Register" on Page 175** through **"Gport RX Watermark Control Register" on Page 179** for details. **Figure 14** shows the watermarks.

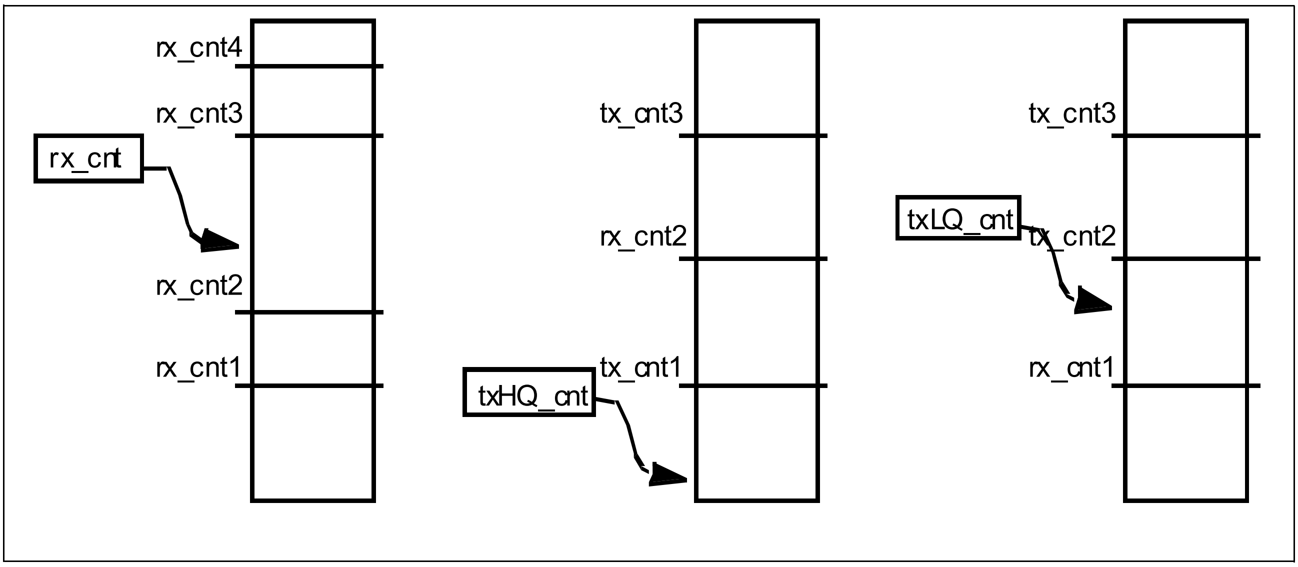


Figure 14 Flow Control Mechanism

When the rx_cnt equals rx_cnt2, flow control is asserted until the rx_cnt drops to rx_cnt1. If the rx_cnt continues to rise above rx_cnt2 and reaches rx_cnt3, the receiving port starts dropping all non-critical packets until rx_cnt drops below rx_cnt2. For example packets like BPDUs are those packets that have critical attribute set (critical packets). Refer to **"MA Table Entry Types" on Page 38** for details. If the rx_cnt continues to rise beyond rx_cnt3 and reaches rx_cnt4 then all packets, including critical packets at the receiving port are dropped until rx_cnt drops below rx_cnt3. The same discussion holds true for gigabit ports, however, the watermark values used are different.

The flow control mechanism is designed to prevent faulty links from hogging packet buffer resources and bringing down the system. Flow control is disabled upon reset and then depending on disable_pause bit and e_hd_cg_ctl bits in Chip configuration Register flow control is controlled. In addition, if the port is configured for full duplex operation, the appropriate *pause_e_cpu* bit in **"Port MII Register" on Page 157** must also be set to 1.

On the Tx side, identical count values and mechanism is used for the HIGH and LOW priority queues. When txHQ_cnt reaches tx_cnt2 non-critical packets are not appended to the TxHQ queue of the transmitting port, until the txHQ_cnt falls below tx_cnt1. If txHQ_cnt continues to rise and reaches tx_cnt3, then no packets are appended to the TxHQ until txHQ_cnt drops below tx_cnt2.

The same discussion holds true for gigabit and CPU ports, however, the watermark values used are different. This feature is disabled upon reset and must be enabled by setting the *en_txq_drop* (see **“Chip Configuration Register” on Page 141**, page 60) bit to 1 for the Ethernet ports. To enable this feature for the CPU ports, *en_cpu_txq_drop* (see **“Switch Configuration Register” on Page 145**) bit must be set to a 1.

5.3 Statistics Registers

There are statistic counters for each port in PLB 2224. Each counter is 32-bit wide. The on-chip SRAM to store the counter is also 32-bit wide. And it is reset to zero at each chip reset. Each Counter is at an 4x relative offset starting at the base address. The CPU can read these registers and all registers are cleared to zero upon reset.

Receiving Counters

Table 24 Rx Counters

Counter #	Counter Description
0	Number of good packets received
1	Number of good octets received
2	Number of good broadcast packets received
3	Number of good multicast packets received
4	Number of received packets with packet length => 64 bytes with CRC or alignment error.
5	Number of received fragments (packet size < 64 bytes) with good or bad CRC
6	Number of received oversized packet (length 1519 bytes to 1536 bytes) with good or bad CRC
7	Number of received jumbo packets (length > 1536 bytes)
8	Number of received packets dropped because of insufficient buffer resources
9	Number of packets received, length = 64 bytes, with good or bad CRC
10	Number of packets received, length > 64 bytes and < 128 bytes, with good or bad CRC
11	Number of packets received, length => 128 bytes and < 256 bytes, with good or bad CRC
12	Number of packets received, length => 256 bytes and < 512 bytes, with good or bad CRC
13	Number of packets received, length => 512 bytes and < 1024 bytes, with good or bad CRC
14	Number of packets received, length => 1024 bytes and <= 1536 bytes, with good or bad CRC
15	Number of good PAUSE packets received

Transmit Counters

Table 25 Tx Counters

Counter #	Counter Description
0	Number of packets transmitted
1	Number of octets transmitted
2	Number of broadcast packets transmitted
3	Number of multicast packets transmitted
4	Number of oversized (lenght => 1519 bytes and <= 1536 bytes) packets transmitted
5	Number of packets transmitted, length = 64 bytes
6	Number of packets transmitted, length > 64 bytes and < 128 bytes
7	Number of packets transmitted, length => 128 bytes and < 256 bytes
8	Number of packets transmitted, length => 256 bytes and < 512 bytes
9	Number of packets transmitted, length => 512 bytes and < 1024 bytes
10	Number of packets transmitted, length => 1024 bytes and <= 1536 bytes
11	Number of late collisions
12	Number of collisions
13	Number of pause frames transmitted
14	Reserved
15	Reserved

VLAN Counters

Table 26 VLAN Rx Counters (per VLAN)

Counter #	Counter Description
0	Number of unicast packets
1	Number of multicast packets
2	Number of packets dropped

Table 27 VLAN Tx Counters (per VLAN)

Counter #	Counter Description
0	Number of unicast packets

1	Number of multicast packets
2	Number of packets dropped

5.4 Packet Classification

The packet classification and filtering feature enables the PLB 2224 to identify patterns within the first 64 B of packet data and take a specific action if a match is found. Packet filtering allows identification of packets such as protocol type (i.e., an IP packet), TOS values within a certain range of the IP packet, and IGMP control packet. Once a packet is classified, one of the following actions can be taken:

- Increment a dedicated counter for management purposes,
- Forward the packet to a port designated for forwarding such packets. Typically the forwarding port is a CPU port, but it could also be some other port,
- Discard the packet, i.e., do not forward it to the normal destination port(s),
- Monitor the packet, i.e., forward it to the monitoring port,
- Assign HIGH priority to the packet.

The data pattern(s) used for identification and handling actions are configured globally and are shared by all ports on the PLB 2224. **Figure 15** shows how packet filtering is accomplished.

Several registers are used to implement packet classification and filtering. They are described in **Chapter 7**. These configuration registers are mapped into the CPU address space and can be loaded using the CPU interface.

The packet identification is done with the help of four groups of pattern registers. The groups are identified as Group #0 through #3, and pattern sets within a group are identified as pattern #0 through #3. Out of the 4 groups, pattern registers in group #0 and #1 are user configurable, while in group #2 and #3 they are hard coded to identify IP protocol packets in 802.3 SNAP and Ethernet II frame formats respectively. Groups #0 and #1 each consist of 4 pattern sets. Each set has a 16-bit pattern value, a 16-bit mask, 5-bit offset, and a flag for the type of comparison (LE or EQ).

When a packet is received by a port on the PLB 2224, the two bytes with the starting half-word (==2-bytes) address of offset[5:1] within the packet are compared with the data stored in pattern[15:0], and a match signal is generated when the results of the comparison are true. In doing the comparison, the pattern[15:0] is qualified by the corresponding mask[15:0]. A 1 at any position in the mask implies that the corresponding bit in the Ethernet packet and the pattern register are not being compared (i.e., they are considered matched).

The first pattern set (i.e., pattern set #0) in each group can be programmed to generate a match when the packet data is less than or equal to (LE) the pattern data. Setting the corresponding comp_le0 bit to a 1 does this. The other three pattern sets can only check if the Ethernet packet data is equal to the pattern. Thus, groups #0 and #1 generate four match signals each, called match0 through match3

The handling of the identified packets, i.e., the action(s) to be performed, is determined by a two-stage lookup process as follows:

The four match signals each from group #0 and #1 are used as a 4-bit address to do a lookup into a 16-entry x 2-bit table associated with each group. So groups #0 and #1 generates a 2-bit output, called opcode[1:0] as a result of the first look up. The hard-coded groups #2 and #3 generate 1-bit output called opcode0. The 6 opcode bits consisting of 2-bit output, op_code[1:0] from each of group#0 and #1 and 1-bit output op_code[0] from each of the hard coded groups #2 and #3, are used as an address to do the second lookup into a 64-entry x 5-bit table to generate five action signals. The action bits and the corresponding actions are listed below. The action signal is asserted when the corresponding bit is a 1.

- Bit 4 - inc_filter_cnt: increment Rx filter count. Statistics Register #14 in the Tx group is used for this purpose.
- Bit 3 - drop: tag the packet to be dropped, i.e., not sent to the normal intended destination port(s) based on the DA lookup.
- Bit 2 - filter_fwd_port: forward the packet to the designated forwarding port for such filtered packets
- Bit 1 - monitor: tag the packet for monitoring
- Bit 0 - priority: tag the packet as high priority

The filtering feature can be enabled on a per port basis, by setting the appropriate bit in **"Enable Filtering Register" on Page 192**. Also, if the filtering is enabled on any of the ports, the forwarding port_id, filter-fwd_port[4:0] in this register must also be specified.

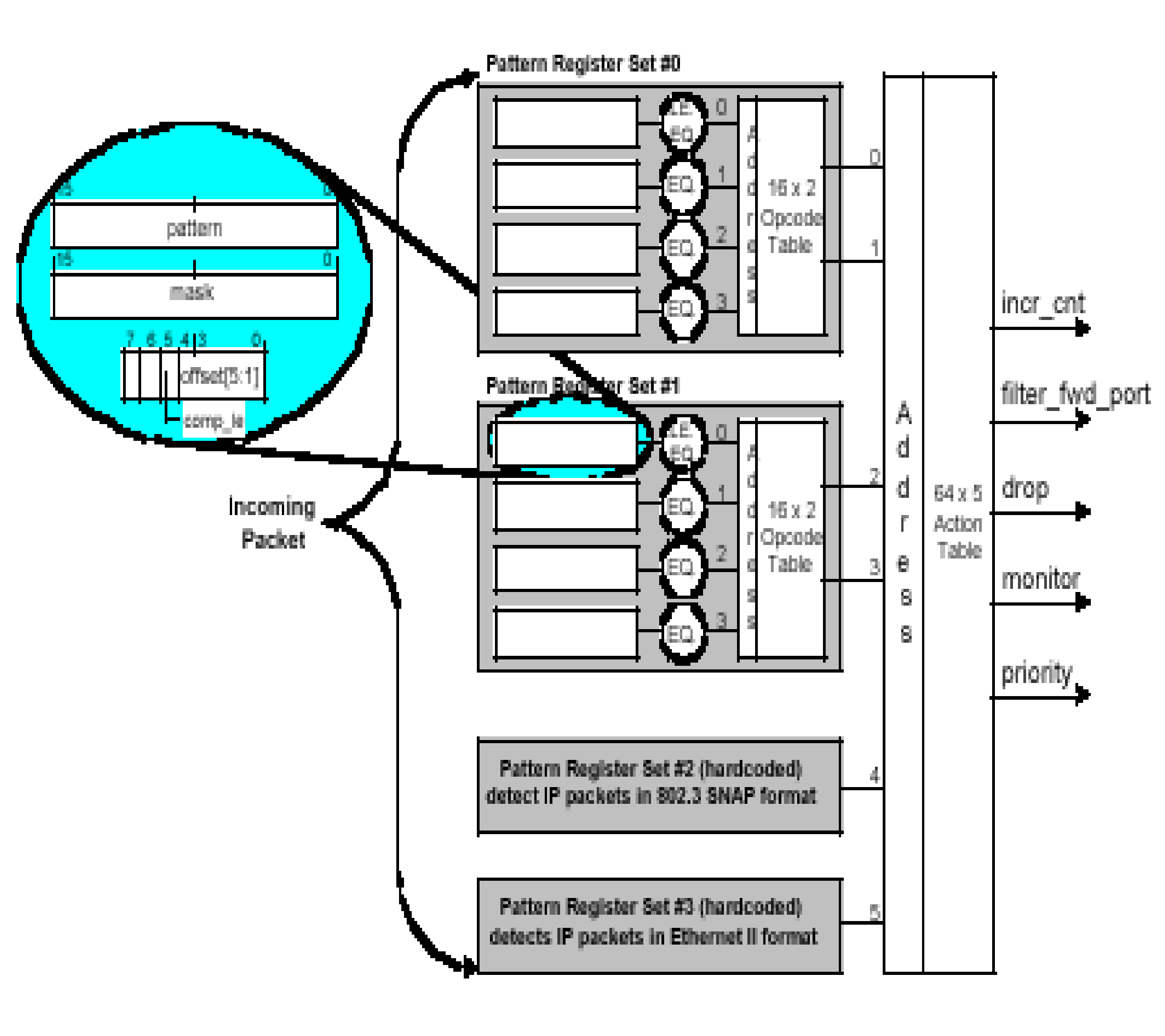


Figure 15 Packet Classification

5.4.1 Examples

The following example shows how the lookup tables are created to perform a typical filtering function.

5.4.1.1 Desired Filters and Actions

Detect Type of Service (TOS) on all the TCP/IP packets and assign them to different priority queues based on TOS value,

Count all the TCP/IP packets received at each port,

Forward all IGMP packets to the CPU.

5.4.1.2 Pattern Register Setup

Group #3 and Group #2 pattern_mask and offset registers are hardcoded as follows:

Table 28 Group[3] Set to Detect IP in Ethernet II Format

Pattern #	Pattern[15:0]	Mask[15:0]	Offset[5:1]	comp_le	Comment
15		0xFFFF			ignored
14		0xFFFF			ignored
13		0xFFFF			ignored
12	0x0800	0x0000	12	0	IP (v4) protocol in Ethernet II

Table 29 Group[2] Set to Detect IP in 802.3 SNAP

Pattern #	Pattern[15:0]	Mask[15:0]	Offset[5:1]	comp_le	Comment
11	0x0800	0x0000	20	0	
10	0x03XX	0x00FF	16	0	
9	0xAAAA	0x0000	14	0	
8	0x1500	0x0000	12	1	

CPU sets up Group #1 and Group #0 pattern_mask and offset registers as follows:

Table 30 Group[1] Used for IP Packets in Ethernet II Format

Pattern #	Pattern[15:0]	Mask[15:0]	Offset[5:1]	comp_le	Comment
11	0xXXXX	0xFFFF		0	ignored
10	0xXX02	0xFF00	22	0	Identify IGMP control frame
9	0xXX06	0xFFF0	22	0	Identify TCP protocol
8	0xXXX2	0xFFF0	16	1	Compare 3 TOS bits

Table 31 Group[0] -Used for IP packets in 802.3 SNAP Format

Pattern #	Pattern[15:0]	Mask[15:0]	Offset[5:1]	comp_le	Comment
7	0xXXXX	0xFFFF		0	Ignored
6	0xXX02	0xFF00	22	0	Identify IGMP control frame
5	0xXX06	0xFFF0	22	0	Identify TCP protocol
4	0xXXX2	0xFFF0	16	1	Compare 3 TOS bits

Opcode Table setup

Group #3 and Group #2 opcodes are hard coded as follows

Table 32 Group[3] -Op Code

Match[3:0]	Op Code[0]	Comment
xxx1	1	IP (v4) protocol
all other	0	non-IP frame

Group[2] - Op Code

1111	1	IP (v4) protocol
all other	0	non-IP frame

The CPU sets up Group #1 and Group #0 opcodes as follows:

Group[1] - Op Code

Match[3:0]	Op Code[1:0]	Comment
x010	11	TCP/IP packet, TOS > 2
x011	10	TCP/IP packet, TOS <= 2
x10x	01	IGMP control frame
all other	00	

Group[0] - Op Code

Match[3:0]	Op Code[1:0]	Comment
x010	11	TCP/IP packet, TOS > 2
x011	10	TCP/IP packet, TOS <= 2
x10x	01	IGMP control frame
all other	00	

Action Code Setup

The CPU sets up the action code table as follows:

Table 33 Action Code Table

Match[3:0]	Action[4:0]	Action Taken
1x11xx	00001	TCP packet; set to high priority
x1xx11	00001	TCP packets, TOS > 2
1x1xxx	10000	TCP/IP packet, increment filter count
x1xx1x	10000	TCP/IP packet, increment filter count
1x01xx	01100	IGMP packet, send to CPU instead of normal DA
x1xx01	01100	IGMP packet, send to CPU instead of normal DA
all other	00000	No special action

Finally, the Enable Filter register is set up to enable filtering on all ports and filter_fwd_port[4:0] is set to Port 26, i.e. the CPU port.

5.5 Packet Monitoring

Packets can be monitored based on the ingress port or egress port, if the monitor bit as described in Register 0 x 40, Port Monitor is set to 1. The id of the monitoring port (i.e., port to which all the monitored packets are sent) is specified in Register 0 x 08, Switch Configuration.

The PLB 2224 can also be configured to monitor:

1. Packets with CRC errors,
2. Packets whose source port number does not match with the port id value in the MA Table entry corresponding to the packet SA.

Setting the appropriate configuration bits in Register 0 x 08, Switch Configuration activates packet monitoring.

5.6 Packet Prioritization

The PLB 2224 architecture supports two priorities – HIGH and LOW per port. The priority can be based on the ingress port and/or DA. Two configuration bits, *pri_at_src* and *pri_at_da*, specified in **“Switch Configuration Register” on Page 145** are used to determine how packet priority is set.

When *pri_at_src* is set, the configuration register in **“Port Priority Register” on Page 158** is looked up to set the priority. When *pri_at_da* bit is set, the MA Table is used to set the priority.

When both *pri_at_src* and *pri_at_da* bits are set, the priority is set to HIGH if either lookup results in HIGH priority (i.e., the result is an OR function).

The following rules are used in determining DA based priority:

- For broadcast packets, priority is always set to LOW
- For unknown packets, priority is always set to LOW
- When MA Table entry corresponding to the DA is not a locked entry, the priority is always set to LOW.

5.7 Trunking

The PLB 2224 supports port trunking among like ports. Two, four or eight 10/100 ports can be grouped together to form a trunk. The two gigabit ports can also be grouped to create a trunk. There is no limit on the number of trunks.

Only consecutively numbered ports can be members of a given trunk. The possible trunk configurations are shown in **Table 34**. The PLB 2224 can be configured to use any combination of 8-Port, 4-Port and 2-Port trunks as long as the same physical port is not a part of two trunks. If any of the port(s) in a trunk group are disabled or have a link failure, the other ports in the group share the traffic.

It is the port with the next higher ID that would transmit the packets in case of failure of one of the ports in a trunk. But if the port that has failed has the highest port ID, then it will be the port with the lowest ID in the trunk group that would be transmitting the packets.

Setting the bits described in **"Port Trunk Register" on Page 159** enables trunking.

Table 34 Trunk Combination

Number	8-Port Trunks	4-Port Trunks	2-Port Trunks
0	Trunk8A	Trunk4A	Trunk2A
1			
2			Trunk2B
3			
4		Trunk4B	Trunk2C
5			
6			Trunk2D
7			
8	Trunk8B	Trunk4C	Trunk2E
9			
10			Trunk2F
11			
12		Trunk4D	Trunk2G
13			
14			Trunk2H
15			

Number	8-Port Trunks	4-Port Trunks	2-Port Trunks
16	Trunk8C	Trunk4E	Trunk2I
17			
18			Trunk2J
19			
20		Trunk4F	Trunk2K
21			
22			Trunk2L
23			
24 (Gport 0)			Gtrunk
25 (Gport 1)			

The two configuration bits, *trunk_da_based* and *trunk_sa_based* in the **"Switch Configuration Register" on Page 145** along with the number of member ports in the trunk is used to determine the physical port over which the packet is transmitted; the first port (i.e., the port with the lowest port number) is considered port 0.

For trunks containing 2, 4 or 8 ports, the least significant 1, 2 or 3 bits respectively, of the SA, or DA, or SA XORed with DA determines the physical port. For proper operation, the Port Trunk Register bits must be set correctly (i.e., they must result in complete, unambiguous and legal trunk definitions). Also, at least one of the bits *trunk_da_based* and *trunk_sa_based* must be set to a 1.

5.8 Address Learning

The Learning Process observes the source MAC Addresses of frames received on each Port and updates the Filtering Database conditionally on the state of the receiving Port. The VLAN ID associated with the frame is used to ensure that the address information is learned relative to the frame's VLAN.

Frames are submitted to the Learning Process by the ingress rules as described in the previous sections. The Learning Process can deduce the Port through which particular end stations in the bridged LAN can be reached by inspecting the source MAC Address field and VLAN ID of received frames. It records such information in the Filtering Database.

The process creates or updates a Dynamic Filtering Entry associated with the frame's VLAN ID, associating the reception Port with the source MAC Address, if the following conditions all apply:

1. The Port on which the frame was received is in a state that allows learning and,
2. The source address field of the frame denotes a specific end station, i.e., is not a group MAC Address and,
3. The resulting number of entries would not exceed the capacity of the Filtering Database, and
4. The Member set for the frame's VLAN ID includes at least one Port.

If the Filtering Database is already filled up to its capacity, where a new entry would otherwise be made, then an existing entry may be removed to make room for the new entry.

Since the learning is performed in the PLB 2224, this processing in the CPU is redundant.

However, the CPU keeps a copy of the MAC addresses for console or other management entities. A copy of the MAC address table is stored and maintained by the CPU. All the sorting, VLAN association, and port association is stored. Address aging is performed through the CPU in coordination with the same function performed in the PLB 2224.

5.9 MAC Address Table and Filtering Table

The MAC address table is used by the Address Resolution Logic to determine the destination ports of the received frame. The MAC addresses are either learned from the Address Learning process or statically configured by the management interface. The addresses are part of the filtering database that is used by the ARL (ingress filtering, frame forwarding, and egress filtering).

5.9.1 Accessing the Address Table

The MAC address table in PLB 2224 can be accessed through the console or other management entities via API's. The following API's are provided by the MAC address:

1. MAC address matching,
2. List all MAC address sorted by MAC address,
3. List MAC address by port number in ascending MAC address order,
4. List MAC address by VLAN in ascending MAC address order.

5.9.2 MAC Table Entry Formats

The format of the unicast/multicast MAC address table entry is.

Table 35 MA Entry Formats for Unicast and Multicast Addresses

Bit Fields	Name	Description
Unicast		
62	Unicast	Denotes whether the entry is Unicast or Multicast
62:57	rsv	
56:52	port_id[4:0]	Index to VLAN/Floodmap for port list lookup
51:48	ma_state[3:0]	Denotes age; plus criticality, priority and lock attributes
47:0	MA[47:0]	MAC address
Multicast		
62	Multicast	Denotes whether the entry is Unicast or Multicast
61:52	flood_ix	Index to VLAN/Floodmap for port list lookup For Multicast entries the number of entries supported is 256 (address 0 to Adress 255)
51:48	ma_state[3:0]	Denotes age; plus criticality, priority and lock attributes
47:0	MA[47:0]	MAC address

5.9.3 Storage of the MAC Tables

The MAC address table is stored in the SRAM controlled by PLB 2224. The CPU can access the MAC address table through indirect addressing by programming the memory access register. Once stored, this table is updated for each new address learned or address change. The table stored in the CPU is the primary source for console or other management entities to access.

5.9.4 Forwarding Frames

To forward frames to one of the two CPU ports, specific MAC addresses (or filters) have to be programmed into the MAC address table (or the filtering database).

The CPU can send a frame to PLB 2224 (CMAC) with the values of specific MAC addresses assigned to the source MAC address field. Before sending it to the PLB 2224, the CPU also needs to program the PLB 2224 ARL to assert the learning option.

5.9.4.1 Frame Forwarding Example with SNMP

An ARP request coming into the port will be broadcast to all ports in the same VLAN (including a CPU port). Since this embedded SNMP is destined for the CPU's IP stack, an ARP response is generated by the ARP module, replying to the ARP request with the MAC address of the switch as the source MAC address. The source MAC address in the frame will be learned and an entry is created.

5.9.4.2 Using the CPU

A MAC address can be assigned by the CPU for each and every port. Then there is no need for a MAC entry in the MAC address table for this purpose. The Filter table is programmed through the configuration of PLB 2224 registers. There are four filters in the PLB 2224 filter table. These four filters can be used to filter (discard) frames, redirect frames to the CPU or to a specific port. One of the examples is to have IP Multicast redirect to the CPU. The filter examines the IP protocol field for frames of the IGMP protocol type. The IGMP packets will be handled by CPU's IP Multicasting process (if supported). IGMP packets originated from the IP Multicasting process in the CPU will be delivered to CMAC without ARL to prevent the packets coming back to the CPU again.

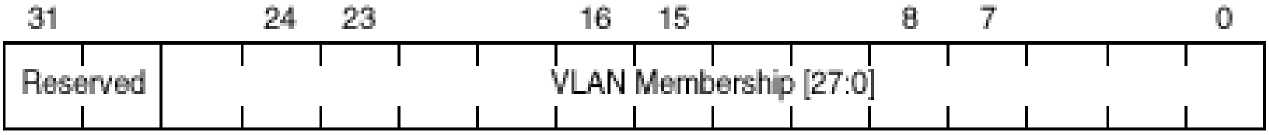
5.10 VLAN

5.10.1 VLAN Introduction

The PLB 2224 supports the VLAN feature. This implementation compliant to 802.1d supports a total of 1024 port-based/tag-based VLANs. VLANs facilitate the easy administration of logical groups of stations that can communicate as if they were on the same LAN. They also facilitate easier administration of members of these groups (moving, adding, and changing).

The VLAN definitions, i.e., the port list comprising a VLAN, are stored in an on-chip table that has 1023 32-bit wide entries. Within each entry, the VLAN membership is specified by setting the corresponding bit for each of the member ports to a 1. The format of the VLAN table entry is shown in **Table 36**.

Table 36 VLAN Table Entry Format



31	24	23	16	15	8	7	0
Reserved		VLAN Membership [27:0]					

5.10.2 VLAN Configuration

Traffic between VLANs is restricted. Bridges forward unicast, multicast, and broadcast traffic only on LAN segments that serve the VLAN to which the traffic belongs. VLANs maintain compatibility with existing bridges and end stations. If all bridge ports are configured to transmit and receive untagged frames, bridges will work in plug-and-play ISO/IEC 15802-3 mode. End stations will be able to communicate throughout the bridged LAN.

A MAC bridge that conforms to this standard:

1. Conforms to the requirements of ISO/IEC 15802-3, as modified by the provisions of this standard,
2. Relays and filters frames,
3. On each Port, supports at least one of the permissible values for the Acceptable Frame Types parameter,
4. Supports the following on each Port that supports untagged and priority-tagged frames:
   a) A Port VLAN Identifier (PVID) value,
   b) The ability to configure at least one VLAN whose untagged set includes that Port,
   c) Configuration of the PVID value via management operations,
   d) Configuration of Static Filtering Entries via management operations.
5. Supports the ability to insert tag headers into, modify tag headers in, and remove tag headers from relayed frames,

6. Supports the ability to perform automatic configuration and management of VLAN topology information by means of Generic Attribute Registration Protocol (GARP) VLAN Registration Protocol (GVRP) on all Ports,
7. Supports the ability for the Filtering Database to contain static and dynamic configuration information for at least one VLAN, by means of Static and Dynamic VLAN Registration Entries,

5.10.3 VLAN Operation

5.10.3.1 VLAN Memories

Table 37 VLAN Memories

Description	Entry Size	Entry #	Total Size	
FLOODMAP	4 bytes	1024	4096 bytes	Destination port list for each entry. This can be vlan_port_list in case of a VLAN aware switch r dst_port list in the case of non-VLAN aware switch.
VLAN Table	40-bits	1 K	40 Kbits	Inputs vid_ix[9:0] Outputs {vid[11:0], port_egress_tagged[27:0]}

VID_IX Table	10-bits	4 K	40 Kbits	Inputs: { vid[11:0] }
				Outputs : { vid_ix[9:0] }
				
				VLAN ID index will be referenced to one entry within the VLAN Table. Multiple VLAN's i.e. multiple vid[], may be assigned to one vid_ix[]

Floodmap memory stores the destination port list for each MA's or VLAN membership. For a non-single destination packet the flood_ix[] index from DA lookup result will be used to reference Floodmap for a destination port list. Lower 28-bits of the floodmap stands for the respective port for the destination. In case of a VLAN aware switch, Floodmap table indicates the membership ports belonging to a particular VLAN. In this case the input lookup is vid_ix. This is used to filter non-member ports at receiving and transmitting.

Registers that are to be used for VLAN operation apart from the above memories

- Tag_pri_table (0 x D4)
- e_pri_table (0 x D8)
- port_ix_register (0 x 60 , 0 x 78)
- vlan_aware / admit tagged only (0 x E0)

- Ingress filter enable (0 x EC)
- Security enable (*e_ucast_secu* bit 0 x 08)

5.10.4 VLAN Ingress Filter

Name:VLAN Ingress filter

Offset:0 x EC

Access:Read/Write

Description:Enables the Ingress filtering parameter on a per port basis.

Table 38 VLAN Ingress Filter

Bit fields	Name	Block (Access) Initial Value	Description
31:28	Rsv	0	
27:0	Ingress Filter [27:0]	(R/W) 0	0:Disable ingress filtering 1:Enable Ingress filtering An Enable Ingress Filtering parameter is associated with each Port. If the Enable Ingress Filtering parameter for a given Port is set, the ingress rules (8.6) as mentioned in 802.1q document, shall discard any frame received on that Port whose VLAN classification does not include that Port in its Member set

An Enable Ingress Filtering parameter is associated with each Port. If the Enable Ingress Filtering parameter for a given Port is set, the ingress rules shall discard any frame received on that Port whose VLAN classification does not include that Port in its Member set. If the parameter is reset for that Port, the ingress rules shall not discard frames received on that Port on the basis of their VLAN classification.The default value for this parameter is reset, i.e., Disable Ingress Filtering, for all Ports. The value of this parameter may be configured by means of the management operations, if management operations are supported by the implementation. PLB 2224 supports the ability to enable Ingress Filtering on any Port, and also supports the ability to disable Ingress Filtering on those Ports.

5.10.5 VLAN Tag Conversion

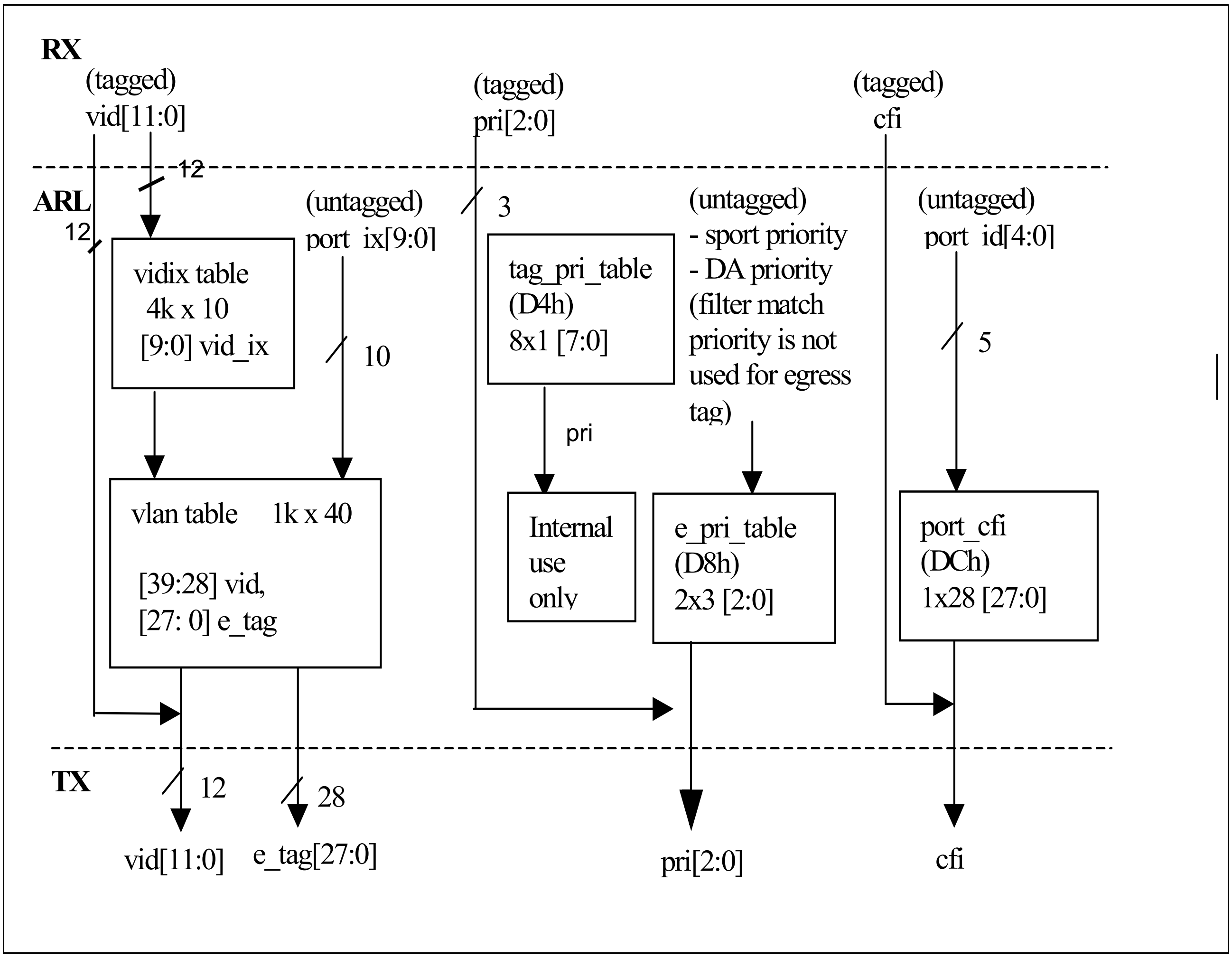


Figure 16 VLAN Tag Conversion

The received frames are checked for the VLAN tag in the RX module. VLAN tagging, untagging and qualification based on VLAN membership are performed only when VLAN aware bit in VLAN aware register is set. When the VLAN aware bit is set to ’0’, PLB 2224 treats all the frames as pure MAC frames.

If the ingress packet is tagged and its size is between 64 bytes and 67 bytes, VLAN padding is performed by TX, before the frame is sent out. For FE port this is done by default. For Gport the bit 22 in g_rxtx is set (Registers 0 x C0 / 0 x E0). This helps in making sure that the egress packet size is guranteed a minimum size of 64 bytes. (The padded data is between the Data field and the CRC field and has a value equal to “0”).

vid_ix table and *vlan_table* is configured by CPU through indirect access, as already explained under the *data structures* section.

5.10.6 VLAN Security

Bit 7 (*e_ucast_sec*) of Switch Configuration register (0x08) is the VLAN security bit for enabling the security features for unicast packets. The packet is dropped if the destination port is not the same VLAN as the one associated with the packet. If this bit is set to '0' the security check is disabled.

5.10.6.1 VLAN Membership/Port List Determination

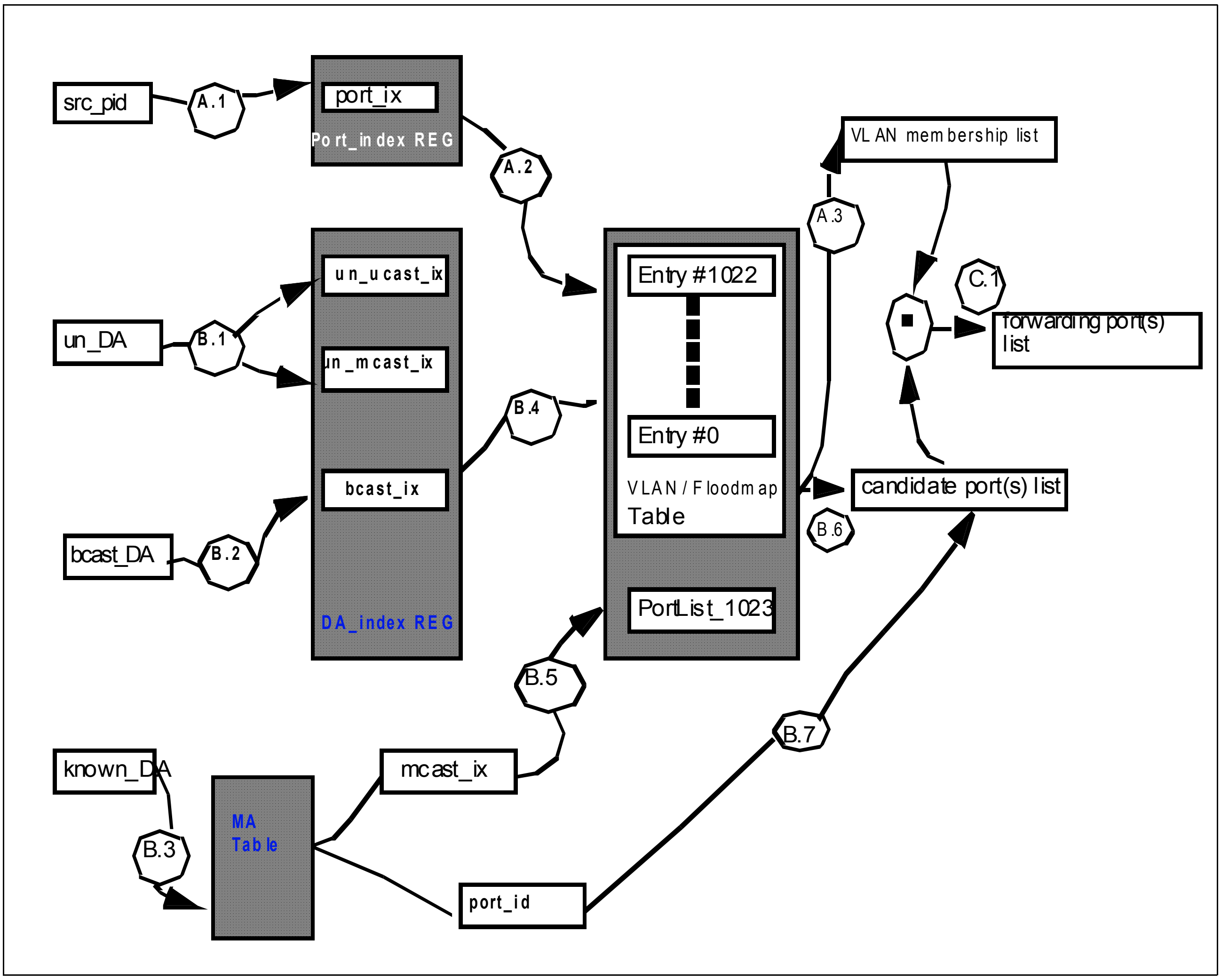


Figure 17 Address Resolution/Destination Port List Creation (VLAN Enabled)

VLAN membership is applied to qualify packet forwarding list so that each packet does not cross VLAN boundary.

First the switch generates candidate forwarding port list [27:0] for each packet. Depending on DA of received packet, switch refers to DA_index reg (0x7C) or MA table, and VLAN/floodmap table, and port list [27:0] is prepared. Last entry for VLAN/floodmap table is 1023, which is not on the memory of this table, instead the portlist1023_reg (0x58) is used.

PLB 2224 also generates the VLAN membership for each packet [27:0]. Depending on whether the ingress packet is tagged, vid >0 and vlan_aware='1', the switch refers to vidix table or port_ix reg (0x60, 0x78) and the VLAN/floodmap table. Then the VLAN membership [27:0] is prepared. The floodmap memory is shared by the VLAN membership information and the floodmap information.

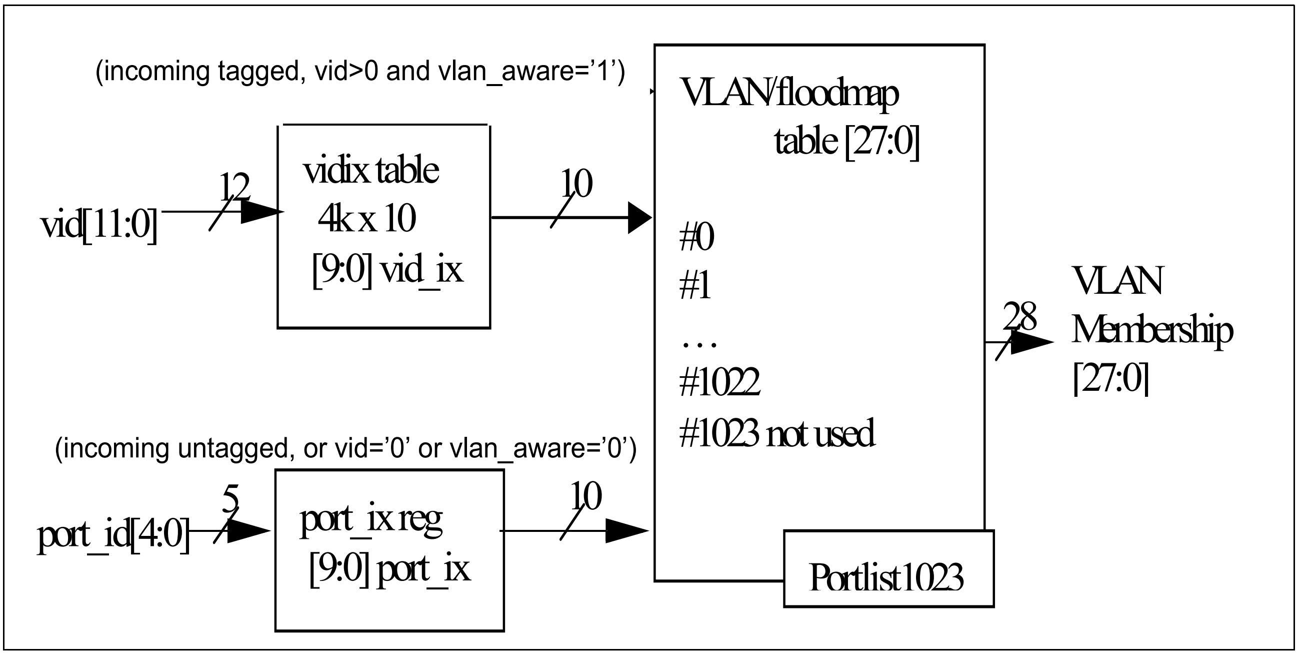


Figure 18 VLAN Membership Determination

Then an AND function is performed between the candidate port list [27:0] and the VLAN membership [27:0]. The switch also checks if the source port is a member of the VLAN. The table below shows the qualification of various packets. The description of the legend **A**/**B**/**C**/**D** in the table is given below the table.

Table 39 Qualification of Packets

Ingress Tagged and vid>0	VLAN Aware	Admit Tagged Only Per Port Basis	Ingress Filter Enable Per Port Basis	Unicast Security Enable	Known Unicast	Unkown Unicast, Known mcast, Unkown mcast, Broadcast
0	0	0/1	0/1	0	A	B
				1	B	B
	1	0		0	A	B
				1	B	B
		1		0/1	D	D
1	0	0/1		0	A	B
				1	B	B
	1		0	0/1	B	B
			1		C	C

Legend

A : No Qualification

B : qualified by destination port (AND function)

C : qualified by destination port (AND function) and source port (membership check).

D : packet is dropped.

An Enable Ingress Filtering parameter is associated with each Port. If the Enable Ingress Filtering parameter for a given Port is set, the ingress rules shall discard any frame received on that Port whose VLAN classification does not include that Port in its Member set. If the parameter is reset for that Port, the ingress rules shall not discard frames received on that Port on the basis of their VLAN classification.The default value for this parameter is reset, i.e., Disable Ingress Filtering, for all Ports. The value of this parameter may be configured by means of the management operations, if management operations are supported by the implementation. PLB 2224 supports the ability to enable Ingress Filtering on any Port, and also supports the ability to disable Ingress Filtering on those Ports.

If the vlan_identifier parameter carried in a received data indication is equal to the null VLAN ID and the Acceptable Frame Types parameter for the Port through which the frame was received is set tothe value *Admit Only VLAN-tagged frames*, then the frame shall be discarded.

Each frame received by a VLAN Bridge shall be classified as belonging to exactly one VLAN by associating a VID value with the received frame. The classification is achieved as follows:

a) If the vlan_identifier parameter carried in a received data indication is the null VLAN ID, then

1. If the implementation supports further VLAN classification rules in addition to Port-based classification and if the application of these rules associates a non-null VID value with the frame, then that VID value is used.
2. If the Enable Ingress Filtering parameter for the Port through which the frame was received is set, and if the Port is not in the Member set for the frame's VLAN classification, then the frame is dis-carded.

Associated with each Port of a VLAN Bridge is an Acceptable Frame Types parameter that controls the reception of VLAN-tagged and non VLAN-tagged frames on that Port. Valid values for this parameter are:

a) *Admit Only VLAN-tagged frames*;
b) *Admit All Frames*.

If this parameter is set to *Admit Only VLAN-tagged frames*, any frames received on that Port that carry no VID (i.e., untagged frames or priority-tagged frames) are discarded by the ingress rules. Frames that are not discarded as a result of this parameter value are classified and processed according to the ingress rules that apply to that Port.

5.11 Spanning Tree

5.11.1 Description of Spanning Tree Support

PLB 2224 supports multiple VLANs. Each VLAN operates over a single Spanning Tree.

All bridges within a bridged LAN infrastructure participate in a single Spanning Tree.

The primary goals of Spanning Tree are as follows:

1. 1. Elimination of loops in a bridged infrastructure,
2. 2. Improved scalability in a large network,
3. 3. Provision of redundant paths, which can be activated upon failure.

5.11.1.1 Spanning Tree Topologies

A Spanning Tree formed in a VLAN environment need not be identical to the topology of the VLAN(s). All VLANs are aligned along the Spanning Tree from which they are formed; a given VLAN is defined by a subset of the topology of the Spanning Tree upon which it operates.

The topology of a VLAN is dynamic. The structure of the VLAN may change due to new devices requesting or releasing the services available via the VLAN. The dynamic nature of VLANs has the advantages of flexibility and bandwidth conservation, at the cost of network management complexity.

PLB 2224 supports up to 1K VLANs, each maps to a single Spanning Tree. In addition, PLB 2224 supports port-based VLAN and active VLAN's. Each port can be configured as a separate VLAN or multiple PLB 2224 ports can be configured as one VLAN. The port-based VLAN is configured through PLB 2224 port register configuration.

The state of each port in the VLAN is derived from the result of the Spanning Tree algorithm, with the resulting port state for each port written into PLB 2224 port register.

Each of the ports in a VLAN can be in the following state:

1. Disabling/Blocking
2. Listening
3. Learning
4. Forwarding

The addition or removal of tag headers by a VLAN-aware bridge is performed only on frames submitted to the relay function of the bridge that are potentially to be forwarded on other Ports. BPDUs are forwarded to one of the CPU ports in PLB 2224. Forwarding is configured through PLB 2224 port register configuration. GARP PDUs destined for any GARP applications are forwarded or filtered depending upon whether the bridge supports the application concerned.

5.11.2 Spanning Tree Protocol Support

The PLB 2224 supports the Spanning Tree Protocol as per the 802.1D specification. As specified in Registers 0x50 - 0x54, **“Port Bridge State Register” on Page 160** each port has two configuration bits that define the bridging state for that port. For the Ethernet (i.e., non-CPU) ports, the following actions are performed, depending on the state of the port. Upon reset, each port is in the default forwarding state (2’b11). Depending on the state, the non-transmitted packets are automatically purged from the Tx queue.

Table 40 Spanning Tree States and Actions

Port State[1:0]	Receive Packets	Learn SA	Lookup DA	Transmit Packets
00 - Disabled	No	No	No	No
01 - Listening	Yes	No	Yes	Only those with critical attributes (treated as a blocking state with only CPU being able to tranmit packets)
10 - Learning	Yes	Yes	Yes	Only those with critical attributes
11 - Forwarding	Yes	Yes	Yes	All packets

5.12 SNMP and RMON MIBs

SNMPv1 is implemented in the PLB 2224 switch as part of the TMS software. The PLB 2224 MIB variables are accessible through the Simple Network Management Protocol (SNMP), which is an application-layer protocol designed to facilitate the exchange of management information between network devices. The SNMP system consists of three parts: SNMP manager, SNMP agent, and MIB.

Instead of defining a large set of commands, SNMP places all operations in a get-request, get-next-request, and set-request format. For example, an SNMP manager can get a value from an SNMP agent or store a value into that SNMP agent. The SNMP manager can be part of a network management system (NMS), and the SNMP agent can reside on a networking device such as a router. The SNMP agent can respond to MIB-related queries being sent by the NMS.

PLB 2224 maintains a set of MIB counters, one for each port in its internal registers.

These counters are classified as transmit & receive counters:

Table 41 SNMP Receiving Counters

Counter No.	Description of the Contents
0	Number of good packets received
1	Number of good octets received
2	Number of good broadcast packets received
3	Number of good multicast packets received
4	Number of packets (>64 bytes) received with CRC error, including the allignment error
5	Number of Fragments received (< 512 bits) with good or bad CRC
6	Number of oversized packets received with good or bad CRC (1519 to 1536 bytes)
7	Number of packets received with size greater than 1536 bytes
8	Number of packets dropped due to lack of resources
9	Number of packets received with size of 64 bytes (with good or bad CRC)
10	Number of packets received with sizes from 65 bytes to 127 bytes (with good or bad CRC)
11	Number of packets received with sizes from 128 bytes to 255 bytes (with good or bad CRC)
12	Number of packets received with sizes from 256 bytes to 511 bytes (with good or bad CRC)
13	Number of packets received with sizes from 512 bytes to 1023 bytes (with good or bad CRC)
14	Number of packets received with sizes from 1024 bytes to 1536 bytes (with good or bad CRC)
15	Receive good pause frame packet

Table 42 SNMP Transmitting Counters

Counter No.	Description of the Contents
0	Number of good packets transmitted
1	Number of good octets transmitted
2	Number of good broadcast packets transmitted
3	Number of good multicast packets transmitted

Counter No.	Description of the Contents
4	Oversized packet (1519 to 1536 bytes) transmitted
5	Number of packets of size 64-bytes transmitted
6	Number of packets transmitted with sizes from 65 bytes to 127 bytes
7	Number of packets transmitted with sizes from 128 bytes to 255 bytes
8	Number of packets transmitted with sizes from 256 bytes to 511 bytes
9	Number of packets transmitted with sizes from 512 bytes to 1023 bytes
10	Number of packets transmitted with sizes from 1024 bytes to 1536 bytes
11	Packets experiencing late collision
12	Packets experiencing collision
13	Number of packets transmitted with pause frame
14	Filter matched packet counter
15	Reserved

The access to these MIB counters is through register access as described in the section **"Indirect Access to the Memories" on Page 62**.

There are 3 MIB counters for VLAN aware switch for each of transmit & receive section

Receive VLAN MIB: Number of Unicast packets received, number of multicast packets received and number of packets dropped

Transmit VLAN MIB: Number of Unicast packets transmitted, number of multicast packets transmitted and the number of packets dropped

5.13 LED

There are four LED's for each of the 26 ports in PLB 2224. Each of the four LED's represents a different meaning based on the LED display mode which are classified as *Mode 0* and *Mode 1*. The table below explains the LED configurations.

- *Mode 0* is a speed based display mode. In this mode, 10 Mbit/s and 100 Mbit/s links use a different LED.
- *Mode 1* is feature based display mode. In this mode, LED0 selects link speed and LED1 indicates link activities.

The LED activities also depend on the chip LED configurations. The related LED configuration bits in the chip configuration register are: mask_coll_led, en_tx_led, and led_mode.

The individual LED's meaning is determined by the led_mode. The interpretation of LED's flashing activities considers whether collision is masked out for LED activities (mask_coll_led) or packet transmitting is configured to have LED activities (en_tx_led). These two work together to determine the actual interpretation of flashing activities.

Table 43 LED Configurations

LED		**Mode 0** **(Speed Based)**	**Mode 1** **(Feature Based)**
LED0	ON	10 Mbit/s	100 Mbit/s link
	FLASHING	10 Mbit/s link activity	-
	OFF	10 Mbit/s link Bad	10 Mbit/s link
LED1	ON	100 Mbit/s	No activity
	FLASHING	100 Mbit/s link activity	-
	OFF	100 Mbit/s link Bad	10 Mbit/s link
LED2	ON	Full Duplex	
	OFF	Half Duplex	
LED3	ON	Port Diabled	
	FLASHING	CRC error, Oversize packet, drop received packet, MAC address update or late collision	
	OFF	10 Mbit/s link Bad	10 Mbit/s link

In addition PLB 2224 offers serial LED interface. This provides the user the flexibility of having an external logic to latch and display the data. The Serial LED interface or the normal Matrix LED mode of operation can be selected on power-on through powerstrap of LED_COL[7] pin. For Serial LED Interface the LED_COL[7] pin is pulled Low.

The following pins are used in serial mode of operation

LED_ROW_N[1]	:	Load or Data Strobe
LED_ROW_N[2]	:	Serial Data Out
LED_ROW_N[3]	:	Clock

The sequence of data flowing out of the Serial Data Out line is in the following order

Port0 Row0, Port0 Row1, Port0 Row2, Port0 Row3, Port1 Row0, Port1 Row2, ... Port27 row0, Port27 Row1, Port27 Row2, Port27 Row3.

Once the last set of data is out, Data strobe signal is issued on LED_ROW_N[1].

All the LED's have an internal pullup (2 K), which is used to prevent open inputs when the chip is reset. During the Reset phase the values on the LED pins are pinstrapped. If a low value has to be pinstrapped, the value of the pull down resistor is K.

6 Interface Description

6.1 External Interfaces

The PLB 2224 interfaces with the following external devices. They are briefly described in this section; the detailed description is given later in the document.

6.1.1 10/100 PHYs

The PLB 2224 has twenty-four 10/100 ports. The PLB 2224 connects to either three octal or six quad PHY devices that support the SMII interface.

6.1.2 Gigabit PHYs

There are two Gigabit-Ethernet ports on the PLB 2224. These ports connect to gigabit PHY devices using either the GMII or the TBI interface.

6.1.3 MDIO Interface

The MDIO interface is compliant with the MII specification and is used to communicate with the PHY devices. All 10/100 Ethernet and Gigabit Ethernet ports share this interface. Using MDIO interface, the CPU can also access the PHY registers.

6.1.4 LED Interface

The PLB 2224 provides LED interface signals that can be used to indicate port status. This interface provides information on link status; speed, duplex and activity can be displayed using the LED signals. Only external drivers (i.e., no decode logic) are needed to drive the LED. Serial LED interface is also supported by PLB 2224.

6.1.5 JTAG Interface

Standard JTAG interface is provided for board-level testing

6.1.6 Clock / Reset Interface

The PLB 2224 requires two clock sources:

100 MHz system clock

125 MHz clock for the SMII and Gigabit interfaces

The PLB 2224 internally generates the clocks required for the internal logic and CPU logic. The PCI Interface uses the PCI Clock which runs at 33 MHz. A single reset signal (RESET_N) is shared by all functions on the PLB 2224.

6.2 CPU and EEPROM Interface

6.2.1 CPU Interface

The PLB 2224 has three options for connecting a CPU subsystem:

- PCI bus - a 32-bit PCI bus (v2.1 compliant) operating at up to 33 MHz,
- Generic Processor bus - a 32-bit data bus operating at up to 33 MHz. CPUs with multiplexed address/data as well as de-multiplexed address/data are supported,
- 2-wire serial bus - for connecting an I^{2}C compatible master mode processor. The bus can operate at up to 3.3 MHz.

The use of the CPU port is optional and is typically required for managed systems or systems with uplink (ATM, FDDI, and DSL) capability. Only one of the three interfaces can be used in the system at one time.

The CPU port can be used to access internal configuration, RMON registers and memory. In addition monitored port traffic, unknown packets and other special packets (uplink) can be directed to this CPU port for further processing.

6.2.2 I^{2}C Interface

An I^{2}C compatible EEPROM is required for configuring the PLB 2224 when a CPU is not present in the system. The PLB 2224 interfaces with the EEPROM using the 2-wire serial bus operating in master mode.

The key features of these three buses are described in the following paragraphs.

6.2.2.1 The PCI Local Bus Interface

The PCI Local Bus is a high performance 32-bit fully synchronous bus with multiplexed address and data lines. The bus is intended for use as an interconnect mechanism between highly integrated peripheral components, peripheral add-in boards, and processor/memory systems. Some of the performance features include:

- High-bus bandwidth, 132 Mbytes/s burst memory transfer on a 33 MHz bus,
- PCI bus bridges give add-on PCI bus masters a high bandwidth path to main memory,
- Cache and exclusive access support,

Typical PCI Applications

Typical PCI applications are add-on boards that require high-speed memory access, including LAN adapters, graphic adapters, hard drive controllers and SCSI cards. Using the PCI bus allows system designers to implement system critical components on a high bandwidth bus using low cost components so enhancing system price and performance.

There are three main types of devices that operate on the PCI bus:

- PCI Bus/System bridge: This Interfaces the PCI bus to the system processor and main memory. This device can act as a PCI master and arbitrate for systems that allow multiple bus masters,
- PCI bus add-on masters: Add-ons are devices that can operate the bus and may need access to other PCI add-ons or main memory on the system,
- PCI Bus Target-only add-ons: These are add-on devices that can only operate as targets. These devices respond to but do not initiate bus cycles.

The current implementation of the CPUIF supports only 33 MHz , 32-bit, target-only PCI Local Bus for 3.3 V operations.

6.2.2.2 The Generic Interface

The GENERIC interface is a simple asynchronous bus, similar to those used in systems or processors of conventional Motorola and Intel products. It supports either multiplexed or separated address and data buses.

It also can support separated read and write control lines or combined read/write control line with a data strobe signal. A programmable number of wait cycles can be configured for delaying the activation of the ready signal to the host processor.

The I^{2}C-Bus Interface

The I^{2}C bus is a 2-way, 2-line serial communication bus between different ICs or modules. The two lines are a serial data line (SDA) and a serial clock line (SCL). It was developed Philips.

On an I^{2}C bus, each device is recognized by a unique address and can operate as either a receiver-only device (for example, an LCD driver) or a transmitter with the capability to both receive and send information (such as memory).

Transmitters and/or receivers can operate in two modes, depending on whether the chip has to initiate a data transfer or is only addressed. These modes are:

- master mode,
- slave mode.

I^{2}C is a multi-master bus, i.e. it can be controlled by more than one IC connected to it.

The current implementation of I^{2}C interface supports a 7-bit address I^{2}C bus with two data transfer rates: 100 Kbits/s and 3.3 Mbits/s. The operating modes include:

- the master mode for reading the EEPROM data (511 data bytes and one check-sum byte),
- the slave transmit and receive mode for interacting with an external CPU connected on the I^{2}C bus.

Input filtering on both SCL and SDA lines are provided to suppress any pulse spikes.

Correspondence between PCI/Generic and I^{2}C Interfaces

The description of the PCI Interface signals, Generic Interface signals and the I^{2}C interface signals.

Table 44 Correspondence between PCI/Generic and IIC Interface

	PCI Interface Signals		Generic Interface Signals		IIC Interface Signals	
Pin Name	**Signal Name**	**Signal Type**	**Signal Name**	**Signal Type**	**Signal Name**	**Signal Type**
PCI_CLK	PCI_CLK	I	Cycle_sel0	I		I
RESET_N	RESET_N	I	RESET_N	I	RESET_N	I
AD[31:0]	AD[31:0]	B(t/s)	AD[31:0] D[31:0]	B(t/s) B(t/s)		
CBE_N[3]	CBE_N[3]	I	ALE AS_N	I I		
CBE_N[2]	CBE_N[2]	I	useDS	I		
CBE_N[1]	CBE_N[1]	I	DS_n RD_n	I I		I
CBE_N[0]	CBE_N[0]	I	RD/WR_n WR_n	I I	Cycle_sel	I
FRAME_N	FRAME_N	I	CS_N	I		
TRDY_N	TRDY_N	O(t/s)	RDY	O(o/d)		
IDSEL	IDSEL	I	A8	I	Device_ID6	I
IRDY_N	IRDY_N	I	A2	I	Device_ID0	I
PAR	PAR	B(t/s)	A3	I	Device_ID1	I
PERR_N	PERR_N	O(t/s)	A4	I	Device_ID2	I
SERR_N	SERR_N	O(o/d)	A5	I	Device_ID3	I
DEVSEL_N	DEVSEL_N	O(s/t/s)	A6	I	Device_ID4	I
STOP_N	STOP_N	O(s/t/s)	A7	I	Device_ID5	
INTA_N	INTA_N	O(o/d)	INTA_N	O(o/d)	INTA_N	O(o/d)
SDA					SDA	B(o/d)
SCL					SCL	B(o/d)
T_RSV1 (SEL_PCI)	SEL_PCI ='1'	I	SEL_PCI ='0'			
T_RSV2	One_wait_ Cycle	I	Muxed AD	I		

	PCI Interface Signals		Generic Interface Signals		IIC Interface Signals	
Pin Name	**Signal Name**	**Signal Type**	**Signal Name**	**Signal Type**	**Signal Name**	**Signal Type**
SEL_IIC	='0'	I	='0'	I	='1'	I
SEL_EEPROM					SEL_EEPROM	

The signal definition for the above interfaces is shown in the table below

Signal Type	Definition
I	Input is a Standard CMOS input only
O	Output is a standard CMOS active driver
B	Bi-directional signal
t/s	Tri-state input / output pin
s/t/s	Sustained tri-state is an active-low tri-state signal owned and driven by one and only one agent at a time. The agent that drives an s/t/s pin low must drive it high for atleast one clock before letting it float A pullup is required to sustain the inactive state until another agent drives it.
o/d	Open drain allows multiple devices to be shared as a wire-OR. A pullup is required to sustain the inactive state until another agent drives it and must be provided by a central source

6.2.2.3 PCI Interface Details

The PCI Interface Signals are shown in the table below

Table 45 PCI Interface Signals

Signal Name	Signal Type	Signal Description
PCI_CLK	I	PCI_CLK provides the reference signal for all other PCIinterface signals, except RESET_n and INTA_n. The frequency of PCI_CLK ranges from DC to 33 MHz.
RESET_N	I	RESET_n initializes the CPUIF' s PCI interface circuitry to a stable state. RESET_n can be asserted asynchronously to the PCI clock edge. When active, the PCI output signals are tri-stated, and the open-drain signals float.

Signal Name	Signal Type	Signal Description
AD[31:0]	I	AD[31..0] is a time-multiplexed address/data bus, with each bus transaction consisting of an address phase followed by one or more data phases. The FRAME_n signal identifies the start of an address phase. The data phases occur when both IRDY_n and TRDY_n are asserted.
C/BE_n[3:0]	I	Command and byte enables are multiplexed on C/BE_n[]. The bus command is indicated during the address phase of a transaction. The byte enables are active during the data phases of a transaction and indicate which bytes of the 32-bit data transferred are valid.
FRAME_n	I	FRAME_n is driven by the current bus master to indicate the beginning and the duration of a bus operation. When FRAME_n is first asserted, the address and command Signals are present on AD[] and C/BE_n[]. FRAME_n remains asserted during the data operation and is de-asserted to identify the end of a data operation.
TRDY_n	O	Target ready. TRDY_n is output by a target towards the initiator to indicate that the target can complete a data operation.
IDSEL	I	Initialization device select. IDSEL is a chip select forconfiguration read and write transactions.
IRDY_n	I	Initiator ready. IRDY_n is output by a bus master towards a target to indicate that the bus master can complete a data operation.
PAR	B	PAR is the signal of even parity calculated on the linking of events in the AD[] and C/BE_n[] fields. It is always calculated by the device providing data and by the bus master during address phases. PAR must be valid one clock following its corresponding data (or address) on the PCI bus.
PERR_n	O	PERR_n indicates data parity errors on data transactions
SERR_n	O	SERR_n indicates system errors and address parity errors (or data parity errors during special cycles).
DEVSEL_n	O	Target asserts DEVSEL_n as a decode acknowledgment when the address and bus command are valid.
STOP_n	O	STOP_n is asserted by a target to request that the bus master stops the current transaction.

Signal Name	Signal Type	Signal Description
INTA_n	O	Interrupt A is an active low interrupt to the host. INTA_n must be used for any single-function device requiring an interrupt capability.
SEL_IIC	I	SEL_IIC = 0
T_RSV1 (SEL_PCI)	I	SEL_PCI = 1

6.2.2.4 Generic Interface Signals Description

Table 46 Generic Interface Signals

Signal Name	Signal Type	Signal Description

PCI_CLK	I	GENERIC mode wait cycle select
		Cycle_sel0 =0: no wait
		Cycle_sel0 =1: wait delay cycle = 3 cycles = 90 ns

RESET_n	I	Reset Input (Asynchronous)
AD[31:0]	B	When muxed_AD = 1, AD[] is multiplexed address and databus. When muxed_AD = 0, D[] is bidirectional databus.
CBE_N[3]	I	Treated as ALE / AS_n for Generic Interface When: muxed_AD = 1, ALE is the active-high address latch enable input for latching the multiplexed address on AD[]. muxed_AD = 0, AS_n is the active-low address strobe input for latching the address lines on A2 thru A8.
CBE_N[2]	I	Treated as useDS for Generic Interface When: useDS = 1, the active-low data strobe signal DS_n and the read/write signal RD/WR_n are used for memory I/O control. useDS = 0, the active-low read strobe signal RD_n and the active-low write strobe signal WR_n are used for memory I/O control.
CBE_N[1]	I	Treated as DS_n for Generic Interface DS_n is the active-low data strobe signal used when useDS = 1. RD_n is the active-low read strobe signal when useDS = 0.

Signal Name	Signal Type	Signal Description
CBE_N[0]	I	Treated as RD/WR_N for Generic Interface When useDS =1 RD/WR_n is the read/write mode indicator (read when 1; write when 0) When useDS = 0 WR_n is the write strobe line.
FRAME_N	I	CS_n is the active low chip select input for GENERIC mode.
TRDY_N	O	When the interface is ready for data transfer, RDY is driven 1. It should be driven 0 as soon as CS_n is active if the interface is not ready for data transfer yet.
IRDY_N	I	Address line # 2 when muxed_AD = 0
PAR	I	Address line # 3 when muxed_AD = 0
PERR_N	I	Address line # 4 when muxed_AD = 0
SERR_N	I	Address line # 5 when muxed_AD = 0
DEVSEL_N	I	Address line # 6 when muxed_AD = 0
STOP_N	I	Address line # 7 when muxed_AD = 0
INTA_N	O (o/d)	Interrupt Output (Active Low)
SEL_IIC	I	SEL_IIC = 0
T_RSV1 (SEL_PCI)	I	SEL_PCI = 0
T_RSV2	I	Multiplexed address and data bus selection input. 1: multiplexed address and data on AD[]. 0: address lines on A2 thru A8 and data on D[].

6.2.2.5 IIC Interface Signals

Table 47 I^2C Interface Signals

Signal Name	Signal Type	Signal Description
RESET_n	I	Reset Input (Asynchronous)
EEPROM_SPEED	I	1 : Data Rate 3.3 MHz 0 : Data Rate 100 kHz
IRDY_N	I	Device ID bit 0
PAR	I	Device ID bit 1

Signal Name	Signal Type	Signal Description
PERR_N	I	Device ID bit 2
SERR_N	I	Device ID bit 3
DEVSEL_N	I	Device ID bit 4
STOP_N	I	Device ID bit 5
IDSEL	I	Device ID bit 6
INTA_N	O (o/d)	Interrupt request to external CPU on IIC Bus
SEL_IIC	I	SEL_IIC = 1
T_RSV1 (SEL_PCI)	I	SEL_PCI = 0
SCL	I/O	I^2C Serial Clock
SDA	I/O	I^2C Serial data

6.2.3 I^2C Bus Concept

Two wires, serial data (SDA) and serial clock (SCL), carry information between the devices connected to the bus. Each device is recognised by a unique address whether it's a microcontroller, LCD driver, memory or keyboard interface and can operate as either a transmitter or receiver, depending on the function of the device.

Table 48 Definition of IIC Bus Terminology

Terminology	Description
Tranmitter	The device which sends the data to the bus
Receiver	The device which receives the data from the bus
Master	The device which initiates a transfer, generates clock signals and terminates a transfer
Slave	The device addressed by a master
Multi-master	More than one master can attempt to control the bus at the same time without corrupting the message
Arbitrator	Procedure to ensure that, if more than one master simultaneously tries to control the buse, only one is allowed to do so and the message is not corrupted
Synchronisation	Procedure to synchronize the clock signals of tow or more devices

6.2.3.1 Start and Stop Conditions

A HIGH to LOW transition on the SDA line while SCL is HIGH indicates a START condition.

A LOW to HIGH transition on the SDA line while SCL is HIGH indicates a STOP condition .

START / STOP conditions are always generated by the master.

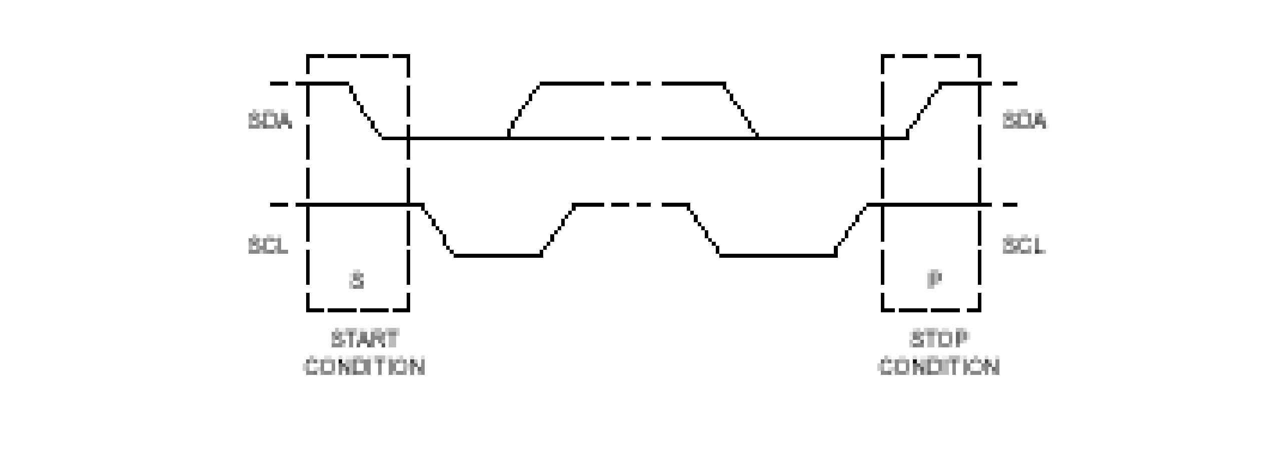


Figure 19 Start & Stop Condition

Every byte transferred on the SDA line must be 8-bits long and each byte has to be followed by an acknowledge bit. Data transfer with acknowledge is mandatory. The acknowledge-related clock pulse is SDA line during the acknowledge clock pulse. The receiver must pull down the SDA line

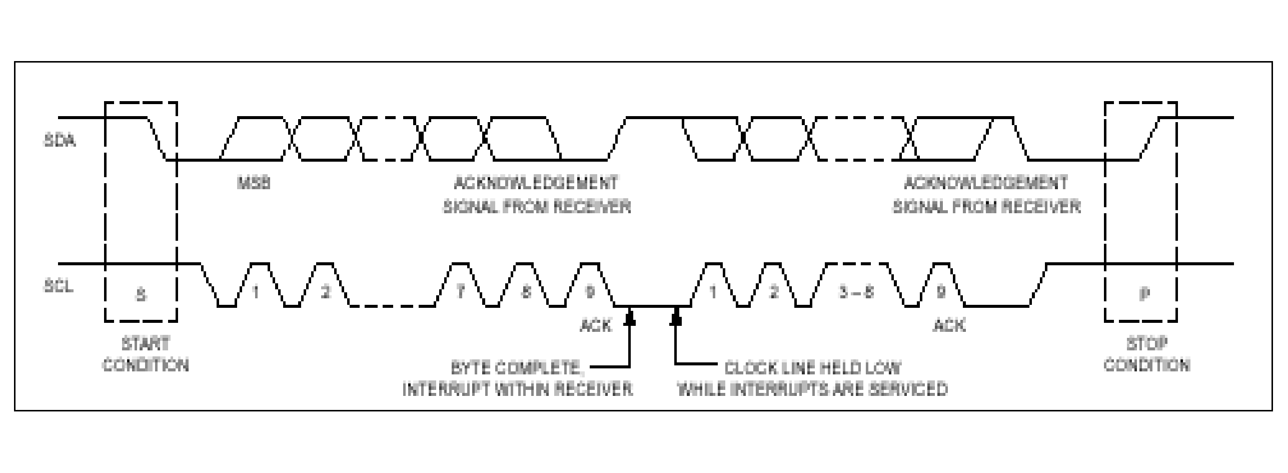


Figure 20 Data Transfer on the IIC Bus

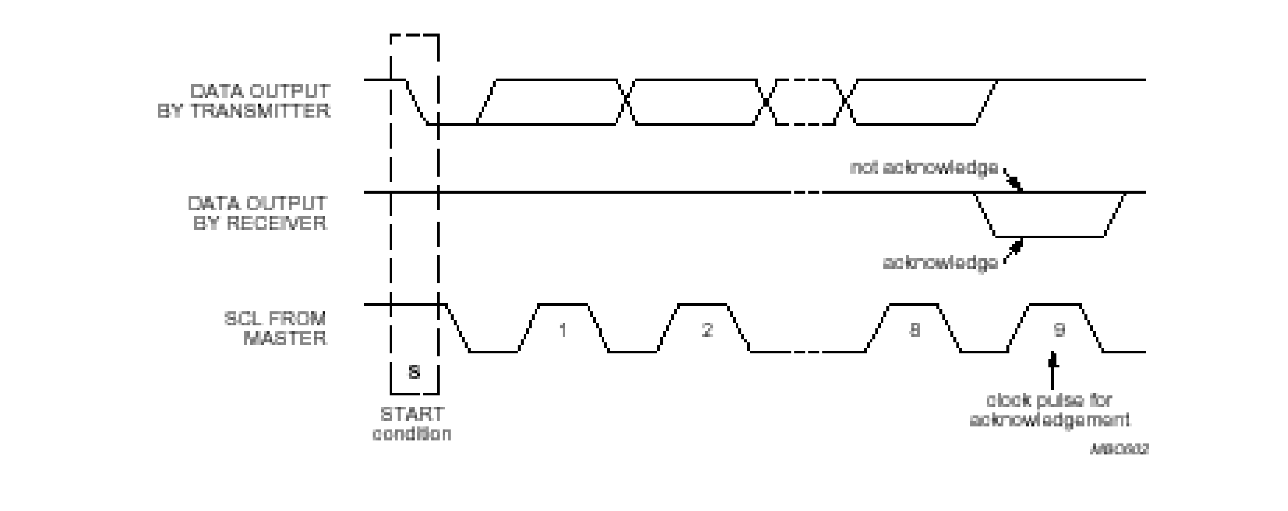


Figure 21 Acknowledge on the I^2C Bus

6.2.3.2 Application Note for IIC Interface of PLB 2224

Read Green Swith_config register
Green I2C Address is set to 1100,000

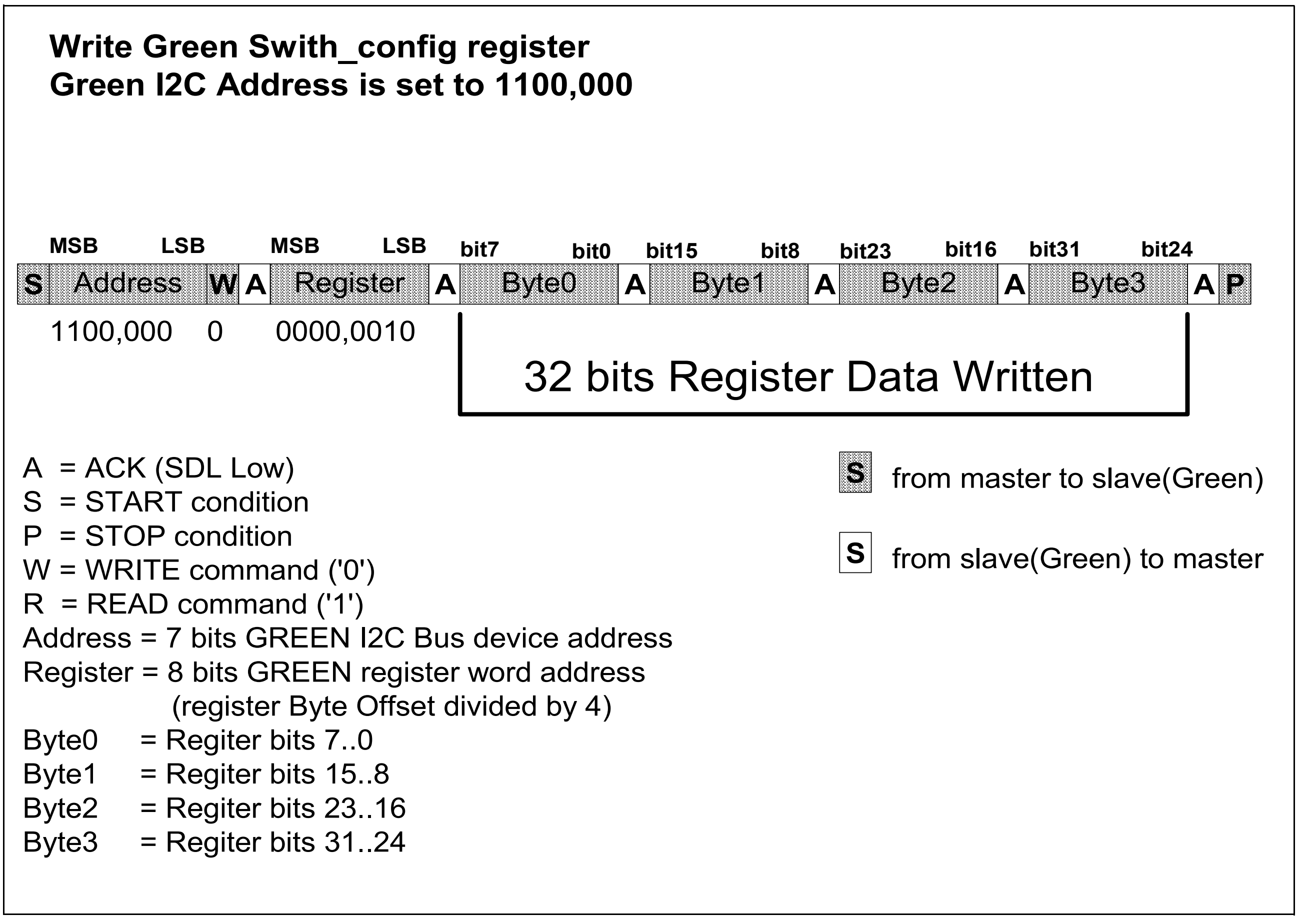
	MSB LSB			MSB LSB			MSB LSB			bit7 bit0		bit15 bit8		23 16		31 24		
S	Address	W	A	Register	A	S	Address	R	A	Byte0	A	Byte1	A	Byte2	A	Byte3	A	P
	1100,000	0		0000,0010			1100,000	1		32 bits Register Data								

A = ACK (SDL Low)
S = START condition
P = STOP condition
W = WRITE command ('0')
R = READ command ('1')
Address = 7 bits GREEN I2C Bus device address
Register = 8 bits GREEN register word address
(register Byte Offset divided by 4)
Byte0 = Regiter bits 7..0
Byte1 = Regiter bits 15..8
Byte2 = Regiter bits 23..16
Byte3 = Regiter bits 31..24

▨S from master to slave(Green)

S from slave(Green) to master

Figure 22 Read Protocol for IIC Bus



Write Green Swith_config register
Green I2C Address is set to 1100,000

A = ACK (SDL Low)
S = START condition
P = STOP condition
W = WRITE command ('0')
R = READ command ('1')
Address = 7 bits GREEN I2C Bus device address
Register = 8 bits GREEN register word address
(register Byte Offset divided by 4)
Byte0 = Regiter bits 7..0
Byte1 = Regiter bits 15..8
Byte2 = Regiter bits 23..16
Byte3 = Regiter bits 31..24

▨ S from master to slave(Green)

□ S from slave(Green) to master

Figure 23 Write Protocol for IIC Interface

1. IIC interface of PLB 2224 support 7-bits device address.

It requires EEPROM's device address be fixed as 7'b1010000.

The device address for PLB 2224 itself is decided by 6 pins' value when SEL_IIC == 1:

device_ID[6:0] = {IDSEL, STOP_N, DEVSEL_N, SERR_N, PERR_N, PAR, IRDY_N}, in which IDSEL is device_ID6, IRDY_N is device_ID0.

2. IIC interface of PLB 2224's data unit is double word (4 bytes).

PLB 2224 has 512-bytes register space and each register has 4 bytes, so totally AR 2224 has 128 registers.

Given register 4 as an example, the serial bits occured sequentially on bus SDA is: reg4[7], reg4[6], ..., reg4[0], reg4[15], ..., reg4[8], ..., reg4[23], ..., reg4[16], reg4[31], ..., reg4[24] so, the least significant byte is first in 4-bytes and the most significant bit is first in one byte.

3. Master mode, or EEPROM mode.

set SEL_IIC = 1 and FRAME_N = 1.

PLB 2224 will automatically issue 512-bytes sequential read in which the first 511-bytes are PLB 2224 register data, the last byte is checksum.

Assuming the register is 7'bxxxxxxx, its data are stored in the corresponding four EEPROM bytes locations with address 9'bxxxxxxx00, 9'bxxxxxxx01, 9'bxxxxxxx10 and 9'bxxxxxxx11, where the byte at 9'bxxxxxxx00 corresponds to the least significant byte of that register.

4. checksum calculation: assuming data[i] is byte i in the 512-bytes of EERPROM.

checksum = data[511] = ~(data[0]+data[1]+...+data[510]) +1

5. Slave mode, or system has a IIC mirocontroller

set SEL_IIC = 1 and FRAME_N = 0.

set device_ID[6:0] as needed. see 1.

IIC interface of PLB 2224 supports random read, sequential read, random write and sequential write.

In the following section, the terms can be referenced in ATMEL's eeprom data sheet.

The operation is much simillar as ATMEL's eerpom. The only difference is that eeprom's address is byte boundary, and PLB 2224's address is 4-byte boundary. So only 7-bits address (or 7-bits register ID, not the 7 bits device address in 1.) can cover PLB 2224's 128 registers.

Random Read Operation on SDA

{start, device_id[6:0], 1'b0 (write), ack};

{reg_id[7:0] (actually the MSB always 0 because AR 2224 only has 128 registers), ack};

{start, device_id[6:0], 1'b1(read), ack};

{data[7:0] (data is the content of the register whose id is reg_id[7:0], ack};

{data[15:8], ack};

{data[23:16], ack};

{data[31:24], no ack, stop}

Sequential Read Operation on SDA

{start, device_id[6:0], 1'b1(read), ack};

{data_n[7:0], ack}, {data_n[15:8], ack}; {data_n[23:16], ack}; {data_n[31:24], ack};

{data_n+1[7:0], ack}, {data_n+1[15:8], ack}; {data_n+1[23:16], ack}; {data_n+1[31:24], ack};

...;

{data_n+x[7:0], ack}, {data_n+x[15:8], ack}; {data_n+x[23:16], ack}; {data_n+x[31:24], no ack, stop}

Random Write Operation on SDA

{start, device_id[6:0], 1'b0 (write), ack};

{reg_id[7:0], ack};

{data[7:0] (data is the content of the register whose id is reg_id[7:0], ack};

{data[15:8], ack};

{data[23:16], ack};

{data[31:24], ack, stop}

Sequential Write Operation on SDA

{start, device_id[6:0], 1'b0 (write), ack};

{n[7:0], ack};

{data_n[7:0], ack}, {data_n[15:8], ack}; {data_n[23:16], ack}; {data_n[31:24], ack};

{data_n+1[7:0], ack}, {data_n+1[15:8], ack}; {data_n+1[23:16], ack}; {data_n+1[31:24], ack};

...;

{data_n+x[7:0], ack}, {data_n+x[15:8], ack}; {data_n+x[23:16], ack}; {data_n+x[31:24], ack, stop}

6.3 PCI Command Definition

Table 49 PCI Command Definition

C_BE_N	Command Type	Support in CPUIF
0000	Interrupt acknowledge	No
0001	Special cycle	No
0010	I/O Read	No
0011	I/O Write	No
0100	Reserved	No
0101	Reserved	No
0110	Memory Read	Yes
0111	Memory Write	Yes
1000	Reserved	No
1001	Reserved	No
1010	Configuration Read	Yes
1011	Configuration Write	Yes
1100	Memory Read Multiple	No
1101	Dual Address cycle	No
1110	Memory Read Line	No
1111	Memory Write and invalidate	No

6.4 PCI Configuration Registers

Table 50 PCI Configuration Register Address Space

Address	Byte			
	3	2	1	0
00	Device ID		Vendor ID	
04	Status Register		Command Register	
08	Class Code			Revision ID
0C	Bist	Header Type	Latency Timer	Cache Line Size
10	Base Address Register 0			
14	Base Address Register 1			
18	Base Address Register 2			
1C	Base Address Register 3			
20	Base Address Register 4			
24	Base Address Register 5			
28	Cardbus CIS Pointer			
2C	Subsystem ID		Subsystem Vendor ID	
30	Expansion ROM Base Address			
34				Capability Pointer
38	Reserved			
3C	Max Latency	Minimun Grant	Interrupt Pin	Interrupt Line

7 Register Description

This chapter describes the PLB 2224 registers configuration, dealing with the register map and the details of each register. The PCI configuration registers are also covered.

In a managed Ethernet switch, the PLB 2224 PCI Configuration Space needs to be programmed first. This is accomplished through the changing of the Configuration Address and Configuration Data registers in QSpan PCI bridge chip.

7.1 PCI Configuration Phase

The address of QSpan ranges from FA210000 to FA21FFFFF. When an address in this range is accessed, the CPU uses CS6 to select QSpan. The reference design will have the IDSEL (ID select) of the PLB 2224 hardwired to one of the upper 16-bits of the address.

Once the device number is determined, CPU can write to CON_ADD (FA210500) for the PLB 2224 chip and one specific PCI Configuration register. The content of the register is written to CON_DATA (FA210504).

The following registers need to be programmed before the PLB 2224 internal registers can be accessed:

- PCI Configuration Space Control and Status,
- PCI Configuration Miscellaneous 0,
- PCI Configuration Base Address,
- PCI Configuration Miscellaneous 1.

The PCI Target channel registers in QSpan need to be programmed before the address translation takes place in addressing the PLB 2224 internal register.

7.2 Switch Configuration Phase

After PCI Configuration phase is completed, the PLB2224 internal registers are exposed to the PCI bus. These registers can be programmed according to the specific functions required by the Ethernet switch.

7.2.1 Chip Level Configuration

Table 51 Chip Level Configration

Function	Register	Field	Comment
Enable/Disable all PLB 2224 ports	**"Chip Configuration Register" on Page 141**	intelligent	This can be used by 802.1D to control the port
Disable 802.3x flow control for full duplex ports	Chip Configuration Register	disable_pause	
MDIO or SMII polling result selection	Chip Configuration Register	use_mdio_mode	Select speed, pause_e, full_duplex, and link_ok source
	"Switch Configuration Register" on Page 145	e_hw_mode	The e_hw_mode should be 1.
Congestion control for Half duplex ports	Chip Configuration Register	e_hd_cg_ctl	Generate collision if encountering congestion.
TX Queue length control	Chip Configuration Register	en_txq_drop	To make sure wire-speed can be achieved and resources are not exceeded.

Function	Register	Field	Comment
LED Control	Chip Configuration Register	led_mode , en_tx_led, mask_coll_led	Control the behavior of LED.
Watermark Setup	Watermark registers for CPU, FE, and GE ports for both TX and RX.	All	Setup watermarks for each type of ports for resource management.
	Switch configuration register	watermark_scale	

7.2.2 Port Level Configuration and Monitoring

Table 52 Port Level Configuration and Monitoring

Function	Register	Field	Comment
Monitor Port Status	**"Port Status Register" on Page 154**	port_active, rxdq_full_drop, crc_err, late_collision	One bit for each port can be read to determine the port status corresponding to the events mentioned in the Field column
	"Port Event Register" on Page 155	txdq_full port_critical_event	One bit for each port can be read to determine the port status corresponding to the events mentioned in the Field column
	"Port Underrun/ Overrun Register" on Page 156	txfifo_underrun rxfifo_overun	One bit for each port can be read to determine the port status corresponding to the underrun/ overrun mentioned in the Field column

Function	Register	Field	Comment
MII Port Control	**"Port MII Register" on Page 157**	speed_cpu pause_e_cpu full_duplex_cpu link_ok_cpu	When autonegotiation is bypassed these values are used to configure the individual ports
Port Priority setup	**"Port Priority Register" on Page 158**	priority	Set priority of individual ports incase of pri_at_src
Monitor the packets sent and received on each port	**"Port Monitor Register" on Page 158**	monitored	Packets are monitored for specific port in both receive and transmit directions. The monitoring port is set based on the setting in switch configuration register (*monitoring_pid*)
Trunking	**"Port Trunk Register" on Page 159**	trunk_opt	Associated to form a trunk between two ethernet switches
	"Switch Configuration Register" on Page 145	trunk_da_based trunk_sa_based	

7.2.3 Switch Level Configuration

Table 53 Switch Level Configuration Setup

Function	Register	Field	Comment
Select Autonegotiation	**"Switch Configuration Register" on Page 145**	e_hw_mode	The CPU does the port configuration or gets the results from auot-negotiation results (Through MDIO interface or SMII interface)
CPU co-ordination	**"Switch Configuration Register" on Page 145**	big_endian, soft_rst, dis_err_rst	Supports CPU endian modes and reset behaviour
Packet priority selection	**"Switch Configuration Register" on Page 145**	e_pri, pri_at_src, pri_at_da, bandwidth_ratio	Enable packets priority for forwarding to different queues
MAC Address Control	**"Switch Configuration Register" on Page 145**	ma_freeze_new, ma_freeze	Manage MAC additiona and deletions
Set up monitoring port	**"Switch Configuration Register" on Page 145**	monitoring_port	Forward packets to the monitoring ports if enabled
Enable Monitoring	**"Switch Configuration Register" on Page 145**	monitor_pkt_err, monitor_pkt_ma_change	Enable Packet monitoring
Queue Appropriation	**"Switch Configuration Register" on Page 145**	watermark_scale, rx_pause_frame, bandwidth_ratio	

Function	Register	Field	Comment
Aging Time Control	**"Switch Configuration Register" on Page 145**	age_tick_sofware, age_tick_sel[2:0]	Control of the aging timer under different situations
Monitor Switch Events	**"Switch Status and Mask Register" on Page 152**	All bits	Events like ma_change, freeq_empty, late_collision etc.

7.2.4 VLAN and Bridge Configuration

Table 54 VLAN and Bridge Configuration

Function	Register/s	Field	Comment
802.1D control Port Bridge state	**"Port Bridge State Register" on Page 160**	bridge_state	For spanning tree control (Each VLAN operates over a single spanning tree)
VLAN aware indicator / Admit tagged only	**"VLAN Aware/ inTag Control Register" on Page 183**	vlan_aware, admit_tagged_only[]	Switch follows 802.1Q with tagging and untagging capability. Please refer to **"VLAN" on Page 90** for more details
Port VLAN membership	**"Port Index Register_20_0" on Page 161** **"Port Index Register 26_21" on Page 184**	port_ix	Each port has a VLAN index, which is used to findout the membership from the VLAN table
VLAN ingress filter	**"VLAN Ingress Filter" on Page 184**	Ingress_Filter	Enable ingress filtering on a per port basis
Security Enable	**"Switch Configuration Register" on Page 145**	e_ucast_secu	VLAN security bit for unicast packets
Priority information for tagged packets	**"Tag Priority Table Register" on Page 181**	tag_pri	to determine the priority of the packets
Egress priority table	**"Egress Priority Table" on Page 182**	E_pri_1, E_pri_0	Priority of the egress packets for a tagged frame

Function	Register/s	Field	Comment
VLAN Table / VID_IX_table / Floodmap tables	**"Memory Upper Address Register" on Page 163**	All	Please refer to respective register details and also section on VLAN for more information
	"Memory Access Register" on Page 165	All	

Table 55 Filtering Configuration

Function	Register	Field	Comment
VLAN port list	**"Portlist_1023 Register" on Page 161**	portlist	Port list 1023 is used when port index is 1023 as a special case. This is typically an all port list.
Broadcast	**"DA Index Register" on Page 163**	unknown_flood_ix, unknow_ucast_ix, bcast_ix	Port list index for broadcast and unknown MAC addresses.

Table 56 MAC Address Accessing

Function	Register	Field	Comment
Read and Write MAC address table	Memory	All	Please set blk_sel_msb[3:0] = 4'b1001 to access ARL dataspace
	Memory access	All	

7.2.5 BPDU Transmission and Receiving

Table 57 BPDU Transmission and Reception

Function	Register	Field	Comment
Configure MAC address for Bridge or transmit BPDU packets	**“CMAC RX Register” on Page 167**	All	Poll crx_cpu_pktid_rdy before sending a packet. Setup crx_sof, crx_eof, crx_bytecnt, crx_crc_err, crx_crc_gen. Write data to cmac_data till the last transaction is complete. In between crx_cpu_fifo_rdy is polled. Before the last transaction the crx_eof & crx_bytecnt is set.ARL register is used to do the DA lookup, destination ports etc. Please refer to **“CPU Port Rx” on Page 44** for more details
	“CMAC Data Register” on Page 167	All	
	“ARL Register” on Page 170ARL	All	
Receiving BPDU or other learning packet	**“CMAC TX Register” on Page 169**	All	Please refer to section **“CPU Port Tx” on Page 45** for more details
	“CMAC_TX1 Register” on Page 170	All	
	“CMAC Data Register” on Page 167	All	

7.3 PLB 2224 PCI Configuration Registers

Table 58 PCI Configuration Registers

Offset	Name	Access	Description
0x000	PCI_ID	R	PCI Configuration Space ID register
0x004	PCI_CS	R/W	PCI Configuration Space Control and Status Register
0x008	PCI_CLASS	R	PCI Configuration Class Register
0x00C	PCI_MISC0	R/W	PCI Configuration Miscellaneous Register 0
0x010	PCI_BSM0	R/W	PCI Configuration Base Address for memory register
0x014	PCI_BSM1	R/W	PCI Configuration Base Address for memory register
0x018	PCI_BSM2	R/W	PCI Configuration Base Address for memory register
0x01C	PCI_BSM3	R/W	PCI Configuration Base Address for memory register
0x020	PCI_BSM4	R/W	PCI Configuration Base Address for memory register
0x024	PCI_BSM5	R/W	PCI Configuration Base Address for memory register
0x028	PCI_CIS	R/W	PCI Configuration Card bus CIS Pointer Register
0x02C	PCI_SID	R	PCI Configuration Subsystem ID register
0x030	PCI_BSROM	R/W	PCI Configuration Expansion ROM Base Address Register
0x034	PCI_RSVD0		Reserved
0x038	PCI_RSVD1		Reserved
0x03C	PCI_MISC1	R/W	PCI Configuration Misc 1 Register

7.3.1 Register 0x00, PCI Configuration Space ID

Table 59 PCI Configuration Space ID

Bit(s)	Name	Length	Block	Access	Default
31:16	dev_id	16	PCI_IF	RO	0x0009
15:0	vendor_id	16	PCI_IF	RO	0x15D1

Description: PCI Configuration Space ID register is used by the PCI device to identify itself. It consists of a unique vendor ID assigned by PCI SIG and a device ID assigned by the vendor.

dev_id : 0x0009

vendor_id : 0x15D1.

7.3.2 Register 0x04, PCI Configuration Space Status and Command

Bit(s)	Name	Length	Block	Access	Default
31	perr	1	PCI_IF	RR	0
30	serr	1	PCI_IF	RR	0
29	rma	1	PCI_IF	RR	0
28	rfa	1	PCI_IF	RR	0
27	sta	1	PCI_IF	RR	0
26:25	dev_sel	2	PCI_IF	RO	0x01
24	dpd	1	PCI_IF	RR	0
23	tfbbc	1	PCI_IF	RO	0x1
22	rsvd	1	PCI_IF	RO	0
21	dev66	1	PCI_IF	RO	0
20:10	rsvd	11	PCI_IF	RO	0
9	mfbbc	1	PCI_IF	RO	0
8	serr_en	1	PCI_IF	R/W	0
7	wait	1	PCI_IF	RO	0
6	peresp	1	PCI_IF	R/W	0
5	vgsps	1	PCI_IF	RO	0
4	mwi_en	1	PCI_IF	RO	0
3	sc	1	PCI_IF	RO	0
2	bm	1	PCI_IF	RO	0
1	ms	1	PCI_IF	RO	0
0	ios	1	PCI_IF	RO	0

Description

The PCI Configuration Space Status and Command register has two sub-registers. The status sub-register records status information for PCI bus related events. The command sub-register provides coarse control to generate and respond to PCI commands.

perr

The perr bit indicates a detected (address or data) parity error. It is set by CPUIF when detected.

0: No parity error,

1: Parity error

serr

The serr bit signals an address parity error. It is set by CPUIF when detected.

0: SERR# not asserted

1: SERR# asserted.

rma

The rma bit indicates the receiving of a master abort. This bit is hardwired to 0 since PLB 2224 does not detect master abort.

rta

The rta bit indicates the receiving of a target abort by a PCI master. This bit is hardwired to 0 since PLB 2224 is not a PCI master device.

sta

This bit is set by CPUIF whenever it terminates a transaction with target abort.

dev_sel

The dev_sel field indicates the slowest time that a device asserts DEVSEL# for any bus command except Configuration Read and Configuration Write. It encodes the timing of DEVSEL#. The

possible values are

0: Fast;

1: Medium

2: Slow

3: Reserved

This field is hardwired to 1 for CPUIF DEVSEL# timing in PLB 2224 is medium.

dpd

This bit is set when three conditions are met:

1. The bus agent asserted PERR# itself (on a read) or observed PERR# asserted (on a write),
2. The agent setting the bit acted as the bus master for the operation in which the error occurred, and
3. The Parity Error Response bit (Command register) is set. This bit is hardwired to 0 since PLB 2224 does not act as a PCI master.

tfbbc

This optional read-only bit indicates whether or not the target is capable of accepting fast back-to-back transactions when the transactions are not to the same agent.

0: PLB 2224 does not accept fast back-to-back transactions

1: PLB 2224 does accept fast back-to-back transactions.

dev66

This optional read-only bit indicates whether or not this device is capable of running at 66 MHz.

0: 33 MHz only

1: 66 MHz capable

mfbbc

This optional read/write bit controls whether or not a master can do fast back-to-back transactions to different devices. Initialization software will set the bit if all targets are fast back-to-back capable.

1. Fast back-to-back transactions are only allowed to the same agent.
2. Master is allowed to generate fast back-to-back transactions to different agents.

This bit is hardwired to 0 as PLB 2224 does not work as a PCI master.

serr_en

This bit is enable parity error to be reported on SERR# pin.

0: Disable the SERR#

1: Enable the SERR#

wait

This bit controls whether or not a device does address/data stepping.

0: Devices that never do stepping.

1: Devices that always do.

Devices that can do either, must make this bit read/write and have it initialize to 1 after reset. This bit is hardwired to 0 in PLB 2224 does not support address stepping.

peresp

This bit enables the device’s response to parity errors.

0: Disable parity error reporting

1: Enable parity error reporting.

Devices that check parity must implement this bit. Devices are still required to generate parity even if parity checking is disabled.

This bit is initialized to 1.

vgsps

This bit controls how VGA compatible and graphics devices handle accesses to VGA palette registers.

0: Disable palette

1: Enable palette snooping is enabled

VGA compatible devices should implement this bit. This bit is hardwired to 0 since PLB 2224 is not a VGA device.

mwi_en

This bit enables Memory Write and Invalidate command.

0: Disable Memory Write and Invalidate

1: Enable Memory Write and Invalidate

This bit is hardwired to 0 since PLB2224 does not support Memory Write and Invalidate command.

sc

This bit controls a device's action on Special Cycle operations.

0: Ignore all Special Cycle operations.

1: Monitor Special Cycle operations.

This bit is hardwired to 0 as PLB 2224 does not monitor special cycles.

bm

This bit controls a device's ability to act as a master on the PCI bus.

0: PCI target only

1: PCI master capable

This bit is hardwired to 0 as PLB 2224 does not act as a PCI master.

ms

This bit controls a device's response to Memory Space accesses.

0: Disables memory space access

1: Enable memory space accesses.

This bit is initialized to 1.

ios

This bit controls a device’s response to I/O Space accesses.

0: Disables I/O space access

1: Enable I/O space accesses.

This bit is initialized to 0 as PLB 2224 supports memory space access only.

7.3.3 Register 0x08, PCI Configuration Class

Table 60 Register 0x08 PCI Configuration Class

Bits	Name	Length	Block	Access	Default
31:24	base_class	8	PCI_IF	Read Only	0x02
23:16	sub_class	8	PCI_IF	Read Only	0x00
15:8	program_if	8	PCI_IF	Read Only	0x00
7:0	revision	8	PCI_IF	Read Only	0x01

Description

The PCI Configuration Class register is read-only and is used to identify the generic function of the PLB 2224 device and, in some cases, a specific register-level programming interface.

base_class

The base class code classifies the type of function the device performs. This field is hardwired to 0x02 – Network Controller.

sub_class

The sub-class code identifies more specifically the function of the device. This field is hardwired to 0x00 – Ethernet Controller.

program_if The programming interface identifies a specific register-level (if any) programming interface that device independent software can interact with the device. This field is hardwired to 0x00 – Ethernet Controller.

revision

This register specifies a device specific revision identifier. This is vendor specific. Zero is an acceptable value. This field should be

viewed as a vendor defined extension to the Device ID. This field is hardwired to 0x01.

7.3.4 Register 0x0C, PCI Configuration Miscellaneous 0

Table 61 Register 0x0C PCI Config Miscellaneous0

Bits	Name	Length	Block	Access	Default
31:24	bist	8	PCI_IF	RO	0x0
23:16	header_type	8	PCI_IF	RO	0x0
15:8	timer	8	PCI_IF	R/W	0x0
7:0	cline	8	PCI_IF	R/W	0x0

Description

The PCI Configuration Miscellaneous 0 register is used to identify the device independent features for supporting BIST, burst, and multiple functions.

bist

BIST is used for control and status of BIST. This field is hardwired to 0x0 since PLB 2224 does not support BIST.

header_type

This byte identifies the layout of the second part of the predefined header and also whether or not the device contains multiple functions.

Bit 7 in this register is used to identify a multi-function device. If the bit is 0, then the device is single function. If the bit is 1, then the device has multiple functions.

Bits 6 through 0 identify the layout of the second part of the predefined header.

The encoding 01_H is defined for PCI-to-PCI bridges. The encoding 02_H is defined for a CardBus bridge. All other codes are reserved. This field is hardwired to 0x00 to indicate a single function device.

ltimer

This field specifies the value (in PCI clock unit) of the Latency Timer for a PCI bus master. This field is hardwired to 0x0 since PLB 2224 does not act as a PCI master.

cline

This read/write field specifies the cache line size in unit of 32-bit words for PCI master to use in generating memory Write and Invalidate commands and to determine whether to use Read, Read Line, or Read Multiple in accessing memory. This field is hardwired to 0x0 since PLB 2224 does not act as a PCI master.

Table 62 Register 0x10, PCI Configuration Base Address

Bit(s)	Name	Length	Block	Access	Default
31:4	base_addr	28	PCI_IF	R/W	0x00
3	prefetch	1	PCI_IF	R	0x00
2:1	type	2	PCI_IF	R	0x00
0	space	1	PCI_IF	R	0x00

Description

The PCI Configuration Base Address register specifies the PCI addresses this PLB 2224 recognizes in Memory Space.

base_addr

Base Address is a 24 bit (16 Mbytes). The first 8 bits are 0’s.

prefetch

PLB 2224 memory space is not pre-fetchable. This field is hardwired to 0.

type

PCI Bus base address type

0: Base register is 32 bits wide and mapping can be done anywhere in the 32-bit Memory Space.

1: Reserved

2: Base register is 64-bits wide and can be mapped anywhere in the 64-bit address space.

3: Reserved

space

PCI Bus Address Space

7.3.5 Register 0x3C, PCI Configuration Miscellaneous 1

Table 63 Register 0x3C, PCI Configuration Miscellaneous 1

Bits	Name	Length	Block	Access	Default
31:24	max_lat	8	PCI_IF	R	0x00
23:16	min_gnt	8	PCI_IF	R	0x00
15:8	int_pin	8	PCI_IF	R	0x01
7:0	int_line	8	PCI_IF	R/W	0x00

Description

The PCI Configuration Miscellaneous 1 register specifies the PCI bus related timing and signals for access, burst, and interrupts.

max_lat

Maximum Latency specifies how often the device needs to gain access to the PCI bus. This field is hardwired to 0x0 since PLB 2224 has no special latency requirements.

min_gnt

Minimum Grant specifies how long of a burst period the device needs assuming a clock rate of 33 MHz. This field is hardwired to 0x0 since PLB 2224 has no major requirements.

int_pin

Interrupt Pin specifies which interrupt pin PLB 2224 uses. PLB 2224 uses INTA# (0x01).

int_line

Interrupt Line is used to communicate interrupt line routing information for device driver and operating systems to determine priority and vector information. This field is initialized to 0x0.

7.4 PLB2224 Internal Registers

7.4.1 Chip Configuration Register

Name: chip_config

Offset: 0x00

Access: Read/Write

Description

Used to store configuration information about the PLB 2224 within the system. Upon hardware reset, bits [23:0], bits [28:26] and bit[31] will reflect values from the the power strapped LED signal pins as indicated in the table. All LED signal pins have an internal pullup.

Table 64 Chip Configuration Register

Bit fields	Name	Access / Initial value	Description in PLB 2224
31	Intelligent	(R/W) x	*0*:After reset, PLB 2224 is ready to do switching function by enabling all the ports. Intended for systems without a CPU. *1*: All the ports stay in disabled mode until CPU enable them. Intended for systems used in managed mode of operation. The bit is latched from pin-strapping input pin ***led_row_n[3]*** during reset.
30:29	rsv	00	

28	disable_pause	(R/W) x	*0*: to enable PAUSE bit in Advertisement Register during auto-negotiation in full duplex mode of operation. *1*: to disable PAUSE feature for all ports and also not to advertise PAUSE capability during auto-negotiation. The bit is latched from pin-strapping input pin ***led_row_n[1]*** during reset. *Note: For each full duplex ports, 802.3x flow control method will be used depending on PAUSE[1:0] bits from the port's Reg. #4 and #5. For all the half duplex ports, collision method is enabled to control the incoming packet rate depending on whether e_hd_cg_ctl (Bit 6 in **"Chip Configuration Register" on Page 141**) is asserted.* If *disable_pause* bit has been set to disable pause in Software mode of operation, Autonegotiation should be restarted for this to take effect.
27:24	Rsv	00	
23:22 21:20 19:18 17:16 15:14 13:12	delay_sel5[1:0] delay_sel4[1:0] delay_sel3[1:0] delay_sel2[1:0] delay_sel1[1:0] delay_sel0[1:0]	(R/W) x	delay_sel[5:0][1:0] is to select the delay for smii_clk input pins, one per 4 SMII ports. delay_sel0[1:0] is for Port 0 to 3, delay_sel1[1:0] is for Port 4 to 7, …, and delay_sel5[1:0] is for Port 20 to 23. Each select has two bit to control the delay. The total delay will be the bit value times 0.8 ns at typical case. The bit is latched from power-strapping input pin ***led_col[23:12]*** during reset. Since PLB 2224 has source synchronous SMII interface, it is very unlikely that the customer will have to make use of the delay pinstrap to control clock delay Please refer to section **"Delay Select for SMII_RCLK" on Page 50** for more details.

11:10	Sim_mode[1:0]	(R/W) xx	Should always be set to '11' for normal operation. *Note: mode 00, 01, 10 will be used for testing purposes only.* The bit is latched from power-strapping input pins ***led_col[11:10]*** during reset.
9	in_ma_en	(R/W) x	To enable the internal MA table. This bit should be strapped to 1. '0' : Disable internal MA table The bit is latched from power-strapping input pin ***led_col[9]*** during reset.
8	Use_mdio_Mode	(R/W) x	This bit is only valid when e_hw_mode (Bit 9 in **"Switch Configuration Register" on Page 145**) bit is 1. *1*: The MDIO polling result from each Fast Ethernet port's speed capability, pause_e capability, full_duplex capability, and link_ok status is used for operation. *0*: The capabilities of the PHY are polled over SMII interface and the pause capability in this case cannot be determined, as Pause information is not available over SMII interface during auto-negotiation. *Note: This bit has no effect on Gb ports.* The bit is latched from power-strapping input pin ***led_col[8]*** during reset.
7	Rsv	0	Reserved
6	e_hd_cg_ctl	(R/W) x	1: Enables collision-based flow control for half duplex ports. 0: Disables collision-based flow control for half duplex ports. The bit is latched from power-strapping input pin ***led_col[6]*** during reset.
5	mask_coll_led	(R/W) x	*0*: The Status LED flashes when the corresponding port experiences collision. *1*: Disables the flashing of the STATUS LED This bit is latched from pin-strapping input pin ***led_col[5]*** during reset.

4	en_tx_led	(R/W) x	*1*: The Link LED flashes when the corresponding port has successful transmission. *0*: Disable the flashing of the Link LED This bit is latched from pin-strapping input pin ***led_col[4]*** during reset.
3	led_mode	(R/W) x	To select display mode. This bit is latched from pin-strapping input pin ***led_col[3]*** when reset. Please refer to section **“LED” on Page 103** for more details
2:0	pcb_config	(R) x	PLB 2224’s PCB configuration bits which are used to identify how PLB 2224 is configured. This is more used for board level identification purposes, where the manufacturer can set these bits to do some specific kind of operation. This bit is latched from pin-strapping input pin ***led_col[2:0]*** during reset.

7.4.2 Chip Type Register

Name: Chip_type

Offset: 0x04

Access: Read Only

Description: Used for chip identification.

Table 65 Chip_type Register

Bit Fields	Name	(Access) Initial Value	Description
31	mem_test_ done	(R) 0	1: indicating memory test is done. 0: indicating memory test is still in progress.
30:12	rsv	0	Reserved
11:3	chip_type	(R) 9’b000000010	PLB 2224 chip type is hardwired internally. 2 is assigned to PLB 2224.
2:0	chip_rev	(R) 3’b010	PLB 2224 chip revision number.

7.4.3 Switch Configuration Register

Name: switch_config

Offset: 0x08

Access: Read/Write

Description

Stores the configuration information for paramaters and are common for all the ports in PLB 2224.

Table 66 Switch Configuration Register

Bit	Name	(Access) Initial Value	Description
31	Endian_Mode	(R/W) 1	The Endian_Mode bit selects the PLB 2224 internal operation mode based on CPU's Endian mode. *1*: CPU is running big endian mode and is consistent with PLB 2224's internal packet format. *0*: CPU is running little endian mode. All the packet data to and from CPU requires byte swapping.
30	soft_rst	(R/W) 0	*1*: To reset all the pointers and state machines, except timer and CPU interface. Resets all the CPU registers except chip_config register, and re-initializes all the memories. *0*: To release software reset. The soft_rst bit will not reset itself after set to 1. CPU has to reset it by writing a '0' to this bit to end the soft reset of PLB 2224.

29	pri_at_src	(R/W) 0	*1*: The priority of the received packet is set to high when the corresponding bit of the Source Port Priority register is 1. *0*: The packet’s priority is determined by pri_at_da. When both the *pri_at_src* and *pri_at_da* bits are set to ’0’, the packet priority flag is set to low. When both *pri_at_src* and *pri_at_da* are ’1’ and if either lookup results in high priority, the packet priority is set to high.
28	pri_at_da	(R/W) 0	*1*: The priority of the received packet is set to high when ‘ma_pri’ bit of DA lookup result is 1.(*ma_state[3:0]=4’b001x*) *0*: The packet’s priority is determined by pri_at_src. When both *pri_at_src* and *pri_at_da* are set to ’1’, either lookup results in higher priority and the packet priority is set to high When both *pri_at_src* and *pri_at_da* bits are ’0’, the packet priority is set to high. If both of them are set, packet’s priority is determined by the OR result of both. For auto-learned unicast packets and broadcast packets, priority is always ’0’
27	ma_freeze	(R/W) 0	The *ma_freeze* bit controls the updating of MA entries. *1*: No new MA will be learned and the MA port number won’t be changed. Non-critical packets (ma_critical set to 0, new MA, or existing MA with port change) will be discarded. MA ageing function is turned off.
26	ma_freeze_new	(R/W) 0	The *ma_freeze_new* controls the creation of new MA only. If set, only port changes with existing MA entry is allowed. No new MA will be learned. All the non-critical packets (ma_critical set to 0, new MA) will be discarded. MA ageing function is turned off.

25:23	Aging_tick_sel [2:0]	(R/W) 3’b001	Aging tick timer is controlled by this field and is defined below. Default is to select 3.18 ms per tick to have 300 s tick-time for each entry service. aging_tick_sel Tick period Entry aging time 000 (always tick) 0 001 tick_time x1 300 s 010 - 20 s 011 tick_time x3 15 minutes 100 tick_time x12 1 hour 101 tick_time x48 4 hours 110 tick_time x288 1 day 111 - No aging
22:20	bandwidth_ratio [2:0]	(R/W) 3’b001	This is to select the bandwidth ratio between high and low priority queues. If one of them is idle, the other queue will have the full bandwidth to forward its packets. Packet burstness for the same queue is reduced. Low queue is served only when there is no High available. **bandwidth_ratio[2:0] High_Q/Low_Q ratio** 000 16/0 001 15/1 010 14/2 011 13/3 100 12/4 101 11/5 110 10/6 111 8/8 Please refer to the section **“Packet Scheduling” on Page 42** for more details on the bandwidth ratio

19	e_pkt_pri	(R/W) 0	The *en_pkt_pri bit* enables packet prioritization. There are two TX queues and two broadcast queues for each port. One for high priority and one for low priority. If packet prioritization is enabled, packet's priority is determined by either pri_by_src or pri_by_da as described above. 0. Packet prioritization disabled. 1. Packet prioritization is enabled.
18	monitor_pkt_err	(R/W) 0	Set to monitor all the packets with CRC Error. Default to 0.
17	monitor_pkt_ma _change	(R/W) 0	The *monitor_pkt_ma_change* enables packet monitoring for all the packets with 'intrusion', 'ma_change', or 'ma_full'. In an 'intrusion' case, the packet should be forwarded only to Monitoring port. In 'ma_change' or 'ma_full' case, the packet is forwarded to destination port and monitoring port. The default is set to 0.
16	fwd_ovsz_pkt	(R/W) 1	The *fwd_ovsz_pkt* controls oversize packet (greater than 1518 bytes) forwarding mechanism. The oversize packet is sent to its destination ports as long as its CRC is still good. The default value is set to 1.
15	dis_err_rst	(R/W) 0	The *dis_err_rst* bit controls the error recovery mechanism. 0. Enable the fatal error status bits in the switch status register to generate a reset command that will reset the chip. 1. Disabled reset when detecting a fatal error. Fatal errors happen when src_pid_err exists and their corresponding interrupt_en bit is enabled. The default value is set to 0.

14	halt_tx	(R/W) 0	The *halt_tx* bit controls port transmission. When set, all Ethernet ports (Port 0 to 25) will stop transmitting by ignoring TXQ's not_empty status. This bit has no effect on the CPU ports.. Default to 0. ***This is only for chip diagnostic purpose.***
13	halt_rx	(R/W) 0	The *halt_rx* bit controls port receiving. When set, all Ethernet ports (Port 0 to 25) drop the received packets while maintaining the correct counters. This bit has no effect on the CPU ports. Default to 0. ***This is only for chip diagnostic purpose.***
12	halt_release	(R/W) 0	The *halt_release* bit controls the releasing of packet buffers after transmission. 0. Normal operation. 1. Stop releasing or updating ser_cnt after each port's transmission. Default to 0. ***This is only for chip diagnostic purposes.***
11	trunk_on_da	(R/W) 0	The *trunk_on_da* bit controls whether to use DA to determine the forwarding trunk port. *0*. DA is not used to determine which trunk port to forward packets. *1*. DA is used to determine which trunk port to forward packets. If both *trunk_on_da* and *trunk_on_sa* are set, both DA and SA are used for the trunk port decision. Either *trunk_da_based* or *trunk_sa_based* should be set. If both are cleared, only one port is used for forwarding in a trunk.

10	trunk_on_sa	(R/W) 0	The *trunk_on_sa* bit controls whether to use SA to determine the forwarding trunk port. *0*. SA is not used to determine which trunk port to forward packets. *1.* SA is used to determine which trunk port to forward packets. If both *trunk_on_da* and *trunk_on_sa* are set, both DA and SA are used for the trunk port decision.
9	e_hw_mode	1	The *e_hw_mode* bit decides whether the autonegotiation results from MDIO/SMII can be used. *0* : Autonegotiation results (speed / pause_e / full_duplex) from either MDIO or SMII is not used. *1* : Enable MDIO's polling result or SMII polling results. (Ethernet's speed / pause capability / full_duplex). This bit when set to '1' is used in conjunction with use_mdio_mode bit in **"Chip Configuration Register" on Page 141**. The results are written to port_mii registers described in **"Port MII Register" on Page 157**.
8	Watermark_ scale	R/W 0	*0*: watermark scale equal 4, which means the values set in the different watermark registers are multiplied by 4 and used in the application for flow control meachnism *1*: watermark scale equal 1, meaning the register setting is used directly for the flow control

7	e_ucast_secu	0	The *e_ucast_secu* bit enables VLAN security for unicast packets. 0. Disable the security check. 1. Enable VLAN security VLAN security is accomplished by dropping unicast packets if the destination port is not in the same VLAN as the one associated with the source port.
6	rx_pause_frame	(R/W) 0	The *rx_pause_frame* bit controls the received pause frames. *0*. Discard the received pause frames. *1*. Forward the received pause frames like anyother packets.
5	age_tick_software	(R/W) 0	Set to generate Aging tick through software. When CPU sets this bit, aging process is initiated for one hash bucket and the age of the MA table entry in that bucket is incremented. The bit is automatically cleared at this point. Typically used in conjunction with the setting the *aging_tick_sel[2:0]* in **“Switch Configuration Register” on Page 145** to 3b’111.
4:0	monitoring_pid[4:0]	R/W 5’b 11111	All the monitored packets will be forwarded to this port corresponding to the value set on *monitoring_pid[4:0]*

7.4.4 Switch Status and Mask Register

Name: switch_status

Offset: 0x0C

Access: Read/Write (in MISC/LED)

Description

All the bits in this register should be set only one time for each event, and will be cleared after reset or CPU read. Bits [31:16] are the interrupt enable bits for the corresponding events indicated in bits [15:0]. The event bits [15:0] in this register are set when the corresponding event(s) accur. All the event bits are cleared when the CPU reads the register. The event bits are read only.

Table 67 Switch Status and Mask Register

Bit Fields	Name	(Access) Initial Value	Description
31:16	Interrupt_en[]	All 1's	Default to all 1. Each interrupt enable bit has a corresponding status bit from bit 15 to 0. All the interrupts are enabled after reset.
15	cpu_txq_avail27	(R)	Set when cpu's txq port#27 changes from "empty" to "not-empty", indicating availability of packet for the CPU to read. It is cleared when CPU reads and there is no more entry in TxQ#27.
14	cpu_txq_avail26	(R) 0	Set when cpu's txq port#26 changes from "empty" to "not-empty", indicating availability of packet for the CPU to read. It is cleared when CPU reads and there is no more entry in txq#26.
13	aged_interrupt	(R) 0	Occurs whenever there is an aging of an entry in the internal cache or the MA table
12	any_seq_err	(R) 0	Set when packet sequencing error is detected by TX/PQC. It is cleared when CPU reads the bit. If this bit is set and the corresponding *interrupt_en* bit set, external INTERRUPT output will be asserted (INTA_N). If both bits (*any_seq_err* and *interrupt_en*) are set and *dis_err_rst* is 0, the chip reset due to fatal error is activated.
11	src_pid_parity_err	(R) 0	Set when src_pid[] from SRAM is read with parity error by PQC/ARL. It is cleared when CPU reads. If this bit is set and the corresponding interrupt_en bit set, external INTERRUPT output will be asserted. If both bits are set and *dis_err_rst* is 0, the fatal error reset will be activated.

10	ma_full	0	Set when a new MA cannot find an invalid entry to replace. It is cleared when CPU reads.
9	ma_intrusion	0	Set if a received packet meets the following conditions *ma_change* set, *ma_freeze* set and the DA in the packet is non-critical. It is cleared when CPU reads this bit. For the packet resulting in *ma_intrusion* condition, should not be forwarded to the destination port.
8	ma_change	(R) 0	Set when either of the 2 conditions is satisfied 1. a new entry is created in the MA table for a newly learnt SA 2. src_pid associated with the SA has changed. It is cleared when CPU reads this bit.
7	freeq_empty	(R) 0	Set whenever FREEQ is empty i,e there are no more buffers available to store an incoming packet. It is cleared when CPU reads this bit and FREEQ is not empty.
6	Any_txdq_full	(R) 0	Set whn TXQ for any port including CPU's becomes full (when the queue hits the largest (final) watermark). It is cleared when CPU reads this bit.
5	Any_rxdq_cg_ctl	(R) 0	Set when a 10/100 Ethernet Port operating in Half duplex mode goes into congestion state. It is cleared when CPU reads this bit.
4	Any_rxdq_full_ drop	(R) 0	Set when an incoming packet is dropped because the RxQ on that port is full.(when the queue hits the largest (final) watermark). It is cleared when CPU reads this bit.
3	any_crc_error	(R) 0	Set when any of Ethernet ports receives a new packet with CRC error. It is cleared when CPU reads this bit.

2	any_fifo_run	(R) 0	Set when any of the Ethernet ports experiences RX fifo overrun or TX fifo underrun. It is cleared when CPU reads this bit.
1	any_late_ collision	(R) 0	Set when a 10/100 Ethernet port operating in half duplex mode experiences a late collision
0	any_link_ok_ change	(R) 0	Set when *link_ok* state of a port changes from either 1 => 0 or 0 => 1

7.4.5 Port Status Register

Name: port_status

Offset: 0x10-1C

Access: Read Only (in RTX)

Description

Stores the status information for the Ethernet ports. Each of 26 ports has a corresponding bit. All the **STATUS** bits will be cleared after reset or after CPU read. In all the registers described below Bit0 is for Port0, Bit1 for Port1 etc.

Table 68 Port Status Register

Bit Fields	Name	(Access) Initial Value	Description
25:0 0x1C	rxdq_full_drop	(R) 0	Set if a new received packet is dropped because RxQ is full. If any bit in this register is being set, the bit *any_rxdq_full_drop* in the **"Switch Status and Mask Register" on Page 152** is set.
25:0 0x18	crc_err	(R) 0	Set if a CRC error is detected on a received packet with length >= 64. If any of the bits in this register is set, then if a crc error is detected, *any_crc_err* bit in the **"Switch Status and Mask Register" on Page 152** is set.

25:0 0x14	late_collision	(R) 0	Set if late collision is detected. Cleared by reset or CPU read. If any bit in this register is being set, then *any_late_collision* bit in the **"Switch Status and Mask Register" on Page 152** is set.
25:0 0x10	port_active	(R) 0	Set when TXE or CRS of the corresponding port is changing to be active.

7.4.6 Port Event Register

Name: port_event

Offset: 0x20-24

Access: Read/Write

Description

Stores information about certain events for the Ethernet ports. Each of the Ethernet ports have a corresponding bit. All the status bits are cleared when the CPU reads the corresponding register.

Table 69 Port Event Register

Bit Fields	Name	(Access) Initial Value	Description
25:0 0x24	txdq_full	(R) 0	Set as long as the port's TxQ is full. It is cleared by reset or CPU read if the TxQ for that port is no longer full.
25:0 0x20	port_critical_event	(R) 0	Set when either *ma_change*, *ma_intrusion* or *ovz_pkt* event occurs. (*ovz_pkt* event occurs when an oversize packet > 1518 bytes is received) It is cleared when the CPU reads the register

7.4.7 Port Underrun/Overrun Register

Name: port_run

Offset: 0x28-2C

Access:Read

Description

Stores the information related to the Rx and Tx FIFO's for the Ethernet ports. Each of the Ethernet ports have a corresponding bit. All the status bits are cleared when the CPU reads the corresponding register.

Table 70 Port Underrun/Overrun Register

Bit Fields	Name	(Access) Initial Value	Description
25:0 0x2C	tx_fifo_underrun	(R) 0	Set if Tx fifo experienced an underrun. If any bit in this register is set, the *any_fifo_run* bit in the Switch Status and Interrupt Mask register is set. This bit is cleared by reset or CPU read.
25:0 0x28	rx_fifo_overrun	(R) 0	Set if Rx fifo experienced overrun. If any bit in this register is set, the *any_fifo_run* bit in the Switch Status and Interrupt Mask register is set. This bit is cleared by reset or CPU read.

7.4.8 Port MII Register

Name: port_mii

Offset: 0x30-3C

Access: Read/Write

Description

The values in these registers determine the link operating parameters when the *e_hw_mode* bit in **"Switch Configuration Register" on Page 145** is '0'. Otherwise the operationg parameters depend on the values read from the PHY using the MDIO interface (*use_mdio_mode* bit in Chip configuration register set to '1') or using the values obtained from the SMII interface (*use_mdio_mode* bit in **"Chip Configuration Register" on Page 141** is cleared to '0'). For gigbit ports the *use_mdio_bit* is ignored and the values are always read from the MDIO interface.

When these registers are read by the CPU, the values that are read are the ones collected from the PHY devices, either from the MDIO or SMII interface.

Table 71 Port MII Register

Bit Fields/ Address	**Name**	**(Access) Initial Value**	**Description**
23:0 0x3C	speed_cpu[]	(R/W) 1	1: FE port is running at 100 Mbit/s/ and Gbit port at 1 Gbit/s. 0: FE port is at 10 Mbit/s and Gbit port at 100 Mbit/s.
23:0 0x38	pause_e_cpu[]	(R/W) 0	Set to enable full-duplex flow control using PAUSE frames. *pause_e_cpu* enforces the flow control specified in 802.3x. As this information is not available on the SMII interface, this register also governs enabling full duplex flow control when *e_hw_mode* = 1 and *use_mdio_mode* = 0 1: Pause frame is enabled for pause frame control. 0: Pause frame is disabled for pause frame control.
23:0 0x34	full_duplex_cpu[]	(R/W) 0	1: 10/100 ports is in full duplex mode. 0: 10/100 ports is in half duplex.
23:0 0x30	Link_ok cpu[]	(R/W) 1	1: Ethernet port has link ok status (operational state). 0: Ehternet port’s link failed.

7.4.9 Port Monitor Register

Name: port_monitor

Offset: 0x40

Access: Read/Write

Description

Specifies if the packets received at or transmitted from these ports are to be monitored, i.e. forwarded to the monitoring port. Any packets from or to the ports with the '*monitored*' bit set, the packet is also forwarded to the monitoring port.

Table 72 Port Monitor Register

Bit Fields	Name	(Access) Initial Value	Description
31:27	rsv	0	Reserved
26:0	monitored[]	(R/W) 0	0 = not monitored. 1 = port needs to be monitored. Default value is 0. Bit 0 is for port 0, and bit 26 is for CPU port 0.

7.4.10 Port Priority Register

Name: port_priority

Offset: 0x44

Access: Read/Write

Description

The packet from the port or to the port will be forwarded to high priority queue if the corresponding priority bit (*pri_at_src* bit in **"Switch Configuration Register" on Page 145**) is set. Please also reference to *pri_at_src* and *pri_at_da* in the switch_config register.

Table 73 Port Priority Register

Bit Fields	Name	(Access) Initial Value	Description
31:27	rsv	0	Reserved
26:0	priority[]	(R/W) 0	Set to '1' for high priority Set to '0' for Low priority Bit 0 is for Port 0, and bit 26 is for CPU ports.

7.4.11 Port Trunk Register

Name: port_trunk

Offset: 0x48-4C

Access: Read/Write

Description

2, 4 or 8 ports can be assigned to a trunk group for 10/100 Mbit/s ports, and 2-Gbit ports can be grouped into a trunk group. Only the adjacent ports with higher *port_id*[] bits equal can be formed for a trunk group. For example, for a 4 port option, only ports 0 to 3, ports 4 to 7, or ports 20 to 23 can be in a trunk group. All the trunk ports should share the traffic as a single logic port. If any of the ports is disabled or link failure, the other ports will assume the traffic.

Table 74 Port Trunk Register

Bit Fields	Name	(Access) Initial Value	Description
31:17	rsv	0	Reserved
16 0x4C	trunk_opt	(R/W) 0	If 1, two Gbit ports are in a trunk. If 0, there is no trunk for Gbit ports.
15:0 0x4C	trunk_opt	(R/W) 0	Each port has two bits for trunk option. Bit 1:0 is for Port 16, bit 3:2 is for Port 17, … and bit 15:14 is for Port 23. The number of total ports in each trunk is same as those in ports 0 to 15.
31:0 0x48	trunk_opt	(R/W) 0	Each port has two bits for trunk option. Bit 1:0 is for Port 0, bit 3:2 is for Port 1, … and bit 31:30 is for Port 15. The number of total ports in each trunk is defined below. Note: All the ports in the same trunk should have identical trunk_opt[]. *Value* *Membership* 00 Not a member of any trunk 01 Member of a 2-port trunk 10 Member of 4 port trunk 11 Member of 8 port trunk

7.4.12 Port Bridge State Register

Name: port_bridge_state

Offset: 0x50-0x54

Access: Read/Write

Description

The Port Bridge State register specifies the forwarding state of each port as per the 802.1D specification.

Table 75 Port Bridge State Register

Bit Fields Offset	Name	(Access) Initial Value	Description
31:24 0x54	rsv	0	Reserved
23:20 0x54	bridge_state[1:0] for each CPU port. (Port 26 and 27) (2-bits per port)	(R/W) 0	00 = disable, 01 = listening, 10 = learning, 11 = forwarding. Cleared to 00 after reset. CPU port's *bridge_state[]* stays at 00 until CPU updates them.
19:0 0x54	bridge_state[1:0] for Port 25 to 16 (2-bits per port)	(R/W) 0	00 = disable, 01 = listening, 10 = learning, 11 = forwarding. Clear to 00 after reset. Bit 1 and 0 are for Port 16, bit 19 and 18 are for Port 25. Same definition as Port 15 to 0.
31:0 0x50	bridge_state[1:0] for Port 15 to 0 (2-bits per port)	(R/W) 0	00 = disable, 01 = listening, 10 = learning, 11 = forwarding. Clear to 00 after reset. Bit 1 and 0 are for Port 0, bit 31 and 30 are for port 15. At the end of the initialisation sequqnce, *bridge_state[]* is set to '11' (forwarding state) when the intelligent bit in **"Chip Configuration Register" on Page 141** is '0', otherwise it stays at '00' and must be configured by the CPU.

7.4.13 Portlist_1023 Register

Name: port_list1023

Offset: 0x58

Access: Read/Write

Description

Sepcifies the destination port list for the index 10’b1111111111. Usually, index is used to find the destination port list from FLOODMAP. However, if index is equal to 1023, it will find out the port list from portlist_1023 register instead of FLOODMAP.

Table 76 Portlist_1023 Register

Bit Fields	Name	(Access) Initial Value	Description
31:28	rsv	0	Reserved
27:26	portlist	(R/W) All 0’s	CPU Ports with the corresponding bit set will be one of the destinations for index 1023. Default to all 0’s.
25:0	portlist	(R/W) All 1’s	Ports with the corresponding bit set will be one of the destinations for index 1023.

7.4.14 Port Index Register_20_0

Name: port_ix

Offset: 0x60-78

Access: Read/Write

Description

The index number is defined for each of the ports. Each index is 10-bits wide and is shown at the respective bit positions for the different ports. The index value is used to lookup the destination ports belonging to the source ports VLAN. The number based on source port will be used to lookup FLOODMAP for the destination port list when a non-unicast packet is received. All the data are initialized to all 1’s.

For Ports 21 to 26, the Port Index Registers can be found in **“Port Index Register 26_21” on Page 184**

Table 77 Port Index Register_20_0

Bit Fields	Name	Initial value	Offset	Description
31:30	rsv	00		Reserved
29:20	port20_ix[]	All 1’s	0x78	Index Value for Port 20

19:10	port19_ix[]	All 1's	0x78	Index Value for Port 19
9:0	port18_ix[]	All 1's	0x78	Index Value for Port 18
31:30	rsv	00		Reserved
29:20	port17_ix[]	All 1's	0x74	Index Value for Port 17
19:10	port16_ix[]	All 1's	0x74	Index Value for Port 16
9:0	port15_ix[]	All 1's	0x74	Index Value for Port 15
31:30	rsv	00		Reserved
29:20	port14_ix[]	All 1's	0x70	Index Value for Port 14
19:10	port13_ix[]	All 1's	0x70	Index Value for Port 13
9:0	port12_ix[]	All 1's	0x70	Index Value for Port 12
31:30	rsv	00		Reserved
29:20	port11_ix[]	All 1's	0x6C	Index Value for Port 11
19:10	port10_ix[]	All 1's	0x6C	Index Value for Port 10
9:0	port9_ix[]	All 1's	0x6C	Index Value for Port 9
31:30	rsv	00		Reserved
29:20	port8_ix[]	All 1's	0x68	Index Value for Port 8
19:10	port7_ix[]	All 1's	0x68	Index Value for Port 7
9:0	port6_ix[]	All 1's	0x68	Index Value for Port 6
31:30	rsv	00		Reserved
29:20	port5_ix[]	All 1's	0x64	Index Value for Port 5
19:10	port4_ix[]	All 1's	0x64	Index Value for Port 4
9:0	port3_ix[]	All 1's	0x64	Index Value for Port 3
31:30	rsv	00		Reserved
29:20	port2_ix[]	All 1's	0x60	Index Value for Port 2
19:10	port1_ix[]	All 1's	0x60	Index Value for Port 1
9:0	Port0_ix[]	All 1's	0x60	Index Value for Port 0

7.4.15 DA Index Register

Name: da_index

Offset: 0x7C

Access: Read/Write

Description

Used to specify the destination port list for the flooding the packet when the DA is either unknown multicast, an unknown unicast or broadcast. Each index is 10-bit wide, and it will be used to find the destination port list from FLOODMAP. Bit 23, 15, and 7 are reserved bits.

Table 78 DA Index Register

Bit Fields	Name	(Access) Initial Value	Description
31:30	rsv	0	Reserved
29:20	unknown_flood_ ix	(R/W) all 1's	This index is used when the DA is a multicast (i.e. bit 40 equal to 1) and is not found in MA table.
19:10	unknown_ucast_ ix	(R/W) all 1's	This index is used when the DA is an unicast (i.e. bit 40 equal to 0) and is not found in MA table.
9:0	bcast_ix	(R/W) all 1's	This index is used when DA's bytes are equal to all 1's.

7.4.16 Memory Upper Address Register

Name: mem_u_addr

Offset: 0x80

Access: Read/Write

Description

This register is used to store the upper address of the memory. It should be combined with the lower bits in mem_access register to have the complete address offset. The final memory address for each access is concatenated from mem_addr[20:11] from mem_u_addr register and mem_addr[10:1] from mem_access register.

Table 79 Memory Upper Address Register

Bits Field	Name	Initial Value	Description
31:25	rsv	0	Reserved

24:21	blk_sel_msb [3:0]	(R/W) x	The value of this field indicates which memory will be accessed for CPU to READ or WRITE mem_access register. The definition of memory access is described below: **0 - 1: to select EDRAM block #0 to #1**, respectively. **2 - 7: reserved**. **8: to select the memory for MIB counters**. **9: to select the memory for ARL data base**. **10: to select the memory for PQC link list or queues**. **11:15: reserved**.
20:11	mem_addr [20:11]	(R/W) x	Higher order address bits (Upper address of memory access). The definition of each access is described in blk_sel_msb[] field above.
10:0	rsv	0	Reserved

The description of the blk_sel_msb[3:0] bits is given below:

The value of this filed indicates which memory will be accessed for CPU to READ or WRITE mem_access register.

0 - 1: to select EDRAM block #0 to #1, respectively. This area is for diagnostic purpose only.

2 - 7: reserved.

8: to select the memory for MIB counters. Mem_addr[20:13] is ignored. Each counter has a unique address defined in mem_addr[12:2]. *Mem_addr[12:2]* consists *{rx_cnt_sel, ‘0’, port_id[4:0], item_num[3:0]}*. *rx_cnt_sel* = 1 for accessing receive MIB counter, and *rx_cnt_sel* = 0 for accessing the MIB transmit counters.

port_id[] is the logical port number for each Ethernet port or CPU port.

And *item_num[]* is defined in Network Management section, which helps in determining the offset for each counter (offset = 4 * item_num).

9: to select the memory for ARL data base.

mem_addr[19:18] is used to select one of the memories

mem_addr[19:18] = 00 : to select ssram 20 k x 32 (used only for diagnostics purposes)

mem_addr[19:18] = 01 : to select internal small ARL cache (used only for diagnostics purposes) *mem_addr[19:18] = 10* : to select vlan ix table 4 k x 10

mem_addr[19:18] = 11 :to select vlan table 1 k x 40

10: to select the memory for PQC link list or queues.

mem_addr[17:15] is used to select one of the memory inside PQC.

mem_addr[17:15] = 000: to select PBL memory in external SSRAM

mem_addr[17:15] = 001: to select internal FloodMap

mem_addr[17:15] = 010 – 011: reserved

mem_addr[17:15] = 100 – 111: to select BCASTQ0 to BCASTQ3 respectively

These area except FloodMap are for diagnostic purpose only.

11:15: reserved.

7.4.17 Memory Access Register

Name: mem_access

Offset: 0x84

Access: Read/Write

Description

After CPU writes this register, a memory access state machine, either Read or Write process, will be activated. This register is used for all the device access including all the memories, and external MII's PHY register. Each access can only be 16-bits. For 32-bits memory data, it is in little endian format. So, the first two bytes are from bit's[15:0] and the second two bytes of data with *mem_addr*[1] equal to 1 are from bit's[31:16].

Table 80 Memory Access Register

Bits Field	Name	Access / Initial Value	Description
31	access_rdy	(R/W) 0	When reading data, this bit indicates that the data is available for the CPU to read. When writing data, this bit indicates that previous write operation is complete and the CPU can write to this register. CPU should clear this bit by writing to '0' before initiating a read or write operation
30	wr	(R/W) 0	This bit is set by CPU to initiate a write operation.
29	rd	(R/W) 0	This bit is set by CPU to initiate a read operation.

28:27	mem_sel	(R/W) 00	00 to select MII's PHY. 01 – reserved. 10 – reserved. 11 for memory access. Upper address bits are needed. Refer to memory upper address register for them.
26	reserved	0	Reserved
25:16	mem_addr [10:1]	(R/W) X	The description of this field is given below the table.
15:0	mem_data [15:0]	(R/W) x	Write data for Write command, and Read data for Read command.

mem_addr[10:1]

Memory access lower address bits.

For each memory Write access, 2-bytes per access will be temporarily stored into a 16-byte register with the offset given by mem_addr[3:1]. The real memory transfer operation won't start until mem_addr[3:1] stands for the last access, i.e. equal to 'e'. *access_rdy* bit won't be set until the receiving memory has acknowledged.

For each 16-bytes of memory Write, there are eight Write's for this register and the value for Bit's[3:1] for the first write is 0. After receiving the eighth Write with mem_addr[3:0] equal to 'e', the memory operation starts.

For a 16-bytes of Read, PLB 2224 issues a command to request for 16-bytes of data from memory after receiving a Read command with mem_addr[3:1] equal to 0. CPU then polls this register for mem_data[] until *access_rdy* is asserted. After detecting *access_rdy*, CPU reads the first two bytes. It continues to issue the Read command with mem_addr[3:1] as a continuous offset, and then polls for *access_rdy* after which *mem_data[]* is valid. During the subsequent polling, *access_rdy* should be always ready because all the 16-bytes of the data are ready when the command with mem_addr[3:1] equal to 0.

For MII's PHY access, each Read or Write command is to access 2-bytes of data. For each Write command, the state machine starts to write mem_data[] onto the external device. For each Read command, the state machine starts to get the data onto mem_data[]. *access_rdy* is asserted when the read or write command is completed.

For MII's PHY register access, mem_addr[10:6] is for PHY device ID and mem_addr[5:1] is for PHY's internal register address.

Details on Indirect memory accesses can be found in **"Indirect Access to the Memories" on Page 62**

7.4.18 CMAC Data Register

Name: cmac_data

Offset: 0x90-0x9C

Access: R/W

Description

When this register is written using any of the four addresses, the data is pushed into CPU Rx FIFO. When this register is read using any of the four addresses, the data is popped from CPU Tx FIFO.

Table 81 CMAC Data Register

Bit Fields	Name	Initial Value	Description
31:0	cmac_data	x	CMAC data register. data[31:0] for CPU Rx FIFO or from CPU Tx FIFO.

7.4.19 CMAC RX Register

Name: cmac_rx

Offset: 0xA0

Access: R/W

Description

Used by the CPU to send packet parameters to PLB 2224

Table 82 CMAC RX Register

Bit Fields	Name	Access/Initial Value	Description
31	crx_cpu_pktid_rdy	(R) 0	1: indicates that CMAC Rx. is ready to accept a new packet from the CPU. This bit should be cleared before CPU writes the last byte of data.
30	crx_cpu_fifo_rdy	(R) 0	1: indicating CMAC RX is ready for CPU to write at least 16 bytes of data.
29:26	rsv	0	Reserved

25:16	filter_match26 [9:0]	(R) x	Matching status for filtering patterns for each packet being forwarded to Port 26, i.e. CPU port. Bit 9 indicates whether all the patterns in group #3 have been matched. Similarly, bit 8 is for group #2. Bit #7 to #4 are the matching results, respectively, for the four patterns #7 to #4 being defined within group#1. Similarly, bit #3 to #0 are for group#0. The field and pbh26[] are valid when cpu_txq_rd_req[26] in cmac_tx0 is cleared.
15:7	rsv	0	Reserved
6	e_cpu_pkt_ padding	(W/R) 0	1: CRX to append the packet to packet length to 64-bytes if less. 0: no padding for small size packet.
5	crx_crc_gen	(W/R) 0	Indicates whether the packet data excludes CRC status. Usually, it is 1 for the packets from CPU port, and destination port's TX will append CRC bytes at the end of packet data.
4	crx_crc_err	(W/R) 0	Indicates the CRC error status for the current packet. This is generally 0 for packets received from the CPU port.
3	crx_sof	(W/R) 0	Indicates that the next data write is the first chunk of the newpacket. This bit is always cleared when packet data is written.
2	crx_eof	(W/R) 0	Indicates that the next data write is the last chunk of data for the current packet. This bit should be set before the last data is written.
1:0	crx_bytecnt[1:0]	(W/R) 00	Indicates the number of data bytes to be written. This field should be set correctly before the last data write. 00 = 4-bytes to be written 01 = 3-bytes to be written 10 = 3-bytes to be written 11 = 1-byte to be written

7.4.20 CMAC TX Register

Name: cmac_tx0

Offset: 0xA4

Access: R/W

Description

Stores the packet buffer header for the current packet available in the TX queue for Port 26 (CPU Port0), as well as some other flags required for the transmit function for both Ports 26 and 27 i.e. CPU Port0, CPU Port1.

Table 83 CMAC TX Register

Bit Fields	Name	Access/Initial Value	Description
31:30	ctx_txq_avail [27:26]	(R) 0	1: indicates one packet being received from Ethernet ports is ready for CPU to process. There are two bits for two CPU logical ports.
29	ctx_cpu_fifo_rdy [27:26]	(R) 0	1: CTX has at least 16 bytes or eof bytes ready for CPU to read. 0: CTX has no data available for CPU to read Which CTX port was selected depends on *cpu_txq_rd_req* bit setting
28:24	rsv		
23:22	cpu_txq_rd_req [27:26]	(R/W) 0	Set by CPU to collect the packet data from one of CPU's ports. It is cleared when PBH[] for the corresponding port is read from SRAM.
21	ctx_flush	(R/W) 0	The current processing packet data in CTX will be flushed when this bit toggles from 0 to 1.
20:0	pbh26[20:0]	(R) 0	Current PBH[] data for Port 26.

7.4.21 CMAC_TX1

Name: cmac_tx1

Offset: 0xA8

Access: R

Description

Stores the packet buffer header for the current packet available in the Tx queue for Port 27 (CPU Port 1)

Table 84 CMAC_TX1 Register

Bit Fields	Name	Access/Initial Value	Description
31:22	filter_match27 [9:0]	(R)	Similar to filter_match26[] above, matching status for filtering patterns for each packet being forwarded to Port 27, i.e. CPU port. The field and pbh27[] are valid when cpu_txq_rd_req[27] in cmac_tx0 is cleared.
21	rsv	0	Reserved
20:0	pbh27[20:0]	(R) 0	Current PBH[] data for Port 27.

7.4.22 ARL Register

Name: arl

Offset: 0xAC

Access: R/W

Description

This register is only for the packets being received from Port 26 or CPU port 0.

Table 85 ARL Register

Bit Fields	Name	Access/Initial value	Description
31	e_da_lookup	(R/W) 0	When set to 1, the ARL does normal DA lookup from the MA Table for destination information. When cleared to 0, ARL does not do normal DA lookup; instead it uses *dst_ports*, *dst_critical* and *dst_pri* as the result of the lookup.

30	e_ma_learn	(R/W) 0	When set to 1, the ARL creates or updates the MA Table entries using the SA contained in the received packet, and the values of the *ma_ports*, *ma_locked, ma_critical, and ma_pri* fields in this register. If *ma_locked = 0*, the *ma_state* is set to 3'b100. If *ma_delete* = 1, the matched MA Table is deleted. When e_ma_learn = 0: ARL does not perform address learning on packets received from the CPU port.
29	dst_critical	(R/W) 0	Used as the lookup result if *e_da_lookup* is not asserted. This bit is ignored if *e_da_lookup* is 1.
28	dst_pri	(R/W) 0	Used as the lookup result if *e_da_lookup* is not asserted. This bit is ignored if *e_da_lookup* is 1.
27	ma_delete	(R/W) 0	When set to 1, the matched entry in the MA Table is deleted if *e_ma_learn* = 1. This bit is ignored if *e_ma_learn* = 0.
26	ma_locked	(R/W) 0	Used as the learned MA entry's attribute '*ma_locked*' if *e_ma_learn* is asserted. This bit is ignored if *e_ma_learn* is 0.
25	ma_critical	(R/W) 0	Used as the learned MA entry's attribute '*ma_critical'* if *e_ma_learn* is asserted. This bit is ignored if *e_ma_learn* is 0.
24	ma_pri	(R/W) 0	Used as the learned MA entry's attribute '*ma_pri*' if *e_ma_learn* is asserted. This bit is ignored if *e_ma_learn* is 0.
23:22	rsv	0	Reserved

21:11	dst_ports [10:0]	(R/W) 0	Used as the lookup result as the destination port information (either as port id or index into the VLAN table, depending on whether the SA is unicast or multicast address) Bit 21 if set to ’0’ then *dst_ports[9:0]* is for port_index and if Bit 21 is set to a ’1’ then *dst_ports[9:0]* correspond to flood_ix. This field is ignored if *e_da_lookup* is 1.
10:0	ma_ports [10:0]	(R/W) 0	Used as the destination port attribute (either as port_id or index into the VLAN Table, depending on whether the SA is unicast or multicast address) for creating MA Table entry when e_ma_learn = 1. This field is ignored if *e_ma_learn* is ’0’.

7.4.23 FREEQ Register

Name: freeq_ptr

Offset: 0xB0

Access: Read only

Description

For test purposes only. This is used to access read and write pointers for the free packet buffer pool.

Table 86 FREEQ Register

Bit Fields	Name	(Access) Initial Value	Description
31:29	rsv	0	Reserved
28:16	freeq_wptr	(R) 0	The current FREEQ write pointer. (Used for test purposes only)
15:13	rsv	0	Reserved
12:0	freeq_rptr	(R) 0	The current FREEQ read pointer. (Used for test purposes only)

7.4.24 FREEQ_CNT

Name: freeq_cnt

Offset: 0xB4

Access: Read only

Description

For test purposes only. This contains the count of buffers currently available in the free packet buffer pool.

Table 87 FREEQ_CNT

Bit Fields	Name	Access/Initial Value	Description
31:12	rsv	0	Reserved
11:0	freeq_cnt [11:0]	(R) 0	The current available PB count in FREEQ.

7.4.25 CPU_TXQ26 Count Register

Name: cpu_txq26

Offset: 0xB8

Access: Read only

Description

Maintains the count for the packets in the HIGH and LOW priority Tx. queues for Port 26 i.e. CPU Port 0

Table 88 CPU TXQ26 Count Register

Bit Fields	Name	Access/Initial Value	Description
31:29	rsv	0	Reserved
28:16	cpuq_cnt_h26	(R) 0	Current txq count for CPU Port 26 in high priority
15:13	rsv	0	Reserved
12:0	cpuq_cnt_l26	(R) 0	Current txq count for CPU Port 26 in low priority

7.4.26 CPU_TXQ27

Name: cpu_txq27

Offset: 0xBC

Access: Read only

Description

Maintains the count for the packets in the HIGH and LOW priority Tx. queues for Port 27. i.e. CPU Port 1

Table 89 CPU_TXQ27 Register

Bit Fields	Name	Access/Initial Value	Description
31:29	rsv	0	Reserved
28:16	cpuq_cnt_h27	(R) 0	Current txq count for CPU Port 27 in high priority
15:13	rsv	0	Reserved
12:0	cpuq_cnt_l27	(R) 0	Current txq count for CPU Port 27 in low priority

7.4.27 CPU Water Mark Register

Name: cpu_wm_cnt

Offset: 0xC0

Access: R/W (PQC)

Description

Water Mark registers are to specify the limit on the receiving queue and transmit queue for each ports and helps in the flow control. These count limits are for each port to compare its own resource usage not to exceed the limit.

The watermark register values can be multiplied by 4 or 1, depending on the *watermark_scale* bit setted in switch_config register. By default the value is set to a scaling of 4 for the watermark register value vs. the limit on the receiving queue / transmit queue.

Table 90 CPU Water Mark Register

Bit Fields	Name	(Access) Initial Value	Description
31:24	rsv	0	Reserved
23:16	ctx_cnt_3[7:0]	(R/W) 0x28	The ctx_cnt_3 field is the turn-off watermark for the CPU port. The ctx_cnt_3 is the cutoff threshold for the CPU port to not accept any packet if either the low or high priority packet counts rise to the value of ctx_cnt_3.
15:8	ctx_cnt_2[7:0]	(R/W) 0x20	The ctx_cnt_2 field is the critical packet watermark for the CPU port. The ctx_cnt_2 is the threshold for dropping non-critical packets when packet count rises. The CPU port accepts only critical packets when either low or high priority packet counts rises to the value of ctx_cnt_2. The ctx_cnt_2 is the threshold for accepting critical packets only when packet count drops. When both low and high priority packet counts drop below ctx_cnt_2 (inclusive), the CPU port starts accepting critical packets.
7:0	ctx_cnt_1[7:0]	PQC (R/W) 0x18	The ctx_cnt_1 field is the non-critical packet watermark for the CPU port. The ctx_cnt_1 is the threshold for accepting non-critical packets (in addition to the critical packets) when both low and high priority packet counts drop below ctx_cnt_1 (inclusive) from above.

7.4.28 FE Watermark Control (TX)

Name: fe_wm_cnt0 (FE Tx watemark control)

Offset: 0xC4

Access: R/W (PQC)

Description

The watermark registers specify the limit on the receiving queue and transmit queue for each ports. These count limits are for each port to compare its own resource usage not to exceed the limit. While comparing, bits 3 to 0 of each counter are ignored. Used for the flow control and can be scaled depending on the *watermark_scale* bit.

Table 91 FE TX Watermark Control Register

Bit Fields	Name	(Access) Initial Value	Description
31:24	Rsv	0	Reserved
23:16	tx_cnt_3[7:0]	(R/W) 0x28	The tx_cnt_3 field is the turn-off watermark for the FE port. The tx_cnt_3 is the cutoff threshold for the FE port to not accept any packet if either low or high priority packet counts rise to the value of tx_cnt_3.
15:8	Tx_cnt_2 [7:0]	(R/W) 0x20	The tx_cnt_2 field is the critical packet watermark for the FE port. The tx_cnt_2 is the threshold for dropping non-critical packets when packet count rises. FE port accepts only critical packets when either low or high priority packet counts rises to the value of ctx_cnt_2. The tx_cnt_2 is the threshold for accepting critical packets only when packet count drops. When both low and high priority packet counts drop below tx_cnt_2 (inclusive), FE port starts accepting critical packets.
7:0	Tx_cnt_1 [7:0]	(R/W) 0x18	The tx_cnt_1 field is the non-critical packet watermark for the FE port. The tx_cnt_1 is the threshold for accepting non-critical packets (in addition to the critical packets) when both low and high priority packet counts drop below tx_cnt_1 (inclusive) from above.

7.4.29 FE Watermark Control (RX)

Name: fe_wm_cnt1 (FE Rx watermark control)

Offset: 0xC8

Access:R/W (PQC)

Description

Maintains watermark values for Rx queues for the 10/100 ports. Used for the flow control and can be scaled depending on the *watermark_scale* bit.

Table 92 FE Rx Watermark Control Register

Bit Fields	Name	(Access) Initial Value	Description
31:24	rx_cnt_4 [7:0]	(R/W) 0x12	The rx_cnt_4 field is the turn-off watermark for the FE port. The rx_cnt_4 is the cutoff threshold for the FE port to not accept any packet if packet counts rise to the value of rx_cnt_4.
23:16	rx_cnt_3 [7:0]	(R/W) 0x10	The rx_cnt_3 field is the critical packet watermark for the FE port. The rx_cnt_3 is the threshold for dropping non-critical packets when packet count rises. The FE port accepts only critical packets when packet counts rises to the value of rx_cnt_3. The rx_cnt_3 is the threshold for accepting critical packets only when packet count drops. When packet counts drop below rx_cnt_3 (inclusive), FE port starts accepting critical packets.

15:8	rx_cnt_2 [7:0]	(R/W) 0x0C	The rx_cnt_2 field is the flow control on watermark for the FE port. The rx_cnt_2 is the threshold for starting flow control when packet count rises. The FE port sends a pause frame with ‘0xffff’ when packet counts rises to the value of rx_cnt_2. The rx_cnt_2 is the threshold for accepting non-critical packets when packet counts drops. When packet counts drop below rx_cnt_2 (inclusive), FE port starts accepting non-critical packets.
7:0	rx_cnt_1 [7:0]	(R/W) 0x08	The rx_cnt_1 field is the flow control off watermark for the FE port. The rx_cnt_1 is the threshold for end of flow control when both packet counts drop below rx_cnt_1 (inclusive) from above. The FE port sends a pause frame with ‘0x0’ when packet counts drop below rx_cnt_1.

7.4.30 Gport Watermark Control (TX)

Name: ge_wm_cnt0 (Gport Tx watermark control)

Offset: 0xCC

Access: R/W (PQC)

Description

Maintains watermark values for Tx queues for the Gigabit ports. Used for the flow control and can be scaled depending on the *watermark_scale* bit.

Table 93 Gport TX Watermark Control Register

Bit Fields	Name	(Access) Initial Value	Description
31:24	Rsv	0	Reserved
23:16	gtx_cnt_3 [7:0]	(R/W) 0x90	The gtx_cnt_3 field is the turn-off watermark for the Gigabit port. The gtx_cnt_3 is the cutoff threshold for the Gigabit port to not accept any packet if either low or high priority packet counts rise to the value of gtx_cnt_3.

15:8	gtx_cnt_2 [7:0]	(R/W) 0x80	The gtx_cnt_2 field is the critical packet watermark for the Gigabit port. The gtx_cnt_2 is the threshold for dropping non-critical packets when packet count rises. The Gigabit port accepts only critical packets when either low or high priority packet counts rises to the value of gtx_cnt_2. The gtx_cnt_2 is the threshold for accepting critical packets only when packet count drops. When both low and high priority packet counts drop below gtx_cnt_2 (inclusive), the Gigabit port starts accepting critical packets.
7:0	gtx_cnt_1 [7:0]	(R/W) 0x70	The gtx_cnt_1 field is the non-critical packet watermark for the Gigabit port. The gtx_cnt_1 is the threshold for accepting non-critical packets (in addition to the critical packets) when both low and high priority packet counts drop below gtx_cnt_1 (inclusive) from above.

7.4.31 Gport Watermark Control (RX)

Name: ge_wm_cnt1 (Gport RX watermark control register)

Offset: 0xD0

Access: R/W (PQC)

Description

Maintains watermark values for Rx queues for the Gigabit ports. Used for the flow control and can be scaled depending on the *watermark_scale* bit.

Table 94 Gport RX Watermark Control Register

Bit Fields	Name	Access/Initial Value	Description
31:24	grx_cnt_4[7:0]	(R/W) 0x8C	The grx_cnt_4 field is the turn-off watermark for the Gigabit port. The grx_cnt_4 is the cutoff threshold for the FE port to not accept any packet if packet counts rise to grx_cnt_4.

23:16	grx_cnt_3 [7:0]	(R/W) 0x70	The grx_cnt_3 field is the critical packet watermark for the Gigabit port. The grx_cnt_3 is the threshold for dropping non-critical packets when packet count rises. The Gigabit port accepts only critical packets when packet counts rises to the value of grx_cnt_3. The grx_cnt_3 is the threshold for accepting critical packets only when packet count drops. When packet counts drop below grx_cnt_3 (inclusive), the Gigabit port starts accepting critical packets.
15:8	grx_cnt_2 [7:0]	(R/W) 0x64	The grx_cnt_2 field is the flow control on watermark for the Gigabit port. The grx_cnt_2 is the threshold for starting flow control when packet count rises. The Gigabit port sends a pause frame with ‘0xffff’ when packet counts rises to the value of grx_cnt_2. The grx_cnt_2 is the threshold for accepting non-critical packets when packet counts drops. When packet counts drop below grx_cnt_2 (inclusive), the Gigabit port starts accepting non-critical packets.
7:0	grx_cnt_1 [7:0]	(R/W) 0x48	The grx_cnt_1 field is the flow control off watermark for the Gigabit port. The grx_cnt_1 is the threshold for end of flow control when both packet counts drop below grx_cnt_1 (inclusive) from above. The Gigabit port sends a pause frame with ‘0x0’ when packet counts drop below grx_cnt_1.

7.4.32 Tag Priority Table

Name: **tag_pri_table**

Offset: 0xD4

Access: Read/Write

Description

Tag priority table contains 8 entries with one bit each entry. It converts 8 priority classes of the received tagged frame to 2 priority classes (High priority and Low priority). For tagged packets, pri[] field will be used to determine the queue class of the egress packet.

Table 95 Tag Priority Table Register

Bit Fields	Name	Block (Access) Initial Value	Description
31:8	Rsv	0	Reserved
7	Tag_pri_7	(R/W) 1	
6	Tag_pri_6	(R/W) 1	
5	Tag_pri_5	(R/W) 1	
4	Tag_pri_4	(R/W) 1	
3	Tag_pri_3	(R/W) 0	
2	Tag_pri_2	(R/W) 0	
1	Tag_pri_1	(R/W) 0	
0	Tag_pri_0	(R/W) 0	

7.4.33 Egress Priority Table

Name: e_pri_table

Offset: 0xD8

Access: Read/Write

Description

Egress priority table contains two entries with 3-bits per entry. It converts the two internal priority classes to one of the 8 priority classes of tagged frame. This determines the pri[] field of the tag control field if the Output tagging is needed and the ingress packet has no tag.

Table 96 Egress Priority Table

Bit Fields	Name	(Access) Initial Value	Description
31:6	Rsv	0	Reserved
5:3	E_pri_1	(R/W) 3’b110	A internal high priority queue data is translated to the value set the 3 bits
2:0	E_pri_0	(R/W) 3’b010	A internal low priority queue data is translated to the value set the 3 bits

7.4.34 Port Cfi Register

Name: port_cfi

Offset: 0xDC

Access: Read/Write

Description

This bit is used as cfi in the tag control field if the Output tagging is needed and ingress packet has no tag Default set to ’0’

Table 97 Port Cfi Register

Bit Fields	Name	(Access) Initial Value	Description
31:29	Rsv	0	
27:0	Port_cfi[27:0]	(R/W) 0	

7.4.35 VLAN Aware/InTag Control Register

Name: vlan_aware/InTag control register

Offset: 0xE0

Access: Read/Write

Description

This register is used to indicate the status of the switch's VLAN awareness and also the type of frame that are accepted.

Table 98 VLAN Aware/inTag Control Register

Bit Fields	Name	Access/Initial Value	Description
31:29	Rsv	0	
28	Vlan_aware	(R/W) 0	0 : Indicates a VLAN Unaware switch 1 : Indicates VLAN aware switch, which means that the switch will follow 802.1Q with tagging and detagging capability.
27:0	Admit_tagged_only [27:0]	(R/W) 0	0: Accept tagged and untagged frame 1: Only tagged frame will be passed through, untagged frame will be dropped If this parameter is set to *Admit Only VLAN-tagged frames*, any frames received on that Port that carry no VID (i.e., untagged frames or priority-tagged frames) are discarded by the ingress rules

7.4.36 VLAN Ingress Filter

Name: VLAN Ingress filter

Offset: 0xEC

Access: Read/Write

Description

Enables the Ingress filtering parameter on a per port basis.

Table 99 VLAN Ingress Filter

Bit Fields	Name	(Access) Initial Value	Description
31:28	Rsv	(R/W) 0	
27:0	Ingress Filter [27:0]	(R/W) 0	*0*:Disable ingress filtering *1*:Enable Ingress filtering An Enable Ingress Filtering parameter is associated with each Port. If the Enable Ingress Filtering parameter for a given Port is set, as per the ingress rules (8.6) as mentioned in 802.1Q document, shall discard any frame received on that Port whose VLAN classification does not include that Port in its Member set

7.4.37 Port Index Register 26_21

Name: port_ix21-23,port_ix24-26

Offset: 0xE4 - 0xE8

Access: Read/Write

Description

The index number is defined for each of the ports. The number based on source port will be used to lookup FLOODMAP for the destination port list when a non-unicast packet is received. All the data are initialized to all 1's. Port_vid_ix[9:0] of each port is assigned as the default VLAN vid_ix[9:0] for a port based VLAN, for the packets being received from the corresponding source port.

For Ports 20 to 0, the Port Index Register can be found in **"Port Index Register_20_0" on Page 161**

Table 100 Port Index Register 26_21

Bit Fields	Name	Offset	Description
31:30	Rsv	0xE8	Reserved
29:20	Port26_ix[]	0xE8	Index Value for Port 26

19:10	Port25_ix[]	0xE8	Index Value for Port 25
9:0	Port24_ix[]	0xE8	Index Value for Port 24
31:30	Rsv	0xE4	Reserved
29:20	Port23_ix[]	0xE4	Index Value for Port 23
19:10	Port22_ix[]	0xE4	Index Value for Port 22
9:0	Port21_ix[]	0xE4	Index Value for Port 21

7.4.38 LED Data Register

Name: led_data

Offset: 0xF0-FC

Access: Writeable

Description

32-bits of LED display data if CPU has the control of LED.

Table 101 LED Data Register

Bit Fields	Name	Access/Initial Value	Description
31:17 0xFC	Rsv	0	Reserved
16 0xFC	Soft_led	(W) 0	Set to let CPU control the display of the LED bits for the Ethernet ports
15:12	Rsv	0	Reserved
11:8 0xFC	cpu_led[3:0]	(W) x	4-bits of LED data for the CPU Ports. 1 = on, 0=off. Unlike the LED bits for the Ethernet ports, these LED Bits are always controlled by software
7:0 0xFC	led_d[25:24] [3:0]	(W) x	4-bits of LED data for Port 25 (GPort1) and Port 24 (GPort0) 1 = on, 0 = off
31:0 0xF8	led_d[23:16] [3:0]	(W) x	4-bits for LED data for Port 23 through Port 16. 1 = on, 0 = off

31:0 0xF4	led_d[15:8] [3:0]	(W) x	4-bits for LED data for Port 15 through Port 8. 1 = on, 0 = off
31:0 0xF0	led_d[7:0] [3:0]	(W) x	4-bits for LED data for Port 7 through Port 0. 1 = on, 0 = off

7.4.39 PATTERN Registers

Name: pattern_mask[7:0]

Offset: 0x100 – 0x11C

Access: R/W

Description

The register with address offset of 0x100 is for pattern and mask #0 in group #0, the next one with address offset of 0x104 is for pattern and mask #1 in group #0, the next register with offset of 0x108 is for pattern and mask #2 in group #0, the register with address offset of 0x10C is for pattern and mask #3 in the group #0, the register with address offset of 0x110 is for pattern and mask #0 in group #1, the register with address offset of 0x114 is for pattern and mask #1 in group #1, the register with offset of 0x118 is for pattern and mask #2 in group #1…and the last one with address offset of 0x11C is for pattern and mask #3 in group #1.

There are eight patterns totally in PLB 2224, which can be configured by the user.

Table 102 PATTERN Registers

Bit Fields	Name	(Access) Initial Value	Description
31:16	pattern[15:0]	(R/W) x	Pattern[] is used to compare with each incoming packet data. Each pattern[] has mask[] as companion.
15:0	mask[15:0]	(R/W) x	*1*: the corresponding bit in pattern[] is skipped for the pattern comparison, and is considered to be compared successfully. *0*: the corresponding bit in pattern[] needs to be compared for the comparison result. Note: If mask[] is equal to all 1's, the whole pattern is always matched.

7.4.40 OFFSET Registers

Name: **offset_03 (offset_group0)**

Offset: 0x120

Access: R/W

Description

Specifies half-word offset within the packet for the 2B packet data that has to be compared with the pattern

Table 103 OFFSET Group0 Registers

Bit Fields	Name	(Access) Initial Value	Description
31:29	Rsv	0	Reserved
28:24	offset3[5:1]	(R/W) x	offset3[] specifies the starting Half word location within each packet for pattern #3 to compare.
23:21	Rsv	0	Reserved
20:16	offset2[5:1]	(R/W) x	offset2[] specifies the starting Half word location within each packet for pattern #2 to compare.
15:13	rsv	0	Reserved
12:8	offset1[5:1]	(R/W) x	offset1[] specifies the starting Half word location within each packet for pattern #1 to compare.
7	comp_le0	(R/W) x	1: Use “LESS THAN or EQUAL” as the condition for comparison result for pattern #0. 0: Use “EQUAL” as the condition for comparison result. Note: only pattern #0 and pattern #4 have this option bit.
6:5	rsv	0	Reserved
4:0	offset0[5:1]	(R/W) x	offset0[] specifies the starting Half word location within each packet for pattern #0 to compare.

7.4.41 Offset Group Register

Name: offset_47 (offset_grp1)

Offset: 0x124

Access: R/W

Description

Specifies the half-word offset within the packet for the 2B packet data that is to be compared with pattern[]

Table 104 Offset Group1 Register

Bit Fields	Name	Access/Initial Value	Description
31:29	rsv	0	Reserved
28:24	offset7[5:1]	(R/W) x	offset7[] specifies the starting Half word location within each packet for pattern #7 to compare.
23:21	rsv	0	Reserved
20:16	offset6[5:1]	(R/W) x	offset6[] specifies the starting Half word location within each packet for pattern #6 to compare.
15:13	rsv	0	Reserved
12:8	offset5[5:1]	(R/W) x	offset5[] specifies the starting Half word location within each packet for pattern #5 to compare.
7	comp_le4	(R/W) x	1: Use “LESS THAN or EQUAL” as the condition for comparison result for pattern #4. 0: Use “EQUAL” as the condition for comparison result. Note: only pattern #0 and pattern #4 have this option bit.
6:5	rsv	0	Reserved
4:0	offset4[5:1]	(R/W) x	offset4[] specifies the starting Half word location within each packet for pattern #4 to compare.

7.4.42 OP_Table Registers

Name: op_table0/1

Offset: 0x128- 0x12C

Access: R/W

Description

The register with Offset 0x128 is the group op code table which is used to convert the results of the pattern #3 to pattern #0 into 2-bits. Similarly the register with Offset 0x12C is for pattern #7 to #4.

Table 105 OP_Table Registers

Bit Fields	Name	Access/Initial Value	Description
31:0 0x12C	op_table1[31:0]	(R/W) x	Similar to op_table0[]. op_table1[] is for pattern #4 to pattern #7.
31:0 0x128	op_table0[31:0]	(R/W) x	There are four inputs representing the comparison results from pattern #3 to pattern #0, respectively. The most significant bit is from pattern #3 result. Four bit inputs are decoded onto 16 select lines from 0 to 15. There are 16 sets of 2-bit op codes, i.e. 32-bits totally. Each 2-bit op code is selected by the value of the four inputs. The first set, bit[1:0], are selected by the select line #0. The second set, bit[3:2], are for line #1. And, the last set, bit[31:30], are for line #15.

7.4.43 Action_Table Registers

Name: action_table0-15

Offset: 0x130- 0x16C

Access: R/W

Description

These registers are to use the combined group_op_code[5:0] as the inputs to select one set of action bits. There are six inputs which are decoded to 64 select lines from 0 to 63. Each select line select one set of action bits starting from the register 0x130.

Note: The description below only show the select lines from 0 to 3. Similarly, for select lines from 4 to 7 are used to select the action4 to action7[] which are stored in the register with offset 0x134. Continuing in the same way the last four select lines from 60 to 63 are to select the action60 to action63 in the register with offset 0x16C.

Table 106 Action_Table Registers

Bit Fields	Name	Access/Initial Value	Description
31:29	rsv	0	Reserved
28:24 0x130	action3[4:0]	(R/W) x	Action3[] are selected as the packet's action bits, when select line #3 is asserted.
23:21	rsv	0	Reserved
20:16 0x130	action2[4:0]	(R/W) x	Action2[] are selected as the packet's action bits, when select line #2 is asserted.
15:13	rsv	0	Reserved
12:8 0x130	action1[4:0]	(R/W) x	action1[] are selected as the packet's action bits, when select line #1 is asserted. *Note: The definition of action bits are identical to those in bit 4:0 field.*

7:5	rsv	0	Reserved
4:0 0x130	action0[4:0]	(R/W) x	action0[] are selected as the packet's action bit,s when select line #0 is asserted. The action bits now are defined below. bit 4: inc_filter_cnt. If asserted, the port with packet received increments its filter_cnt[]. bit 3: pkt_drop. If asserted, the packet is forced to drop instead of forwarding to any port. bit 2: fwd_cpu. If fwd_cpu and ~(pkt_drop), the packet will be forced to forward to filter_fwd_port[] port instead of the normal destination ports from lookup result. bit 1: monitor. Add 'monitor' attribute to the packet so that it will also be forwarded to the monitor port besides the destination ports from lookup result. bit 0: pri. Add 'pri' attribute to the packet so that the packet will be forwarded through high priority queue.

7.4.44 Enable Filtering Register

Name: e_filter

Offset: 0x170

Access: R/W

Description

Specifies the ports that are enabled for packet filtering functionality. Also specifies the port_id used for the forwarding the filtered packets.

Table 107 Enable Filtering Register

Bit Fields	Name	Access/Initial Value	Description
31:27	filter_fwd_port [4:0]	(R/W) x	If fwd_cpu action bit is asserted, the packet will be forwarded to the port assigned by this register instead of the normal destination ports from lookup result.
26:0	e_filter[]	(R/W) 0	1: to enable filtering function by taking the action for the corresponding ports. 0: to ignore the action bit results for the corresponding ports. *Note: port 26 is the CPU port.*

7.4.45 GMAC Registers

Name: g_rxtx

Offset: 0x1C0/0x1E0

Access: R/W

Description

Register with offset 0x1C0 is for GMAC Port 24, and 0 x 1E0 is for Port 25.

Table 108 G_RxTx Register

Bit Fields	Name	Access/Initial Value	Description
31:30	Rsv	0	reserved
29:24	Ipgt[5:0]	(R/W) 6'h0A	Back-to-Back Transmit IPG Inter frame gap between successive transmission of packets.
23	Adpad	(R/W) 0	Auto Detect Pad Enable Automatically detects a VLAN tagged frame and pads the necessary data to make it a legal 68 Byte frame

22	paden	(R/W) 0	Pad Enable Pads data on to a packet to make it legal frame depending on whether Adpad bit is set or VLAN Pad Enable is set. 68-byte Padding is enabled when paden = ’1’ and (Adpad = ’1’ or Vlpad = ’1’) on a VLAN Tagged Frame 64-byte Padding is enabled when paden = ’1’ and (Adpad = ’0’ and Vlpad = ’0’) on a non VLAN tagged frame
21:19	Rsv		Reserved
18	Vlpad	(R/W) 0	VLAN Pad Enable When set has the same functionality as Adpad except that it does not look for VLAN Tagged Frame and makes all the packets which are of size less than 68-bytes, to a packet size of 68-bytes. This bit is activated only if paden = ’1’
17:0	Rsv		Reserved

7.4.46 G_Mode Register

Name: **g_mode (Mode of operation for the G Ports)**

Offset: 0x1CC / 0x1EC

Access: R/W

Description

Register with offset 0x1CC is for GMAC Port 24, and 0x1EC is for Port 25. This register is used in conjunction with *link_ok* bit (Bit 1) of **“G_PCS_0 / Gport Link Status Register” on Page 196** register.

Table 109 G_Mode Register

Bit Fields	Name	(Access) Initial Value	Description
31:11	rsv	0	Reserved
10	Speed1000	(R) 0	GPORT selected speed 1: Gport running in 1000 Mbit/s 0: Gport running either in 10/100 Mbit/s depending on *speed* (bit 9)

9	Speed	(R) 0	GPORT selected speed 1 : Gport running in 100 Mbit/s 0 : Gport run in 10 Mbit/s
8	Full_duplex_cpu	(R/W) 0	Set to run the Gport in full duplex Clear to run the Gport in half duplex This bit is written by CPU to determine whether the gigabit port in half duplex or full duplex if *e_hw_mode* in **"Switch Configuration Register" on Page 145** is 0. If *e_hw_mode* is asserted, this bit is ignored. The CPU READ data is always from the mode finally used. *Note: PLB 2224 supports half duplex only for 100 Mbit/s and 10 Mbit/s mode of operation for Gports*
7	Rx_pause_e Cpu	(R/W) 0	Set to enable full duplex flow control using PAUSE frame for reception Clear to disable full duplex flow control using PAUSE frame for reception This bit is written by CPU to determine the Gport's rx pause capability if *e_hw_mode* is 0. If *e_hw_mode* is asserted, this bit is ignored. The CPU READ data is always from the mode finally used.
6	tx_pause_e_ cpu	(R/W) 0	Set to enable full duplex flow control using PAUSE frame for transmission Clear to disable full duplex flow control using PAUSE frame for transmission This bit is written by CPU to determine the port's tx pause capability if *e_hw_mode* is 0. If *e_hw_mode* is asserted, this bit is ignored. The CPU READ data is always from the mode finally used.

5	sel_gmii	(R/W) 0	*1*: to select GMII instead MII interface. *0*: to select MII interface for 100BASE PHY. This field is valid only when *gmac_mode* (bit 0) is 1. These bits should be pin strapped to 1 if *gmac_mode* (bit 0) is 0. The bit is latched from pin-strapping input during the reset. *led_col[24]* for GPort0 & *led_col[25]* for GPort1
4	Sw_reset	(R/W) 0	Software Reset
3	l10b	(R/W) 0	Loopback 10-bit Symbols in GMAC
2	l8b	(R/W) 0	Loopback 8-bit Data in GMAC
1	ewrap	(R/W) 0	Enable PHY loopback
0	gmac_mode	(R/W) 0	*0*: to select Gb serdes based PHY instead GMII or MII interface. *1*: to select GMII or MII interface instead of serdes based PHY. Used as sel_pma bit. This bit is initialized through the value of power strapped input at reset. *led_col[26]* for GPort0 & *Led_row_n[0]* for GPort1

7.4.47 G_PCS0 Register / Gport Link Status Register

Name: g_pcs_0

Offset: 0x1D0 / 0x1F0

Access: R/W

Description

Register with offset 0x1D0 is for GPort 24, and 0x1F0 is for GPort 25. This register has information related to link_ok status for the Gigabit ports

Table 110 G_PCS_0 / Gport Link Status Register

Bit Fields	Name	(Access) Initial Value	Description
31:8	rsv	0	Reserved
7	autos	(R/W)	Auto-sense bit
6	anen	(R/W) 1	Auto-Negotiation Enable Should be written to ’0’ through CPU interface or EEPROM to get the TBI interface working
5	rstan	(R/W) 0	Restart Auto-Negotiation
4	rfind	(R/W) 0	Remote fault Indicator
3	ancplt	(R/W) 0	Auto-Negotiation Completed
2	pgrx	(R/W) 0	Auto-Negotiation Page Received

1	link_ok	(R/W) 0	The link_ok bit is to indicate the link status for a GMAC port. The link status determines whether the port has good link and is able to transmit and receive normally and to assert external LED's. This bit is equivalent to the *link_ok_cpu* bit in the FE port. If *e_hw_mode* is 0, the link state of the GMAC port is based on the value written by CPU. If *e_hw_mode = '1'*, link_ok bit is ignored. When the CPU reads data, it is always based on the mode latched from MDIO or internal PCS (depending on gmac_mode). If *e_hw_mode* is 0, link_ok takes the value written by the CPU. Otherwise, (e_hw_mode is 1), the link_ok bit indicates the result of polling the PHY or CPU's read state. CPU can force the link_ok status to '1' when *e_hw_mode* is 0. When reading link_ok bit, link_ok indicates pma_linkok bit if *gmac_mode* is 0. It indicates mii_linkok bit when *gmac_mode* is 1.
0	syncok	(R/W) 0	The syncok bit indicates the sync status from internal PCS when GMAC is running in TBI mode. It is ignored otherwise. This bit is the result of receiving a good code group.

7.4.48 G_PCS_1 Register

Name: **g_pcs_1**

Offset: 0x1D4 / 0x1F4

Access: R/W

Description

Register with offset 0x1D4 is for GPort 24, and 0x1F4 is for GPort 25.

Table 111 G_PCS_1 Register

Bit Fields	Name	(Access) Initial Value	Description
31	annp	(R/W) 0	Auto-Negotiation Additional Next Page to follow
30	Ack	(R/W) 0	Auto-Negotiation acknowledge
29	anmsg	(R/W) 0	Auto-Negotiation Message Page in Code Field
28	anack2	(R/W) 0	Auto-Negotiation Acknowledge 2
27	antog	(R/W) 0	Auto-Negotiation Toggle Flag
26:16	npcf[10:0]	(R/W) 11'h0	Next Page Message/Unformatted Code Field
15	nextp	(R/W) 0	Next Page Capable
14	rsv	0	Reserved
13:12	anerr [1:0]	(R/W) 2'b0	Auto-Negotiation Error {RF2, RF1}
11:9	rsvd		Reserved
7:8	pause [1:0]	(R/W) 2'b0	Pause capable {ASM_DIR, PAUSE}
6	hd	(R/W) 0	Half Duplex
5	fd	(R/W) 0	Full Duplex
0:4	Rsv	0	Reserved

7.4.49 G_PCS_2 Register

Name: **g_pcs_2**

Offset: 0x1D8 / 0x1F8

Access: R/W

Description

Register with offset 0x1D8 is for GMAC Port 24, and 0x1F8 is for Port 25.

Table 112 G_PCS_2 Register

Bit Fields	Name	(Access) Initial Value	Description
31	Lpannp	(R) 0	Link Partner Auto-Negotiation Additional Next Page to follow
30	Lpack	(R) 0	Link Partner Auto-Negotiation acknowledge
29	Lpanmsg	(R) 0	Link Partner Auto-Negotiation Message Page in Code Field
28	Lpanack2	(R) 0	Link Partner Auto-Negotiation Acknowledge 2
27	Lpantog	(R) 0	Link Partner Auto-Negotiation Toggle Flag
26:16	Lpnpcf[10:0]	(R) 11'h0	Link Partner Next Page Message/ Unformatted Code Field
15	Lpnextp	(R) 0	Link Partner Next Page Capable
14	Rsv	0	Reserved
13:12	Lpanerr[1:0]	(R) 2'b0	Link Partner Auto-Negotiation Error (Remote Fault) {RF2, RF1}
11:9	Rsvd		Reserved
8:7	Lppause[1:0]	(R) 2'b0	Link Partner Pause capable {ASM_DIR, PAUSE}
6	Lphd	(R) 0	Link Partner Half Duplex
5	Lpfd	(R) 0	Link Partner Full Duplex
4:0	rsv	0	Reserved

8 Electrical Characteristics

Table 113 Absolute Maximum Ratings

Parameter	Symbol	Limit Values		Unit
		min.	max.	
Ambient temperature under bias	T_A	0	70	°C
Storage temperature	T_{stg}	-65	125	°C
IC supply voltage	V_{DD}	-0.4	1.89	V
PAD (I/O) supply voltage	V_{DDP}	-0.5	3.65	V
Voltage on any pin with respect to ground	V_S	-0.4	V_{DDP} + 0.4	V
Maximum current on all lines connected to the backplane when the DOC is without power supply; at 5.5 V external signal level	I_{max}		2.3	mA
ESD robustness[1)] HBM: 1.5 kΩ, 100 pF	$V_{ESD,HBM}$	2000		V

[1)] According to MIL-Std 883D, method 3015.7 and ESD Ass. Standard EOS/ESD-5.1-1993. The RF Pins 20, 21, 26, 29, 32, 33, 34 and 35 are not protected against voltage stress > 300 V (versus V_S or GND). The high frequency performance prohibits the use of adequate protective structures.

Note: Stresses above those listed here may cause permanent damage to the device. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Table 114 Operating Range

Parameter	Symbol	Limit Values		Unit	Test Condition
		min.	max.		
Ambient temperature	T_A	0	70	°C	
Supply voltage	V_{DD}	1.65	1.89	V	
Pad supply voltage	V_{DDP}	3.0	3.6	V	
Ground	V_{SS}	0	0	V	

Note: In the operating range, the functions given in the circuit description are fulfilled.

Table 115 DC Characteristic

Parameter	Symbol	Limit Values		Unit	Notes
		min.	max.		
Input low voltage	V_{IL}	– 0.4	0.8	V	
Input high voltage	V_{IH}	2.0	V_{DD} + 0.4	V	
Output low voltage	V_{OL}		0.45	V	I_{OL} = 7 mA[1] I_{OL} = 2 mA[2]
Output high voltage	V_{OH}	2.4		V	I_{OH} = – 1.0 mA
Avg. power supply current	I_{CC} (AV)		TBD	mA	V_{DD} = 3.3 V, T_A = 25 °C:
Input leakage current	I_{IL}		1	µA	V_{DD} = 3.3 V, GND = 0 V; all other pins are floating; V_{IN} = 0 V, V_{DDP} + 0.4
Output leakage current	I_{OZ}		1	µA	V_{DD} = 3.3 V, GND = 0 V; V_{OUT} = 0 V, V_{DDP} + 0.4

[1] The listed characteristics are ensured over the operating range of the integrated circuit.

[2] The listed characteristics are ensured over the operating range of the integrated circuit. Typical characteristics specify mean values expected over the production spread. If not otherwise specified, typical characteristics apply at T_A = 25 °C and the given supply voltage.

8.1 AC Characteristics

8.1.1 MDIO Interface Timing Details

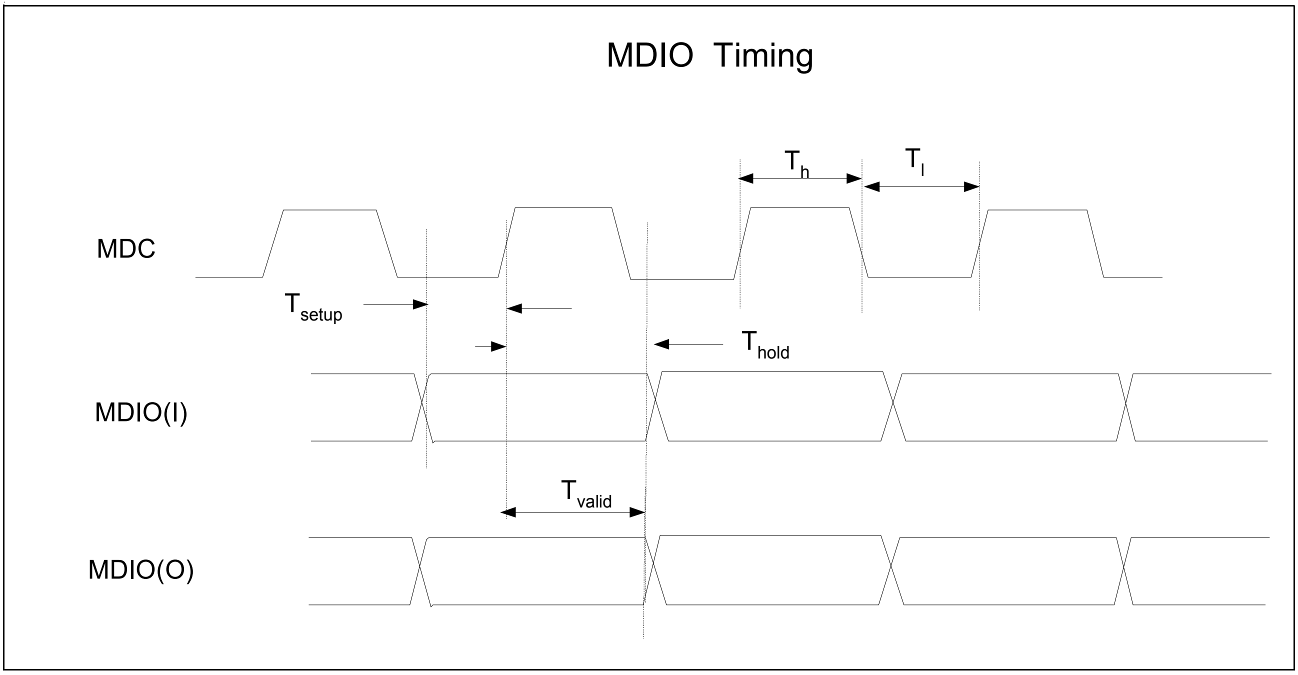


Figure 24 MDIO Timing

Table 116 AC Characteristics MDIO

Parameter	Symbol	Limit Values		Unit
		min.	max.	
MDC High Pulse Width	T_H	40	-	ns
MDC Low Pulse Width	T_l	40	-	ns
MDIO (I) Setup with respect to MDC rising edge	T_{setup}	20	-	ns
MDIO (O) Hold time from MDC rising edge	T_{hold}	20	-	ns
MDIO (O) valid from MDC rising edge	T_{valid}	0	-	ns

8.1.2 SMII Interface Timing Details

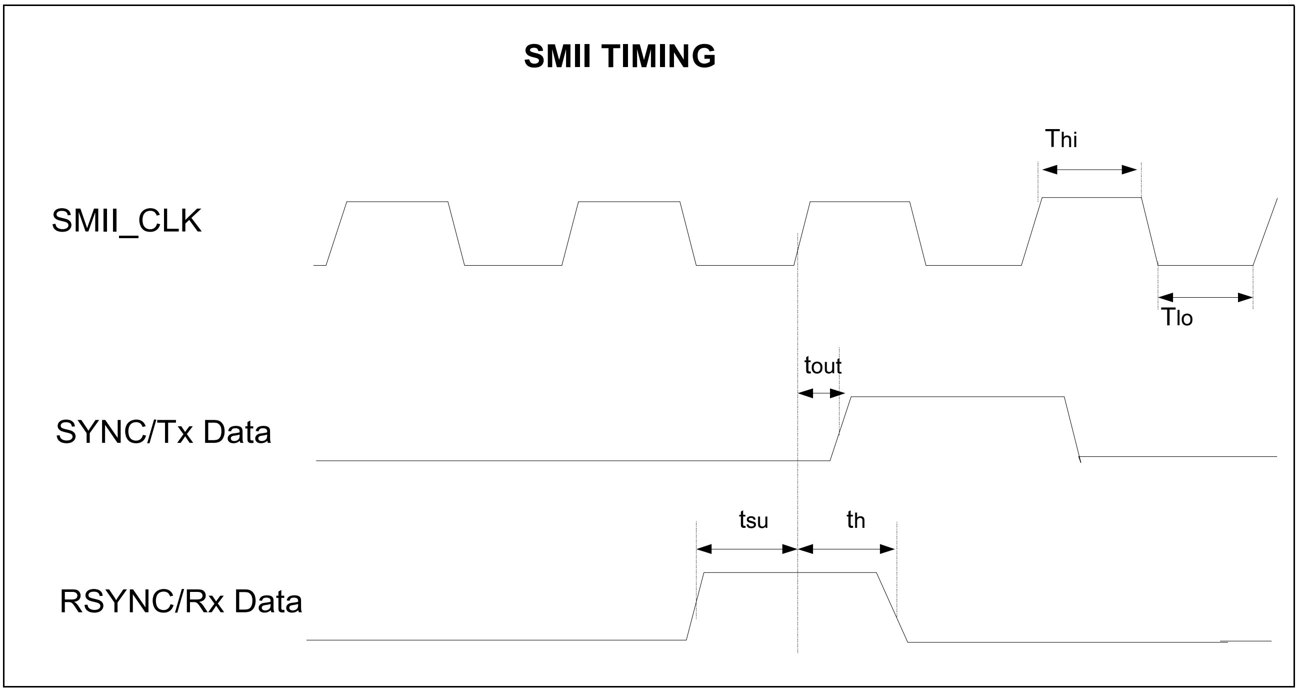


Figure 25 SMII Interface Timing

Table 117 AC Characteristics - SMII

Parameter	Symbol	Limit Values		Unit
		min.	max.	
SMII Clock High period	T_{hi}	3	-	ns
SMII Clock Low period	T_{lo}	3	-	ns
SMII Clock to output data valid	t_{out}	1.5	6	ns
Rx. Data Input Setup time	t_{su}	0.8	-	ns
Rx. Data Input Hold time	t_{h}	0.5	-	ns

8.1.3 GMII/TBI Interface Timing

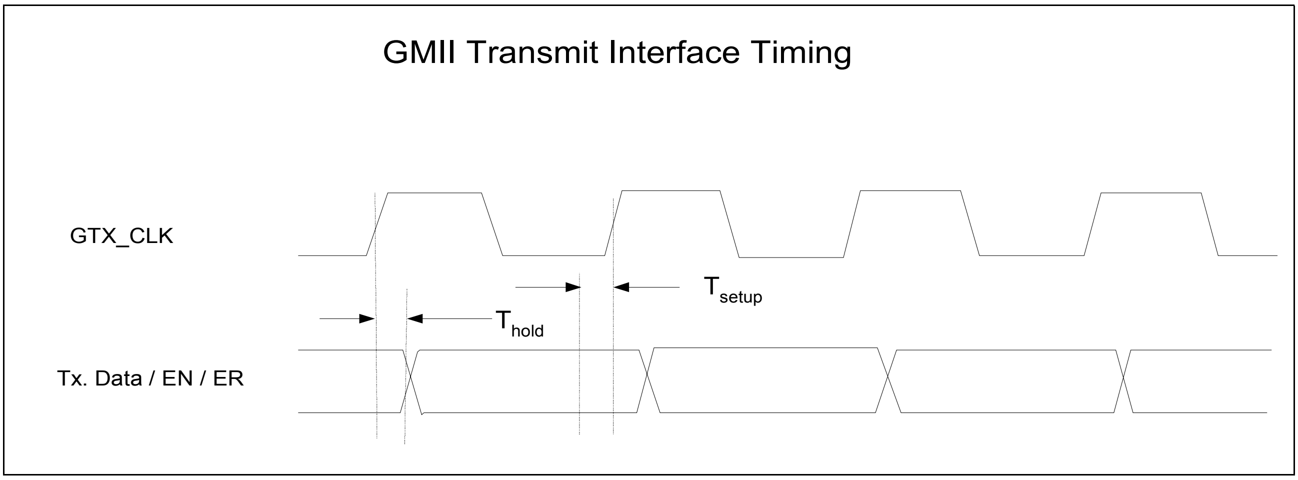


Figure 26 GMII/TBI Tx. Interface Timing

Table 118 AC Characteristics GMII Tx. Interface

Parameter	Symbol	Limit Values		Unit
		min.	max.	
Setup from valid Tx. Data / EN / ER to the rising edge of the GTX clock	T_{setup}	2	-	ns
Hold from the rising edge of the GTX clock to the Tx. data / EN / ER	T_{hold}	1	-	ns

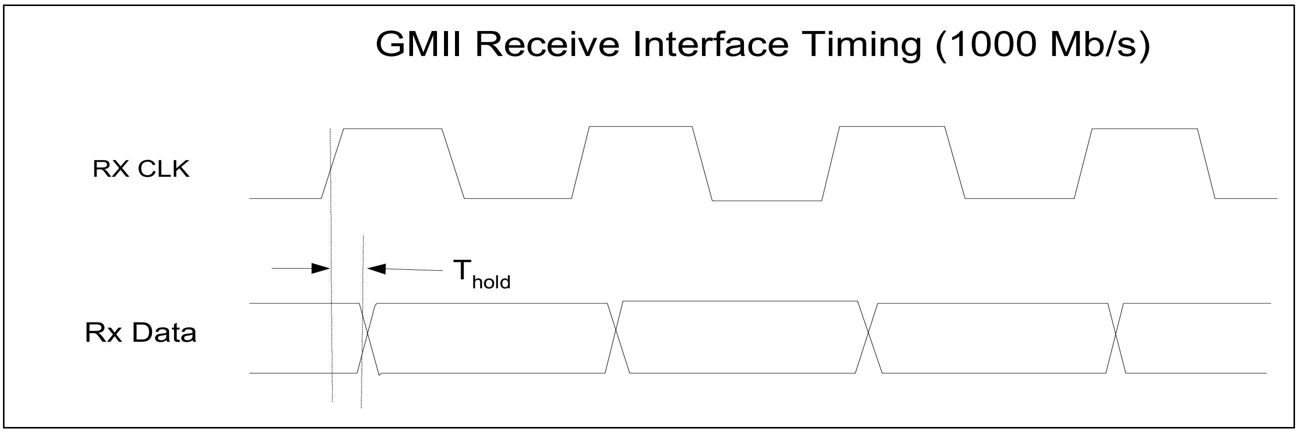


Figure 27 GMII/TBI Rx. Interface Timing

Table 119 AC Characteristics GMII Rx. Interface

Parameter	Symbol	Limit Values		Unit
		min.	max.	
Setup from the rising edge of RX clock to Rx. data / EN / ER	T_{setup}	1.5	-	ns
Hold from the rising edge of RX clock to Rx. data / EN / ER	T_{hold}	1	5.5	ns

8.1.4 PCI Interface / Generic Interface Timing

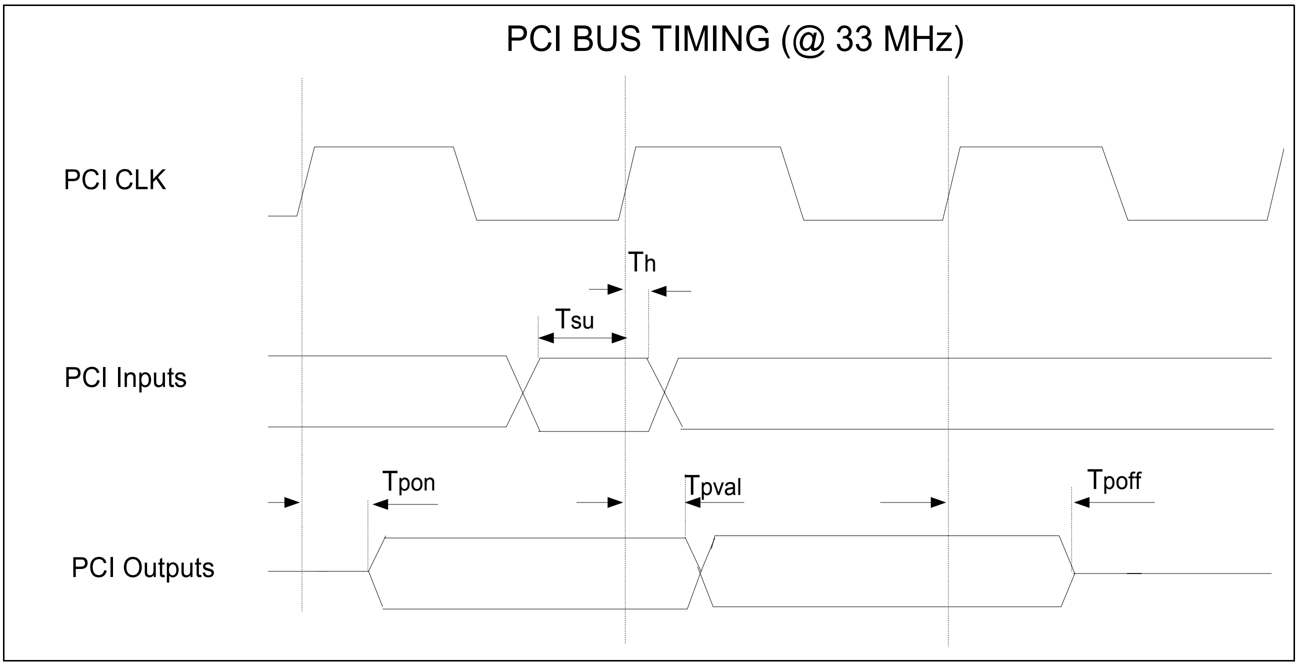


Figure 28 PCI / Generic Interface Timing

Table 120 AC Characteristics PCI

Parameter	Symbol	Limit Values		Unit
		min.	max.	
Data Input Setup time	T_{su}	6.8	-	ns
Data Input Hold time	T_h	0	-	ns
Data Output valid	T_{pval}	3	20.2	ns
Float to active delay	T_{pon}	3	-	ns
Active to float delay	T_{poff}	-	15	ns

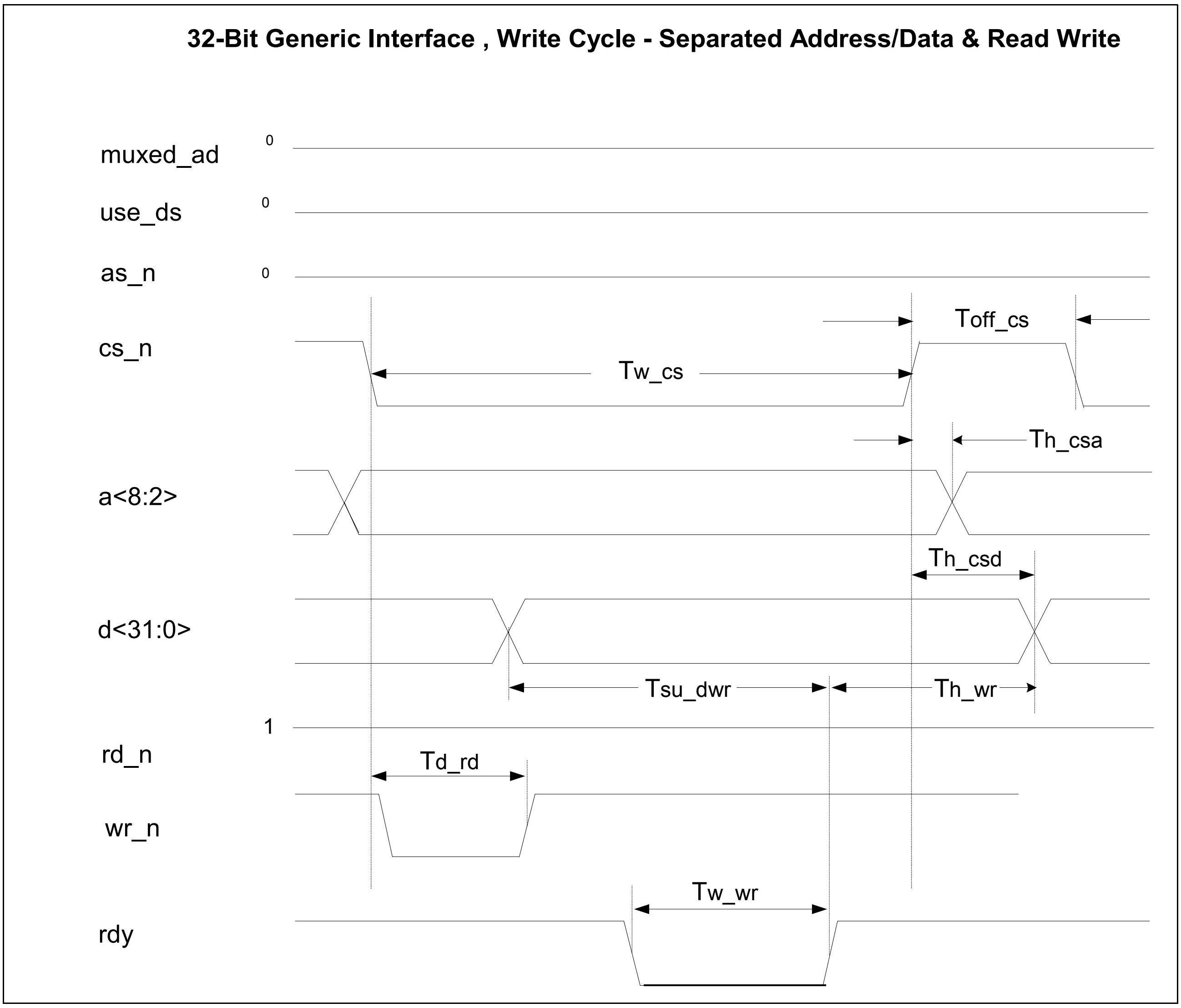


Figure 29 32-Bit Generic Interface Write Cycle - Separate Addr/Data / R / W

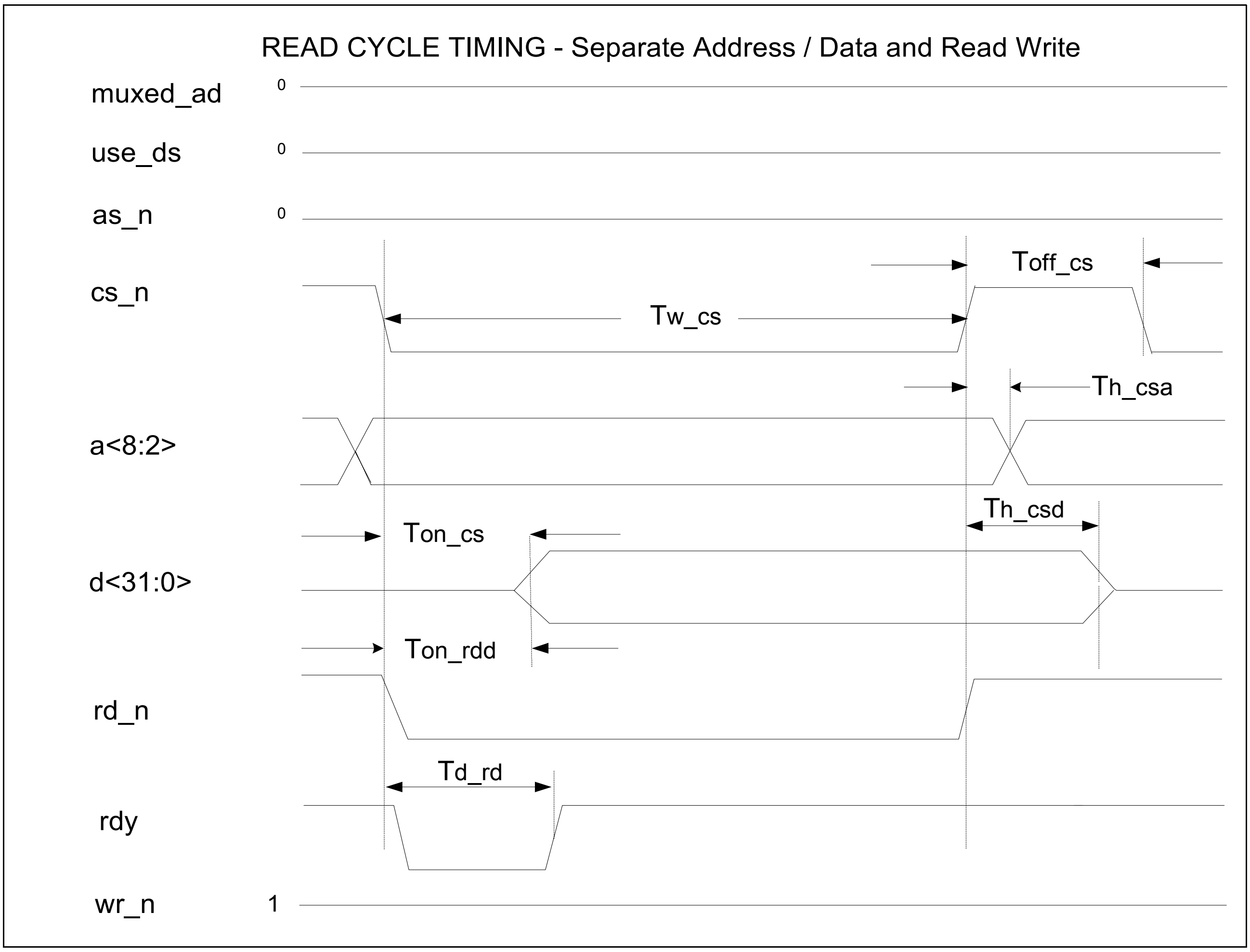


Figure 30 32-Bit Generic Interface Read Cycle - Separate Addr / Data / R / W

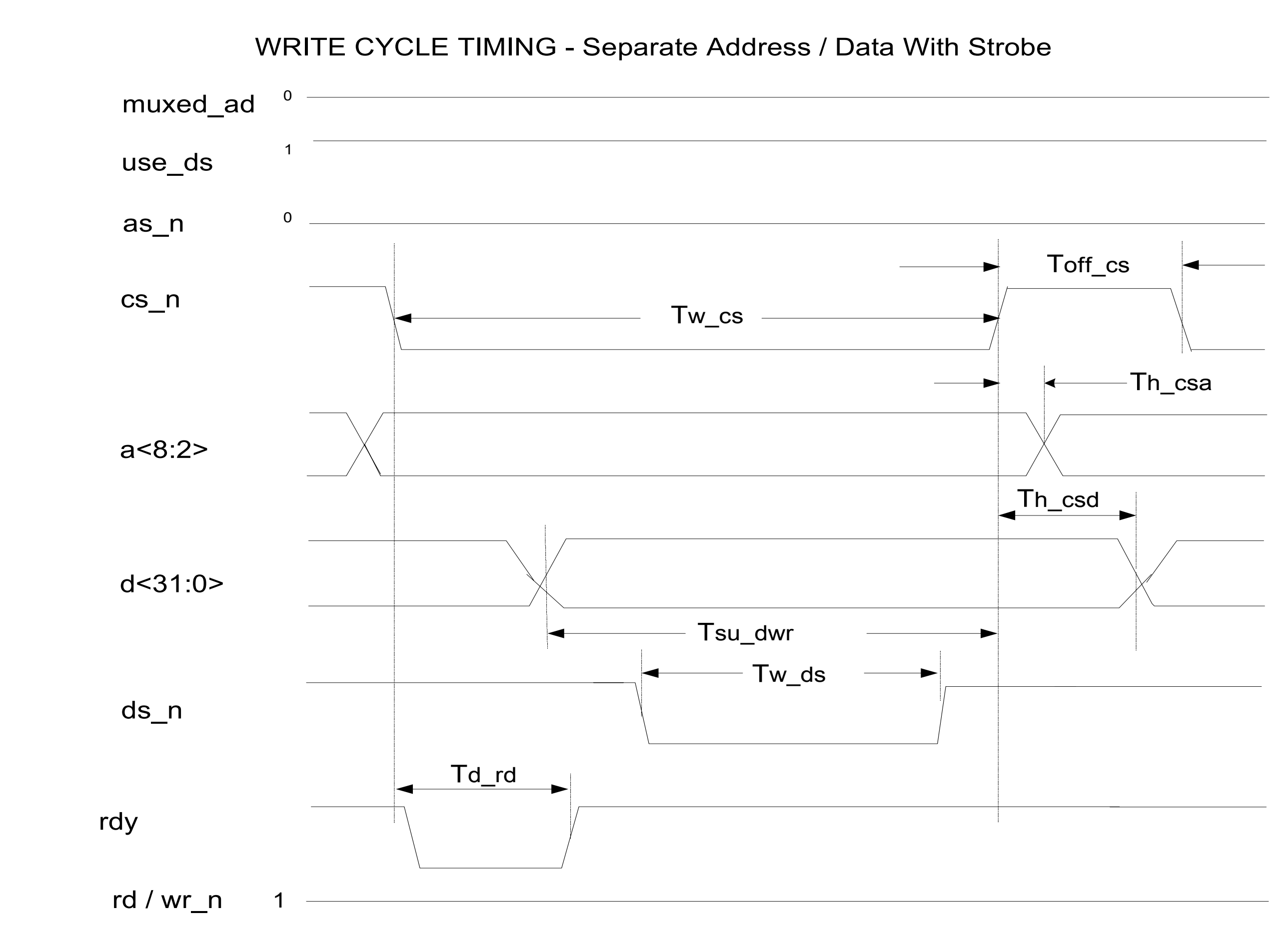


Figure 31 **32-Bit Generic Interface Write Cycle - Separate Addr / Data with Strobe**

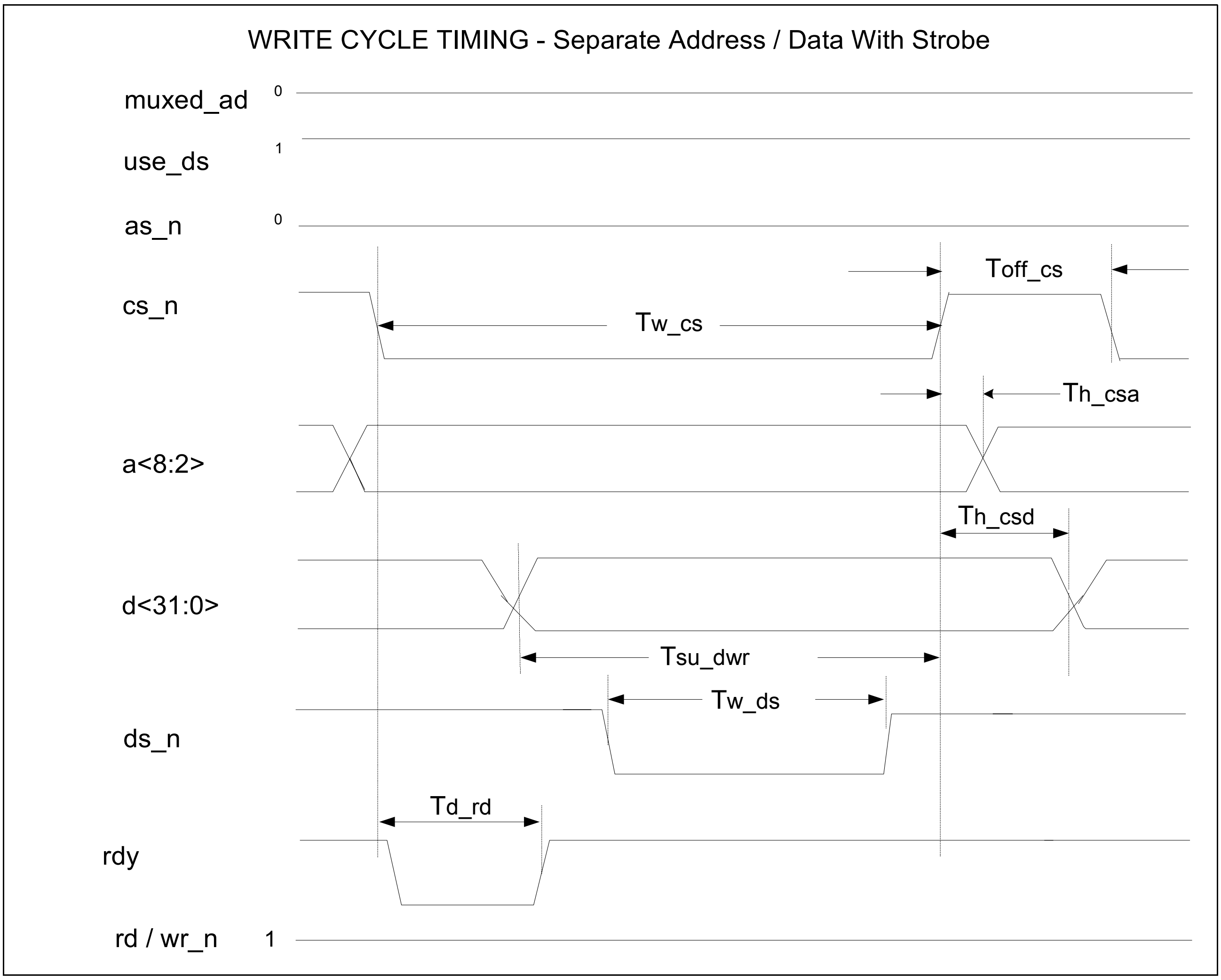


Figure 32 32-Bit Generic Interface Write Cycle - Separate Addr / Data with Strobe

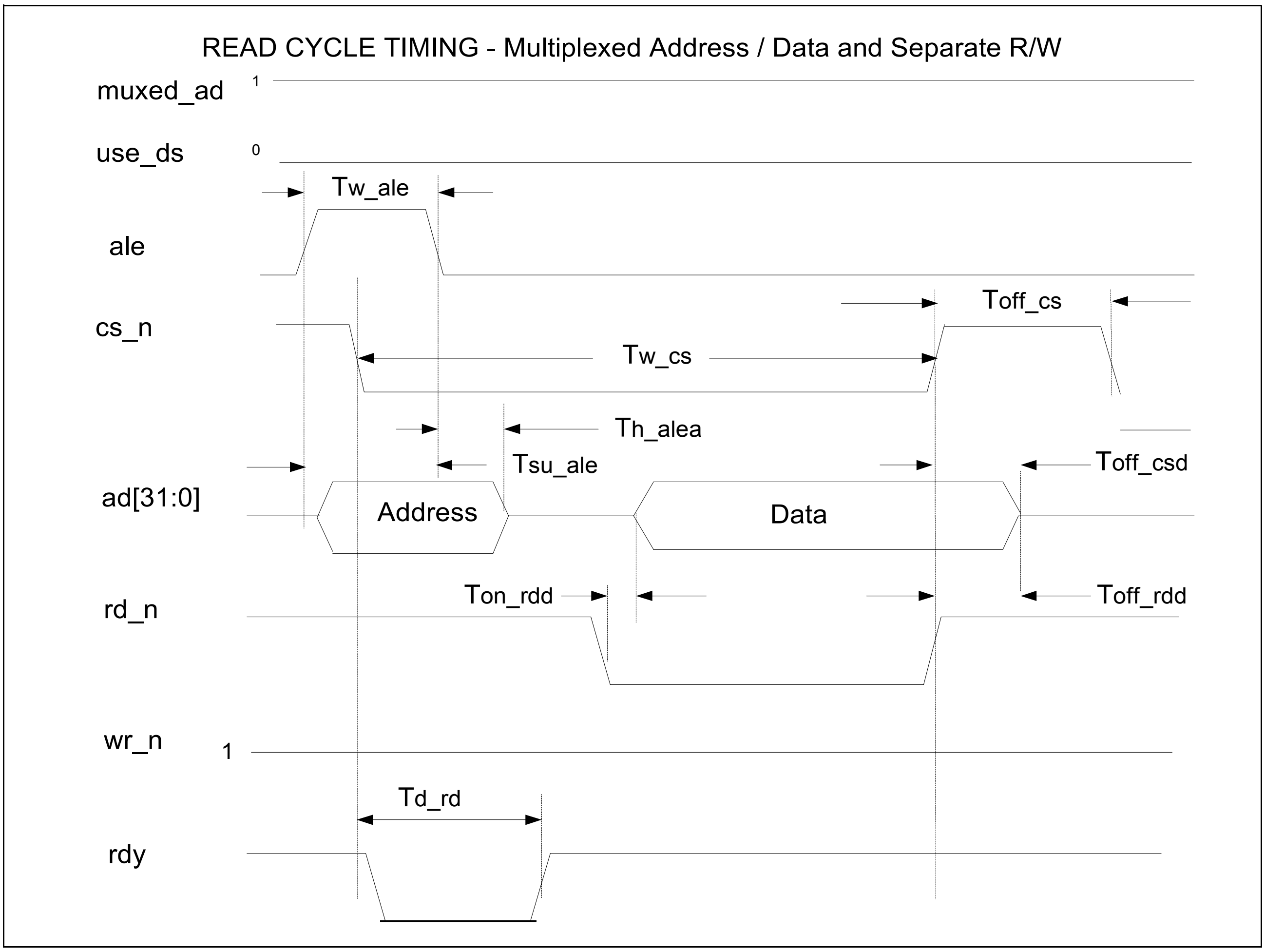


Figure 33 32-Bit Generic Interface Read - Muxed Addr / Data & Separate R/W

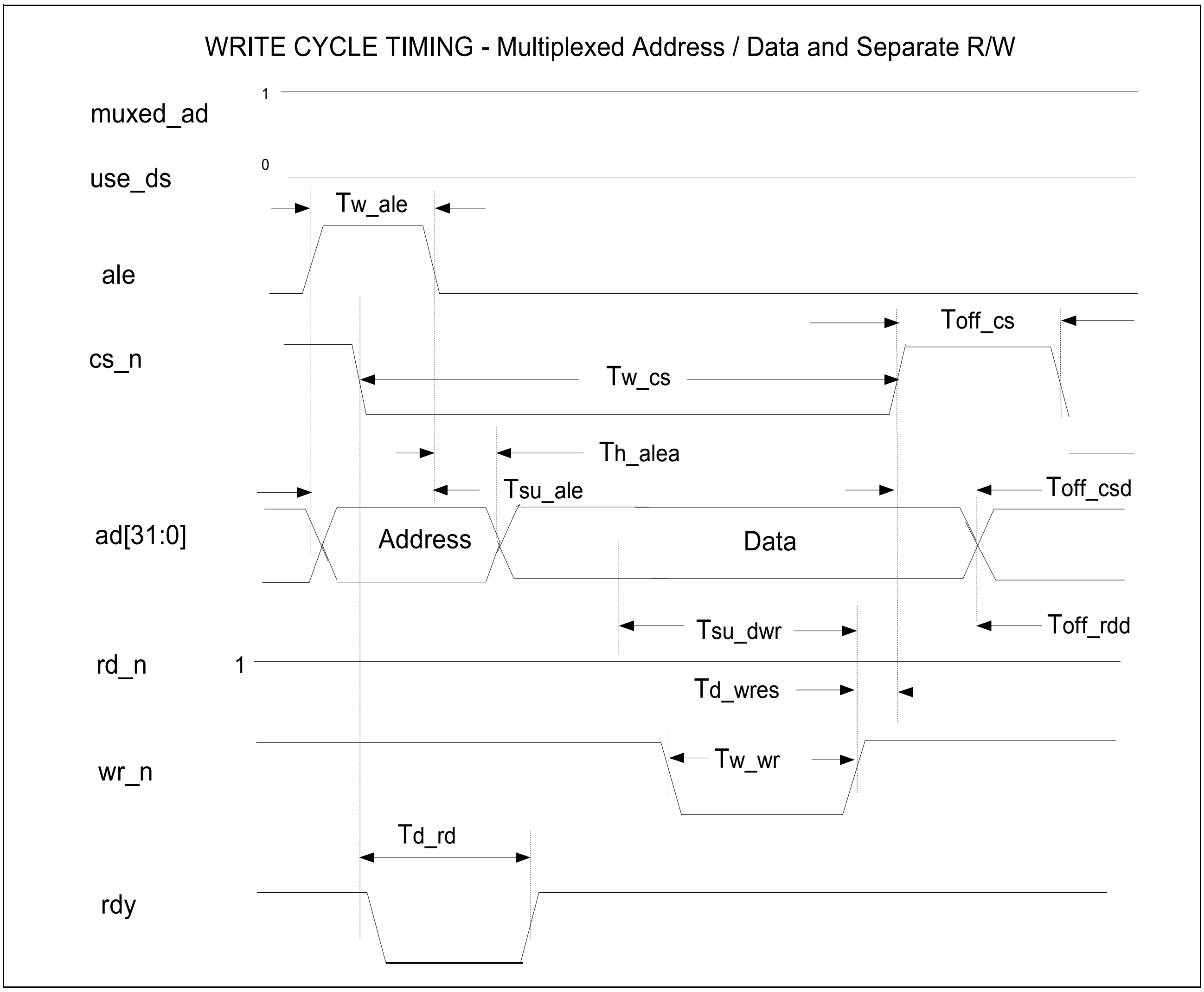


Figure 34 32-Bit Generic Interface Write - Muxed Addr / Data & Separate R/W

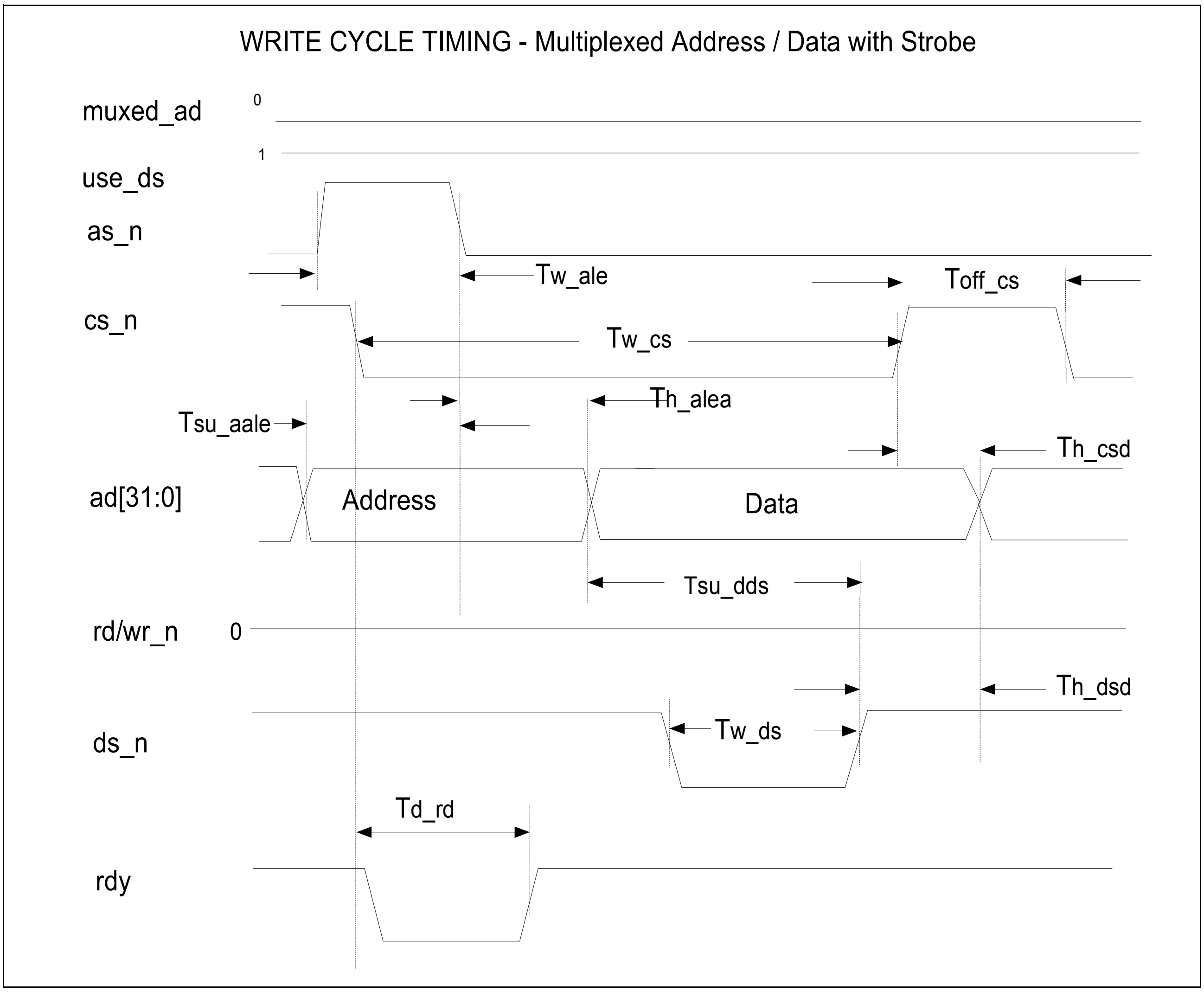


Figure 35 32-Bit Generic Interface Write Cycle - Muxed Addr / Data with Strobe

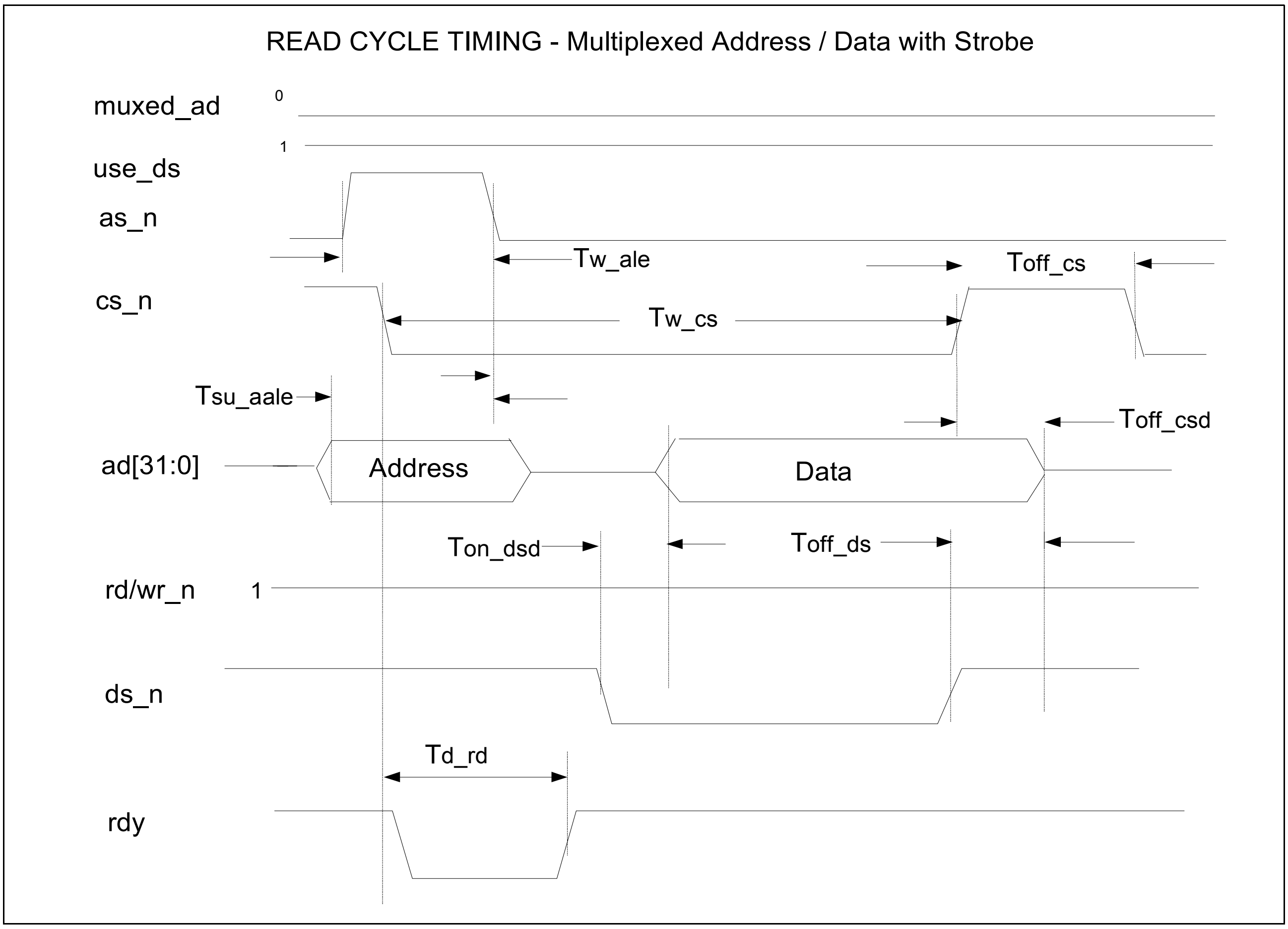


Figure 36 32 Bit Generic Interface Read Cycle - Muxed Addr/Data with Strobe

8.1.5 Timing Parameters (Figure 29 to Figure 36)

Table 121 32-Bit Generic Interface Timing Parameters

Parameter	Symbol	Limit Values		Unit
		min.	max.	
cs_n fall to rdy rise delay	$T_{\text{d\_rd}}$	10	-	ns
wr_n rise to cs_n rise delay	$T_{\text{d\_wrcs}}$	-	0	ns
ale fall to address invalid hold time	$T_{\text{h\_alea}}$	0	-	ns
cs_n rise to address invalid hold time	$T_{\text{h\_csa}}$	0	-	ns
cs_n rise to data invalid hold time	$T_{\text{h\_csd}}$	0	-	ns
wr_n rise to data invalid hold time	$T_{\text{h\_wrd}}$	0	-	ns
cs_n off time	$T_{\text{off\_cs}}$	30	-	ns
cs_n rise to read data invalid delay	$T_{\text{off\_csd}}$	4	-	ns

Table 121 32-Bit Generic Interface Timing Parameters (cont'd)

Parameter	Symbol	Limit Values		Unit
		min.	max.	
ds_n rise to read data invalid delay	$T_{\text{off\_dsd}}$	4	-	ns
rd_n rise to read data invalid delay	$T_{\text{off\_rdd}}$	4	-	ns
cs_n fall to read data valid delay	$T_{\text{on\_csd}}$	-	45	ns
ds_n fall to read data valid delay	$T_{\text{on\_dsd}}$	-	45	ns
rd_n fall to read data valid delay	$T_{\text{on\_rdd}}$	-	45	ns
address valid to ale fall setup time	$T_{\text{su\_aale}}$	2.5	-	ns
data valid to ds_n rise setup time	$T_{\text{su\_dds}}$	30	-	ns
data valid to wr_n rise setup time	$T_{\text{su\_dwr}}$	30	-	ns
width of ale high time	$T_{\text{w\_ale}}$	2.5	-	ns
width of cs_n low time	$T_{\text{w\_cs}}$	60	-	ns
width of data write ds_n low time	$T_{\text{w\_ds}}$	32	-	ns
width of wr_n low time	$T_{\text{w\_wr}}$	32	-	ns

8.1.6 Serial LED Interface Timing

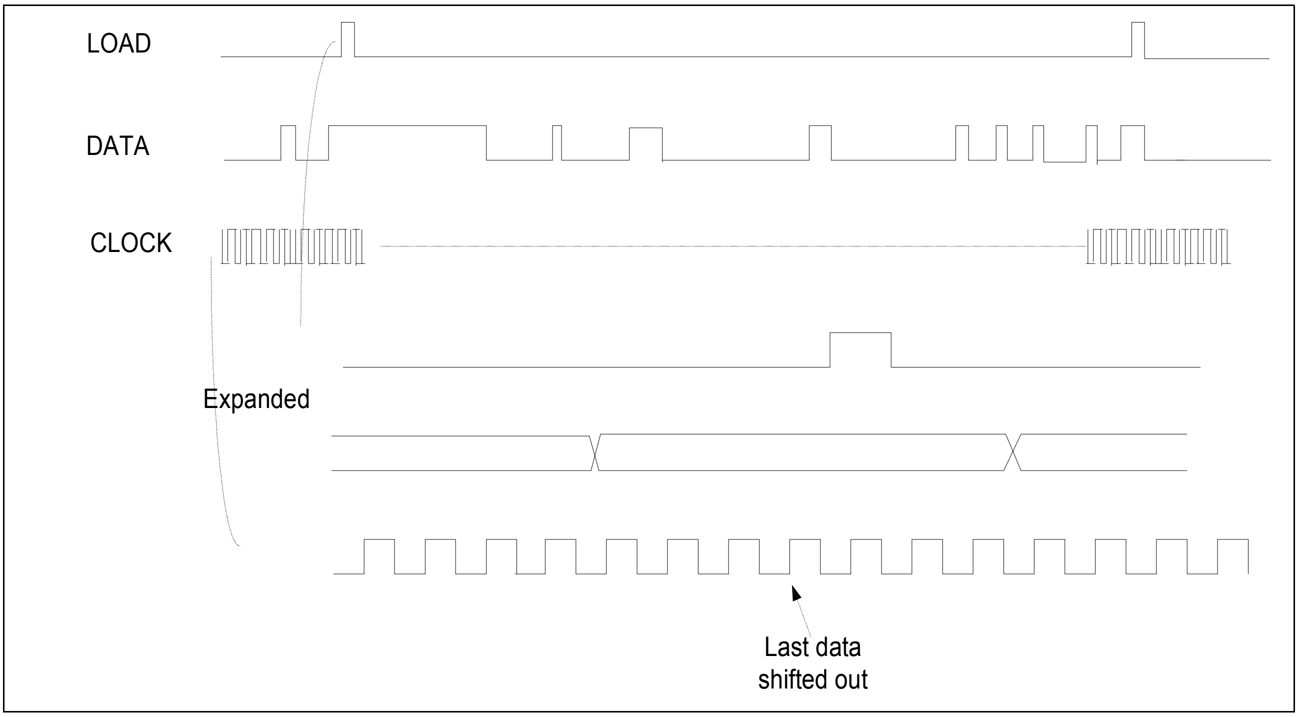


Figure 37 Serial LED Interface

Table 122 Serial LED Interface Timing

Parameter	Value	Unit
LOAD	20	ms
CLK	163.8	µs

The order of data shifted out is port0/row0, port0/row1, port0/row2, ... port27/row3

8.1.7 IIC Timings

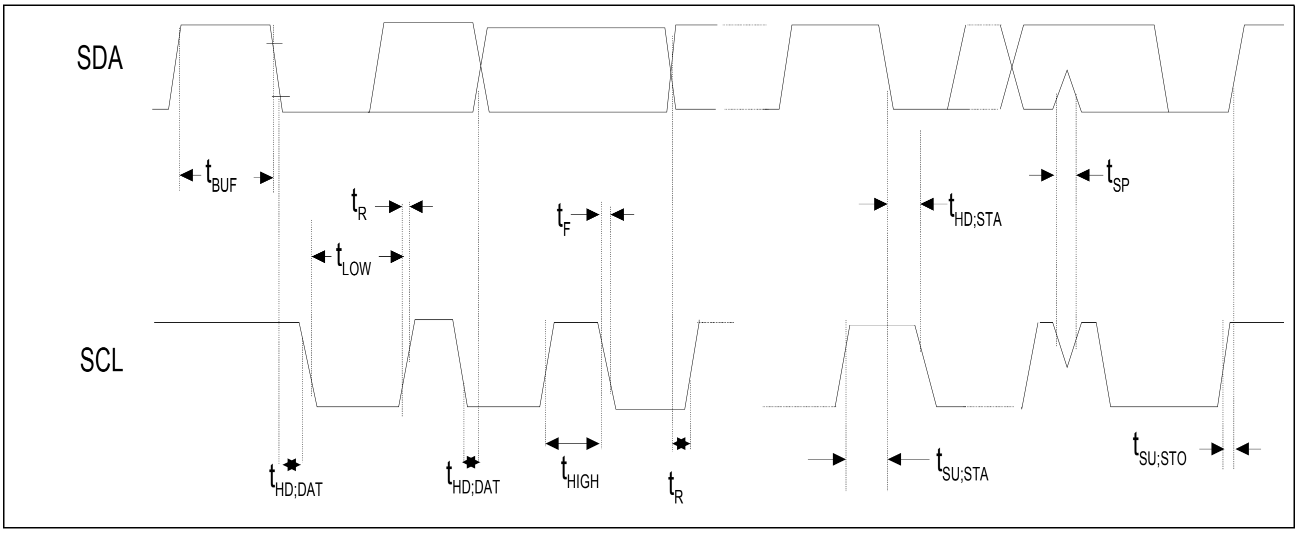


Figure 38 IIC Interface Timing

Table 123 IIC Bus Timing Parameters

Parameter	Symbol	Clock Rate								Unit
		100 kHz		400 kHz		1 MHz		3.3 MHz		
		min.	max.	min.	max.	min.	max.	min.	max.	
SCL Clock frequency	f_{SCL}	0	0.1	0	0.4	0	1	0	3.3	MHz
Hold Time Start Condition. After this period the first clock pulse is generated	$t_{HD;STA}$	4.0	-	0.6	-	0.2	-	0.06	-	µs
Low period of the SCL clock	t_{LOW}	4.7	-	1.3	-	0.47	-	0.14	-	µs
High period of the SCL clock	t_{HIGH}	4.0	-	0.6	-	0.4	-	0.12	-	µs

Table 123 IIC Bus Timing Parameters (cont'd)

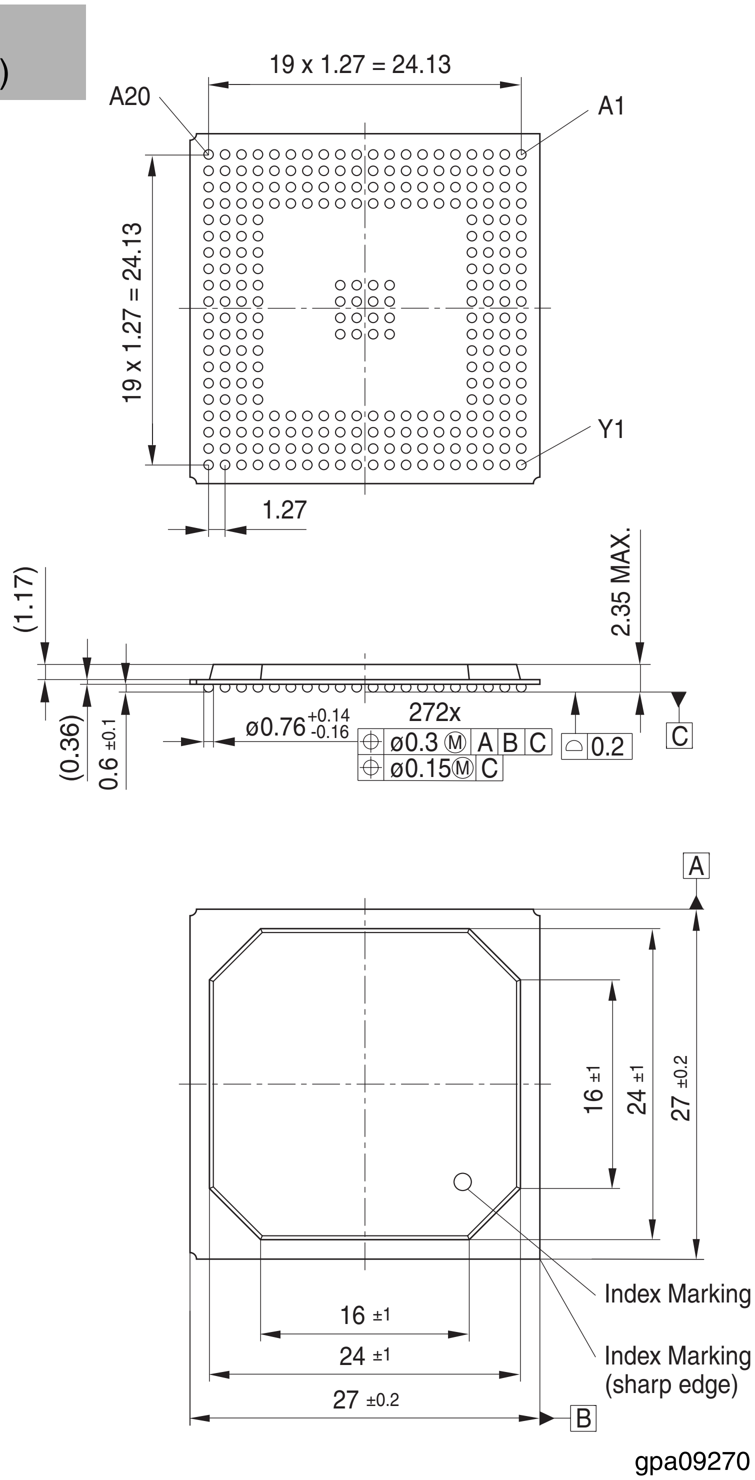
Parameter	**Symbol**	**Clock Rate**								**Unit**
		100 kHz		**400 kHz**		**1 MHz**		**3.3 MHz**		
		min.	**max.**	**min.**	**max.**	**min.**	**max.**	**min.**	**max.**	
Setup time for a repeated START condition	$t_{SU;STA}$	4.7	-	0.6	-	0.2	-	0.06	-	µs
Data Hold Time	$t_{HD;DAT}$	0	3.45	0	0.9	0	0.3	0	0.09	µs
Data Setup time	$t_{SU;DAT}$	250	-	100	-	40	-	12	-	µs
Rise time of both SDA & SCL signals	t_r	-	1000	-	300		120	-	30	ns
Fall time of both SDA & SCL signals	t_f	-	300	-	300		120	-	30	ns
Bus free time between a STOP & START condition	t_{BUF}	4.7	-	1.3	-	0.47	-	0.14	-	µs
Setup time for STOP condition	$t_{SU;STO}$	4.0	-	0.6	-	0.4	-	0.12	-	µs
Pulse width of spikes suppressed	t_{SP}	10 to 1000	-	10 to 250	-	10 to 100	-	10 to 33	-	ns
Capacitive load for each bus line	C_b	-	400	-	400	-	200	-	100	pF

Table 124 Capacitances

Parameter	Symbol	Limit Values		Unit	Notes
		min.	max.		
Clock input capacitance	C_{XIN}	-	5	pF	f_C = 1 MHz The pins, which are not under test, are connected to GND
Input capacitance	C_{IN}	-	7	pF	
Output capacitance	C_{OUT}	-	7	pF	

9 Package Details (P-BGA-272)

P-BGA-272
(Plastic Ball Grid Array Package)



gpa09270

You can find all of our packages, sorts of packing and others in our Infineon Internet Page “Products”: http://www.infineon.com/products.

SMD = Surface Mounted Device

Dimensions in mm

Infineon goes for Business Excellence

“Business excellence means intelligent approaches and clearly defined processes, which are both constantly under review and ultimately lead to good operating results.
Better operating results and business excellence mean less idleness and wastefulness for all of us, more professional success, more accurate information, a better overview and, thereby, less frustration and more satisfaction.”

Dr. Ulrich Schumacher

http://www.infineon.com

Published by Infineon Technologies AG