



## GT-482xx

GT-48212 / GT-48208 / GT-48207

Advanced Switched Ethernet Controllers for 10+10/100 BaseX

Preliminary

Revision 1.2

1/27/99

Please contact Galileo Technology for possible updates before finalizing a design.

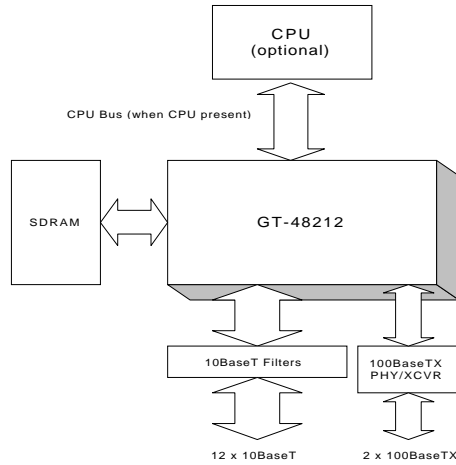
### FEATURES

- Single-chip Switched Ethernet Controllers for 10 and 10/100Base-X
  - Provides packet switching functions between eight or 12 Ethernet ports and two Auto-Negotiated on-chip Fast Ethernet ports
  - Switch expansion via Fast MII port
- Three versions for different cost/performance points
  - GT-48212: 12 10BaseT ports, two 100BaseX ports and advanced management features
  - GT-48208: eight 10BaseT ports, two 100BaseX ports and advanced management features
  - GT-48207: eight 10BaseT ports, two 100BaseX ports with no management features
- Low-cost 32-bit CPU interface for management
  - Glueless interface to IDT 3041, Motorola ColdFire, Intel i960@R/Jx CPUs, and GT-641xx controllers.
  - Simple interface to other 32/64-bit CPUs
- Management CPU not required
  - Allows for cost sensitive unmanaged designs
- Eight or Twelve 802.3 compliant Ethernet ports
  - 10Mbps Half-Duplex or 20Mbps Full-Duplex
  - Serial mode selectable per port: 10Base-T or FL
- Two Fast Ethernet Media Access Controllers
  - Direct Interface to MII
  - Half/Full Duplex Support
  - IEEE 802.3 100Base-TX, T4, and FX compatible
  - Full MII Management Support (MDC/MDIO)
  - Auto-Negotiation supported through MII Interface
- Flow Control on all ports
  - Standard 802.3x flow control for Full Duplex mode
  - Back pressure for Half Duplex mode
- Direct support for packet buffering
  - 1Mbyte: using one device - 256Kx32-bit Synchronous graphics RAM (SGDRAM)
  - 4Mbyte: using two devices - 1Mx16bit SDRAM
  - Up to 2K buffers, 1536-bytes each, dynamically allocated to the receive queues and CPU
- High observability LED interface
  - Three pin serial LED interface for additional status information per port
- Advanced address recognition on-chip
  - Intelligent address recognition mechanism enables forwarding rate at full wire speed
  - Self-learning mechanism
  - Supports up to 8K Unicast addresses and unlimited Multicast/Broadcast addresses
  - Multicast address support in Address Table
  - Broadcast storm filtering
- Extensive network management support
  - Repeater MIB counters allowing implementation of four RMON groups
  - Hardware assist for Spanning Tree algorithm
  - CPU access to Address Table
  - CPU Query - Ability to read the information from the Address Table
  - Ability to define static addresses
  - Monitoring (sniffer) mode
- Port locking for security
- Automatic address aging support
- Priority queuing based on MAC address or 802.1Q tag
- Port and MAC address based VLAN
- IP Multicast support
- Flexible software or hardware intervention in packet routing decisions
- Packet sampling management technology

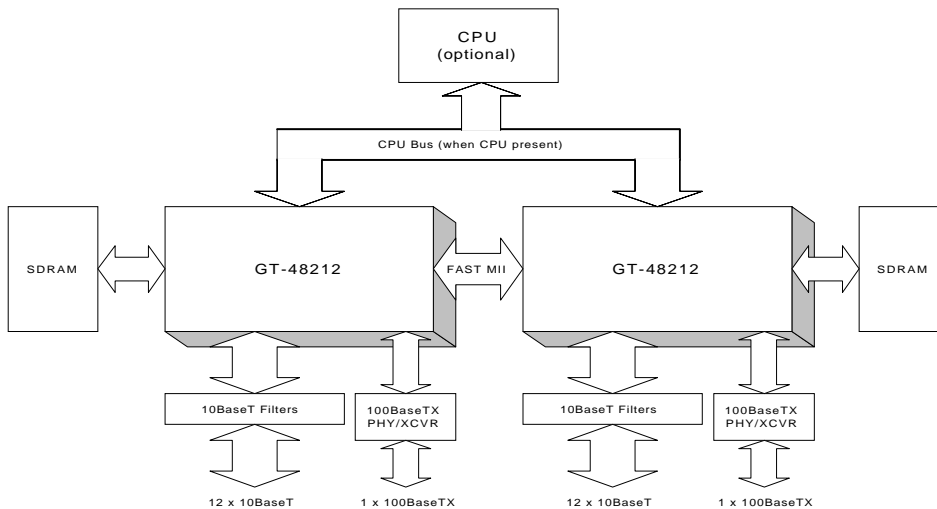
GALILEO TECHNOLOGY CONFIDENTIAL -- DO NOT REPRODUCE

- Takes "snapshots" of packets and counters at programmable intervals
- Allows for the implementation of HP-EASE or sampled RMON with low-cost CPUs
- 3.3V with 5V tolerant I/Os
- 208 pin PQFP package
- 12 General Purpose Output pins (LEDs, etc.)

**Block Diagram of Typical Managed Switch  
(Two 100 Mbit Ports + 12 10 Mbit Ports)**



**Block Diagram of Typical Managed Switch  
(Two 100 Mbit Ports + 24 10 Mbit Ports)**



## Table of Contents

<b>1.</b>	<b>General Description .....</b>	<b>9</b>
1.1	Fast Ethernet Ports .....	9
1.2	Ethernet Ports .....	9
1.3	Flow Control and Back Pressure .....	10
1.4	CPU Interface .....	10
1.5	Synchronous GRAM/DRAM Interface .....	10
1.6	Address Recognition .....	10
1.7	IP Multicast and VLAN Support .....	11
1.8	Priority Queueing .....	11
1.9	Network Management Features .....	11
1.10	Differences Between the GT-48212, GT-48208 and GT-48207 .....	12
<b>2.</b>	<b>Pinout .....</b>	<b>13</b>
2.1	Pin Functions and Assignment .....	14
<b>3.</b>	<b>Galaxy Family Overview .....</b>	<b>19</b>
3.1	Basic Operation .....	19
3.2	Address Learning .....	19
3.3	Packet Buffering .....	20
3.4	Packet Forwarding .....	20
3.5	Terminology .....	20
<b>4.</b>	<b>Microarchitectural Overview .....</b>	<b>21</b>
<b>5.</b>	<b>Buffers and Queues .....</b>	<b>23</b>
5.1	Rx Buffer Threshold Programming .....	24
5.2	Head-of-Line Blocking .....	25
<b>6.</b>	<b>MAC Address Table .....</b>	<b>26</b>
6.1	Forwarding Mask .....	26
6.2	Port Number .....	26
6.3	Address Learning Process .....	27
6.4	Locked Port .....	27
6.5	Address Entry Update and Query from CPU .....	28
6.6	Address Recognition .....	28
6.7	Address Aging .....	30
6.8	Static Addresses .....	31
6.9	Address Recognition Failure .....	31
6.10	Forwarding Priority .....	31
<b>7.</b>	<b>Packet Forwarding .....</b>	<b>33</b>
7.1	Forwarding a Unicast Packet to a Local Port .....	33
7.2	Forwarding a Multicast Packet .....	33
7.3	Forwarding a Packet to the CPU .....	33
7.4	Forwarding a Packet from the CPU to the GT-482xx .....	34
7.5	Intervention Mode .....	35
7.6	IGMP Packet Support .....	35
7.7	CRC Generation .....	35
7.8	Tx Watchdog Timer .....	36

<b>8.</b>	<b>Fast Ethernet Interfaces .....</b>	<b>37</b>
8.1	10/100 MII Compatible Interface .....	37
8.2	Media Access Control (MAC) .....	37
8.3	Auto-Negotiation .....	37
8.4	Backoff Algorithm Options .....	38
8.5	Data Blinder .....	39
8.6	Inter-Packet Gap (IPG) .....	39
8.7	10/100 Mbps MII Transmission .....	39
8.8	10/100 Mbps MII Reception .....	39
8.9	10/100 Mbps Full-duplex Operation .....	40
8.10	Illegal Frames .....	40
8.11	Partition Mode .....	40
8.12	Back Pressure .....	41
8.13	Flow Control .....	41
8.14	802.1q VLAN Tagging Support .....	42
8.15	MI Management Interface (SMI) .....	42
8.16	Link Detection and Link Detection Bypass (ForceLinkPass) .....	43
8.17	Using the MII Interfaces to Connect Two (or More) Galaxy Devices .....	44
<b>9.</b>	<b>Ethernet (10Mbps) Interfaces .....</b>	<b>45</b>
9.1	Media Access Control (MAC) .....	45
9.2	Illegal Frames .....	45
9.3	Duplex Mode Selection .....	45
9.4	Backoff Algorithm Options .....	45
9.5	Manchester Encoder/Decoder .....	45
9.6	Link Integrity .....	45
9.7	Data Blinder .....	45
9.8	Inter-Packet Gap (IPG) .....	45
9.9	Partition Mode .....	46
9.10	802.1q VLAN Tagging Support .....	46
9.11	Back Pressure .....	46
9.12	Flow Control .....	46
9.13	Serial Modes .....	47
9.14	Physical Interface Circuitry .....	47
9.15	Serial Link Status Indication .....	47
<b>10.</b>	<b>Enabling/Disabling Ports .....</b>	<b>48</b>
<b>11.</b>	<b>Network Management Support .....</b>	<b>49</b>
11.1	Repeater MIB Counters .....	49
11.2	Monitoring (Sniffer) Mode .....	49
11.3	Spanning Tree (BPDU) Support .....	50
11.4	Broadcast Storm Filtering .....	50
<b>12.</b>	<b>Packet Sampling Technology (HP-EASE) .....</b>	<b>51</b>
12.1	Packet Sampling Overview .....	51
12.2	EASE Functionality on the GT-482xx .....	51
12.3	Ease Register .....	51
12.4	EASE Interrupts .....	52
12.5	Sampled Packet Indication .....	52

12.6	Error Source Indications . . . . .	52
12.7	Enabling/Disabling EASE Functionality . . . . .	52
<b>13.</b>	<b>LED Support . . . . .</b>	<b>54</b>
13.1	LED Indications Interface Description . . . . .	54
13.2	Detailed LED Signal Description . . . . .	54
13.3	LED Signal Timing Types . . . . .	56
13.4	LED Interface Description . . . . .	57
<b>14.</b>	<b>Interrupts . . . . .</b>	<b>65</b>
<b>15.</b>	<b>RESET Configuration . . . . .</b>	<b>66</b>
15.1	Configuration Pins . . . . .	66
15.2	Configuration Input Timings . . . . .	68
<b>16.</b>	<b>CPU Hardware Interface and Address Mapping . . . . .</b>	<b>69</b>
16.1	Register and Memory Mapping . . . . .	69
16.2	CPU Interface Modes . . . . .	69
16.3	CPU Interface Pin Definitions . . . . .	70
16.4	Selecting the CPU Mode . . . . .	70
16.5	GT-482xx Base Address . . . . .	70
16.6	CPU Interface Applications . . . . .	71
16.7	CPU Interface Priority . . . . .	79
16.8	Memory Endianness . . . . .	79
<b>17.</b>	<b>SDRAM Interface . . . . .</b>	<b>80</b>
17.1	DRAM Configuration . . . . .	80
17.2	DRAM Initialization . . . . .	80
<b>18.</b>	<b>Register Tables . . . . .</b>	<b>81</b>
18.1	Register Description . . . . .	84
18.2	Port MIB Counters (14 Blocks), Offset (start): 0x600, 0xA00, 0xE00, 0x1200 . . . . .	107
<b>19.</b>	<b>GT-482xx Pinout Differences . . . . .</b>	<b>112</b>
19.1	Pinout Differences between GT-48207, GT-48208, and GT-48212 Devices . . . . .	112
19.2	Using a GT-48212 in a GT-48208/7 Socket: Disabling Unused Ethernet Ports . . . . .	112
19.3	Using a GT-48212 or GT-48208 in a GT-48207 Socket: Disabling Unused CPU Interface . . . . .	112
19.4	CCLK in an Unmanaged System . . . . .	113
<b>20.</b>	<b>GT-482xx Pinout Tables, 208-PQFP . . . . .</b>	<b>114</b>
<b>21.</b>	<b>DC Characteristics - PRELIMINARY/SUBJECT TO CHANGE ) . . . . .</b>	<b>120</b>
21.1	Thermal Data . . . . .	121
<b>22.</b>	<b>AC Timing - TARGET/SUBJECT TO CHANGE . . . . .</b>	<b>122</b>
<b>23.</b>	<b>Packaging . . . . .</b>	<b>127</b>
<b>24.</b>	<b>Document History . . . . .</b>	<b>129</b>
	<b>Appendix A . . . . .</b>	<b>135</b>

## List of Figures

Figure 1: Logic Symbol . . . . .	13
Figure 2: Block Diagram of the GT-48212 . . . . .	22
Figure 3: GT-482xx Buffers and Queues . . . . .	24
Figure 4: MII Transmit Signal Timing . . . . .	39
Figure 5: MMII Receive Signal Timing . . . . .	40
Figure 6: MDIO Output Delay . . . . .	43
Figure 7: Required MDIO Setup and Hold Time . . . . .	43
Figure 8: Expansion MII Wiring Diagram . . . . .	44
Figure 9: Example of Serial Link Status Indicator of Port 1 Link Fail. . . . .	47
Figure 10: Primary Status LED Timing (One Blink). . . . .	55
Figure 11: Primary Status LED Timing (Two Blinks) . . . . .	55
Figure 12: Serial LED Interface Timings . . . . .	57
Figure 13: i960 Write Single Long Word . . . . .	71
Figure 14: i960 Write Burst of Four Long Words . . . . .	72
Figure 15: i960 Read Single Long Word . . . . .	72
Figure 16: i960 Read Burst of Four Long Words . . . . .	73
Figure 17: ColdFire 5202 Write Single Long Word . . . . .	73
Figure 18: ColdFire 5202 Write Burst of Four Long Words . . . . .	74
Figure 19: ColdFire 5202 Read Single Long Word . . . . .	74
Figure 20: ColdFire 5202 Read Burst of Four Long Words . . . . .	75
Figure 21: R3041 Write Single Long Word . . . . .	75
Figure 22: R3041 Read Single Long Word . . . . .	76
Figure 23: R3041 Read Burst of Four Long Words . . . . .	76
Figure 24: GT Write Single Long Word . . . . .	77
Figure 25: GT Write Burst of Four Long Words . . . . .	77
Figure 26: GT Read Single Long Word . . . . .	78
Figure 27: GT Read Burst of Four Long Words . . . . .	78
Figure 28: Serial Clock Waveform (SClk) . . . . .	124
Figure 29: Output Delay from Rising Edge . . . . .	125
Figure 30: Input Setup and Hold . . . . .	125
Figure 31: Output Delay from Clock . . . . .	125
Figure 32: Output Float and Drive Delay . . . . .	126
Figure 33: 208 Lead PQFP Package Outline . . . . .	127

## List of Tables

Table 1: Pinout Differences . . . . .	12
Table 2: Pin Functions . . . . .	14
Table 3: Terminology . . . . .	20
Table 4: GT-482xx DRAM Address Mapping . . . . .	24
Table 5: Address Table Entry Format . . . . .	26
Table 6: Address Table Entry Field Description . . . . .	26
Table 7: Forwarding of Unicast Destination Address Packet . . . . .	29
Table 8: Forwarding of Multicast Destination Address Packet . . . . .	29
Table 9: Priority Queuing Options . . . . .	32
Table 10: SMI Bit Stream Format . . . . .	42
Table 11: Enabling/Disabling Ports of the GT-482xx . . . . .	48
Table 12: Spanning Tree Enable Bit Definition . . . . .	50
Table 13: LED Signals Available . . . . .	54
Table 14: LED Signals for Mode0 . . . . .	57
Table 15: LED Signals for Mode1 . . . . .	61
Table 16: RESET Pin Strapping Options . . . . .	66
Table 17: GT-482xx CPU Support . . . . .	69
Table 18: CPU Interface Pin Mappings . . . . .	70
Table 19: CPU Mode Selection . . . . .	70
Table 20: Burst Size for Different CPU Modes . . . . .	70
Table 21: Register Map Table . . . . .	81
Table 22: Base Address, Offset: 0x00 . . . . .	84
Table 23: Global Control, Offset: 0x04 . . . . .	84
Table 24: Status Register, Offset: 0x08 . . . . .	87
Table 25: Sniffer and Aging Timer, Offset: 0x0C . . . . .	88
Table 26: Serial Parameters 10 Register, Offset: 0x10 . . . . .	88
Table 27: Serial Parameters 100 Register, Offset: 0x78 . . . . .	88
Table 28: Watchdog and Tx Threshold Register, Offset: 0x14 . . . . .	89
Table 29: Interrupt Cause, Offset: 0x18 . . . . .	90
Table 30: Interrupt Mask, Offset: 0x1C . . . . .	91
Table 31: CPU Tx High Desc1 - Packet Descriptor, Offset 0x20 . . . . .	91
Table 32: CPU Tx High Desc2 - Packet Descriptor, Offset 0x24 . . . . .	92
Table 33: CPU Tx High Desc1 - New Address, Offset 0x20 . . . . .	92
Table 34: CPU Tx High Desc2 - New Address, Offset 0x24 . . . . .	92
Table 35: CPU EL Free Req, Offset: 0x30 . . . . .	93
Table 36: CPU Empty Buffer, Offset: 0x34 . . . . .	94
Table 37: CPU Enqueue1, Offset: 0x38 . . . . .	94
Table 38: CPU Enqueue2, Offset: 0x3C . . . . .	94
Table 39: CPU New Address1, Offset: 0x40 . . . . .	95
Table 40: CPU New Address2, Offset: 0x44 . . . . .	95
Table 41: CPU New Address3, Offset: 0x48 . . . . .	96
Table 42: CPU Query, Offset: 0x4C . . . . .	96
Table 43: SMI Register, Offset: 0x50 . . . . .	98
Table 44: 802.1Q Ethertype Register, Offset: 0x54 . . . . .	98
Table 45: General Purpose Register1, Offset: 0x58 . . . . .	98
Table 46: General Purpose Register2, Offset: 0x5C . . . . .	99
Table 47: Rx_10 Threshold, Offset: 0x60 . . . . .	99

Table 48:	Rx_100 Threshold, Offset: 0x64	99
Table 49:	CPU Threshold, Offset: 0x68	100
Table 50:	LED Override, Offset: 0x6C	100
Table 51:	Flow Control Source Address Low, Offset: 0x70	100
Table 52:	Flow Control Source Address High, Offset: 0x74	101
Table 53:	CPU Time Out Register, Offset: 0x7C	101
Table 54:	DRAM Configuration, Offset: 0x1448	101
Table 55:	DRAM Parameters, Offset: 0x144C	101
Table 56:	SDRAM Operation, Offset: 0x1474	102
Table 57:	Address Decode, Offset: 0x147C	102
Table 58:	Port Control (10M ports), Offset: 0x400-0x40C, 0x800 - 0x80C, 0xC00 - 0xC0C	102
Table 59:	Port Control 12 (100M ports), Offset: 0x1000	104
Table 60:	Port Control 13 (100M ports), Offset: 0x1004	106
Table 61:	EASE Register, Offset: 0x410-0x41C, 0x810-0x81C, 0xC10-0xC1C, 0x1008-0x100C	107
Table 62:	Definitions Used in Counter Descriptions	108
Table 63:	Port MIB Counters	109
Table 64:	Pinout Differences	112
Table 65:	GT-48212 Pinout Table (Sorted by Pin Number)	114
Table 66:	GT-48208 Pinout Table (Sorted by Pin Number)	115
Table 67:	GT-48207 Pinout Table (Sorted by Pin Number)	117
Table 68:	Absolute Maximum Ratings	120
Table 69:	Recommended Operating Conditions	120
Table 70:	DC Electrical Characteristics Over Operating Range	120
Table 71:	208 PQFP Thermal Data	121
Table 72:	CPU Interface Timings	122
Table 73:	Switch Engine Interface Timings	122
Table 74:	MII, LED and MDC/MDIO Timings	123
Table 75:	Serial Clock Timings	124
Table 76:	208 PQFP Package Dimensions	127
Table 77:	Document History	129



**NOTE:** This document applies to all the Galaxy GT-482xx devices. All of the information is relevant specifically for the GT-48212 device (12 Ethernet ports and two Fast Ethernet ports). However, all of the functional descriptions are also relevant for all GT-482xx devices. The specific deviations and differences between the devices are described in section "Differences Between the GT-48212, GT-48208 and GT-48207" on page 12.

## 1. GENERAL DESCRIPTION

The GT-482xx is a high-performance, low-cost, Switched Ethernet Controller for 10+10/100Base-X that provides packet switching functions between 12 or 8 10Mbps and two 10/100Mbps ports. Switch expansion is possible by using one of the 100BaseX ports working with a separate oscillator. Switch expansion can reach up to 60Mhz clock frequency to achieve up to 240Mbps full duplex bandwidth. The GT-482xx can be used in both managed and unmanaged configurations.

The basic operation of the GT-482xx is quite simple. The GT-482xx receives incoming packets from the Ethernet or Fast Ethernet wire, searches in the Address Table for the Destination MAC Address and then forwards the packet to the appropriate port. If the Destination Address is not found, the GT-482xx treats the packet as a Multicast packet. The GT-482xx optionally forwards the packets to specified ports (or all ports in the VLAN, if enabled) and optionally to the CPU.

The GT-482xx automatically learns the port number of attached network devices by examining the Source MAC Address of all incoming packets. If the Source Address is not found in the GT-482xx Address Table, the device adds it to the table indicating on which port the address resides. The GT-482xx then notifies the CPU of the new address via a New\_Address message and an interrupt.

### 1.1 Fast Ethernet Ports

The GT-482xx integrates two Fast Ethernet ports. Each port works at 10/100Mbps (half duplex) or 20/200Mbps (full duplex). Two Media Independent Interfaces (MII) are provided for glueless connection to off-the-shelf PHY chips. Full Auto-Negotiation for both managed and unmanaged switches is supported.

One of the Fast Ethernet ports can be used for switch expansion. In this mode, it can operate at 60Mhz clock frequency to achieve up to 480Mbps full-duplex bandwidth. No special messages other than the regular Ethernet packets are passed between two Galaxy devices when connected with an expansion port. Management CPU treats two devices as separate devices which cannot share learning information. However, the CPU can control the learning (Section 6.4.1 "Port-Based VLAN Support" on page 28), and VLANs can be created by the CPU across the two devices so that the same VLAN can share ports on different devices.

The GT-482xx incorporates full MII management support. The MDC/MDIO pins are directly controlled by the CPU (and the Auto-Negotiation state machine, when enabled.)

### 1.2 Ethernet Ports

The GT-482xx integrates 12 or 8 10Mbps Ethernet ports. Each port works at 10Mbps (half duplex) or 20Mbps (full duplex) and includes the Media Access Control (MAC), Manchester encoder/decoder, link integrity logic, and a LED interface.

The GT-482xx Ethernet ports are compliant with both the 802.3 and Ethernet specifications. Each port contains three pins allowing direct interface with the AMD QuIET (see URL: [www.amd.com](http://www.amd.com)) or the Tamarack TC3001 (see URL: [www.tmi.com.tw](http://www.tmi.com.tw)) digital filters. Link indication from the PHY is serially shifted into the GT-482xx on the Link-Status pins.

Port #0 includes five pins (TxEn and Link Status are added) allowing interface to AUI, coax, or fiber-optic media.

### 1.3 Flow Control and Back Pressure

IEEE standard 802.3x flow control (for full duplex) and proprietary back pressure (for half duplex) are supported on both the 100Mbps and 10Mbps ports. Back pressure or flow control is activated when the port or device buffer budget is almost exhausted.

### 1.4 CPU Interface

The GT-482xx provides a simple interface for low-cost 32-bit bus processors operating at a 16-50Mhz clock rate. The GT-482xx provides glueless interface to the following processors:

- IDT 3041
- MIPS 64-bit CPUs (via the Galileo GT-64010A, GT-64011/14 and GT-64120 components)
- Motorola ColdFire CPUs (small amount of glue logic required for demuxed bus versions)
- Intel i960<sup>®</sup>Jx and i960<sup>®</sup>Rx
- Other CPUs may be attached to the GT-482xx via a PCI bus through the GT-64111 PCI Bridge/Bridge Memory Controller available from Galileo

In addition, the GT-482xx provides interface to the following processors requiring a minimum amount of glue logic:

- 80486 and derivatives
- i960<sup>®</sup>Cx and i960<sup>®</sup>Hx processors
- PowerPC 401/403 family

The CPU performs management functions such as:

- SNMP and RMON
- VLAN programming
- IP Multicast session initiation
- Spanning tree BDPU processing
- Packet trapping/transmission/reception
- Address table access and query.
- Layer 3 routing

**NOTE:** A CPU is not required in unmanaged GT-482xx configurations.

### 1.5 Synchronous GRAM/DRAM Interface

The GT-482xx interfaces directly to 1 or 4 Mbytes of synchronous graphics RAM (SGRAM) or standard SDRAM. The DRAM is used to store the incoming/outgoing packets as well as the Address Table and other device data structures. The interface to the SGRAM is glueless; all signals needed to control the memory are provided.

The GT-482xx's SGRAM configurations are:

- 1 MByte (one 256K x 32 device): Address table contains up to 2K addresses, 512 Rx buffers.
- 4 Mbyte (two 1Mb x 16 device): Address table contains up to 8K addresses, 2048 Rx buffers.

### 1.6 Address Recognition

The GT-482xx can recognize up to 8192 (2,024 in 1Mbyte configuration) different Unicast MAC addresses and unlimited Multicast/Broadcast MAC addresses. An intelligent address recognition mechanism enables filtering and forwarding packets at full Fast Ethernet wire speed. Hardware address aging and static address support is also included.

The GT-482xx provides an address self-learning mechanism. Each device has a private AddressTable located in its DRAM array. As the GT-482xx learns new addresses, it updates the CPU (if present) by sending a New\_Address message.

The GT-482xx performs Unicast address aging on its Address Table. This can be done automatically or when triggered by a CPU request. See Section 6.7 "Address Aging" on page 30

## 1.7 IP Multicast and VLAN Support

When a management CPU is present, the GT-482xx traps IGMP packets over Ethernet/802.3 (on IPv4 and partial IPv6) and passes these to the CPU. The CPU may then program the port-mask (added to each entry in the Address Table) and the GT-482xx will lookup the Multicast and the Broadcast entries in the table and forward them according to the port-mask.

In addition, the CPU can program the port masks for Source Addresses. The GT-482xx forwards the Multicast and Broadcast packets according to the Source Address port masks only if the Multicast address is not present in the Address Table.

For Unicast packets, the GT-482xx checks that the source port appeared in the destination address port-mask bit and forwards the packet only if this bit is set. See also Section 6.4 "Locked Port" on page 27.

When unmanaged, the GT-482xx treats Multicast packets as Broadcast, and passes them to all ports, except for the source port.

## 1.8 Priority Queueing

Each port, including the CPU has two transmit queues for high and low priority. The GT-482xx inserts the packet to the high priority queue if one of the following conditions occur:

- Priority bit in the Source Port Control register is set
- Priority bit in the Address Table is set for the destination address
- Priority bit in the Address Table is set for the source address
- The incoming packet contains a 802.1Q tag and the most significant bit of the Quality of Service field in the tag is set.

The CPU is able to force the priority regardless of the result of the above conditions. This is done by setting the ForcePri bit in the Port Control register. When ForcePri is set, the priority is defined only by the priority bit in the Source Port Control register.

In addition to the above conditions, all IGMP, BPDU, EASE sampled packets and New\_Address messages are entered into the CPU high-priority queue. All Unknown (both Unicast and Multicast) and Broadcast packets are entered into the low-priority CPU queue.

## 1.9 Network Management Features

The GT-482xx provides comprehensive management capabilities enabling the switch OEM to implement a wide range of network management features.

For OEMs offering RMON capability, the GT-482xx provides per-port statistic counters to implement the first four groups of RMON.

The GT-482xx includes a unique packet sampling capability invented by the Hewlett-Packard Company called HP-EASE (Embedded Advanced Sampling Environment.) Each port has the ability to take "snapshots" of packet data at programmable intervals. These samples are forwarded to the management CPU for processing. The samples can be used to implement HP-EASE compatible messages for OpenView environments, or to create custom management information bases such as statistical RMON. Since packets are sampled using this technology, less local processing is required over standard RMON implementations. The source addresses of packet errors are also sent to the CPU allowing the switch OEM to support error counters in RMON host and matrix groups.

The GT-482xx includes hardware assistance for bridge-spanning tree algorithm and full hardware support for address aging.

## 1.10 Differences Between the GT-48212, GT-48208 and GT-48207

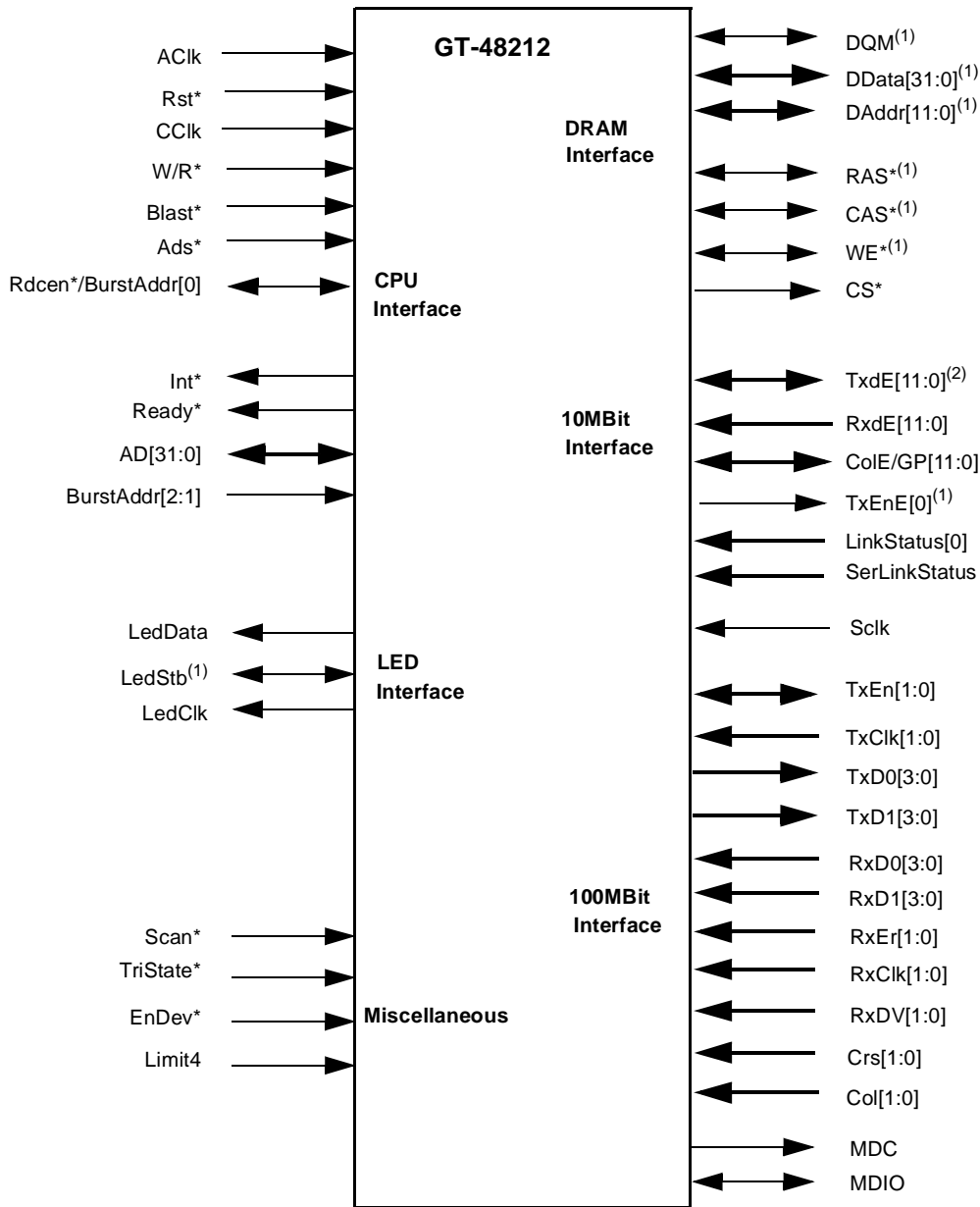
The differences between the three Galaxy devices are summarized in Table 1.

**Table 1: Pinout Differences**

Galaxy Device	Functions not implemented	Pins Deleted	Note
GT-48212	N/A	N/A	Baseline pinout
GT-48208	Ethernet ports 1, 5, 7 and 11	All pins related to these ports	To use a GT-48212 in a GT-48208 socket, the four unused Ethernet ports must be properly disabled.
GT-48207	Ethernet ports 1, 5, 7 and 11 Management CPU interface	All pins related to the these Ethernet ports and the CPU interface	To use a GT-48212 in a GT-48207 socket, the four unused Ethernet ports and the CPU interface must be properly disabled.

## 2. PINOUT

Figure 1: Logic Symbol



<sup>(1)</sup> - Pins sampled at RESET to establish the GT-482xx Parameters  
<sup>(2)</sup> - Pins sampled at RESET to establish Duplex mode in the 10Mbps ports

## 2.1 Pin Functions and Assignment

Table 2: Pin Functions

Symbol	Type	Description
<b>CPU Interface</b>		
Rst*	I	<b>RESET:</b> Active Low. Resets the GT-482xx to its initial state. This signal must be asserted for at least 10 MII clock cycles. Upon reset deassertion, the GT-482xx clears the internal empty list and the Address Table.
AClk	I	<b>Internal Clock:</b> This clock provides the timing for the GT-482xx internal units. All units, except for the Serial interface and the CPU interface, use this clock. This clock can vary between CClk and 66Mhz. AClk is also used to clock the synchronous DRAM. AClk frequency must be no higher than 4x CClk frequency.
CClk	I	<b>CPU Clock:</b> This clock provides the timing for the GT-482xx CPU interface. The clock can vary between 16Mhz to 50Mhz. In unmanaged switch operation, CClk should be tied to the 25MHz 100Mbit PHY clock. CClk frequency must not be lower than 25% of AClk frequency, and not higher than total AClk frequency.
AD[31:0]	I/O	<b>Address Data:</b> 32-bit multiplexed CPU address and data lines. During the first clock of the transaction, AD<31:2> contains a physical word address (30 bits). During subsequent clock cycles, AD<31:0> contains data.
Blast*	I	<b>LAST in Burst:</b> Indicates the last word in the burst. The maximum burst size is 8 words. Blast* has alternate meanings depending on CPU mode.
Ready*	O	<b>READY:</b> Indicates that AD[31:0] lines contain valid data. Ready* has alternate meanings depending on CPU mode. This output pin features an open-collector driver and should be tied as "wired-OR" in multiple GT-482xx designs.
Ads*	I	<b>Address Strobe:</b> Indicates that AD[31:0] holds addresses (when deasserted it holds data). Ads* has alternate meanings depending on CPU mode.
W/R*	I	<b>Write Read:</b> Indicates Write/Read transaction. The polarity of this pin is programmable depending on CPU mode.
Int*	O	<b>Interrupt:</b> Interrupt request line. Int* is asserted by the GT-482xx when one (or more) of the unmasked bits in the Interrupt Cause registers are set.
BurstAddr[2:1]	I	<b>BurstAddr:</b> In 3041 mode, contains bits [3:2] of the physical-byte address. In 64010/11 mode, contains bits [2:1] of the address. (See Table 16, "RESET Pin Strapping Options," on page 66.)
RdCen*/BurstAddr[0]	I/O	<b>RdCen*/BurstAddr:</b> Read Buffer Clock Enable / Burst Address - In 3041 mode, this output pin indicates to the 3041 that the GT-482xx placed valid data on the AD bus. In 64010/11 mode, contains bit [0] of the address. When programmed to be an output, this pin features an open-collector driver and should be tied as "wired-OR" in multiple GT-482xx designs.
<b>DRAM Interface</b>		

Table 2: Pin Functions (Continued)

Symbol	Type	Description
DQM	I/O	<b>Data output mask:</b> In normal operation, used in read and write cycles to DRAM. During reset, this pin is sampled by the GT-48212 to indicate the DRAM size. The GT-482xx always accesses 32-bit values and does not require a separate DQM for each byte. (See Table 16, "RESET Pin Strapping Options," on page 66.)
DAddr[11:0]	I/O	<b>DRAM Address:</b> In normal operation, Addr lines contains the DRAM address and bank selection. During reset, pins <11:0> are sampled by the GT-48212. (See Table 16, "RESET Pin Strapping Options," on page 66.)
Ras*	I/O	<b>Row Address Strobe:</b> In normal operation, indicates the Row Address. During reset, this pin in conjunction with Cas* and WE* indicate the CPU Type. (See Table 19, "CPU Mode Selection," on page 70.)
Cas*	I/O	<b>Column Address Strobe:</b> In normal operation indicates the Column Address. During reset, this pin in conjunction with Ras* and WE* indicates the CPU Type. (See Table 19, "CPU Mode Selection," on page 70.)
CS*	O	<b>Chip Select:</b> DRAM Chip Select. Indicates the DRAM chip select.
WE*	I/O	<b>Write Enable:</b> In normal operation indicates DRAM write transaction. During reset, this pin in conjunction with Ras* and Cas* indicates the CPU Type. (See Table 19, "CPU Mode Selection," on page 70.)
DData[31:0] also used for configuration parameters	I/O	<b>DRAM Data/GT-482xx parameters:</b> Multiplexed 32-bit SGRAM data bus and the GT-482xx parameters. In normal operation DData[31:0] connect directly to the data input/output pins of the SGDRAM devices. During reset, some of these pins are sampled by the GT-482xx. (See Table 16, "RESET Pin Strapping Options," on page 66.)
<b>10Mbps Interface</b>		
TxDE[11:0]	IO	<b>Transmit Data (Ethernet) / Duplex Mode.</b> In normal operation carries the Transmit Data for the 10Mbps ports. During reset, these pins are sampled by the GT-482xx to indicate the duplex mode. (See Table 16, "RESET Pin Strapping Options," on page 66.)
RxDE[11:0]	I	<b>Receive Data (Ethernet):</b> This pin carries the receive data for the 10Mbps ports
ColE/GP[11:0]	I	<b>Collision (Ethernet)/General Purpose pins:</b> Collision detect in AUI mode. In 10Base-T or 10Base-FL these pins are used as general purpose pins. They are sampled or driven according to the General Purpose register values.  These pins are GP input pins by default, therefore, in 10BaseT and 10BaseF modes, these pins should be pulled HIGH or LOW through a 4.7K Ohm resistor.
TxEn[0]	IO	<b>Transmit Enable:</b> In normal operation indicates that the packet is being transmitted on port 0. During reset, this pin is sampled by the GT-482xx to indicate AUI type. (See Table 16, "RESET Pin Strapping Options," on page 66.)
Sclk	I	<b>Serial Clock:</b> 80Mhz clock. This clock is used to recover the receive data and to generate the transmit clock for the 10Mbps ports.

Table 2: Pin Functions (Continued)

Symbol	Type	Description
SerLinkStatus	I	<b>Serial Link Status Indication:</b> This pin is used to serially shift the Link Status of ports 1 to 11 from the external PHY. LEDClk and LEDStb are used to clock and strobe the data (assertion of LedStb indicates the beginning of the link data stream). SerLinkStatus pin is sampled at the rising edge of LedClk. It should be driven at the falling edge of the LEDClk. Polarity: High SerLinkStatus: link is good Low SerLinkStatus: link fail. (See Figure 9: Example of Serial Link Status Indicator of Port 1 Link Fail on page 47).
LinkStatus[0]	I	<b>Link Status Indication:</b> This pin is used to indicate the Link Status of port 0 from the external PHY. Polarity: High LinkStatus[0]: link is good Low LinkStatus[0]: link fail. (See Figure 9: Example of Serial Link Status Indicator of Port 1 Link Fail on page 47).
<b>10/100Mbps Interface (MII)</b>		
TxEn[1:0]	O	<b>Transmit Enable:</b> Active HIGH. This output indicates that the packet is being transmitted. TxEn is synchronous to TxClk.
TxClk[1:0]	I	<b>Transmit Clock:</b> Provides the timing reference for the transfer of TxEn, TxData signals. TxClk frequency is one fourth of the data rate (25 MHz for 100Mbps, 2.5 MHz for 10Mbps, 60MHz for 240Mbps). TxClk nominal frequency should match the nominal frequency of RxClk for the same port.
TxD0[3:0]	O	<b>Transmit Data 0:</b> Outputs the Port0 Transmit Data. Synchronous to TxClk[0].
TxD1[3:0]	O	<b>Transmit Data 1:</b> Outputs the Port1 Transmit Data. Synchronous to TxClk[1].
Col[1:0]	I	<b>Collision detect:</b> Active HIGH. Indicates a collision has been detected on the wire. This input is ignored in full-duplex mode. Col is not synchronous to any clock.
RxD0[3:0]	I	<b>Receive Data 0:</b> Port 0 Receive Data. Synchronous to RxClk[0].
RxD1[3:0]	I	<b>Receive Data 1:</b> Port 1 Receive Data. Synchronous to RxClk[1].
RxEr[1:0]	I	<b>Receive Error.</b> Active HIGH. Indicates that an error was detected in the received frame. This input is ignored when RxDV for the same port is inactive.
RxCik[1:0]	I	<b>Receive Clock.</b> Provides the timing reference for the transfer of the RxDV, RxData, RxError signals (per port). Operates at either 25 MHz (100Mbps), 2.5 MHz (10Mbps) or 60MHz (240Mbps). The nominal frequency of RxClk (per port) should match the nominal frequency of that port's TxClk.
RxDV[1:0]	I	<b>Receive Data Valid:</b> Active HIGH. Indicates that valid data is present on the RxData lines. Synchronous to RxClk.
CrS[1:0]	I	<b>Carrier Sense:</b> Active HIGH. Indicates that either the transmit or receive medium is non-idle. CrS is not synchronous to any clock.



Table 2: Pin Functions (Continued)

Symbol	Type	Description															
MDC	O	<b>Management Data Clock:</b> Provides the timing reference for the transfer of the MDIO signal. This output may be connected to the PHY devices of both ports.															
MDIO	I/O	<b>Management Data Input/Output:</b> This bidirectional line is used to transfer control information and status between the PHY and the GT-482xx. It conforms with IEEE Std 802.3. This signal may be connected to the PHY devices of both ports. When not driven by an MII compliant PHY, this pin must be connected to a pull-up or pull-down resistor. PHY register reads performed by the GT-482XX will be decoded as all 1's or all 0's respectively."															
<b>LED Interface Pins</b>																	
LEDStb/LEDMode	I/O	<b>LED Strobe/LED Mode:</b> Multiplexed LED Strobe and LED mode. In normal operation, envelopes the full-duplex and Link-status data stream (bit #0 to bit#27) of a valid data frame on LEDData output. During reset, indicates the LED mode. (See Table 16, "RESET Pin Strapping Options," on page 66.)															
LEDClk	O	<b>LED Clock:</b> 1 MHz clock. This output is used to clock the LEDStb and LEDData outputs. During RESET, LEDClk output is tri-stated.															
LEDData	O	<b>LED Data:</b> Active LOW. Serial data bit stream which contains the LED indicators per port. The data is shifted out using the LEDClk. LEDStb is used to mark the first data bit.															
<b>Miscellaneous Interface Pins</b>																	
EnDev*	I	<p><b>Enable Device:</b> This pin together with EnDev bit in the Global Control register activate the ports as follows:</p> <table border="1"> <thead> <tr> <th>EnDev*</th><th>EnDev bit</th><th>Mode</th></tr> </thead> <tbody> <tr> <td>0</td><td>x</td><td>Enable</td></tr> <tr> <td>1</td><td>0</td><td>Disable (default)</td></tr> <tr> <td>1</td><td>1</td><td>Enable</td></tr> </tbody> </table> <p><b>Note:</b> EnDev* is not synchronous to any clock</p>	EnDev*	EnDev bit	Mode	0	x	Enable	1	0	Disable (default)	1	1	Enable			
EnDev*	EnDev bit	Mode															
0	x	Enable															
1	0	Disable (default)															
1	1	Enable															
Scan*	I	<p><b>Scan:</b> This pin together with TriState* indicate the GT-482xx mode of operation as follows:</p> <table border="1"> <thead> <tr> <th>Scan*</th><th>TriState*</th><th>Mode</th></tr> </thead> <tbody> <tr> <td>1</td><td>1</td><td>Normal operation</td></tr> <tr> <td>0</td><td>1</td><td>Factory test mode (Reserved)</td></tr> <tr> <td>1</td><td>0</td><td>The GT-482xx drives all outputs and I/O pins to high impedance.</td></tr> <tr> <td>0</td><td>0</td><td>Factory test mode (Reserved)</td></tr> </tbody> </table> <p>Factory test modes are Reserved and are not to be used in system. Failure to observe this restriction could result in damage to the device.</p>	Scan*	TriState*	Mode	1	1	Normal operation	0	1	Factory test mode (Reserved)	1	0	The GT-482xx drives all outputs and I/O pins to high impedance.	0	0	Factory test mode (Reserved)
Scan*	TriState*	Mode															
1	1	Normal operation															
0	1	Factory test mode (Reserved)															
1	0	The GT-482xx drives all outputs and I/O pins to high impedance.															
0	0	Factory test mode (Reserved)															

Table 2: Pin Functions (Continued)

Symbol	Type	Description
TriState*	I	<b>Tri-State:</b> This pin together with Scan* indicate the GT-482xx mode of operation as described above.
Limit4	I	<p><b>Backoff Algorithm:</b> This pin together with Limit4 pin in the Global Control register selects the number of consecutive packet collisions that will occur before the collision counter is reset. When the logical OR of this pin and the bit is LOW, 16 consecutive collisions must occur before the collision counter is reset (802.3 standard). When HIGH, four consecutive collisions must occur before the collision counter is reset (more aggressive.)</p> <p><b>Note:</b> Limit4 is not synchronous to any clock.</p>

### 3. GALAXY FAMILY OVERVIEW

The Galaxy Family of switching devices is designed to provide the lowest cost solutions for desktop and workgroup Ethernet switching.

The Galaxy Family currently includes the following devices:

- **GT-48212** Twelve ports of 10BaseX, two ports of 100BaseX with management interface
- **GT-48208** Eight ports of 10BaseX, two ports of 100BaseX with management interface
- **GT-48207** Eight ports of 10BaseX, two ports of 100BaseX but without any management capability. The GT-48207 is designed for cost-sensitive "hub replacement" applications.

During the design of the Galaxy Family, Galileo Technology had the following goals:

- **Produce a Product Yielding the Lowest Possible System Cost.** This is evident in the Galaxy Family's high level of integration and lower overall bill-of-materials. For example, Galaxy Family devices do not require the external address Lookup/VLAN engines and PHY devices required by other solutions. We have also made it very simple to interface a CPU.
- **Make it Simple for the Switch OEM to Build a Modular/Flexible Product Line.** The GT-48212, GT-48208, and GT-48207 all share a common pinout. This allows the OEM to build a variety of port densities and management options on the same printed circuit board. This flexibility allows the OEM to manage production planning more effectively.
- **Provide Software Compatibility to Future Generations.** Future generations of Galaxy devices that support different port speeds/densities will strive for backwards software compatibility. This allows you to write your management code once, and only modify it as your end-user requirements change.

Galileo Technology will continue to extend the Galaxy Family to meet future needs for workgroup/desktop LANs. The Galaxy Family uses a "store-and-forward" switching approach. Store-and-forward was chosen for the following reasons:

- Store-and-forward switches allow switching between different speed media (e.g. 10BaseX and 100BaseX.) Such switches require the large elastic buffers that are provided by the SDRAM arrays.
- Store-and-forward switches improve overall network performance by acting as a "network cache", effectively buffering packets during periods of heavy congestion.
- Store-and-forward switches prevent the forwarding of corrupted packets by analyzing the frame check sequence (FCS) before forwarding to the destination port.

#### 3.1 Basic Operation

The basic operation of the GT-482xx is quite simple. The GT-482xx receives incoming packets from the Ethernet wire, searches in the Address Table for the Destination MAC Address and then forwards the packet to the appropriate port. The destination port can either be local (one of the GT-482xx's ports) or in a different GT-482xx device that is connected via the Fast MII expansion bus. If the destination address is not found, the GT-482xx treats the packet as a Multicast packet and forwards the packet to all ports of all the devices in the system specified to forward unknown packets.

The GT-482xx automatically learns the port number of attached network devices by examining the Source MAC Address of all incoming packets. If the Source Address is not found in the GT-482xx's Address Table, the device adds it to the table (indicating on which port the address resides). The GT-482xx then notifies the CPU (if present) of the new address via a New\_Address message.

#### 3.2 Address Learning

The GT-482xx can learn up to 8K unique MAC addresses. Addresses are stored in the Address Table located in the DRAM. The Address Table is managed automatically by the GT-482xx (i.e. a new address is automatically added to the Address Table). The GT-482xx's address learning process is detailed in Section 6. "MAC Address Table" on page 26.

The Address Table includes information regarding target port, aging status, static/dynamic status, VLAN, and flags to force processor intervention. The management CPU has the ability to insert, remove or modify the entries.

### 3.3 Packet Buffering

Incoming packets are buffered in the SDRAM array. These buffers provide elastic storage for transferring data between low-speed and high-speed segments. The GT-482xx automatically manages packet buffers.

### 3.4 Packet Forwarding

Once an address has been learned and the packet is buffered, it must be forwarded. The packet forwarding mechanism for the GT-482xx is handled automatically based on the destination address. Optionally, the CPU can be involved in Unicast packet forwarding decisions by using Intervention Mode. For more information about Intervention mode see Section 7.5 on page 35. If a CPU is utilized for system management functions, Multicast packets are forwarded to the CPU for forwarding decisions.

### 3.5 Terminology

It is important to understand the basic terminology used to describe the Galaxy Family before getting into a detailed description. Table 3 explains the terms used throughout this document.

**Table 3: Terminology**

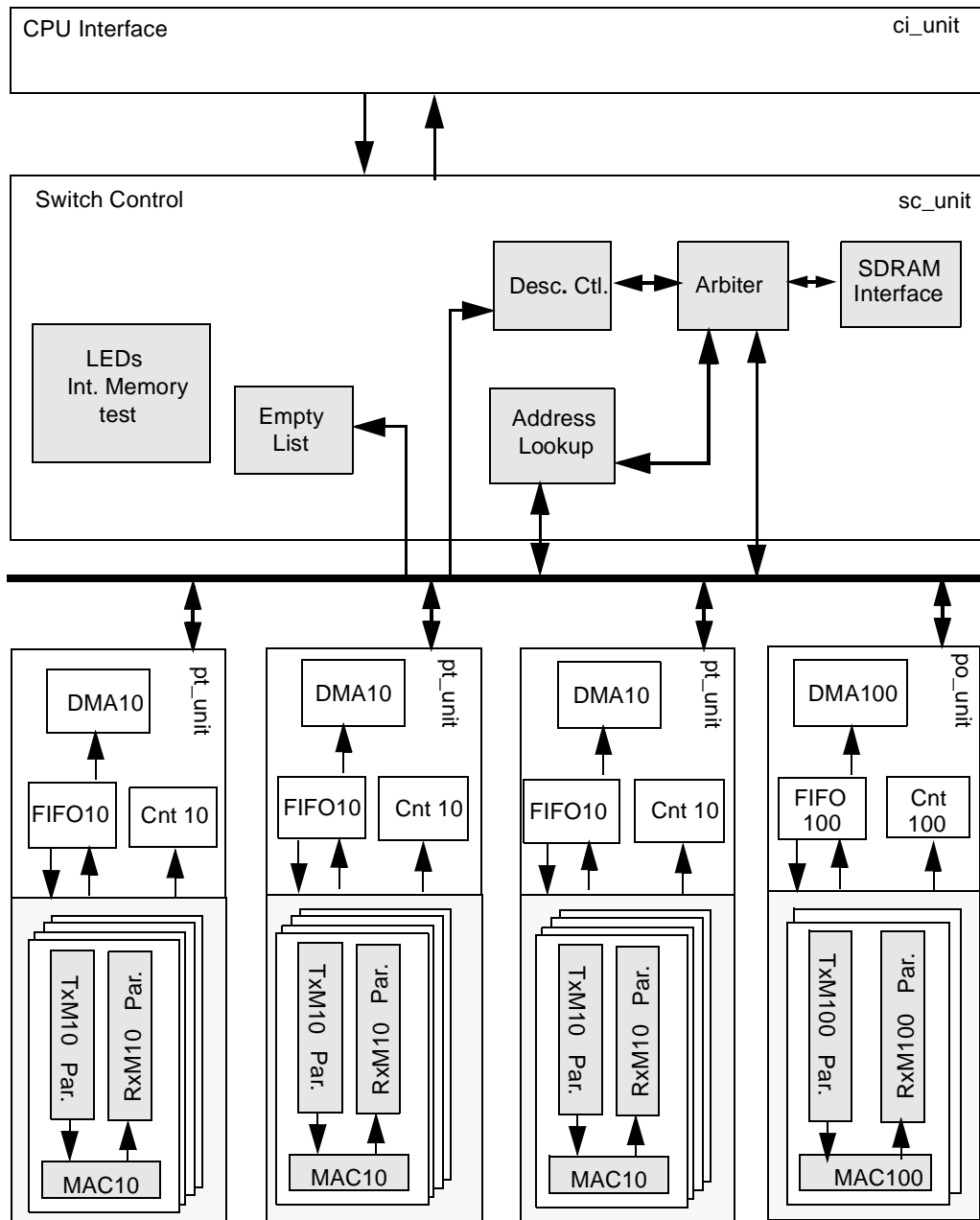
Term	Definition
Address Table	The Address Table is a data structure in the GT-482xx's DRAM that contains all learned MAC addresses, and routing information associated with those addresses.
Source Address	The Source Address (SA) is the MAC address from which a received packet was sent.
Destination Address	The Destination Address (DA) is the MAC address to which a received packet is directed.
Port Number	Each Ethernet port on a GalNet device has an associated port number. The GalNet device associates Port Numbers with the MAC addresses located on those ports.

#### 4. MICROARCHITECTURAL OVERVIEW

The GT-482xx is comprised of the following hardware units:

<b>CPU Interface</b>	The CPU Interface Includes all interface functions to the CPU bus. It handles all interconnect control signals and executes CPU slave transactions. The CPU interface unit controls the packet transfer between the GT-482xx and the CPU. It includes simple registers for passing messages to/from the CPU. The GT-482xx is always a slave on the CPU interface.
<b>SDRAM Interface</b>	Includes all the interface functions to 1 or 4Mbyte SGRAM or SDRAM. It generates the control signals and drives the address/data lines.
<b>Switching Core</b>	Performs the switching functions between the 12 or 8 10Mbit ports and the two 100Mbit ports and the CPU.
<b>10Mbit Port Unit</b>	There are three 10Mbit port units in the GT-48212 (two in the GT-48207 and GT-48208). Each handles four 10Mbit ports and consists of DMA, four 64x32-bit word FIFOs (one for each port), counters and serial interface blocks. Packets are transferred between the DRAM interface and the Ethernet 10Mbit serial interface. It uses the GT-482xx RXM, TXM and SIA with the addition of the MAC-CONTROL block that performs IGMP trapping and flow control.
<b>100Mbit Port Unit</b>	There is one such unit. Each handles two 100Mbit ports and consists of DMA, two 128x32-bit FIFOs, counters and serial blocks. Packets are transferred between the DRAM interface and the Ethernet 10Mbit serial interfaces. It uses the GT-482xx RXM, TXM and MII with the addition of the MAC-CONTROL block that performs IGMP trapping and flow control.

**Figure 2: Block Diagram of the GT-48212**



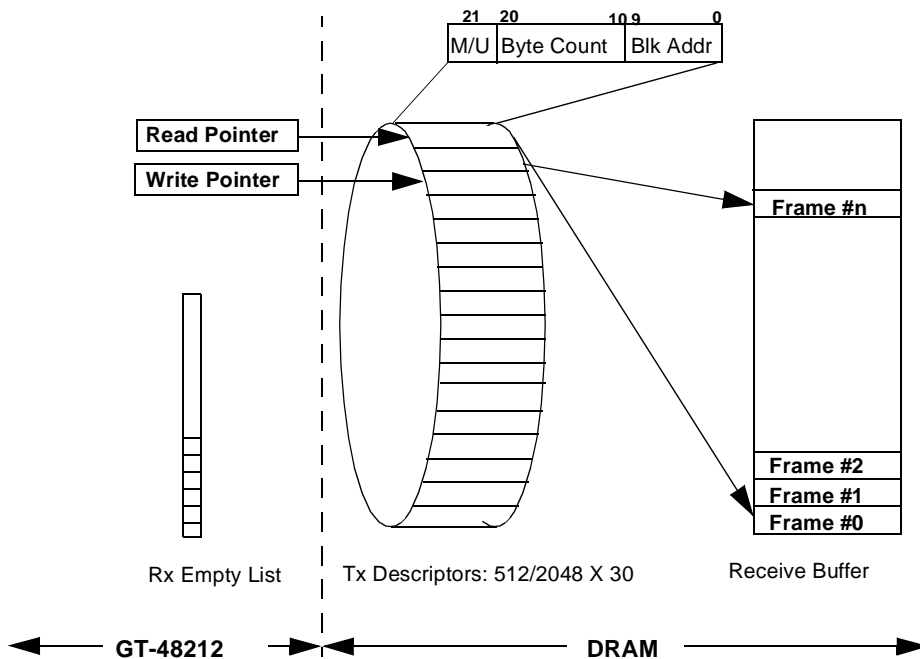
## 5. BUFFERS AND QUEUES

The GT-482xx incorporates 30 transmit queues for the 12 or 8 Ethernet ports, two Fast Ethernet ports and the CPU bus port. In each port there are two queues, one for high and one for low priority. The GT-482xx also contains a common receive buffer area. The receive buffers are allocated to the receive ports and the CPU. The GT-482xx incorporates a simple mechanism to prevent head-of-line blocking (See Section 5.2 "Head-of-Line Blocking" on page 25). The receive buffers as well as the transmit queues are located in the DRAM with the Address Table. See Table 4 on page 24 for DRAM address mapping in the GT-482xx.

The GT-482xx data structure has the following components:

- **Receive Buffer** - A common Receive Buffer area for all ports. The buffer is divided into 512 or 2048 blocks (depending on the DRAM size) of 1.5KBytes (1536 bytes) each. Each block contains the entire packet.
- **Rx Empty List** - A list of 512 or 2048 bits. Each bit contains the status of its appropriate receive block in the DRAM (empty or occupied).
- **Tx Descriptors** - A set of 30 transmit descriptor rings. Each ring contains 512 or 2048 descriptors. The descriptor size is one 32-bit word and contains the Block Address divided by 0x600 (1.5K), the byte count and the packet type (Multicast or Unicast).
- **Read/Write Pointers** - 30 pairs of pointers to the transmit descriptors.

**Figure 3: GT-482xx Buffers and Queues**



**Table 4: GT-482xx DRAM Address Mapping**

Memory	Description	1Mbyte	4Mbyte
Rx Buffers	512 buffers for 1Mbyte 2048 buffers for 4 Mbyte	0x00000 - 0xBFFFF	0x00000 - 0x2FFFFFF
Multicast Indicators	4 bytes per buffer	0xC0000 - 0xC07FF	0x300000 - 0x301FFF
Tx Descriptors	28 descriptor queues	0xC0800 - 0xCE7FF	0x302000 - 0x339FFF
CPU Descriptors	2 descriptor queues	0xCE800 - 0xD6800	0x33A000 - 0x341FFF
Reserved	-	0xD0800 - 0xE7FFF	0x342000 - 0x39FFFF
Address Table	2K addresses for 1Mbyte 8K addresses for 4Mbyte	0xE8000 - 0xFFFFF	0x3A0000 - 0x3FFFFFF

## 5.1 Rx Buffer Threshold Programming

The number of receive buffers allocated to each port of the GT-482xx is controlled by the RxBufThr fields in the Rx Buffer Threshold registers. The default value is 32 (10Mbps) or 64 (100Mbps) buffers per port for 1MB DRAM and 64 (10Mbps) or 592 (100Mbps) buffers per port for 4MB DRAM. If the buffer threshold is disabled by clearing the



BufThrEn bit in the Global Control register or the DAddr[10] pin is held LOW during reset, the GT-482xx dynamically allocates the buffers to the 14 Ethernet ports and the CPU. Therefore, there are no limits on each buffers' allocation. The Rx Buffer Threshold value can be used for performance tuning during development. For more details about the RX Threshold register see Table 47 on page 99, Table 48 on page 99, and Table 49 on page 100.

If a received packet overflows the Receive Buffer allowance, the packet is discarded and the Dropped Packets counter incremented.

## 5.2 Head-of-Line Blocking

The GT-482xx incorporates a simple mechanism to prevent head-of-line (HOL) blocking. This is done by passing an indication from the transmit queues to the buffer allocation machine. A packet destined to a transmit port will be discarded if the following conditions are met:

- The transmit queue of destination port exceeds a programmable threshold (as defined in the TxThr field in the Tx Threshold Register)
- Number of buffers allocated to this source port is equal to or greater than the HolLimit value (in Rx Threshold registers).

HOL prevention is enabled only when the GT-482xx is in Fixed Rx Buffer Mode (BufThrEn = '1').

## 6. MAC ADDRESS TABLE

The GT-482xx's Address Table resides in the local DRAM array. Table 5 and Table 6 show the Address Table structure. The Address Table structure occupies approximately 320Kbytes (in 4Mbyte configuration) or 80Kbytes in 1Mbyte configuration) and is entirely controlled and initialized by the GT-482xx. Following RESET, the GT-482xx initializes the Address Table by invalidating all entries.

Modifications to the Address Table are normally made through New\_Address message requests by the CPU. The Address Table can be accessed directly, however, this mode is not recommended. Contact Galileo Technology if your application requires direct Address Table access.

### 6.1 Forwarding Mask

The Forwarding Mask (forw<14:0>) controls forwarding options in managed systems. Each bit in the mask corresponds to a specific port in the GT-482xx device. Port 0 corresponds to bit 0 (least significant bit), Port 13 corresponds to bit 13. Bit 14 corresponds to the CPU port (if present). When the GT-482xx performs new address learning, the forw[14:0] bits are set to 0x7FFF (all 1's).

The forw<14:0> operation is described in Table 6.

### 6.2 Port Number

The Port Number field in the Address Table corresponds to the port on which a specific address was detected (learned). Port 0 = 0x0 in this field, CPU port (port 14) corresponds to 0xE.

**Table 5: Address Table Entry Format**

forw<14:0>	MAC<47:0>	Is	Id	mul	st	port #	Ps	Pd	A	Sk	V
15	48	1	1	1	1	4	1	1	1	1	1
Number of bits in each field is shown above.											

**Table 6: Address Table Entry Field Description**

Bit	General Description
V	Valid - Indicates that the entry is valid 1 - Valid 0 - Not Valid
Sk	Skip - Skip this entry. It is being used to delete an entry 1 - Skip this entry 0 - Don't skip this entry
A	Aging - This bit is used in the address aging process. - Cleared by the GT-482xx upon receiving a packet from this station. - Set by the GT-482xx during aging processing (For more information see the Aging algorithm in Section 6.7)
Pd	Destination address effect on priority. 1 - enqueue to high priority queue.

**Table 6: Address Table Entry Field Description (Continued)**

Bit	General Description
Ps	Source address effect on priority. 1 - enqueue to high priority queue.
Port#	Port number for this MAC address.
St	Static 1 - The entry is static. The Port# can not be modified 0 - The entry can be modified
Mul	Multiple - meaning when bit St is set 1 - Forward this packet as Multicast destination address entry. 0 - Forward this packet only to the destination port.
Id	Forward to CPU only when it is Destination Addresses 1 - Forward packet to the CPU when address is Destination. 0 - no effect
Is	Forward to CPU only when it is Source Addresses 1 - Forward packet to the CPU when address is Source. 0 - no effect
MAC	48 bit of MAC address.
Forw	Forward the packet ALSO to the following ports/CPU (bit mask).

### 6.3 Address Learning Process

The GT-482xx has a self-learning mechanism for learning the MAC addresses of attached Ethernet and Fast Ethernet devices in real time. The GT-482xx searches for the Source Address (SA) of an incoming packet in the Address Table and acts as follows:

1. If the SA was not found in the table (a new address) at the end of the packet, the GT-482xx updates its Address Table. The GT-482xx can also send a New\_Address message to the CPU (if present), and if programmed to do so.
2. If the SA was found, the GT-482xx compares the port number to the existing port number. If they are different and the static bit is clear, it updates the entry with the new information (and optionally notifies the CPU). Else, no action is taken.
3. When a source address exists in the table the Aging bit is set.

When the Static bit in the Address Table is set, the GT-482xx does not modify the entry. When both the Static and Multiple bits are set, the GT-482xx does not modify the entry and forwards the packet as Multicast. It forwards the packet according to the Forw bits of the Destination Address.

When two Galaxy devices managed by the same CPU are configured, the CPU may receive new address information from both devices, since each device reports learning on its own port. The first learning is on the network port, and the second learning is on the expansion port.

### 6.4 Locked Port

The CPU has the ability to lock one or more Ethernet/Fast Ethernet ports. This is done by setting the LockPort bit in the Port Control register. In this mode, the GT-482xx does not learn any new addresses received from this port.

The GT-482xx does not modify the Address Table and sends a New\_Address message only to the CPU. The GT-482xx decides whether to discard or forward the packet based on the DisLock bit in the Global Control register. If this bit is reset, then packets received from new SA on a locked port are discarded.

The CPU can also prevent sending unknown packets to this port. This is done by setting the ForUnk bit in the Port Control Register.

**Note:** In Locked Port mode, a station that moved to a new port receives a New Address, therefore, packets coming from the station that moved are discarded until the CPU reprograms the GT-482xx with the new port number.

#### 6.4.1 Port-Based VLAN Support

The user can implement port-based VLANs using the GT-482xx for both one chip and two chip configurations using the locked-port feature:

- The CPU programs all ports as locked ports.
- The GT-482xx sends New\_Address messages only to the CPU.
- The CPU sends the forward mask for each source address (that is legal for that port) back to the GT-482xx.

VLANs that span ports in two-chip GT-482xx configurations can be defined. Since all ports are locked ports, the GT-482xx checks the forwarding "legality" from source to destination, and assures security.

The user can define port-based VLANs with common ports and still maintain security. For more information see Section 6.6 "Address Recognition". The user can optionally define the forwarding masks to represent port-based VLANs that do not share ports.

#### 6.5 Address Entry Update and Query from CPU

This interface is used to receive New\_Address messages from the CPU. The messages can be new Address Table entries, updates to existing ones, or queries from the CPU. The query from the CPU is a mechanism in which the CPU can read the data from the Address Table. The GT-482xx responds by searching the Address Table for the requested MAC address, and returns the result to the CPU in the QUERY Register.

To perform these tasks the GT-482xx maintains three 32-bit registers called NA\_From\_CPU registers. These three 32-bit registers are written by the CPU to update the GT-482xx with a new/modified address entry, or with a query message

CPU initiates a Query by sending a New\_Address message with a Query bit set. The GT-482xx responds by searching the Address Table for the entry and places the address information in the QUERY register. The GT-482xx then generates an interrupt.

#### 6.6 Address Recognition

The GT-482xx forwards incoming packets to their appropriate port(s) according to the Destination Address (DA), and Source Address (SA) in the Address Table and the programming modes of each port's Port Control register (PCR) and Global Control register (GCR) as follows:

1. The following principles are always valid:
  - A packet is not forwarded to any port if the port is disabled, link is down, or in span-mode (Except BPDU to a span port).
  - A packet is never forwarded on the port from which it was received.
  - If GCR.DisPktLock =0 then any received packet (Unicast or Multicast) from a port that has PCR.Locked=1, with a new SA (or SA that moved to this port), the packet is discarded, regardless of the DA.
2. Otherwise, the forwarding of a packet received on any of the 14 ports can be Unicast or Multicast Destination Addresses as follows:
  - 2.1 Unicast Destination Address:

If (SA has Is=1 or the DA has Id=1 (in the Address Table entries). Forward only to CPU (as intervention)

Else forward as shown in Table 7

**Table 7: Forwarding of Unicast Destination Address Packet**

SA found ?	DA found?	DA's forward-mask includes receive port?	DA is static & multiple	Forward to...
X	Yes	1	No	DA
X	Yes	0	X	Packet Dropped
Yes	No	-	-	All ports in SA<Forward_Mask> that have PCR.Forw_Unk=0 (and also to CPU if it is in the forward mask and GCR.Forw_Unk=1)
No	No	-	-	All ports that have PCR.Forw_Unk=0, and also CPU if GCR.Forw_Unk=1)
Yes	Yes	1	Yes	SA<Forward_Mask> bitwise AND DA<Forward_Mask>
No	Yes	1	Yes	DA<Forward_Mask>

Note: "SA found" represents a valid Source Address table entry found in the AddressTable.  
"X" represents "Don't Care"

## 2.2 Multicast Destination Address:

If the packet is BPDU, see Section 11.3 "Spanning Tree (BPDU) Support" on page 50.

If the packet is an IGMP packet, see Section 7.6 "IGMP Packet Support" on page 35.

If DA=FFFFFFFF and PCR.FilBroad=1 in the receiving port, the packet is discarded.

If DA=FFFFFFFF and PCR.FilBroad=0 and PCR.ForwBroadCPU=1, the packet is forwarded only to CPU.

If (SA has Is=1 or the DA (multicast) has Id=1 (in the Address Table entries) or GCR.ForwMulti-CPU=1, forward only to CPU (as intervention)

Else forward as shown in Table 8.

**Table 8: Forwarding of Multicast Destination Address Packet**

SA found?	Multicast DA found?	DA's forward-mask includes receive port?	Forward to...
Yes	Yes	1	SA<Forward_Mask> bitwise AND DA<Forward_Mask>
No	Yes	1	DA<Forward_Mask>

**Table 8: Forwarding of Multicast Destination Address Packet (Continued)**

SA found?	Multicast DA found?	DA's forward-mask includes receive port?	Forward to...
X	Yes	0	Packet Dropped
Yes	No	-	SA<Forward_Mask>
No	No	-	All ports (Including CPU)

Note: "SA found" represents a valid Source Address table entry found in the AddressTable.  
 "X" represents "Don't Care"

**Note:** The algorithms detailed in this section do not describe the handling of incoming PAUSE packets (for 802.3x flow-control). These packets are never forwarded to other ports, once they are recognized as PAUSE packets. If flow control is disabled, these packets are not recognized as PAUSE packets, and treated as regular Multicast packets.

## 6.7 Address Aging

The GT-482xx performs Unicast address aging on its Address Table. Aging can be done automatically or when triggered by a CPU request. Once a MAC address has been aged or removed from the Address Table, packets with this MAC address as its destination address are broadcasted to all ports. There are three aging modes programmed using bits [3:2] in the Global Control register as follows:

- GCR[3:2] = 00 No hardware aging support.
- GCR[3:2] = 01 Automatic aging support - the GT-482xx includes an internal aging timer. When the timer expires, the GT-482xx scans the Address Table and ages out (removes) all addresses for which the aging bit is clear.
- GCR[3:2] = 10 Trigger mode.
- GCR[3:2] = 11 Reserved.

In automatic aging mode, the GT-482xx automatically sets and clears the Aging bit in the Address Table. Static addresses are not aged out.

### 6.7.1 Determining Aging Time

The GT-482xx uses two variables to determine the aging time when aging is enabled:

- The frequency value of ACIk
- The Aging Timer variable, N, bits[14:9] at offset 0x0C.

The Source Address is removed (aged) from the Address Table if a packet with a learned Source Address is not received by the switch as follows:

$[(500/\text{ACIk Freq}) * N]$  seconds

and

$[(1000/\text{ACIk Freq}) * N]$  seconds

The default value of the Aging Timer is 30d (0x1E). Therefore, if ACIk is 50MHz, the default aging time will be 300 to 600 seconds. If ACIk is 66MHz, the default aging time will be 227 to 455 seconds.

The aging time range is determined by a scan mechanism that scrubs the entire table within  $[(500/\text{ACIk freq}) * N]$  seconds. The Age Scan pointer looks up all entries within this time range. A given entry can be located anywhere in the Address Table, therefore, the Age Scan pointer will look up the same entry twice in no less than  $[(500/\text{ACIk Freq}) * N]$  seconds, and no more than  $[(1000/\text{ACIk Freq}) * N]$  seconds.

### 6.7.2 CPU Trigger Mode

When the CPU Trigger Mode is enabled, the Address Table is scanned once for every trigger. The time for this scan is  $[(500/\text{ACIk Freq}) * N]$  seconds.

## 6.8 Static Addresses

The GT-482xx provides support for "Static" MAC addresses. IEEE 802.1d Chapter 3.9.1: "Static entries may be added to and removed from the filtering database under explicit management control. They are not automatically removed by any time-out mechanism". This means that when an address is selected as static, it is not removed from the Address Table during aging.

During normal address recognition, if an address is static and the port is unlocked, the GT-482xx does not update its Address Table parameters and does not send a New\_Address message to the CPU (if the address has changed ports). When the port is locked, the GT-482xx sends the New\_Address message to the CPU only.

## 6.9 Address Recognition Failure

An address recognition cycle can fail when more than 8K addresses have been entered into the Address Table. In the case of an address recognition failure the packet is treated as unknown and forwarded to all ports. An interrupt is also generated to the CPU (if present).

Address recognition failures are not fatal and do not require handling. Therefore, designers of unmanaged systems need not be concerned with such a failure. When such a failure occurs in managed systems, the Address Table may be "cleaned" of old addresses.

## 6.10 Forwarding Priority

The GT-482xx supports two levels of forwarding priority (high and low) for all 14 transmit ports, and for transmission to the CPU. Priority is supported by maintaining two packet queues per port.

### 6.10.1 Priority Assignment

BPDUs, IGMP and sampled HP-EASE packets are enqueued to High priority. New\_Address messages to the CPU are also entered into the High priority queue. Unknown Unicast, Unknown Multicast and Broadcast packets are enqueued to the CPU's Low priority queue when their reception to the CPU is enabled.

Known Unicast and known Multicast packets are enqueued according to the priority mode. The packet is entered to the High priority queue when one of the following conditions are met:

- Priority bit in the Source Port Control register is set for the input port
- Priority bit in the Address Table is set for the Destination Address
- Priority bit in the Address Table is set for the Source Address
- The incoming packet contains an 802.1Q tag and the most significant bit of the Quality of service field in the tag is set (priority is greater or equal to 4).

The CPU can force the priority regardless of the results of the above conditions by setting ForcePri bit in the Port Control register. When ForcePri is set, the priority is defined only by the priority bit in the Source Port Control register.

The priority of packets enqueued by the CPU are set by the CPU.

### 6.10.2 Priority Throttling

The Global Control register contains a "Priority Weight" field that controls the priority of the high priority queue versus the low priority queue. Options range from fair round robin (i.e. no difference in priority) to high priority (high priority packets are always forwarded first). There are six priority weights between the highest and lowest extremes as shown in Table 9.

**Table 9: Priority Queuing Options**

Priority Weight	Forwarding Behavior	Weight as% H/L
000	One packet transmitted from the high priority queue, and one packet transmitted from the low priority queue	50/50
001	Two packets from the high queue, one from the low	66/33
010	Four packets from the high queue, one from the low	80/20
011	Six from the high queue, one from the low	85.7/14.3
100	Eight from the high queue, one from the low	88.8/12.2
101	10 from the high queue, one from the low	90.9/9.1
110	12 from the high queue, one from the low	92.3/7.7
111	All packets from the high queue.	100/0



## 7. PACKET FORWARDING

This section details the procedures used to forward packets with the following conditions:

- Unicast packet to a local port
- Multicast packet to local ports and the CPU
- Packet destined for the CPU (Multicast, or Unicast packet from the GT-482xx to the CPU)
- Unicast/Multicast packet from the CPU to the GT-482xx

### 7.1 Forwarding a Unicast Packet to a Local Port

The sequence for forwarding a Unicast packet to a local port is as follows:

1. The incoming packet is fed to the Rx FIFO (there is a 64x32-bit Rx FIFO for the 10M ports and 128x32-bit Rx FIFO for the 100M ports) and is transferred to an empty block in the Receive Buffer area of DRAM.
2. In parallel, an address recognition cycle is performed for both the DA and the SA. The GT-482xx uses the DA's corresponding Port Number to queue the packet to the appropriate local port.
3. At the end of an error-free packet transfer, packet information is written to the appropriate port's transmit descriptor. This information includes the Byte Count and the Receive Buffer Block Address which is pointed to by the Write Pointer.
4. The Write Pointer of the outgoing port's transmit descriptor is incremented. The GT-482xx transmits whenever the Write Pointer is not equal to the Read Pointer.
5. At the end of the packet transmit process, the GT-482xx increments the Read Pointer and clears the appropriate bit in the Empty List.

### 7.2 Forwarding a Multicast Packet

The GT-482xx forwards Multicast packets to all local ports and the CPU using the same mechanism as described for Unicast packets. The GT-482xx has the ability to forward Multicast packets to a management CPU for intervention routing, if desired.

### 7.3 Forwarding a Packet to the CPU

Systems that utilize a CPU (CpuEn of the Global Control register) can receive packets from the GT-482xx using a simple slave interface. This includes the following packet types:

- Unicast packets destined for the CPU (port number in the Address Table equal to 14d)
- Multicast packets
- Unknown packets (if set by ForwUnk bit in the Global Control Register)
- IGMP packets
- BPDU messages
- Sniffer packets when the CPU is the target sniffer
- EASE packets

The CPU transmit descriptor queues are used to store the outgoing packets from the GT-482xx ports to the CPU, and to pass New\_Address messages to the CPU. The GT-482xx does not DMA packets to the CPU. The CPU reads the descriptors and then accesses the buffer contents directly from the GT-482xx's SDRAM. The GT-482xx provides the CPU with one descriptor at a time (in a dedicated, single register). The CPU can then access the packet in the SDRAM (read/write) and provide an End\_Of\_Packet message to the GT-482xx, in a dedicated register so that the GT-482xx will free the buffer in DRAM.

There are two packet queues (high and low priority). Each queue has the following data structure components:

- **CPU\_Tx\_Hi/Low\_Desc register.** These two registers (one for high priority and one for the low-priority queue) are loaded with the next descriptor. Each 64-bit register holds the next descriptor, a New\_Address message or an Errored Source address to the CPU. When the GT-482xx has a new CPU descriptor, it attempts to write it into one of the CPU\_Tx\_Hi/Low\_Desc registers (depending on priority). The new descriptor is written to the register provided that bit 31 of the second word is cleared. Bit 31 of

the second word (Valid bit) is automatically cleared when the CPU reads the register. If this bit is not cleared when the GT-482xx attempts to write, the next descriptor is reloaded when the valid bit is reset. An interrupt is asserted (if not masked) when a new descriptor is successfully written.

- **CPU\_EL\_Free\_Req register.** This 32-bit register is written by the CPU to signal to the GT-482xx that the buffer can be cleared in the SDRAM (typically this is done after the CPU has read the packet from the GT-482xx memory). The GT-482xx clears the busy bit in this register when it is ready to receive another command. The CPU reads this register before writing to it to check that it is clear.

The sequence for forwarding a Unicast packet to the CPU is as follows:

1. The incoming packet is fed to the Receive FIFO and transferred to an empty block in the DRAM.
2. In parallel, an address recognition cycle is performed for the SA and the DA (Unicast or Multicast).
3. At the end of a good packet transfer, the packet is forwarded to the CPU Tx queue. The packet is forwarded by writing the following information to the Tx-descriptor pointed to by the queue Producer: type (packet/NA), source port#, byte count, buffer address. In addition the relevant bits are set: Multicast, Unicast, Broadcast, Intervention, Unknown, New\_Address, EASE. In the case of Multicast/Broadcast packets, the GT-482xx sets the Multicast Indicator bits in the SDRAM.
4. The CPU Tx High/Low Queue Producer index is incremented. When the Write pointer is not equal to the Read pointer, the GT-482xx checks if the CPU\_Tx\_High/Low\_Desc register can be loaded, if it can be loaded, the descriptor pointed by the Read pointer is loaded into the CPU\_Tx\_High/Low\_Desc register and the Read pointers incremented. The CPU is then interrupted.
5. The CPU reads the CPU\_Tx\_High/Low\_Desc register. The CPU can read/write to the packet buffer directly. When completed the CPU writes an End\_Of\_Packet message to the CPU\_EL\_Free\_Req register.
6. After getting the End\_Of\_Packet message, the GT-482xx clears the Multicast indicator bit of the CPU (if it was Multicast) and if cleared - resets the appropriate bit in the empty list and decrements the rx\_block\_counter of the source port.
7. Instead of stages (5) and (6) above, the CPU can enqueue the packet for transmission (without copying to another buffer, but after modifying it). This mechanism applies only to Unicast packets, or to Multicast-packets that were forwarded only to the CPU. This may be useful when the CPU wants to do intervention on traffic and redirect it (after optionally modifying the packet contents), without copying the entire packet to its memory. The Source port number written to the CPU Enqueue register should be the original source port number and not the CPU (port number 0xE).

## 7.4 Forwarding a Packet from the CPU to the GT-482xx

This interface is used to transmit packets from the CPU to the Ethernet ports. The CPU requests the next empty buffer, writes to the DRAM directly, and then requests queueing of a single packet to a port Tx queue (equivalent to End\_Of\_Packet message in the GalNet Architecture Family).

CPU packet transmission is performed with a simple register interface that allows handling one packet enqueue at a time. Getting empty buffer addresses and enqueueing packets are two independent mechanisms - for example: the CPU can re-enqueue packets that it received from the GT-482xx for transmission back to port(s) using the same buffer (without getting an empty buffer). Also the CPU can "take" several empty-buffers (one at a time) write to them directly, and enqueue them in arbitrary order.

The following registers are used for this process:

- **CPU\_empty\_buffer register.** This register is loaded by the GT-482xx with the next EL address. The register is 32-bits and holds EL address and valid bit. Reading from it signals the GT-482xx that the CPU owns the buffer (resets valid bit). A new EL request will be issued automatically to the EL block (same format as before), unless CPU\_block\_counter register reached the allowed maximum. It is the CPU's responsibility to release the buffer (the GT-482xx does not record which buffers are owned by the CPU) - this is done when the CPU requests enqueueing packet for transmission.
- **Enqueue\_From\_CPU register.** This 64-bit register is used by the CPU to request enqueueing a packet to be transmitted over the GT-482xx ports. The CPU writes the End\_Of\_Packet message here (in the for-

mat that the Descriptor\_Control block is expecting). The register holds the information needed to enqueue the packet (as the CPU has already wrote the buffer contents) and a busy-bit in the second word (bit 31). The busy-bit is set by the GT-482xx whenever the CPU writes to the register and reset by the GT-482xx when it is ready to receive a new enqueue request.

The CPU is allowed to send a packet to a disabled port. The Tx watchdog timer will eventually remove the packet from the Tx queue and release its buffer.

The recommended structure of an interrupt service routine that handles packets that are forwarded to the CPU is as follows:

1. Mask the interrupt for new packets by writing to the Interrupt Mask register the appropriate value.
2. Clear the interrupt cause bit.
3. Read CPU\_Tx\_Desc1
4. Read CPU\_Tx\_Desc2
5. Read the interrupt\_cause register, and see if there is a new packet. If yes then repeat steps (1)-(4), otherwise enable interrupt receiving by clearing the mask register and exit the ISR.

## 7.5 Intervention Mode

Intervention Mode permits software or hardware intervention in the packet routing decision mechanisms. A packet that is determined to require "intervention" will be enqueued only to the CPU (overriding the DA port#, forw<14:0> fields and VL decisions). This is supported for Multicast and for Unicast packets and is performed when the CPU has set the SA <ls> or the DA <ld> intervention fields in the Address Table.

## 7.6 IGMP Packet Support

When Global\_Control[IGMP\_en]=1, the GT-482xx traps IGMP packets that are received on any of its ports using IPv4 and partial IPv6 over Ethernet V2 or 802.3 SNAP encoded (LLC is AA-AA-03-00-00-00) and passes them to the CPU. See Appendix A on page135 for an illustration of the Ethernet packet formats. The CPU can then program the GT-482xx to efficiently switch/filter IP Multicast traffic. The CPU can send the GT-482xx a New\_Address message containing a Multicast-MAC entry that will include the correct forwarding mask for this Multicast address.

The GT-482xx performs the following algorithm on each of its 14 ports when forwarding IGMP packets:

```
If ((Global_control<lgmp_en>==1) AND (DA is Multicast or Broadcast) AND (Source is NOT CPU) AND
(Packet is IGMP)
THEN forward packet ONLY TO CPU, on high priority queue.
Packet can be identified as IGMP by the following algorithm:
(Format is Ethernet or SNAP) AND
(Ethertype is IP (0x0800)) AND (
(IP version == 4 AND Protocol ==IGMP (0x02)) OR
(IP version == 6 AND Payload type == IGMP (0x02)
)
```

## 7.7 CRC Generation

The GT-482xx includes a CRC generator for packets transmitted by the CPU. The CRC generator enhances system performance by implementing the CPU intensive packet FCS calculation in hardware. The CRC generation is not required for packets transferred between ports or devices, since the CRC is already appended to these packets.

CRC generation for CPU generated packets is enabled through the EnCRC bit in the Global Control register. In addition, the CPU must set the GenCRC bit in the EnQueue1 register.

## 7.8 Tx Watchdog Timer

The GT-482xx includes a transmit watchdog timer for each transmit queue. For 100Mbps operation, the default value of the timer is 63msec; the range is between 10.5 to 168msec. For 10Mbps operation, the default value of the timer is 630msec and the range is between 105msec to 1680msec. The timer measures the time between the transmission of two consecutive outgoing packets. When the timer expires, the GT-482xx clears the appropriate used blocks and sends an interrupt to the CPU via Int\*.

The transmit watchdog timer prevents transmission problems on one port from blocking traffic to other ports. In managed systems, the timers also provide a mechanism to notify the CPU of a possible system problem.

The Tx Watchdog Timer can be externally disabled by holding the DisWD\* pin LOW. The default is to leave the Tx Watchdog Timer (DisWD\* HIGH) enabled.

## 8. FAST ETHERNET INTERFACES

The GT-482xx interfaces directly to two MII (Media Independent Interface) ports which are compliant with the IEEE standard (please see 802.3u Fast Ethernet standard for detailed interface information and timing parameters). Each MII port has the following characteristics:

- Capable of supporting both 10 Mbps and 100 Mbps data rates in half or full duplex modes
- Data and delimiters are synchronous to clock references
- Provides independent 4-bit wide transmit and receive paths
- Uses TTL signal levels
- Provides a simple management interface (common to all ports)
- Capable of driving a limited length of shielded cable
- Supports full Auto-Negotiation
- Supports flow control for full duplex and back pressure for half duplex

In addition, for switch expansion, port #13 can work at a clock frequency of up to 60Mhz. In this mode the ForceLinkPass (DAddr[5 and 6] at reset) pin must be pulled down.

The GT-482xx incorporates all of the required digital circuitry to interface to 100BaseTX, 100BaseT4, and 100BaseFX. Two Fast Ethernet ports are integrated in the GT-482xx and only a small amount of external logic is needed to implement standard physical interfaces.

### 8.1 10/100 MII Compatible Interface

The GT-482xx MAC allows it to be connected to a 10Mbps or 100Mbps network. The GT-482xx interfaces to an IEEE 802.3 10/100 Mbps MII compatible PHY device. The data path consists of a separate nibble-wide stream for both transmit and receive activities. The GT-482xx can switch automatically between 10 or 100 Mbps operation depending on the speed of the network. Data transfers are clocked by 25 MHz transmit and receive clocks in 100 Mbps operation, or by 2.5 MHz transmit and receive clocks in 10 Mbps operation. The clock inputs are driven by the PHY, which controls the clock rate based on Auto-Negotiation.

### 8.2 Media Access Control (MAC)

The GT-482xx MAC performs all of the functions of the 802.3 protocol such as frame formatting, frame stripping, collision handling, deferral to link traffic, etc. The GT-482xx ensures that any outgoing packet complies with the 802.3 specification in terms of preamble structure. The GT-482xx transmits 56 preamble bits before Start of Frame Delimiter (SFD). The GT-482xx operates in half-duplex or full-duplex modes. In half-duplex mode, the GT-482xx checks that there is no competitor for the network bus before transmission. In addition to listening for a clear line before transmitting, the GT-482xx handles collisions in a pre-determined way. If two nodes attempt to transmit at the same time, the signals collide and the data on the line is garbled. The GT-482xx listens while it is transmitting, and it can detect a collision. If a collision is detected, the GT-482xx transmits a 'JAM' pattern and then delays its re-transmission for a random time period determined by the backoff algorithm. In full-duplex mode, the GT-482xx transmits unconditionally.

### 8.3 Auto-Negotiation

#### 8.3.1 Disabled Auto-Negotiation

When DData[27:26] are LOW at reset, Auto-Negotiation is disabled for both ports and each port can be selected as half- or full-duplex mode independently. After RESET the port mode is set by the state sampled on the Ddata[25:24] pin. This value can be overridden in each port's Port Control register. The operation speed for each port (10Mbps or 100Mbps) is determined by the frequency of TxClk[x] and RxClk[x] generated by the PHY. When the port is operating at 10Mbps, the PHY generates a 2.5MHz clock for both TxClk and RxClk. When the port is operating at 100Mbps, the PHY generates a 25MHz clock for both TxClk and RxClk.

### 8.3.2 Enabled Auto-Negotiation

When DData[27:26] are HIGH at reset, Auto-Negotiation is enabled for each port. The GT-482xx decodes the duplex mode for each port from the values of the Auto-Negotiation Advertisement register and the Auto-Negotiation Link Partner Ability registers at the end of the Auto-Negotiation process. Once the duplex mode is resolved, the GT-482xx updates the Port Control registers with that duplex mode. The GT-482xx continuously performs the following operations for each port (PHY addresses 1 and 2 alternately), implemented as READ commands issued via the MDC/MDIO interface:

1. Reads the PHY Auto-Negotiation Complete status. When PHY bit 1.5 (Register 1, bit 5) is '0' switches to half-duplex mode and continues to read PHY register bit 1.5. When PHY bit 1.5 is '1', signaling Auto-Negotiation is complete and step two is performed.

#### Notes:

- Steps 2 through 6 are performed once every time the PHY bit 1.5 transitions from '0' to '1'. Once PHY bit 1.5 remains '1' and PHY registers 4 and 5 are read, the GT-482xx continues to read PHY register 1, and monitors PHY bit 1.5. Steps 2 to 6 are performed once, if after Rst\* de-assertion, the PHY bit 1.5 is read as '1', to update the GT-482xx duplex mode.
- PHY bit 1.2 (Link Status) is read and latched during this register read operation, regardless of the Auto-Negotiation status.

2. Reads the Auto-Negotiation Advertisement register, PHY Register 4 and continues to step 3.
3. Reads the Auto-Negotiation Link Partner Ability register, PHY Register 5 and continues to step 4.
4. Resolves the highest common ability of the two link partners as follows (according to the 802.3u Priority Resolution clause 28B.3):

```

if (bit 4.8 AND bit 5.8) == '1' then ability is 100BASE-TX full duplex
else if (bit 4.9 AND bit 5.9) == '1' then ability is 100BASE-T4 half duplex
else if (bit 4.7 AND bit 5.7) == '1' then ability is 100BASE-TX half duplex
else if (bit 4.6 AND bit 5.6) == '1' then ability is 10BASE-T full duplex
else ability is 10BASE-T half duplex;

```

Continues to step 5.

5. Resolves the duplex mode of the two link partners as follows:
 

```

if ((ability == "100BASE-TX full duplex") or (ability == "10BASE-T full duplex")) then
    duplex mode = FULL DUPLEX
else duplex mode = HALF DUPLEX;

```

Note: The value of the duplex mode indication changes only after reading both PHY registers 4 and 5. Continues to step 6.

6. Updates the Port Control register by writing the correct duplex mode bit.

### 8.4 Backoff Algorithm Options

The GT-482xx implements the truncated exponential backoff algorithm defined by the 802.3 standard. Aggressiveness of the backoff algorithm used by all of the ports is controlled by the Limit4 pin or bit in the Global control register. Limit4 controls the number of consecutive packet collisions that occur before the collision counter is reset. When Limit4 is LOW, the GT-482xx resets the collision counter after 16 consecutive retransmit trials, restarts the backoff algorithm, and continues to try to retransmit the frame. The retransmission is performed from the data stored in the DRAM. In the case of a successful transmission, the GT-482xx is ready to transmit any other frames queued in its transmit FIFO within the minimum IPG of the link.

When Limit4 is HIGH, the GT-482xx resets its collision counter and restarts the backoff algorithm after 4 consecutive transmit trials. As a result the GT-482xx is more aggressive in acquiring the media following a collision. This results in better overall switch throughput (less packet loss) in standardized tests. Limit4 can be toggled during switch operation.

## 8.5 Data Blinder

The data blinder field (DataBlind in the Serial Parameters register) sets the period of time during which the port does not look at the wire to decide to transmit (inhibit time.) The default value is 32 bit times.

## 8.6 Inter-Packet Gap (IPG)

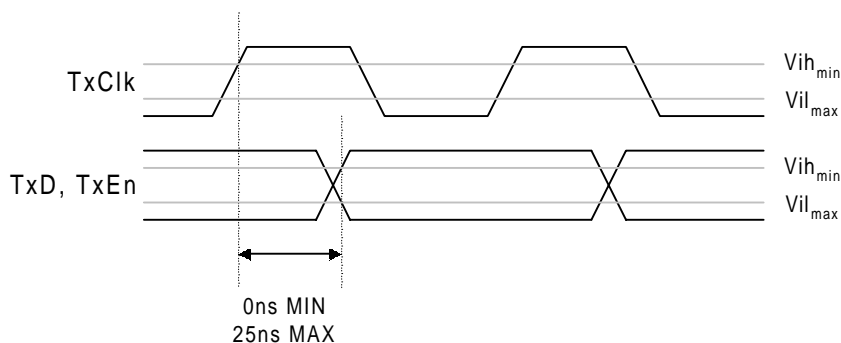
IPG is the idle time between any two successive packets from the same port. The default (from the standard) is 9.6 $\mu$ s for 10Mbps Ethernet and 960nsec for 100-Mbps Fast Ethernet. The IPG can be programmed to be smaller or larger than the Ethernet standards. Making the IPG smaller can improve test scores at the cost of Ethernet compatibility (a technique used by many vendors during head-to-head magazine tests.) This mode of operation is not recommended due to violation of IEEE standards.

IPG is programmable in the Serial Parameters register.

## 8.7 10/100 Mbps MII Transmission

When the GT-482xx has a frame ready for transmission, it samples the link activity. If the CrS signal is inactive (no activity on the link), and the Inter-packet gap (IPG) counter has expired, frame transmission begins. The data is transmitted via pins TxD[3:0] of the transmitting port, clocked on the rising edge of TxClk. The signal TxEn is asserted at the same time. In the case of collision, the PHY asserts the CoL signal on the GT-482xx which then stops transmitting the frame and appends a JAM sequence onto the link. After the end of a collided transmission, the GT-482xx backs off and attempts to retransmit once the backoff counter expires. Per IEEE 802.3 specification, the clock to output delay must be a minimum of 0ns and a maximum of 25ns as shown in Figure 4.

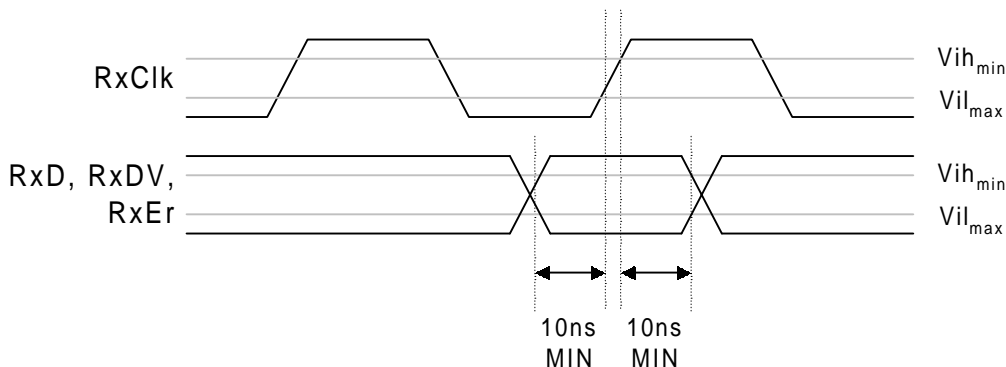
**Figure 4: MII Transmit Signal Timing**



## 8.8 10/100 Mbps MII Reception

Frame reception starts with the assertion of CrS (while the GT-482xx is not transmitting) by the PHY. Once RxDV is asserted, the GT-482xx begins sampling incoming data on pins RxDV[3:0] on the rising edge of RxClk. Reception ends when the RxDV is deasserted by the PHY. The last nibble sampled by the GT-482xx is the nibble present on RxD[3:0] on the last RxClk rising edge in which RxDV is still asserted. During reception, RxDV is asserted. If, while RxDV is asserted, the GT-482xx detects the assertion of RxEr, it designates this packet as corrupted. While no reception is taking place, RxDV should remain deasserted. The input setup time should be a minimum of 10ns and the input hold time must be a minimum of 10ns and shown in Figure 5.

**Figure 5: MMII Receive Signal Timing**



## 8.9 10/100 Mbps Full-duplex Operation

When operating in full-duplex mode the GT-482xx can transmit and receive frames simultaneously. In full-duplex mode, the CrS signal is associated with received frames only and has no effect on transmitted frames. The Col signal is ignored by the GT-482xx while in full-duplex mode. Transmission starts when TxEn goes active. Transmission starts regardless of the state of CrS. Reception starts when the CrS and RxDV signals are asserted indicating traffic on the receive port of the PHY.

## 8.10 Illegal Frames

The GT-482xx discards all illegal frames and increments the appropriate error MIB counters. For example, runts (less than 64 bytes), oversize (greater than 1518/1536 bytes), and bad FCS (bad checksum).

## 8.11 Partition Mode

A port enters Partition Mode when more than 32 consecutive collisions are seen on a 10Mbps port, or more than 60 consecutive collisions are seen on a 100Mbps port. In Partition Mode the port continues to transmit but it will not receive. The corresponding bit in the Status register is set when a port is partitioned. A port is returned to normal operation mode when a good packet is seen on the wire.

### 8.11.1 Enabling Partition Mode

Partition is enabled (for all ports) by setting the enable bit in the GT-482xx Control Register. The default value is Partition disabled for all ports. You must have a CPU in the system to enable partition mode; there is no pin strap-ping option.

### 8.11.2 Entering Partition State

When Partition is enabled, a port enters Partition state for any of the following cases:

- The port detects a collision on every one of 32 consecutive retransmit attempts of the same packet on a 10Mbps port
- The port detects a collision on every one of 60 consecutive retransmit attempts of the same packet on a 100Mbps port
- The port detects a single collision which occurs for more than 512 bit times.

While in Partition state:

- If the interrupt is not masked, the GT-482xx issues an interrupt to the CPU upon entering Partition state, and sets the partition bit of that port in the Status register.



- The port continues to transmit its pending packet, regardless of the collision detection, and does not follow the usual Backoff Algorithm. Additional packets pending for transmission, are transmitted, while ignoring the internal collision indication. This frees the port's transmit buffers which would otherwise be filled up at the expense of the other ports' buffers. The assumption is that Partition state represents a system failure situation (bad connector/cable/station), thus dropping packets is a small price to pay vs. the cost of halting the switch due to full buffers.
- The Partition Indication is available via the LED interface (both the status led - blinking twice, and a dedicated led - on constantly).

### 8.11.3 Exiting from Partition State

The port exits from Partition state at the end of a successful packet transmission or packet reception. A successful packet transmission is declared if no collisions were detected during packet transmit or receive. If the interrupt is not masked, the GT-482xx issues an interrupt to the CPU upon exiting from Partition state, and clears the partition bit of that port in the Status register.

### 8.12 Back Pressure

The GT-482xx implements a back-pressure algorithm when it is operating only in **half-duplex** mode. Back pressure is enabled by setting the BackPressureEn bit in the Port Control register (also sampled at reset from pin). The GT-482xx enters into back-pressure mode when the buffers allocated to a port exceed the RxThreshold value (The user should not disable the Per-port Rx Buffer threshold when using Backpressure). In back-pressure mode, the GT-482xx transmits a JAM pattern for a programmable value of time (JAM\_LENGTH). The IPG between two consecutive JAM patterns (or between the last transmit and the first JAM) is also a programmable value (JAM\_IPG). When a packet is transmitted during back pressure it is transmitted with a gap of IPG\_JAM\_TO\_DATA after the transmission of the JAM pattern.

### 8.13 Flow Control

The GT-482xx implements the IEEE 802.3x flow control standard on any **full-duplex** port. This mode is activated by setting the FlowContEn bit in the Port Control register (sampled at reset from pins). When the GT-482xx receives a PAUSE packet it avoids transmitting a new packet from this port for the period of time specified in the received PAUSE packet.

A received packet is recognized as flow control if it was received without errors and is either one of the following:

- DA = 01-80-C2-00-00-01 and type=88-08 and MAC\_Control\_Opcode=01
- DA = (The port address) and type=88-08 and MAC\_Control\_Opcode=01. The 48bit port address is in the registers Source\_Address\_Low, Source\_Address\_High and Port\_Control[Sa<3:0>]. This address is used as source address for PAUSE packets that the GT-482xx generates (to DA=01-80-C2-00-00-01)

The GT-482xx sends a PAUSE packet on a port, when the number of buffers allocated to this port exceed a programmable threshold (as specified by FlowContThrHigh field in Rx Threshold Register. The user should not disable the Per-port Rx Buffer threshold when using Flow-Control. The PAUSE packet will be sent with the parameter equal to 0xFFFF. The transmitting device is expected to stop transmitting to the specific GT-482xx port for sufficient time to enable the GT-482xx to release its buffers. When the number of buffers drops below the FlowContThrLow number, the GT-482xx sends a PAUSE packet with its parameter equal to 0x0 to indicate to the transmitter that the GT-482xx is ready to receive packets on that link.

To prevent deadlocks on the network (due to loss of PAUSE packets), the GT-482xx will periodically send PAUSE packets every 50mSec (for the 10Mbps ports, every 500mSec). PAUSE packets with parameters equal to 0xFFFF are sent every 50mSec when the number of buffers allocated to this port exceed a programmable threshold (FlowContThrHigh). PAUSE packets with parameters equal to 0x0 are sent every 50mSec when the number of buffers allocated to this port is less than or equal to FlowContThrLow number.

Each port holds its own Source Address for flow control packets. The Source Address consists of 44 common bits for all ports and four dedicated bits per port. The common bits (47:4) are held in the Source Address High and Low registers. The four least significant bits are held in the Port Control registers.

## 8.14 802.1q VLAN Tagging Support

The GT-482xx has the ability to receive and transmit Ethernet frames up to 1536 bytes in length, thereby accommodating the IEEE 802.1q standard VLAN tagging bytes. This feature is enabled/disabled by bit 13 in the Port Control register (also sampled from DAddr[8] at RESET.) The default is to disable this feature. Bytes longer than 1536 are discarded as over-size frames.

## 8.15 MII Management Interface (SMI)

The GT-482xx MAC includes an MII Management Interface (SMI) for MII compliant PHY devices. This enables the passing of control and status parameters between the GT-482xx and the PHY (parameters specified by the CPU) by one serial pin (MDIO) and a clocking pin (MDC), and reduces the number of control pins required for PHY mode control. Typically, the GT-482xx continuously queries the PHY devices for their link status, without CPU intervention. The predefined PHY addresses for the link query are 1 and 2 (out of possible 32 addresses). This protocol complies with the National DP83840 PHY device as well as other available PHYs.

A CPU connected to the GT-482xx has access to all of the PHY addresses/registers, by writing and reading to/from a dedicated set of the GT-482xx SMI control registers. The SMI allows the CPU to have direct control over an MII-compatible PHY device via the GT-482xx SMI control register. This allows the driver software to place the PHY in specific modes such as Full Duplex, Loopback, Power Down, 10/100 speed selection as well as control of the PHY device's Auto-Negotiation function, if it exists. The CPU writes commands to the GT-482xx SMI register and the GT-482xx reads or writes control/status parameters to the PHY device via a serial, bi-directional data pin called MDIO. These serial data transfers are clocked by the GT-482xx MDC clock output.

The delay time between two consecutive SMI write transactions is at least (4x64=256) MDC clock cycles.

### 8.15.1 SMI Cycles

The SMI protocol consists of a bit stream driven or sampled by the GT-482xx on each rising edge of the MDC clock. The bit stream format of the SMI frame is described in Table 10.

**Table 10: SMI Bit Stream Format**

	PRE	ST	OP	PhyAd	RegAd	TA	Data	IDLE
READ	1...1	01	10	AAAAA	RRRRR	Z0	D..D(16)	Z
WRITE	1...1	01	01	AAAAA	RRRRR	10	D..D(16)	Z

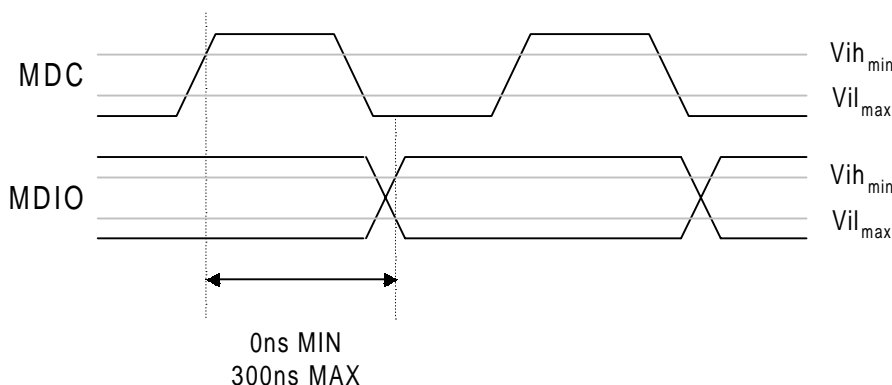
- PRE (Preamble). At the beginning of each transaction, the GT-482xx sends a sequence of 32 contiguous logic one bits on MDIO with 32 corresponding cycles on MDC to provide the PHY with a pattern that it can use to establish synchronization.
- ST (Start of Frame). A Start of Frame pattern of 01.
- OP (Operation Code). 10 - Read; 01 - Write
- PhyAd (PHY Address). A 5 bit address of the PHY device (32 possible addresses). The first PHY address bit transmitted by the GT-482xx is the MSB of the address.
- RegAd (Register Address). A 5 bit address of the PHY register (32 possible registers in each PHY). The first register address bit transmitted by the GT-482xx is the MSB of the address. The GT-482xx always queries the PHY device for status of the link by reading register 1, bit 2.
- TA (Turn Around). The turnaround time is a 2 bit time spacing between the Register Address field and the Data field of the SMI frame to avoid contention during a read transaction. During a Read transaction the PHY should not drive MDIO in the first bit time and drive '0' in the second bit time. During a write transaction, the GT-482xx drives a '10' pattern to fill the TA time.

- Data (Data). The data field is 16 bits long. The PHY drives the data field during Read transactions. The GT-482xx drives the data field during write transactions. The first data bit transmitted and received will be bit 15 of the PHY register being addressed.
- IDLE (Idle). The IDLE condition on MDIO is a high impedance state. The MDIO driver is disabled and the PHY should pull-up the MDIO line to a logic one.

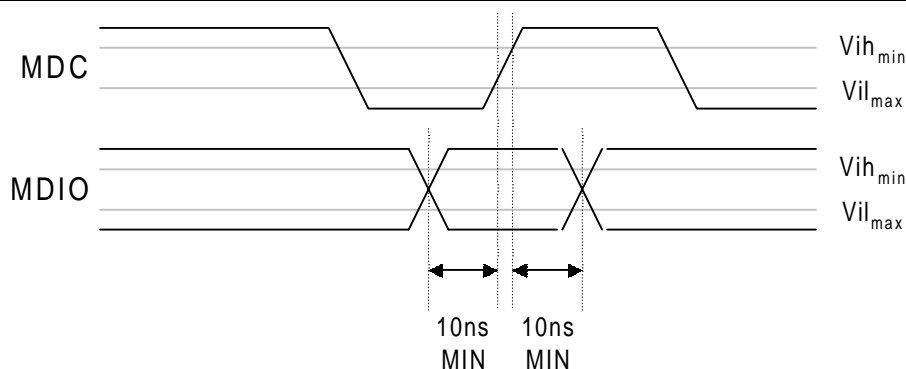
### 8.15.2 SMI Timing Requirements

When the MDIO signal is driven by the PHY, it is sampled by the GT-482xx synchronously with respect to the rising edge of MDC. Per IEEE 802.3 specification, the MDC to output delay must be a minimum of 0ns and a maximum of 300ns as shown in Figure 6. In addition, when the MDIO signal is driven by the GT-482xx, the GT-482xx provides a minimum of 10ns of setup time and minimum of 10ns of hold time as shown in Figure 7.

**Figure 6: MDIO Output Delay**



**Figure 7: Required MDIO Setup and Hold Time**



### 8.16 Link Detection and Link Detection Bypass (ForceLinkPass)

Typically, the GT-482xx will continuously query the PHY devices for its link status, without CPU intervention. The predefined PHY addresses for the link query are 1 and 2 (out of possible 32 addresses). The GT-482xx alternately

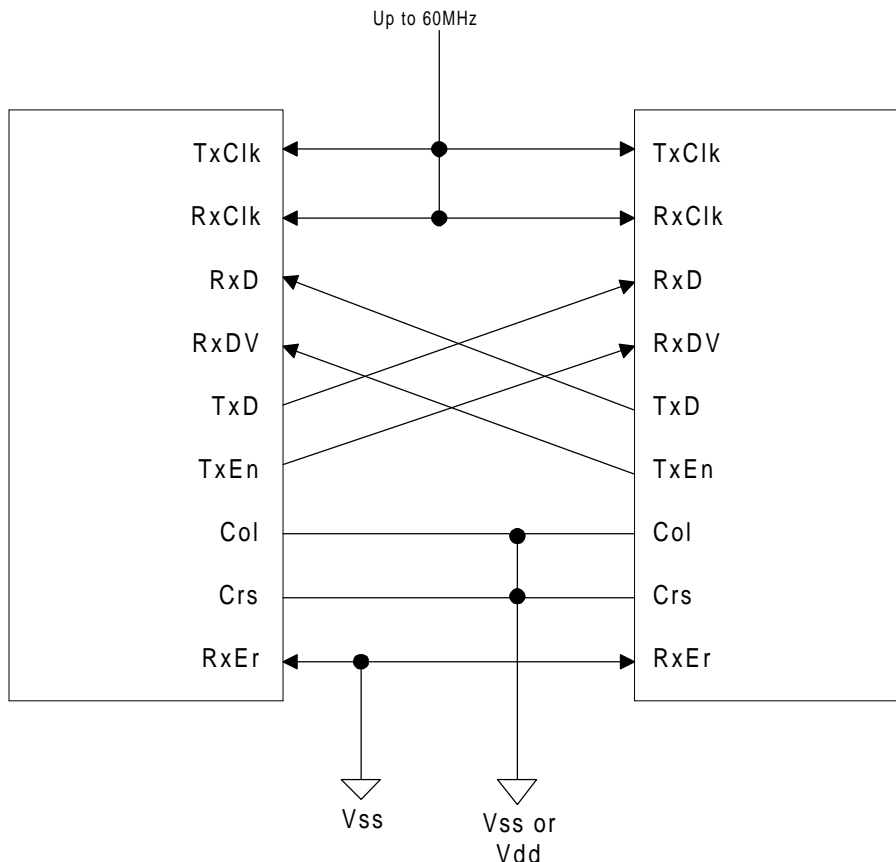
reads register 1 from PHY1 and PHY2 and updates the internal link bits according to the value of bit 2 of register 1. In the case of “link is down” (bit 2 is ‘0’), that port will enter link test fail state. In this state, all of the port’s logic is reset. The port will exit from link test fail state only when the “link is up” (bit 2 of register 1) is read from the port’s PHY as ‘1’.

The GT-482xx provides an option to disable the link detection mechanism by forcing the link state of both ports to the link test pass state. **This mode is used also to operate the expansion port.** This is done with the DAddr[6:5] pins which are sampled at RESET. When DAddr[6:5] are LOW on RESET, the link status of both ports remains in the “link is up” state regardless of the PHY’s link bit value. When DAddr[6:5] are HIGH on RESET, the link status of the ports is read through the SMI from the PHY devices (register 1, bit 2).

### 8.17 Using the MII Interfaces to Connect Two (or More) Galaxy Devices

Either MII interface can be used to cascade two GT-482xx devices to create a higher port density switch. In such an application, the MII interface can be clocked up to 60MHz providing a non-blocking interconnect for configurations of up to 24 Ethernet and two Fast Ethernet ports. More than two Galaxy devices may be cascaded, however, such configurations will exhibit blocking in highly-loaded environments/test setups.

**Figure 8: Expansion MII Wiring Diagram**



## 9. ETHERNET (10Mbps) INTERFACES

The GT-482xx incorporates all the required digital circuitry to interface to 10Base-T and 10Base-FL. One of the 12 ports can work in AU1 mode.

### 9.1 Media Access Control (MAC)

The GT-482xx operates in half-duplex or full-duplex modes. In half-duplex mode, the GT-482xx checks that there is no competitor for the network bus before transmission. In addition to listening for a clear line before transmitting, the GT-482xx handles collisions in a pre-determined way. If two nodes attempt to transmit at the same time, the signals collide and the data on the line is garbled. The GT-482xx listens while it is transmitting, and can detect a collision. If a collision is detected, the GT-482xx transmits a 'JAM' pattern and then delays its re-transmission for a random time period determined by the backoff algorithm. In full-duplex mode, the GT-482xx transmits unconditionally.

### 9.2 Illegal Frames

the GT-482xx discards all illegal frames and increments the appropriate error MIB counters. For example, runs (less than 64 bytes), oversize (greater than 1536 bytes), and bad FCS (bad checksum.)

### 9.3 Duplex Mode Selection

Each port can be independently selected to function in half- or full-duplex mode. Following reset the port mode is set by the state sampled on the TxDE[x] pin. This value can be overridden in each port's Port Control register.

### 9.4 Backoff Algorithm Options

The backoff algorithm used by all ports is controlled by the Limit4 pin or bit in the Global Control register. Limit4 controls the number of consecutive packet collisions that occur before the collision counter is reset. As a result of asserting the pin or bit results, the GT-482xx is more aggressive in acquiring the media following a collision. This results in improved overall switch throughput (less packet loss) in standardized tests. Limit4 can be toggled during switch operation.

### 9.5 Manchester Encoder/Decoder

The Manchester Encoder receives clocked data from the transmit engine and uses an internal 20MHz clock (80MHz divided by 4) to provide the Manchester-encoded data to the Physical interface. The Manchester Decoder uses the 80MHz clock to recover the receive clock and to sample the incoming data.

### 9.6 Link Integrity

The GT-482xx implements the Link Integrity test as specified in the IEEE 802.3 10Base-T and 10Base-FL supplements.

### 9.7 Data Blinder

The data blinder field (DataBlind in the Serial Parameters register) sets the period of time. The port does not look at the wire to decide to transmit (inhibit time.) The default value of the Data Blinder is 32uS. The programmed value in the register is the difference between 96uS and the required Data Blinder length.

### 9.8 Inter-Packet Gap (IPG)

IPG is the idle time between any two successive packets transmitted from the same port. The default (from the standard) is 9.6uS for 10Mbps Ethernet and 960nsec for 100-Mbps Fast Ethernet. The IPG can be programmed to be smaller or larger than the Ethernet standards. Making the IPG smaller can improve test scores at the cost of Ethernet compatibility (a technique used by many vendors during head-to-head magazine tests). However, this mode of operation is not recommended due to violation of IEEE standards.

IPG is programmable in the Serial Parameters register.

## 9.9 Partition Mode

A port enters Partition Mode when more than 32 consecutive collisions are seen on the port. When in Partition Mode, the port continues to transmit but does not receive. The corresponding bit in the Status register is set when a port is partitioned. A port is returned to normal operation mode when a good packet is seen on the wire.

### 9.9.1 Enabling Partition Mode

Partition is enabled (for all ports) by setting the enable bit in the GT-482xx Control Register. The default value is Partition disabled for all ports. You must have a CPU in the system to enable partition mode; there is no pin strap-ping option.

### 9.9.2 Entering Partition State

When Partition is enabled, a port will enter Partition state when either of the following two situations occur:

- The port detects a collision on every one of 32 consecutive retransmit attempts of the same packet.
- The port detects a single collision which occurs for more than 2048 bit times (i.e. most likely to occur in AUI mode).

While in Partition state:

- If the interrupt is not masked, the GT-482xx will issue an interrupt to the CPU upon entering Partition state, and will set the partition bit of that port in the Status register.
- The port will continue to transmit its pending packet, regardless of the collision detection, and will not follow the usual Backoff Algorithm. A more aggressive Backoff Algorithm is in progress which is similar to the Backoff Algorithm when Limit4 is activated. Additional packets pending for transmission are transmitted, while ignoring the internal collision indication. This frees the port's transmit buffers which would otherwise be filled up at the expense of the other ports' buffers. The assumption is that Partition is a system failure situation (bad connector/cable/station), thus losing the transmitted packets is a small price to pay versus the cost of halting the switch by filling up all of its buffers.
- The Partition Indication is available via the LED interface (both the status led - blinking twice, and a dedicated led - on constantly).

### 9.9.3 Exiting from Partition State

The port will exit from Partition state at the end of a successful packet transmission. A successful packet transmission will be declared, provided no collisions were detected on the first 512 bits of the transmission. If the interrupt is not masked, the GT-482xx issues an interrupt to the CPU upon exiting from Partition state, and clears the partition bit of that port in the Status register.

## 9.10 802.1q VLAN Tagging Support

The GT-482xx has the ability to receive and transmit Ethernet frames up to 1536 bytes in length, thereby accommodating the IEEE 802.1q standard VLAN tagging bytes (four extra bytes or more.) This feature is enabled/disabled by bit 13 in the Port Control register. The default is to disable this feature. Bytes longer than 1536 are discarded as over-size frames.

## 9.11 Back Pressure

Back-pressure support in the 10Mbps ports is identical to the support for the 100Mbps ports, described previously.

## 9.12 Flow Control

Flow control support in the 10Mbps ports is identical to the support for the 100Mbps ports, described previously.

### 9.13 Serial Modes

Each Ethernet port can operate in one of three serial modes: 10BaseT, 10BaseFL, or AUI. The serial mode is set after RESET by the state of DData[23:0]. These values can be overwritten in the Port Control register.

### 9.14 Physical Interface Circuitry

Since the digital portion of the Ethernet PHY is integrated in the GT-482xx, only a small amount of external logic is needed to implement the standard physical interfaces. The GT-482xx interfaces directly with the AMD's QuIET® chip (by setting the GT-482xx to work in pseudo-AUI mode) or with the Tamarack TC3001 (setting the GT-482xx to work in 10Base-T mode).

Information about the Tamarack TC3001 can be found at the following URL: [www.tmi.com.tw](http://www.tmi.com.tw). Information about the AMD QuIET chip can be found at the following URL: [www.amd.com](http://www.amd.com). Schematics for both implementations are available upon request.

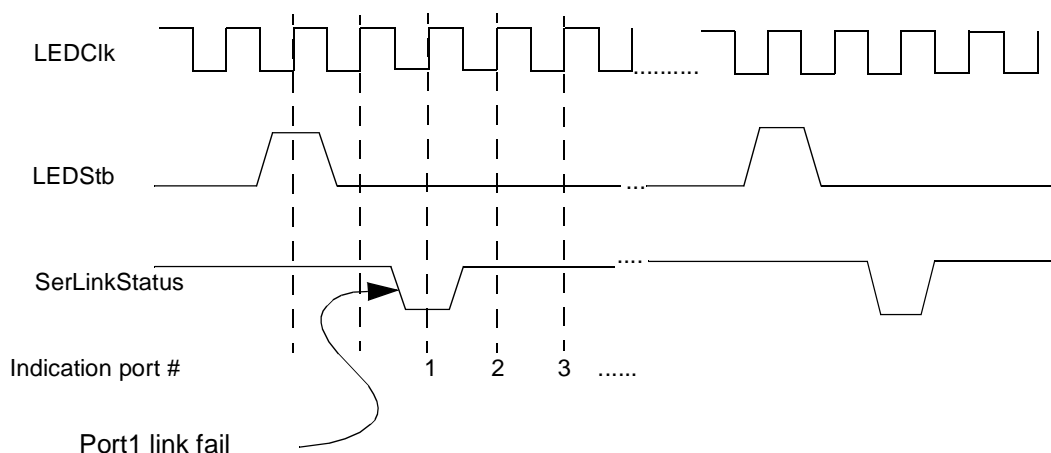
**NOTE:** Galileo Technology may support other physical interfaces in the future. Please contact your local Galileo Technology FAE for further updates.

### 9.15 Serial Link Status Indication

This link is used to serially shift the Link Status of ports 1 to 11 from the external PHY. LEDClk and LEDStb are used to clock and strobe the data (assertion of LEDStb indicates the beginning of the link data stream). The serial link status indication of port 0 is provided separately at LinkStatus[0] pin.

Figure 9 illustrates a port 1 link fail.

**Figure 9: Example of Serial Link Status Indicator of Port 1 Link Fail**



## 10. ENABLING/DISABLING PORTS

The GT-482xx ports can be enabled and disabled depending by a combination of:

- An external hardware pin, EnDev\* (LOW - device enabled, HIGH - device disabled)
- EnableDevice, bit 20 of the Global Control Register ('0' - device status based on EnDev\*, '1' - device enabled)
- PortEn, bit 0 of each Port Control register, ('0' - port disabled, '1' - port enabled)

When a port is disabled, no packets are received or transmitted from the serial ports or the CPU bus. Even though ports are disabled, the CPU can read from and write to the GT-482xx's registers. See Table 11 for the enabling or disabling of the GT-482xx ports.

**Table 11: Enabling/Disabling Ports of the GT-482xx**

EnDev* Pin	EnableDevice Bit	PortEn Bit	Port Status
LOW	0	0	Disabled
LOW	0	1	Enabled
LOW	1	0	Disabled
LOW	1	1	Enabled
HIGH	0	0	Disabled
HIGH	0	1	Disabled
HIGH	1	0	Disabled
HIGH	1	1	Enabled



## 11. NETWORK MANAGEMENT SUPPORT

The GT-482xx supports the following management features:

- Repeater MIB counters
- Port monitoring (sniffer) mode
- Spanning Tree BPDU detection
- Broadcast Storm filtering
- HP-EASE packet sampling technology

The packet sampling (HP-EASE) functions are described in Section 12.

### 11.1 Repeater MIB Counters

The GT-482xx incorporates a full set of Repeater MIB counters for each Ethernet port. Counters are accessed by the management CPU through the CPU interface.

The Repeater MIB counters include the following data:

- Bytes Received
- Bytes Sent
- Frames Received
- Frames Sent
- Total Bytes Received (Good and Bad)
- Total Frames Received (Good and Bad)
- Multicast Frames Received
- Broadcast Frames Received
- CRC + Alignment Error
- Oversize Frames
- Fragments
- Jabber Frames
- Collision
- Late Collision
- Frames with length of 64 Bytes
- Frames with length of between 65-127 Bytes
- Frames with length of between 128-255 Bytes
- Frames with length of between 256-511 Bytes
- Frames with length of between 512-1023 Bytes
- Frames with length of between 1024-1518/1536 Bytes
- MAC Receive Error (received packets with RxEr asserted) - only for 100Mbps ports
- Dropped Frames

See the Table 63, "Port MIB Counters," on page 109 for detailed information on the repeater MIB counter registers.

### 11.2 Monitoring (Sniffer) Mode

The CPU can program the GT-482xx to work in Monitoring mode for one of the Ethernet ports. In Monitoring mode, the GT-482xx sends all received (including local traffic) to the CPU, or to another port in the same GT-482xx device which is assigned to be the Sniffer. The packets that are forwarded to the Sniffer are not necessarily in a linear time order. Monitored packets are also forwarded as normal to their output port.

Sniffer packets (packets that are being sent to target sniffer) are queued to the same priority queue as the original packet.

Monitoring Mode is enabled by setting the Sniffer Register fields. Only one port of a single GT-482xx device can work in monitoring mode at a time.

When Monitoring is implemented in a multi-device configuration and connected together through the expansion port, the user cannot set the source sniffer and the target sniffer in the different devices. This requires using the expansion port as target sniffer in the source device, and as source sniffer in the target device and will cause unrelated traffic between the devices to be forwarded to the target-sniffer port on the other device.

For such configuration the recommended operation should be one of the following:

- a) Designate the CPU as the target sniffer,
- or
- b) Restrict the source and the target sniffer port to reside in the same device

### 11.3 Spanning Tree (BPDU) Support

The GT-482xx provides hardware assistance for the Bridge Spanning Tree Algorithm. The Spanning Tree algorithm itself is performed by a management CPU.

The GT-482xx includes a SpanEn bit in the Global Control register and additional SpanEn bits in each of the 14 Port Control registers. Table 12 summarizes the hardware assistance for the Spanning Tree algorithm.

**Table 12: Spanning Tree Enable Bit Definition**

SpanEn (Global)	SpanEn (Port)	Logic State	Remarks
0	x	Port Enable	No Spanning Tree. Treat BPDU as regular Multicast
1	1	Blocking, Listening, Learning	Transfer BPDUs to CPU. All receive/transmit packets should be rejected. Accept BPDU messages from the CPU. No address learning
1	0	Forward	Transfer BPDU to the CPU. Accept all packets. Address learning

**Note:** The GT-482xx does not learn MAC addresses during the Spanning Tree 'Learning' stage (it is 'learning' the bridge topology while in this mode.) The GT-482xx only learns MAC addresses in the Forward mode.

### 11.4 Broadcast Storm Filtering

Excessive Broadcast packets (Broadcast "storms") can be filtered in a managed switch by setting the FilBroad bit in the Port Control register. In this mode, the inbound Broadcast packets will be discarded. Broadcast packets can be re-enabled once the loops causing the Broadcast storm are eliminated via the spanning tree algorithm.

## 12. PACKET SAMPLING TECHNOLOGY (HP-EASE)

The GT-482xx supports a packet sampling technology developed by Hewlett-Packard called the Embedded Advanced Sampling Environment (HP EASE). This packet sampling technology can be used to implement many of the management functions that RMON provides, but at a greatly reduced implementation cost. With the GT-482xx device, the switch OEM has the freedom to implement "true" RMON, sampled RMON, EASE, or any of the above.

**NOTE:** Customers wishing to use the packet sampling technology to implement a true HP-EASE agent should contact Galileo directly for additional information and MIB specifications.

### 12.1 Packet Sampling Overview

HP EASE sampling requires a counter for each network segment. This counter indicates the number of packets to be skipped before a sample is taken. When the counter reaches zero, the next packet on the network segment is captured by the network device. Software then truncates the sampled packet, to some small fixed length, and appends a snapshot of specific MIB counters for that segment. The counter snapshot does not have to be taken simultaneously with the sample. Software may introduce a delay of some milliseconds after the packet is sampled by hardware, however minimizing this delay makes EASE more accurate. The newly created datagram is sent off to the network management station as an SNMP trap. The network management station records the sample and counters in a database, and uses the information to obtain traffic load estimates, top talker matrices, high-level protocol flows, and other useful sets of information. After the sample has been taken, the CPU loads the count-down counter with the next skip count to capture the next sampled packet. The skip count is a random value loaded by software.

EASE software in the network device must keep track of the last receive error sources and the associated error conditions. The network device keeps track of errors associated with received packets and informs the CPU of the source address (SA) of these error packets.

### 12.2 EASE Functionality on the GT-482xx

Support for EASE sampling is directly integrated in the GT-482xx chip, but requires the presence of a CPU in order to function, for enabling the EASE support as well as the sample packet processing. Each GT-482xx device supports 14 network segments (one per port) as well as the CPU system bus. Sampling will occur only on the network segments, and sampled packets will be sent to the CPU. Sampling is not performed on the CPU bus. It may, however, be performed on packets received from the CPU, but only as a function of the counters for the destination ports (i.e. packets entering the GT-482xx via the CPU bus and being transmitted through one or more ports). There is no counter for the CPU interface itself. Only good packets of valid length are sampled. All other packets are not sampled and do not affect the skip count. All counters and registers implemented in the GT-482xx chip in order to support EASE functionality, may be accessed by the CPU.

### 12.3 Ease Register

A register is defined for each external port supported on the GT-482xx device. This register is used by the CPU to load the internal count down counter, described above, with a random skip count. The count-down counter is 20 bits in length and is used to actually determine when a sample is to be made. The GT-482xx implements a shadow register for each of the Ease registers. The shadow register address is the same address as the Ease\_Register address. After a value has been written to the Ease register it is transferred to an internal 1 word deep FIFO (the shadow register) or directly to the actual count-down counter if that counter is currently idle and empty. If the value can not be transferred to the count-down counter, the value will be held in the Ease\_Register shadow register until space becomes available (i.e. a sample has been taken). If the Ease\_Register shadow register was written and the CPU does attempt to write a new value, the new value will silently replace the existing value. If the Ease\_Register is empty at the time a new value needs to be loaded into the internal counter or the shadow register, the GT-482xx will load the previous value. EASE is effectively disabled on that port.

## 12.4 EASE Interrupts

A status bit indicating the full/empty status of the Ease\_Register for each external port supported on the GT-482xx, is maintained as part of the Interrupt Cause register. When a value is moved from the Ease\_Register into an internal counter or shadow register, a bit is reset in the Interrupt Cause register indicating that the Ease\_Register is now empty. Setting this bit should also generate a processor interrupt. The Interrupt Cause register may be read to determine the state of the Ease\_Registers, and may be written to clear the interrupt condition described above. It is possible for the CPU to mask the interrupt condition as well as clear the interrupt condition. The GT-482xx implements a mask bit in the Interrupt Mask register for each EASE status bit in the Interrupt Cause register. Masking and clearing the interrupts are executed in a way that is consistent with the other interrupts supported by the GT-482xx.

## 12.5 Sampled Packet Indication

Sampled packets are sent to the CPU. The sample indication bits in the descriptor specify which ports on the particular GT-482xx this sample is associated with. It is possible for a single sample to be associated with more than one port at a time. For example, a Broadcast packet flooded to all ports may be sampled on several ports if each of their skip counters had previously been decremented to zero.

Each GT-482xx device operates independently, therefore the CPU can receive the same sample from different GT-482xx devices. For example, a Broadcast packet flooded to all ports in the system may be sampled by several GT-482xx's at the same time. Each sample results in a separate copy of the packet being sent to the CPU. Packets which would normally be received by the CPU can also be sampled. In this case, only a single copy of the packet can be sent to the CPU. The CPU is responsible for determining if a sampled packet should be accepted as a normal receive packet. In the case where a normally received packet is also a sample from multiple the GT-482xx devices (e.g. a Broadcast packet). The GT-482xx must provide an indication to avoid the CPU from processing duplicate packets.

## 12.6 Error Source Indications

EASE software in the network device must keep track of the last receive error sources and the associated error conditions. The GT-482xx informs the CPU of error source conditions by sending a New\_Address message.

Two types of errors are defined for this procedure: FCS error and frames too long. When the GT-482xx receives a packet with any of the above conditions, it will send an Error\_Source message to the CPU. The Error\_Source message will contain the 48-bit source address of the error packet, the source port number and an indication of the error type.

## 12.7 Enabling/Disabling EASE Functionality

An explicit HP EASE enable/disable bit is provided in the Global Control register for the GT-482xx device. When HP EASE is disabled using this bit, no EASE samples nor Error\_Source messages are sent to the CPU. HP EASE packet sampling can be disabled on a port anytime the internal counter can not be reloaded with a new skip count because the CPU has not provided any new values via the Ease\_Register. Interrupt conditions generated by an empty Ease\_Register can be masked by appropriate bits in the Ease\_Full\_Mask and/or Interrupt Cause registers.

The following algorithm enables EASE for the first time:

1. Enable EASE in the Global Control register.
2. Enable EASE in the Port Control register per port (10Mbit ports - bit [8], 100Mbit ports - bits [3:2] of PCR12).
3. Write a skip value to the EASE register of each port.

The following algorithm disables EASE:

1. Disable EASE per port in the PCR.
2. Disable EASE in the Global Control register.

**Note:** If some of the ports have their EASE enable bit enabled, the GCR bit must be enabled. If the CPU failed loading a new value to the EASE register skip counter, the old value will be used and the EASE mechanism will continue sampling.

## 13. LED SUPPORT

The serial LED interface in the GT-482xx is similar to the 3-pin LED interface of the GT-48001A device which requires a PAL to interpret the LED bit stream. Galileo Technology provides reference designs and example PAL equations in the LED interface application note available from our website.

### 13.1 LED Indications Interface Description

Table 13 on page 54 details the accessible data on the LED Indications Serial Interface for each of the GT-482xx ports.

**Table 13: LED Signals Available**

Data Description	Symbolic Signal Name	Type
Primary Port Status LED	primary_port_status	n/a
Transmit data in progress	transmit	dynamic
Receive data in progress	receive	dynamic
Transmit/Receive data in progress	activity	dynamic
Collision active	collision	dynamic
Forwarding of unknown packets enabled	unknown_enable	static
The port is configured as Sniffer	port_is_sniffer	static
Full/Half duplex	full_duplex	static
Receive Buffer Full	rx_buffer_full	dynamic

### 13.2 Detailed LED Signal Description

The LED signals and their activation conditions are described in the following sections.

#### 13.2.1 Primary Port Status LED

The Primary Port Status LED indicates the port status in two operation modes, selectable via the LEDMode input (LedStb pin sampled at reset).

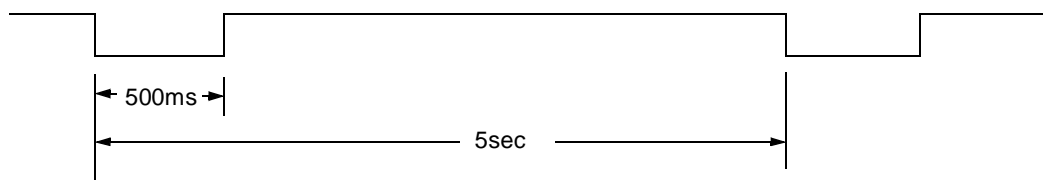
##### 13.2.1.1 Primary Port Status LED in Mode 0: (LEDMode input is LOW)

In this mode, the Port Status LED provides the following information:

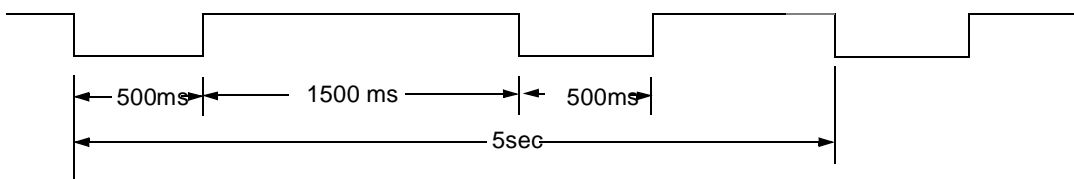
- if Port is disabled
  - Port Status LED is OFF;
- else if Link Integrity test failed
  - Port Status LED blinks once;
- else if Partition State detected
  - Port Status LED blinks twice;
- else Everything is OK (Port Status LED is ON)

##### 13.2.1.2 Status LED blink timing (Mode 0)

Link Integrity test failed, Status LED blinks once. Primary status bit is active for 500 ms every 5 secs. Partition, Status LED blinks twice. Primary status bit is activated twice every 5 secs for 500 ms each time, with a period of 1500 ms between two consecutive activations.

**Figure 10: Primary Status LED Timing (One Blink).**

Link Integrity Test Fails: Primary Status LED Blinks 500ms every 5sec

**Figure 11: Primary Status LED Timing (Two Blinks)**

Port Partitioned: Primary Status LED Twice for 500 ms every 5 sec

### 13.2.2 Transmit data in progress

This signal indicates the port is transmitting data.

### 13.2.3 Receive data in progress

This signal indicates port receive activity.

### 13.2.4 Collision active

This signal indicates the port collision event detected by the port.

### 13.2.5 Full/Half duplex

This signal indicates the port duplex: active - full duplex, inactive - half duplex.

### 13.2.6 Receive Buffer Full

This signal indicates the port receive buffer status: active - the buffer exceeds its programmed threshold, inactive otherwise.

### 13.2.7 Forwarding of unknown packets enabled

This signal indicates the port mode of forwarding unknown packets: active - forwarding unknown packets enabled, inactive otherwise.

### 13.2.8 Port Configured as Sniffer

This signal indicates if the port mode is configured as a sniffer target port: active - port is a target sniffer, inactive otherwise.

### 13.2.9 Link Fail State

This signal indicates the port link status: active - link is down, inactive - link is up.

### 13.2.10 Partition State

This signal indicates the port partition status: active - port entered partition state, otherwise inactive.

### 13.2.11 Secondary Port Status LED

Indicates the secondary status mode as per the inverted value of LEDMode input.

### 13.2.12 Pure Port Status LED

This signal will be inactive for any of the following events:

- Port is disabled
- Link Integrity Test Failed
- Partition State Detected

Otherwise, this signal is active.

## 13.3 LED Signal Timing Types

### 13.3.1 Static LED Signals

These signals are stable for relatively long time periods. The LED indication directly reflects their current value. The static signals are:

- Port Status (LEDMode 0)
- Forwarding of Unknown packets enabled
- Port Configured as Sniffer
- Full/Half Duplex

### 13.3.2 Dynamic Internal Signals

These signals are typically active for short time periods. In order to be visible through the LED Indication Interfaces, the GT-482xx includes a "monostable" function per each of these dynamic signals so they can be viewed on the LED indication output for a period of about 62 ms in LEDMode 0 and 7.5 ms in LEDMode 1. The dynamic signals are:

- Port Status (LEDMode 1)
- Transmit data in progress (TxEn)
- Receive data in progress (RxDV)
- Collision active (Col)
- Receive Buffer Full



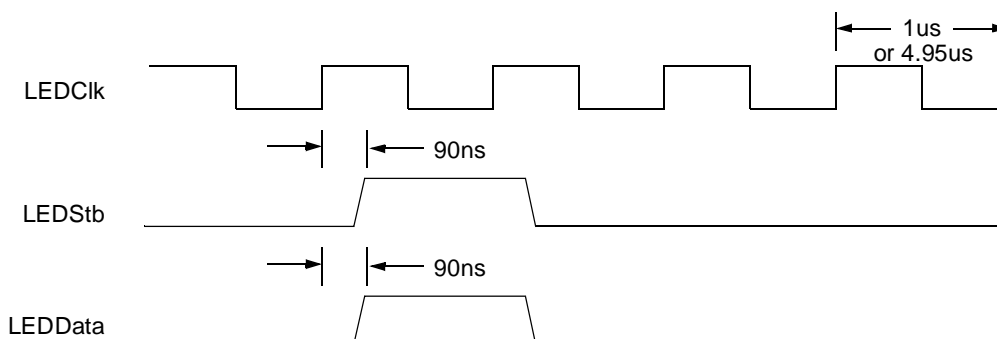
### 13.4 LED Interface Description

The LED serial interface consists of three outputs:

- **LEDClk:** LEDClk is the primary timebase of the LED Indications Interface. It is a 50% duty cycle free running clock at a fixed frequency selectable via the LEDMode input. If LEDMode is LOW, LEDClk will be 1 MHz. If LEDMode is HIGH, LEDClk will be 202 KHz. LEDClk is active when Rst\* is asserted. LEDClk will be floated during Rst\*.
- **LEDStb:** LEDStb (active HIGH) in Mode0, indicates the beginning of the data frame. It is activated for a duration of one LEDClk cycle once every 256 LEDClk cycles, starting from Rst\* deactivation. This signal marks the beginning of the 256 bit long LED data frame. LEDStb in Mode1 is asserted for the first 56 bit stream. LEDStb transitions occur 90 ns after LEDClk rising edge.
- **LEDData:** The internal signals are multiplexed on the LEDData output for every data frame. LEDData transitions occur 90 ns after LEDClk rising edge. All internal signals accessible via LEDData are active HIGH internally and are inverted on the LEDData output (i.e. when an internal signal is active, the data bit on the LEDData output will be LOW). For example: If port 0 transmits data, the internal\_event\_transmit[0] signal is active HIGH and the corresponding bit 9 in the LEDData serial stream is LOW.

The timings for the LED serial interface are shown in Figure 12.

**Figure 12: Serial LED Interface Timings**



#### 13.4.1 Table of Internal Activities/Status Driven via the Serial LED Interface

The following tables provide a bit description of the internal signals driven through the LED Indications Serial Interface. The bit numbers refer to the activation of LEDStb. LEDStb is active for bit# 1. RESERVED bit contents are not defined (i.e. can be either HIGH or LOW).

**Table 14: LED Signals for Mode0**

Bit Number	Signal	Bit Number	Signal
1	primary_port_status[0]	106	transmit[7]
2	primary_port_status[1]	107	receive[7]
3	primary_port_status[2]	108	collision[7]
4	primary_port_status[3]	109	rx_buffer_full[7]
5	primary_port_status[4]	110	unknown_enable[7]

Table 14: LED Signals for Mode0 (Continued)

Bit Number	Signal	Bit Number	Signal
6	primary_port_status[5]	111	port_is_sniffer[7]
7	primary_port_status[6]	112	full_duplex[7]
8	primary_port_status[7]	113	transmit[8]
9	primary_port_status[8]	114	receive[8]
10	primary_port_status[9]	115	collision[8]
11	primary_port_status[10]	116	rx_buffer_full[8]
12	primary_port_status[11]	117	unknown_enable[8]
13	primary_port_status[12]	118	port_is_sniffer[8]
14	primary_port_status[13]	119	full_duplex[8]
15	Reserved	120	transmit[9]
16	Reserved	121	receive[9]
17	Reserved	122	collision[9]
18	Reserved	123	rx_buffer_full[9]
19	Reserved	124	unknown_enable[9]
20	Reserved	125	port_is_sniffer[9]
21	Reserved	126	full_duplex[9]
22	Reserved	127	transmit[10]
23	Reserved	128	receive[10]
24	Reserved	129	collision[10]
25	Reserved	130	rx_buffer_full[10]
26	Reserved	131	unknown_enable[10]
27	Reserved	132	port_is_sniffer[10]
28	Reserved	133	full_duplex[10]
29	Reserved	134	transmit[11]
30	Reserved	135	receive[11]
31	Reserved	136	collision[11]
32	Reserved	137	rx_buffer_full[11]
33	Reserved	138	unknown_enable[11]
34	Reserved	139	port_is_sniffer[11]
35	Reserved	140	full_duplex[11]
36	Reserved	141	transmit[12]
37	Reserved	142	receive[12]
38	Reserved	143	collision[12]
39	Reserved	144	rx_buffer_full[12]
40	Reserved	145	unknown_enable[12]

Table 14: LED Signals for Mode0 (Continued)

Bit Number	Signal	Bit Number	Signal
41	Reserved	146	port_is_sniffer[12]
42	Reserved	147	full_duplex[12]
43	Reserved	148	transmit[13]
44	Reserved	149	receive[13]
45	Reserved	150	collision[13]
46	Reserved	151	rx_buffer_full[13]
47	Reserved	152	unknown_enable[13]
48	Reserved	153	port_is_sniffer[13]
49	Reserved	154	full_duplex[13]
50	Reserved	155	link_test_fail[0]
51	Reserved	156	link_test_fail[1]
52	Reserved	157	link_test_fail[2]
53	Reserved	158	link_test_fail[3]
54	Reserved	159	link_test_fail[4]
55	Reserved	160	link_test_fail[5]
56	Reserved	161	link_test_fail[6]
57	transmit[0]	162	link_test_fail[7]
58	receive[0]	163	link_test_fail[8]
59	collision[0]	164	link_test_fail[9]
60	rx_buffer_full[0]	165	link_test_fail[10]
61	unknown_enable[0]	166	link_test_fail[11]
62	port_is_sniffer[0]	167	link_test_fail[12]
63	full_duplex[0]	168	link_test_fail[13]
64	transmit[1]	169	partition[0]
65	receive[1]	170	partition[1]
66	collision[1]	171	partition[2]
67	rx_buffer_full[1]	172	partition[3]
68	unknown_enable[1]	173	partition[4]
69	port_is_sniffer[1]	174	partition[5]
70	full_duplex[1]	175	partition[6]
71	transmit[2]	176	partition[7]
72	receive[2]	177	partition[8]
73	collision[2]	178	partition[9]
74	rx_buffer_full[2]	179	partition[10]
75	unknown_enable[2]	180	partition[11]

Table 14: LED Signals for Mode0 (Continued)

Bit Number	Signal	Bit Number	Signal
76	port_is_sniffer[2]	181	partition[12]
77	full_duplex[2]	182	partition[13]
78	transmit[3]	183	secondary_port_status[0]
79	receive[3]	184	secondary_port_status[1]
80	collision[3]	185	secondary_port_status[2]
81	rx_buffer_full[3]	186	secondary_port_status[3]
82	unknown_enable[3]	187	secondary_port_status[4]
83	port_is_sniffer[3]	188	secondary_port_status[5]
84	full_duplex[3]	189	secondary_port_status[6]
85	transmit[4]	190	secondary_port_status[7]
86	receive[4]	191	secondary_port_status[8]
87	collision[4]	192	secondary_port_status[9]
88	rx_buffer_full[4]	193	secondary_port_status[10]
89	unknown_enable[4]	194	secondary_port_status[11]
90	port_is_sniffer[4]	195	secondary_port_status[12]
91	full_duplex[4]	196	secondary_port_status[13]
92	transmit[5]	197	pure_port_status[0]
93	receive[5]	198	pure_port_status[1]
94	collision[5]	199	pure_port_status[2]
95	rx_buffer_full[5]	200	pure_port_status[3]
96	unknown_enable[5]	201	pure_port_status[4]
97	port_is_sniffer[5]	202	pure_port_status[5]
98	full_duplex[5]	203	pure_port_status[6]
99	transmit[6]	204	pure_port_status[7]
100	receive[6]	205	pure_port_status[8]
101	collision[6]	206	pure_port_status[9]
102	rx_buffer_full[6]	207	pure_port_status[10]
103	unknown_enable[6]	208	pure_port_status[11]
104	port_is_sniffer[6]	209	pure_port_status[12]
105	full_duplex[5]	210	pure_port_status[13]
		211-256	Reserved

Table 15: LED Signals for Mode1

Bit Number	Signal	Bit Number	Signal
1	link_test_pass/override[0]	106	transmit[7]
2	link_test_pass/override[0]	107	receive[7]
3	link_test_pass/override[1]	108	collision[7]
4	link_test_pass/override[1]	109	rx_buffer_full[7]
5	link_test_pass/override[2]	110	unknown_enable[7]
6	link_test_pass/override[2]	111	port_is_sniffer[7]
7	link_test_pass/override[3]	112	full_duplex[7]
8	link_test_pass/override[3]	113	transmit[8]
9	link_test_pass/override[4]	114	receive[8]
10	link_test_pass/override[4]	115	collision[8]
11	link_test_pass/override[5]	116	rx_buffer_full[8]
12	link_test_pass/override[5]	117	unknown_enable[8]
13	link_test_pass/override[6]	118	port_is_sniffer[8]
14	link_test_pass/override[6]	119	full_duplex[8]
15	link_test_pass/override[7]	120	transmit[9]
16	link_test_pass/override[7]	121	receive[9]
17	link_test_pass/override[8]	122	collision[9]
18	link_test_pass/override[8]	123	rx_buffer_full[9]
19	link_test_pass/override[9]	124	unknown_enable[9]
20	link_test_pass/override[9]	125	port_is_sniffer[9]
21	link_test_pass/override[10]	126	full_duplex[9]
22	link_test_pass/override[10]	127	transmit[10]
23	link_test_pass/override[11]	128	receive[10]
24	link_test_pass/override[11]	129	collision[10]
25	link_test_pass/override[12]	130	rx_buffer_full[10]
26	link_test_pass/override[12]	131	unknown_enable[10]
27	link_test_pass/override[13]	132	port_is_sniffer[10]
28	link_test_pass/override[13]	133	full_duplex[10]
29	activity[0]	134	transmit[11]
30	full_duplex[0]	135	receive[11]
31	activity[1]	136	collision[11]
32	full_duplex[1]	137	rx_buffer_full[11]
33	activity[2]	138	unknown_enable[11]
34	full_duplex[2]	139	port_is_sniffer[11]
35	activity[3]	140	full_duplex[11]

Table 15: LED Signals for Mode1 (Continued)

Bit Number	Signal	Bit Number	Signal
36	full_duplex[3]	141	transmit[12]
37	activity[4]	142	receive[12]
38	full_duplex[4]	143	collision[12]
39	activity[5]	144	rx_buffer_full[12]
40	full_duplex[5]	145	unknown_enable[12]
41	activity[6]	146	port_is_sniffer[12]
42	full_duplex[6]	147	full_duplex[12]
43	activity[7]	148	transmit[13]
44	full_duplex[7]	149	receive[13]
45	activity[8]	150	collision[13]
46	full_duplex[8]	151	rx_buffer_full[13]
47	activity[9]	152	unknown_enable[13]
48	full_duplex[9]	153	port_is_sniffer[13]
49	activity[10]	154	full_duplex[13]
50	full_duplex[10]	155	link_test_fail[0]
51	activity[11]	156	link_test_fail[1]
52	full_duplex[11]	157	link_test_fail[2]
53	activity[12]	158	link_test_fail[3]
54	full_duplex[12]	159	link_test_fail[4]
55	activity[13]	160	link_test_fail[5]
56	full_duplex[13]	161	link_test_fail[6]
57	transmit[0]	162	link_test_fail[7]
58	receive[0]	163	link_test_fail[8]
59	collision[0]	164	link_test_fail[9]
60	rx_buffer_full[0]	165	link_test_fail[10]
61	unknown_enable[0]	166	link_test_fail[11]
62	port_is_sniffer[0]	167	link_test_fail[12]
63	full_duplex[0]	168	link_test_fail[13]
64	transmit[1]	169	partition[0]
65	receive[1]	170	partition[1]
66	collision[1]	171	partition[2]
67	rx_buffer_full[1]	172	partition[3]
68	unknown_enable[1]	173	partition[4]
69	port_is_sniffer[1]	174	partition[5]
70	full_duplex[1]	175	partition[6]

Table 15: LED Signals for Mode1 (Continued)

Bit Number	Signal	Bit Number	Signal
71	transmit[2]	176	partition[7]
72	receive[2]	177	partition[8]
73	collision[2]	178	partition[9]
74	rx_buffer_full[2]	179	partition[10]
75	unknown_enable[2]	180	partition[11]
76	port_is_sniffer[2]	181	partition[12]
77	full_duplex[2]	182	partition[13]
78	transmit[3]	183	secondary_port_status[0]
79	receive[3]	184	secondary_port_status[1]
80	collision[3]	185	secondary_port_status[2]
81	rx_buffer_full[3]	186	secondary_port_status[3]
82	unknown_enable[3]	187	secondary_port_status[4]
83	port_is_sniffer[3]	188	secondary_port_status[5]
84	full_duplex[3]	189	secondary_port_status[6]
85	transmit[4]	190	secondary_port_status[7]
86	receive[4]	191	secondary_port_status[8]
87	collision[4]	192	secondary_port_status[9]
88	rx_buffer_full[4]	193	secondary_port_status[10]
89	unknown_enable[4]	194	secondary_port_status[11]
90	port_is_sniffer[4]	195	secondary_port_status[12]
91	full_duplex[4]	196	secondary_port_status[13]
92	transmit[5]	197	pure_port_status[0]
93	receive[5]	198	pure_port_status[1]
94	collision[5]	199	pure_port_status[2]
95	rx_buffer_full[5]	200	pure_port_status[3]
96	unknown_enable[5]	201	pure_port_status[4]
97	port_is_sniffer[5]	202	pure_port_status[5]
98	full_duplex[5]	203	pure_port_status[6]
99	transmit[6]	204	pure_port_status[7]
100	receive[6]	205	pure_port_status[8]
101	collision[6]	206	pure_port_status[9]
102	rx_buffer_full[6]	207	pure_port_status[10]
103	unknown_enable[6]	208	pure_port_status[11]
104	port_is_sniffer[6]	209	pure_port_status[12]

Table 15: LED Signals for Mode1 (Continued)

Bit Number	Signal	Bit Number	Signal
105	full_duplex[5]	210	pure_port_status[13]
		211-256	Reserved



## 14. INTERRUPTS

The GT-482xx signals interrupts to a management CPU via the Int\* pin. Interrupts are maskable through the Interrupt Mask register and the interrupt source is determined through the Interrupt Cause register. Interrupts are cleared by writing '0' to the corresponding bit in the Interrupt Cause register. Writing '1' to a bit in the Cause register has no effect.

## 15. RESET CONFIGURATION

The GT-482xx uses several pins as configuration inputs to set certain parameters following a RESET. The definition of the configuration pins changes immediately after RESET to their normal function.

### 15.1 Configuration Pins

Configuration pins must be externally pulled up or down at RESET to select the desired operational parameter. The recommended value of the pull-up/down resistors is 4.7K Ohms. Table 16 shows the configuration pins for the GT-482xx. It is recommended to pull up the pins for unmanaged systems unless the user chooses to set a specific feature, for example, pull down the DData[23:0] to set the ports to 10Base-T.

**Table 16: RESET Pin Strapping Options**

Pin	Configuration Function
DAddr[4:0]	GT-482xx Base Address
DAddr[5]	ForceLinkPass12
0- 1-	Force link Do not force link
DAddr[6]	ForceLinkPass13
0- 1-	Force link Do not force link
DAddr[7]	Disable Transmit Watchdog
0- 1-	disable enable
DAddr[8]	Virtual LAN Tagging Packet Extension mode
0- 1-	Accept packets up to 1518 bytes in length Accept packets up to 1536 bytes in length
DAddr[9]	Skip Init
0- 1-	Skip initialization Do not skip initialization
DAddr[10]	Disable Buffer Threshold
0- 1-	disable (dynamic buffer allocation) enable (fixed buffer allocation)
DAddr[11]	BackPressure Enable/Disable
0- 1-	disable enable
DData[23:0]	10Mbps serial mode (two bits per port, DData[1:0] for Port 0, DData[3:2] for Port 1,...,DData[23:22] for Port 11)
00 01 10 11	10Base-T 10Base-FL AUI Reserved
DData[24]	Full duplex mode for port #12

Table 16: RESET Pin Strapping Options (Continued)

Pin	Configuration Function
0- 1-	half duplex full duplex
DData[25]	Full duplex mode for port #13
0- 1-	half duplex full duplex
DData[26]	Auto-Negotiation Enable for port #12
0- 1-	disable enable
DData[27]	Auto-Negotiation Enable for port #13
0- 1-	disable enable
DData[28]	Flow Control Enable for 10Mbps ports
0- 1-	disable enable
DData[29]	Flow Control Enable for port #12
0- 1-	disable enable
DData[30]	Flow Control Enable for port #13
0- 1-	disable enable
DData[31]	Head-of-Line Blocking Prevention
0- 1-	enable disable
BurstAddr[1]	Enable Expansion Priority on port 13.
0- 1-	enable disable
TxDE[11:0]	Full duplex mode for the 10Mbps ports
0- 1-	half full
DQM	DRAM Size
0- 1-	1Mbyte 4Mbyte
Ras*, Cas*, WE*	CPU type
000	See text. Not used in GT-48207.
LEDStb	LED mode
0- 1-	LED mode 0 LED mode 1

Table 16: RESET Pin Strapping Options (Continued)

Pin	Configuration Function
TxEn[0]	AUI Type
0-	The inter-packet gap will restart at the end of TxEn, ignoring any loopback of TxD back to RxD.
1-	This mode is compatible with the GT-48001. The inter-packet gap will restart when there is no transmit or receive activity.

## 15.2 Configuration Input Timings

The configuration inputs have two timing requirements:

- Setup/hold time to clock (as any synchronous input)
- Setup of at least 10 clock cycles before RESET de-assertion (rising edge).

To ensure that these parameters are set, use resistors to strap the configuration pins and delay RESET de-assertion for at least 10 clock cycles after the clock is stable.

## 16. CPU HARDWARE INTERFACE AND ADDRESS MAPPING

The GT-482xx can work in both managed and unmanaged implementations. When managed, the GT-482xx interfaces directly with a 32-bit slave bus. The GT-482xx supports single and burst (up to eight 32-bit word) read/write CPU operations. A single interrupt line is used to indicate interrupt requests to the CPU.

### 16.1 Register and Memory Mapping

The GT-482xx's internal registers and SDRAM are memory mapped. Bit 22 in the address is used to specify DRAM area ("0" - DRAM) or the internal registers ("1" - Internal registers). The GT-482xx SDRAM can be accessed by the CPU.

### 16.2 CPU Interface Modes

The GT-482xx provides direct interface to 32-bit CPUs listed in Table 17. Multiple GT-482xx devices can be connected to the same CPU bus.

**Table 17: GT-482xx CPU Support**

Manuf.	CPU	Support
IDT	R3041	Glueless interface in R3041 mode.
IDT	R4640	Glueless interface through Galileo's GT-64011 System Controller. Use GT Mode.
IDT QED	RV4650 RV4700 RV5000 RM5260 RM5270	Direct interface through Galileo's GT-64010A and GT-64120 System Controllers. Use GT Mode.
Intel	i960 <sup>®</sup> Jx	Glueless interface in i960Jx Mode.
Intel	i960 <sup>®</sup> Cx/Hx	Requires glue logic to multiplex the address and data buses. Use i960Jx Mode.
Intel	i960 <sup>®</sup> Rx	Glueless interface in i960Jx Mode.
Intel AMD Cyrix IBM	80486	Requires glue logic to multiplex the address and data buses. Must terminate bursts not aligned to quad word (16 bytes) since 80486 uses sub-block burst ordering (use Ready/BReady pins). Use i960Jx Mode.
Motorola	ColdFire 5202	Glueless interface in ColdFire Mode.
Motorola	ColdFire 5206	Requires glue logic to multiplex the address and data buses. Use ColdFire Mode.
IBM	PowerPC 401	Minimal glue logic (if any) as this device is compatible with i960Jx protocol. Use i960Jx mode.
-	PCI Bus	Use GT-64111 which has a simple direct connection to GT-482xx. An Application Note will be available. Use GT mode.

### 16.3 CPU Interface Pin Definitions

The CPU interface pins change definitions depending on the CPU mode chosen. Table 18 shows a summary of how to connect CPU pins to the GT-482xx.

**Table 18: CPU Interface Pin Mappings**

CPU Mode	CPU Interface Pins/Mode Definition					
	Ads*	W/R*	Blast*	BurstAddr [2:1]	Ready*	RdCEn*
i960®Jx	Ads*	W/R*	Blast*	pull up	RdyRcv*	pull up
R3041	Wr*	Rd*	Burst*	Addr[3:2]	Ack*	RdCEn*
GT	Ads*	DevRW*	CSTiming*	BAdr[2:1]	Ready*	BAdr[0]
ColdFire	TS*	RW*	SIZE[0]	pull up	DA*[0]	pull up

### 16.4 Selecting the CPU Mode

The CPU Mode is selected at RESET via the Ras\*, Cas\* and DRAM WE\* signals, as shown in Table 19. These pins should be pulled the appropriate level by pull-up/down resistors. Strapping options not shown are reserved and must not be used.

**Table 19: CPU Mode Selection**

CPU Mode	Ras*	Cas*	DRAM WE*
i960®Jx	0	0	0
ColdFire	0	0	1
R3041	0	1	0
GT	0	1	1

**Table 20: Burst Size for Different CPU Modes**

CPU Mode	Number of Long Word Reads	Number of Long Word Writes
i960®Jx	1,2,3,4	1,2,3,4
ColdFire	1,4	1,4
R3041	1,4	1
GT	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8

### 16.5 GT-482xx Base Address

Each GT-482xx in a system has a base address in the memory map. The GT-482xx compares bits 31:27 of the address to its own base address. The GT-482xx will respond only if bits AD[31:27] in the address phase of the cycle are equal to the GT-482xx Base Address Register.

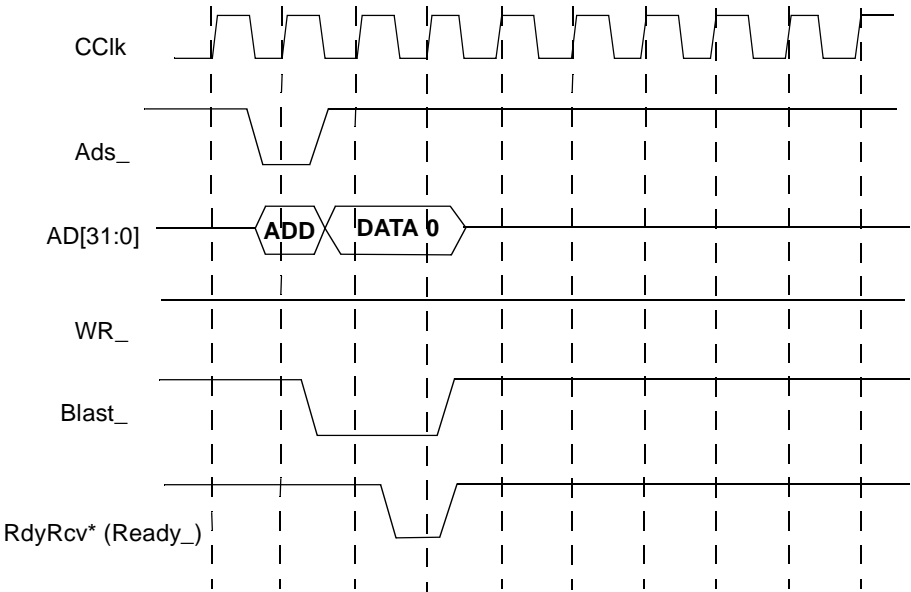
The value of the Base Address Register is set by DAddr[4:0] pin strapping at RESET. Each GT-482xx connected to the same bus in a managed system must have a different Base Address.

16.6 CPU Interface Applications

The following sections provide GT-482xx waveform information illustrating the read/write single long word and read/write bursts for each of the different CPU modes.

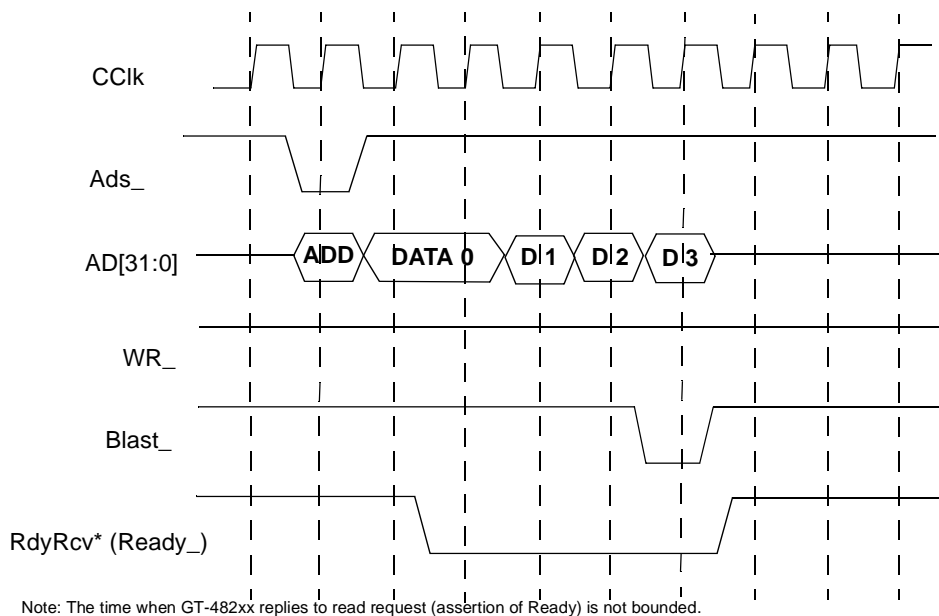
16.6.1 i960® Jx Mode

Figure 13: i960 Write Single Long Word



Note: The CPU can terminate a burst by assertion of Blast.

**Figure 14: i960 Write Burst of Four Long Words**



**Figure 15: i960 Read Single Long Word**

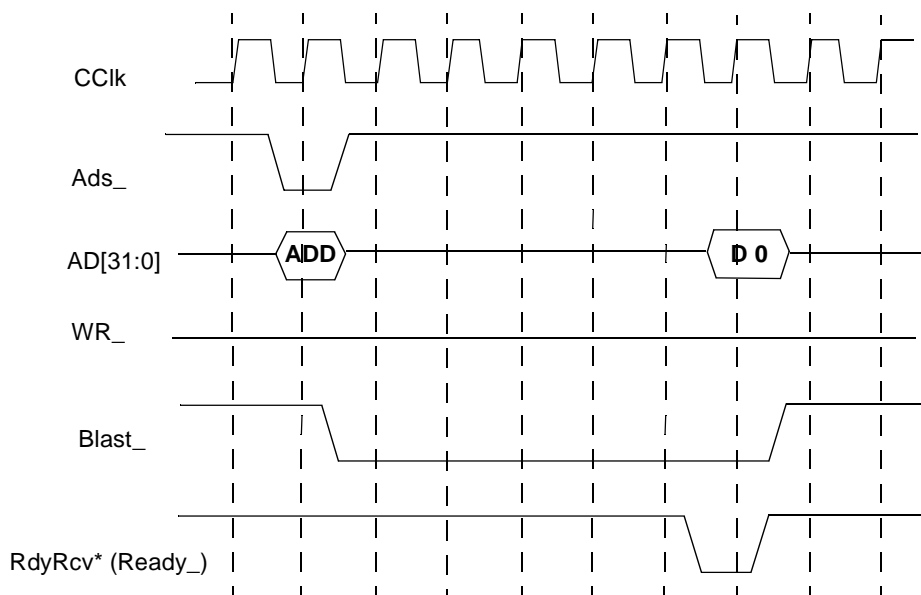
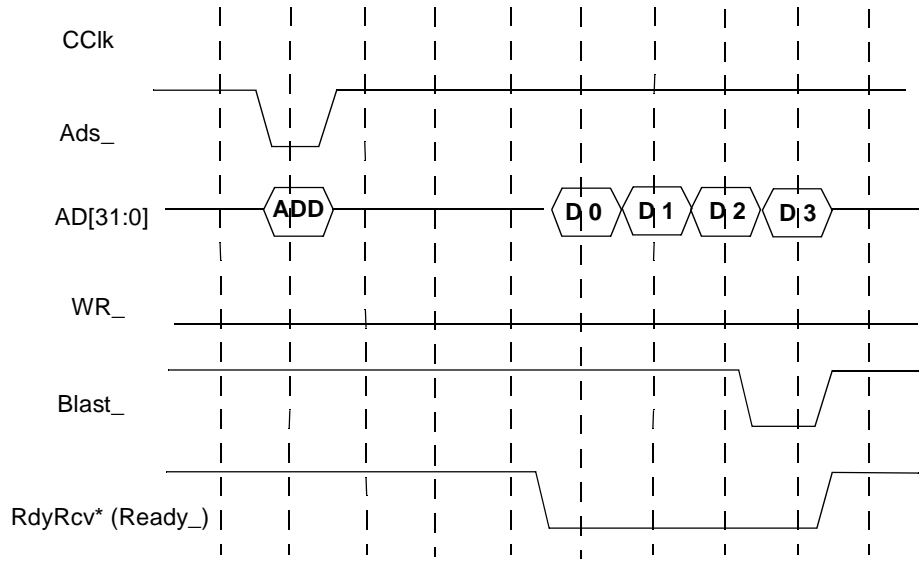


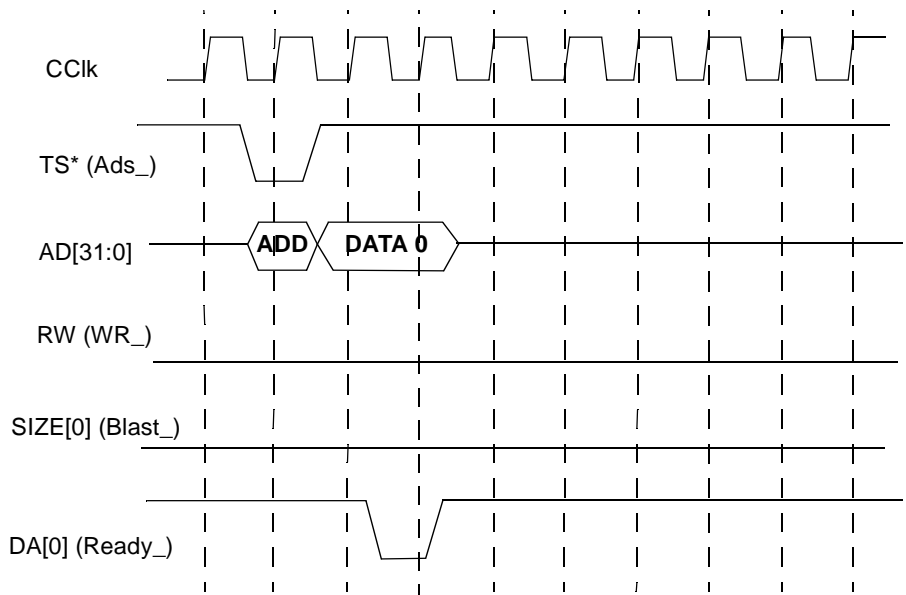


Figure 16: i960 Read Burst of Four Long Words

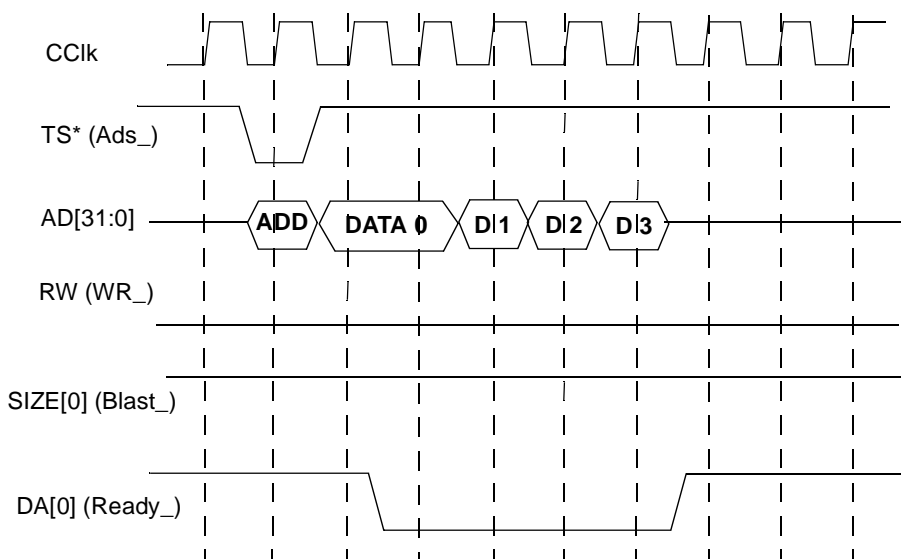


16.6.2 ColdFire (Motorola 5202) Mode

Figure 17: ColdFire 5202 Write Single Long Word



**Figure 18: ColdFire 5202 Write Burst of Four Long Words**



**Figure 19: ColdFire 5202 Read Single Long Word**

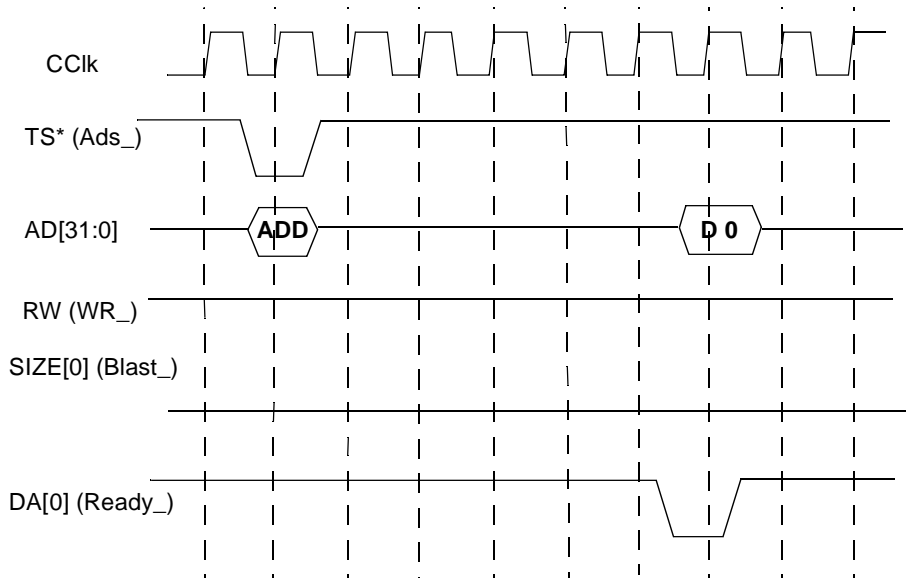
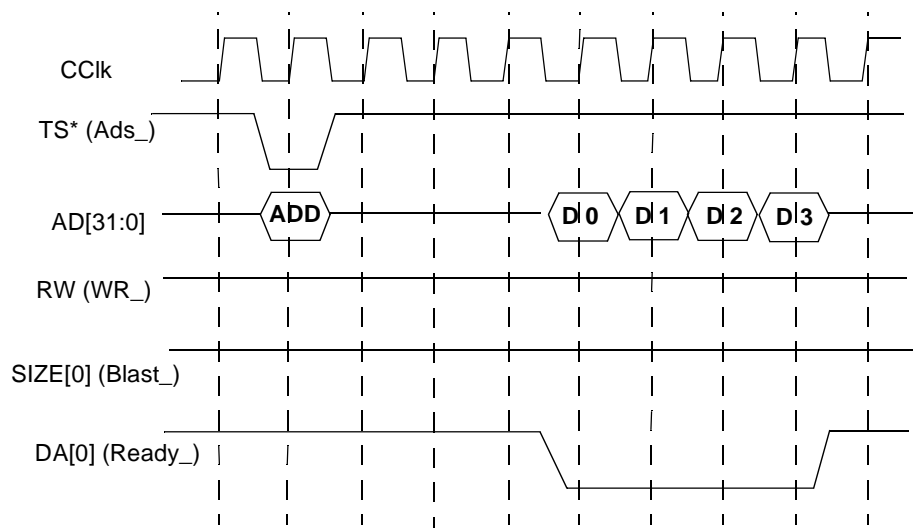


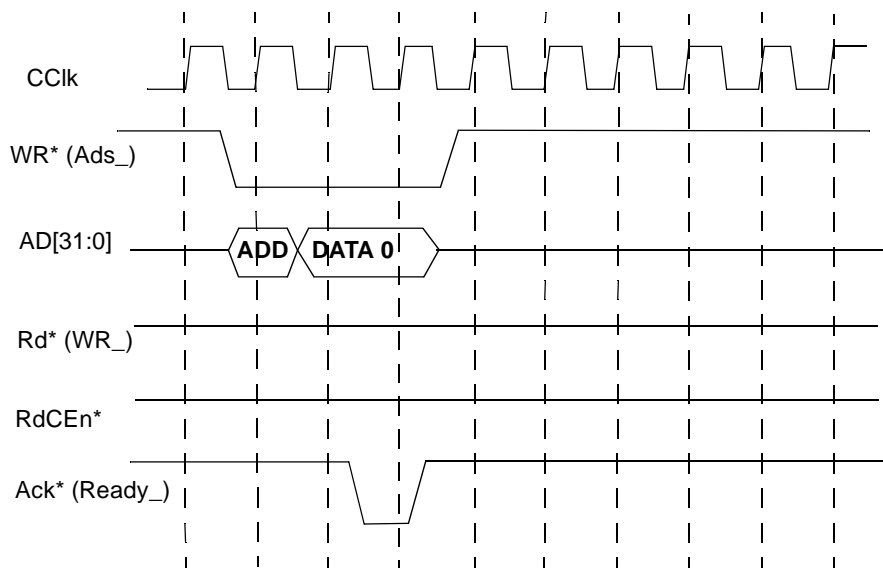
Figure 20: ColdFire 5202 Read Burst of Four Long Words



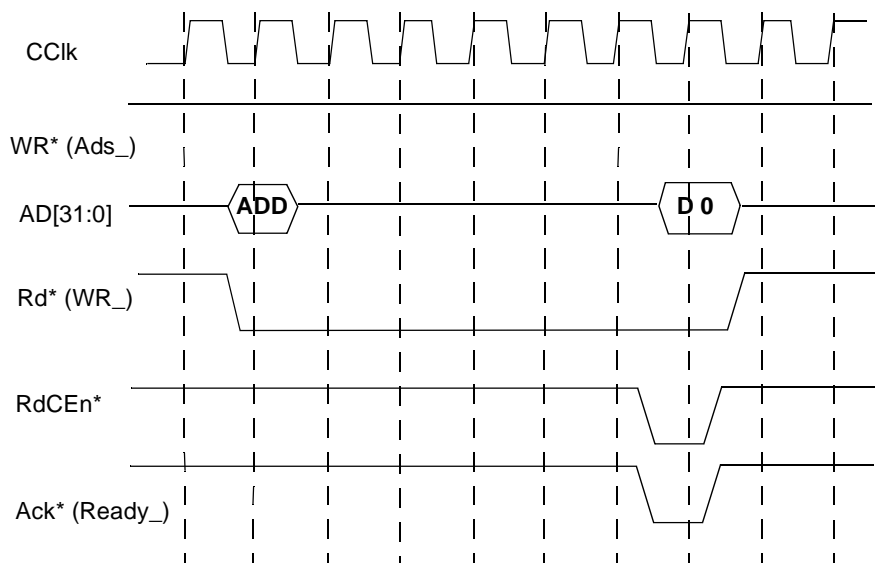
Note: The time when GT-482xx replies to read request (assertion of Ready) is not bounded

### 16.6.3 R3041 Mode

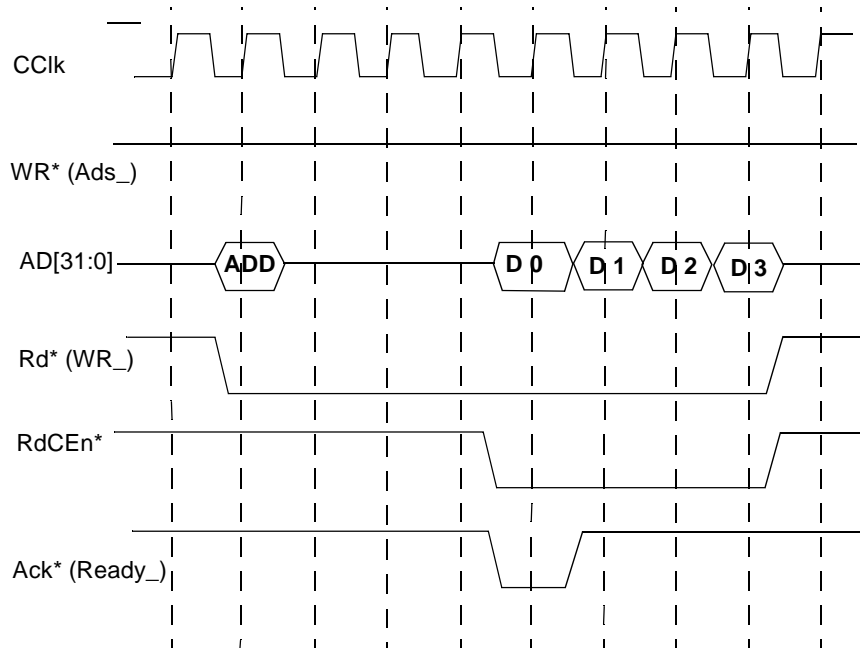
Figure 21: R3041 Write Single Long Word



**Figure 22: R3041 Read Single Long Word**



**Figure 23: R3041 Read Burst of Four Long Words**



Notes: The time when GT-482xx replies to read request (assertion of Ready) is not bounded.  
On the 3041 Writes to GT-482xx is done only at single Long Word Write.

### 16.6.4 Galileo (GT) Mode

Figure 24: GT Write Single Long Word

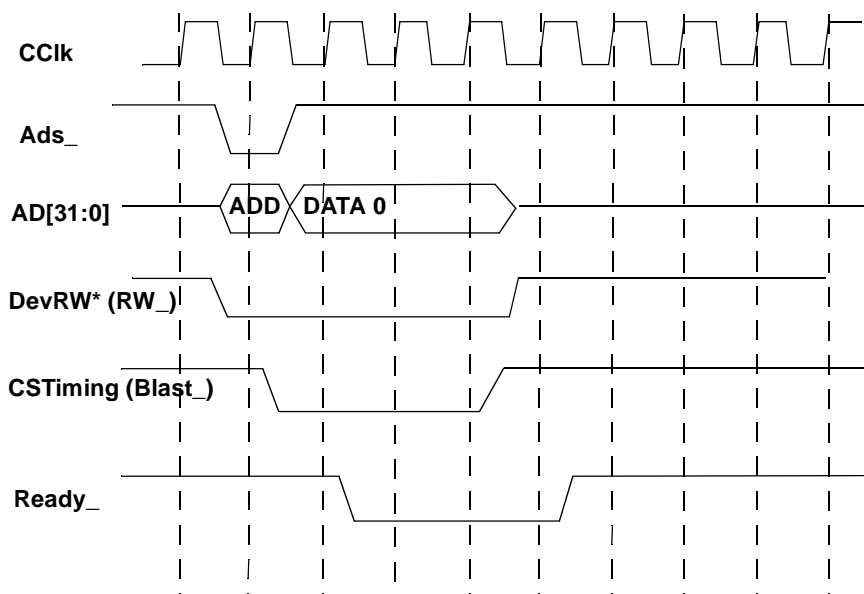


Figure 25: GT Write Burst of Four Long Words

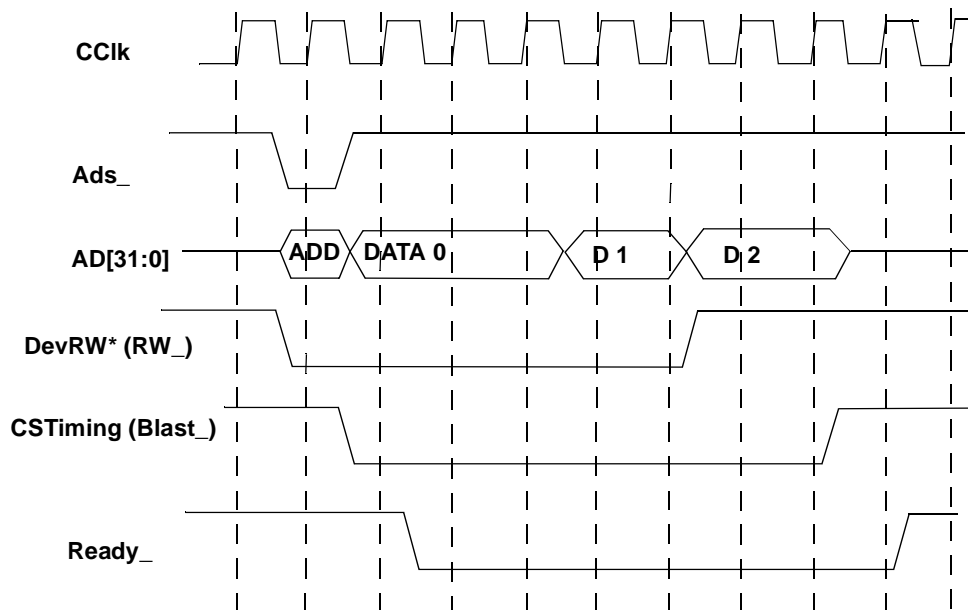


Figure 26: GT Read Single Long Word

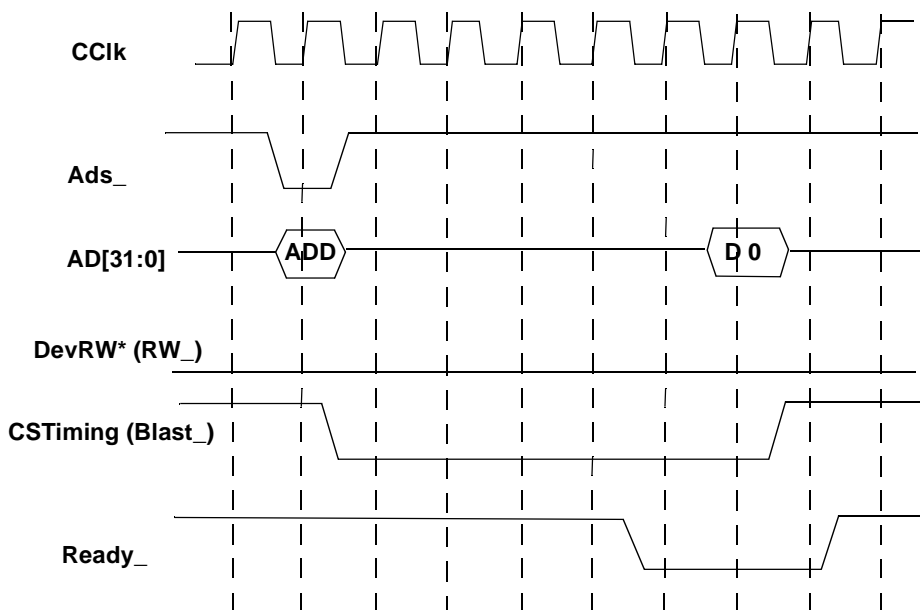
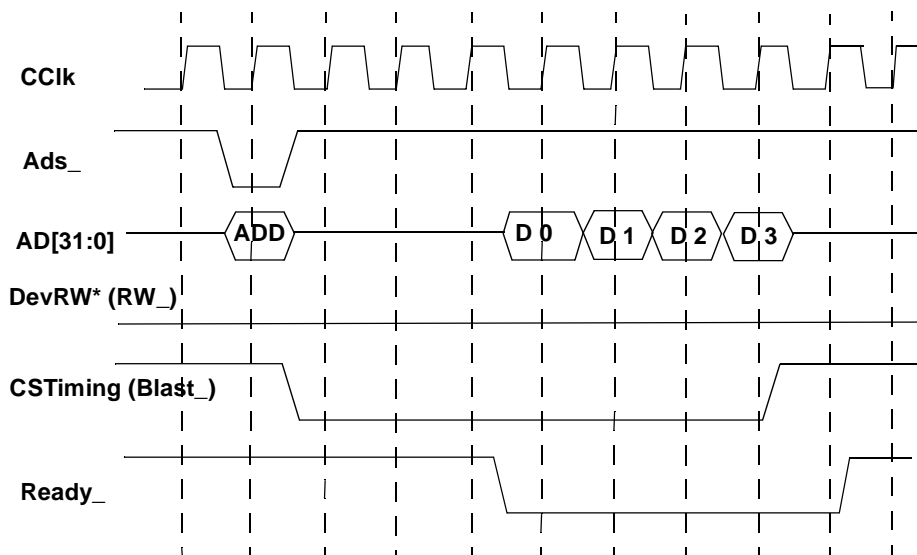


Figure 27: GT Read Burst of Four Long Words



Note: The time when GT-482xx replies to read request (assertion of Ready) is not bounded.

## 16.7 CPU Interface Priority

CPU Access to the GT-482xx's DRAM is the lowest priority in the DRAM arbiter. In order to prevent deadlocks, the GT-482xx incorporates a CPU-to-DRAM access timer. This timer starts counting when the CPU requests an access to the DRAM. When the timer expires, the CPU will be the next master to access the DRAM. The timer can be enabled and programmed via CPU Time Out Register.

## 16.8 Memory Endianess

The GT-482xx treats SDRAM as a little-endian array. When data is fetched from a 32-bit word for transmission it is fetched in the following sequence: bits [7:0], bits [15:8], bits [23:16], bits [31:24]. The software engineer should keep this in mind when creating/modifying packets within GT-482xx memory.

## 17. SDRAM INTERFACE

The GT-482xx includes all of the logic needed for a glueless interface to 1-4 megabytes of synchronous DRAM or synchronous graphics RAM. (SGRAM is supported because of the lower density that allows the creation of 1Mbyte arrays.) The SDRAM interface runs at the switching core frequency as determined by ACIk. Speeds up to 66MHz are supported. Refresh is handled automatically by the GT-482xx as well.

### 17.1 DRAM Configuration

The user can set the DRAM refresh time according to Table 54, "DRAM Configuration, Offset: 0x1448," on page 101. The Cas and Ras parameters can be set as shown in Table 55, "DRAM Parameters, Offset: 0x144C," on page 101. See Table 16, "RESET Pin Strapping Options," on page 66 for DRAM size sampled at the DQM pin.

### 17.2 DRAM Initialization

The GT-482xx automatically resets all Address Tables after reset deassertion. For 1 Mb DRAM mode the process takes 50,000 ACIk cycles, for 4 Mb DRAM mode the process takes 200,000 ACIk cycles. During Address Table reset the GT-482xx will not accept any incoming packets. The GT-482xx automatically initializes the SDRAM after reset. This initialization process can be performed by the CPU using the DRAM configuration registers.



## 18. REGISTER TABLES

The GT-482xx incorporates Command registers and various counters for management purposes. The GT-482xx can work in stand-alone mode. In this mode there is no requirement for CPU Intervention (a system with no CPU) in which case the default values of the control registers are used. The GT-482xx internal registers have 32-bit accesses.

Table 21 lists the registers, their offset, read/write attributes and the corresponding tables and page numbers that provide detailed descriptions for each register.

These registers must be accessed as single data accesses only (burst accesses are not allowed).

**Table 21: Register Map Table**

Description	Offset	Read/Write	Table/Page Number
GT-482xx Base address	0x0	R/W	Table 22 on page 84
Global Control register	0x4	R/W	Table 23 on page 84
Status Register	0x8	Read only	Table 24 on page 87
Sniffer + Aging Timer	0xC	R/W	Table 25 on page 88
Serial Parameters for 10Mbps	0x10	R/W	Table 26 on page 88
Watchdog and Tx Threshold	0x14	R/W	Table 28 on page 89
Interrupt Cause	0x18	R/W	Table 29 on page 90
Interrupt Mask	0x1C	R/W	Table 30 on page 91
Cpu Tx High Desc1	0x20	R/W	Table 31 on page 91
Cpu Tx High Desc2	0x24	R/W	Table 32 on page 92
Cpu Tx Low Desc1	0x28	R/W	Section 18.1.2 page 93
Cpu Tx Low Desc2	0x2C	R/W	Section 18.1.2 page 93
Cpu EL Free Req	0x30	R/W	Table 35 on page 93
Cpu Empty Buffer	0x34	Read only	Table 36 on page 94
Cpu Enqueue1	0x38	R/W	Table 37 on page 94
Cpu Enqueue2	0x3C	R/W	Table 38 on page 94
Cpu New Address1	0x40	Write only	Table 39 on page 95
Cpu New Address2	0x44	Write only	Table 40 on page 95
Cpu New Address3	0x48	R/W	Table 41 on page 96
Cpu Query	0x4C	Read only	Table 42 on page 96
SMI Register	0x50	R/W	Table 43 on page 98
802.1Q Ethertype	0x54	R/W	Table 44 on page 98

Table 21: Register Map Table (Continued)

Description	Offset	Read/Write	Table/Page Number
General Purpose register1	0x58	R/W	Table 45 on page 98
General Purpose register2	0x5C	R/W	Table 46 on page 99
Rx_10 Threshold register	0x60	R/W	Table 47 on page 99
Rx_100 Threshold register	0x64	R/W	Table 48 on page 99
CPU Threshold register	0x68	R/W	Table 49 on page 100
LED Override register	0x6C	R/W	Table 50 on page 100
Source Address Low register	0x70	R/W	Table 51 on page 100
Source Address High register	0x74	R/W	Table 52 on page 101
Serial Parameters for 100Mbps	0x78	R/W	Table 27 on page 88
CPU Time out register	0x7C	R/W	Table 53 on page 101
Port Control Register[3:0]	0x400 - 0x40C	R/W	Table 58 on page 102
EASE Register[3:0]	0x410 - 0x41C	Write only	Table 61 on page 107
Port 0 Counter	0x600 - 0x654	Read only	Table 63 on page 109
Port 1 Counter	0x680 - 0x6D4	Read only	Table 63 on page 109
Port 2 Counter	0x700 - 0x754	Read only	Table 63 on page 109
Port 3 Counter	0x780 - 0x7D4	Read only	Table 63 on page 109
Port Control Register[7:4]	0x800 - 0x80C	R/W	Table 58 on page 102
EASE Register[7:4]	0x810 - 0x81C	Write only	Table 61 on page 107
Port 4 Counter	0xA00 - 0xA54	Read only	Table 63 on page 109
Port 5 Counter	0xA80 - 0xAD4	Read only	Table 63 on page 109
Port 6 Counter	0xB00 - 0xB54	Read only	Table 63 on page 109
Port 7 Counter	0xB80 - 0xBD4	Read only	Table 63 on page 109
Port Control Register[11:8]	0xC00 - 0xC0C	R/W	Table 58 on page 102
EASE Register[11:8]	0xC10 - 0xC1C	Write only	Table 61 on page 107
Port 8 Counter	0xE00 - 0xE54	Read only	Table 63 on page 109
Port 9 Counter	0xE80 - 0xED4	Read only	Table 63 on page 109
Port 10 Counter	0xF00 - 0xF54	Read only	Table 63 on page 109
Port 11 Counter	0xF80 - 0xFD4	Read only	Table 63 on page 109
Port Control Register 12	0x1000	R/W	Table 59 on page 104
Port Control Register 13	0x1004	R/W	Table 60 on page 106

Table 21: Register Map Table (Continued)

Description	Offset	Read/Write	Table/Page Number
EASE Register[13:12]	0x1008 - 0x100C	Write only	Table 61 on page 107
Port 12 Counter	0x1200 - 0x1254	Read only	Table 63 on page 109
Port 13 Counter	0x1280 - 0x12D4	Read only	Table 63 on page 109
DRAM Configuration	0x1448	R/W	Table 54 on page 101
DRAM Parameters	0x144C	R/W	Table 55 on page 101
SDRAM Operation	0x1474	R/W	Table 56 on page 102
ExtraMrsBits	0x1478	Reserved	Reserved
Address Decode	0x147C	Read only	Table 57 on page 102

## 18.1 Register Description

**Table 22: Base Address, Offset: 0x00**

Bits	Field Name	Function	Initial Value
31:27	BaseAddress	Device Base Address	DAddr[4:0] at reset
26:0	Reserved	Reserved	0x0

**Table 23: Global Control, Offset: 0x04**

Bits	Field Name	Function	Initial Value
0	SwReset	Software Reset <sup>1</sup> 0 - No effect 1 - Reset all internal units except for the CPU unit to their initial state	0x0
1	AgingTrigger	Aging Trigger (only valid if 3:2 are written '10'). When this bit is changed from low to high, the GT-482xx will scan the Address Table and age out (remove old addresses) CPU should clear this bit. NOTE: If Aging Mode is set to trigger mode and this bit is set to '1' and is not cleared, it is equivalent to automatic aging mode.	0x0
3:2	AgingMode	Aging mode: 00 - No Aging support 01 - Automatic aging 10 - Trigger mode 11 - Reserved	0x01
4	Learning	Address MAC Learning Mode 0 - Mode 0 1 - Mode 1 In most applications, this bit does not need to be changed. Designers of managed systems may want to experiment with this bit to improve address learning performance. Initial Value should be 0x1 for P-0/1/2 and 0x0 for P-3/4.	0x1 for P-0/1/2 0x0 for P-3/4
5	ForwUnk	Forward Unknown Packets. Defines whether the GT-482xx will forward Unknown packets to the <b>CPU</b> or not. 0 - Do not forward to CPU 1 - Forward to CPU Note: For exact description see "Address Recognition" on page 28	0x0

Table 23: Global Control, Offset: 0x04 (Continued)

Bits	Field Name	Function	Initial Value
6	ForwNewAdd	Forward New Address. Defines whether the GT-482xx will forward New_Address messages to the CPU or not. 0 - Do not forward 1 - Forward	0x0
7	ForwAged Addr	Forward Aged out address Defines whether the GT-482xx forwards New_Address messages for aged out addresses to the CPU or not. 0 - Do not forward 1 - Forward	0x0
8	CPUEn	CPU Enable. This bit indicates that there is a CPU in the system. 0 - CPU does not exist 1 - CPU exists	0x0
11-9	Priority weight	These bits indicates the priority weight as follows: 000 - One packet is transmitted from the high priority queue, and one packet is transmitted from the low priority queue. 001 - 2 from the high queue, 1 from the low 010 - 4 from the high queue, 1 from the low 011 - 6 from the high queue, 1 from the low 100 - 8 from the high queue, 1 from the low 101 - 10 from the high queue, 1 from the low 110 - 12 from the high queue, 1 from the low 111 - All packets from the high queue.	0x000
12	BufThrEn	Buffer Threshold Enable. 0 - There is no limitation on the buffers' allocation (other than physical memory size.) 1 - The buffers allocated to the ports and the CPU are limited to the number which is written in the Rx Buffers Threshold register. This bit is meaningful only when DisBufThr* pin is disabled.	DAddr[10] at reset
13	Reserved	Must be written '0'	0x0
14	ForwMulti	Forward Multicast to <b>CPU</b> only. 0 - the GT-482xx forwards Multicast packets to all the ports and to the CPU. 1 - Multicast packets forwarded only to the CPU. Note: For exact description see "Address Recognition" on page 28.	0x0

Table 23: Global Control, Offset: 0x04 (Continued)

Bits	Field Name	Function	Initial Value
15	ParEn	Partition Enable. When more than 32 collisions occur on a 10Mbps port, or more than 60 collisions occur on a 100Mbps port while transmitting, the GT-482xx enters the Partition mode. It waits for the first good packet from the wire, and then goes back to Normal mode. Under Partition mode it continues transmitting, but not receiving. 0 - Partition mode Disabled 1 - Partition mode Enabled	0x0
16	SpanEn	Spanning Tree Enable. 0 - The BPDU (Bridge Protocol Data Unit) packets are treated as Multicast packets, and therefore are forwarded to all ports. 1 - the GT-482xx forwards BPDU packets only to the CPU.	0x0
17	EnEASE	EASE sampling enable/disable. 0 - EASE sampling disabled. 1 - EASE sampling enabled.  Must be set if EASE is initialized and any of the ports are enabled.  NOTE: Bit 8 must be set to '1' (CPU Enabled) prior to setting this bit to '1'..	0x0
18	EnErrSource	Errored Source Address enable/disable. 0 - Error Source disabled. 1 - Error Source enabled.	0x0
19	MIBCtrMode	0 - MIB counters reflect forwarded packets only. 1 - MIB counters reflect local and forwarded packets. NOTE: A local packet is a packet which is destined to a station on the same port and is not switched.	0x0
20	EnableDevice	Used in conjunction with EnDev* pin to enable or disable the GT-482xx. Refer to EnDev* description for more information.	0x0
21	MIBCIRMode	MIB Counter Clear-on-Read mode 0 - MIB counters will be cleared after they have been read. 1 - MIB counters will not be cleared after they have been read.	0x0
22	IgmpEn	IGMP Enable 0 - Treats IGMP packets as Multicast and send to all ports 1 - Send IGMP only to the CPU	0x0

**Table 23: Global Control, Offset: 0x04 (Continued)**

Bits	Field Name	Function	Initial Value
23	CrcEn	CRC Enable - Enable CRC generation. 0 - Disable 1 - Enable	0x0
24	Limit4	Limit4 - This bit together with Limit4 pin selects the number of consecutive collisions which will occur before the collision counter is reset. When the logical OR of this bit and Limit4 pin is LOW, 16 consecutive collisions must occur before the collision counter is reset (802.3 standard). When HIGH, 4 consecutive collisions must occur before the collision counter is reset (more aggressive).	0x0
25	SyncMode	Synchronous mode 0 - Cclk and Aclk are not synchronized 1 - Cclk and Aclk are synchronized and should be driven from the same clock source  Note: the GT-482xx can work in asynchronous mode even if the Aclk and Cclk are the same.	0x0
26	DisPktLock	Discard packet in Lock mode 0 - Discards the packet with a new address in locked mode 1 - Forwards the packet with new address in locked mode	0x0
31:27	Reserved		-

1. On software RESET, any register bits that are sampled by configuration pins are NOT affected. All other register bits are reset to their initial value.

**Table 24: Status Register, Offset: 0x08**

Bits	Field Name	Function	Initial Value
2:0	Revision number	Indicates the GT-482xx revision number	0x0
3	DRAM_size	Indicates the DRAM size: 0 - 1Mbyte. 1 - 4Mbyte.	DQM at reset
17:4	Port 13:0 link	These bits indicate link on the ports.  Link on the 10Mbps ports (0-11) is indicated by a '0'. Link on the 100Mbps ports (13-12) is indicated by a '1'.	0x0
31:18	Port 13: 0 partition	These bits indicate partition on the ports	0x0

Table 25: Sniffer and Aging Timer, Offset: 0x0C

Bits	Field Name	Function	Initial Value
0	SniffEn	Sniffer Enable 0 - Sniffer mode is disabled 1 - Sniffer mode is enabled	0x0
4:1	SrcSniff	These bits specify the Source Sniffer channel	0x0
8:5	TrgSniff	These bits specify the Target Sniffer channel <b>Note:</b> 0xE (14d) specifies the CPU as the target sniffer	0x0
14:9	Aging Timer	These bits specify the value of the aging timer. See "Address Aging" on page 30 for more information.	0x1E

Table 26: Serial Parameters 10 Register, Offset: 0x10

Bits	Field Name	Function	Initial Value
3-0	JAM_LENGTH	4 bits to determine the JAM_Length (in BackPressure). The step is 0.8mSec. The value of the JAM_Length can vary between 0.8mS to 12mSec	0100b (3.2mSec)
10-4	JAM_IPG	7 bits to determine the JAM_IPG. The step is 100nSec (1 bit time). The JAM_IPG vary between 1 bit time to 128.	0010000b (16-bit time)
17-11	IPG_JAM_TO_DATA	7 bits to determine the IPG JAM to DATA. The step is 100nSec (1 bit time). The JAM_IPG vary between 1 bit time to 128.	0011110b (30-bit time)
24:18	IPGData	Inter-Packet Gap (IPG) Data. The step is 100ns (1 bit-time). The default value is 96 decimal (9.6 $\mu$ s). Note that the IPG during jam varies between 1 bit-time and 128 bit-times.	096d
31:25	DataBlind	Data Blinder. DataBlind is the inhibit time. The time the GT-482xx port does not look at the line to decide to transmit. The range is 0 to 96 in 100ns increments. The default is 64 decimal (6.4 $\mu$ s).	064d

Table 27: Serial Parameters 100 Register, Offset: 0x78

Bits	Field Name	Function	Initial Value
1:0	JAM_Length	2 bits to determine the JAM_Length (in BackPressure) as follows: 00 - 2K nibbles 01 - 4K nibbles 10 - 8K nibbles 11 - 10K nibbles	11b (10K nibbles)



Table 27: Serial Parameters 100 Register, Offset: 0x78 (Continued)

Bits	Field Name	Function	Initial Value
6:2	JAM_IPG	5 bits to determine the JAM_IPG. The step is 10nSec (1 bit time). The JAM_IPG vary between 1 bit time to 32.	01000b (8 bit time)
11:7	IPG_JAM_TO_DATA	5 bits to determine the IPG JAM to DATA. The step is 10nSec (1 bit time). The JAM_IPG vary between 1 bit time to 32.	10000b (16 bit time)
16:12	IPGData	Inter-Packet Gap (IPG): The IPG varies between 12 bit-times (0x3) and 124 bit-times (0x1F). The step is 40ns@100mbs or 400 ns@10mbs (4 bit-times). The default value is 18 hex, or 0.96us@100mbs (9.6uS@10mbs). Value should be written in hexadecimal format. <b>Note:</b> These bits may be changed only when PortEn bits are set to 0 in all Port Control registers (Port is disabled).	0x18 = 24 decimal
21:17	DataBlind	Data Blinder: The number of nibbles from the beginning of the IFG, in which the GT-482xx will restart the IFG counter when detecting a carrier activity. Following this value, the GT-482xx will enter the Data Blinder zone and will not reset the IFG counter to ensure fair access to the medium. Value should be written in hexadecimal format. The default is 10 hex (64 bit times - 2/3 of the default IFG). The step is 40ns (4 bit-times). Valid range is 3 to 1F hex nibbles. <b>Note:</b> These bits may be changed only when PortEn bits are set to 0 in all Port Control registers (Port is disabled).	0x10 = 16 decimal
25:22	Reserved	Reserved	0x0

Table 28: Watchdog and Tx Threshold Register, Offset: 0x14

Bits	Field Name	Function	Initial Value
3:0	TxWatTim	Tx Watchdog Timer. For 100 Mbps operation, the default value of the timer is 63msec and the range is between 10.5mSec and 168msec, in 10.5 ms steps. For 10 Mbps operation, the default value of the timer is 630msec and the range is between 105ms to 1680ms, in 105 ms steps. Valid values: 1 to F hex. Value should be written in hexadecimal format.	0x6 (63 ms @ 100Mbps; 630 ms @ 10 Mbps)
8:4	ChainNu	Chain Number. These bits specify the number of entries in a chain in the Address Table	0x08 (16 entries)

Table 28: Watchdog and Tx Threshold Register, Offset: 0x14

Bits	Field Name	Function	Initial Value
11:9	TxQueThr	<p>Transmit Queue Threshold</p> <p>These bits determine the threshold of the Transmit Queue. When the Transmit queue exceeds this limit, a packet that destined to this queue will be discarded.</p> <p>000: Threshold is 64 in 1M, 256 in 4M</p> <p>001: Threshold is 128 in 1M, 512 in 4M</p> <p>010: Threshold is 192 in 1M, 768 in 4M</p> <p>011: Threshold is 256 in 1M, 1024 in 4M</p> <p>100: Threshold is 320 in 1M, 1280 in 4M</p> <p>101: Threshold is 384 in 1M, 1536 in 4M</p> <p>110: Threshold is 448 in 1M, 1792 in 4M</p> <p>111: Threshold is 512 in 1M, 2048 in 4M</p>	0x3

Table 29: Interrupt Cause, Offset: 0x18

Bits	Field Name	Function	Initial Value
0	IntSumm	<p>Interrupt Summary.</p> <p>Logical 'OR' of all the interrupt bits and their mask.</p> <p>IntSumm is exactly the inverse polarity of Int* pin.</p>	0x0
1	TxDescH	This bit is set by the GT-482xx upon loading the CPU Tx High Priority Desc register	0x0
2	TxDescL	This bit is set by the GT-482xx upon loading the CPU Tx Low Priority Desc register	0x0
3	EmptyBuff	This bit is set by the GT-482xx upon loading the CPU Empty List register	0x0
4	Query	This bit is set by the GT-482xx upon loading the Query register	0x0
5	AddrRecF	<p>Address Recognition Failed.</p> <p>This bit is set by the GT-482xx when the address recognition cycle fails (due to a large number of MAC addresses).</p>	0x0
6	FlushTxQ	<p>Flush Tx Queue.</p> <p>This bit is set by the GT-482xx when one of the Tx queues is flushed due to the Watchdog Timer.</p>	0x0
7	LinkChange	<p>Link State Change.</p> <p>This bit is set by the GT-482xx upon a change in the link state (down-&gt;up, up-&gt;down) for any port.</p>	0x0
8	Part	<p>Partition.</p> <p>This bit is set by the GT-482xx upon entering Partition state in one of the ports.</p>	0x0
22:9	EaseRegister	Ease_Register of port 13 to 0 has reached terminal count of 0	0x0

**Table 29: Interrupt Cause, Offset: 0x18**

Bits	Field Name	Function	Initial Value
23	ErrorSASent	Error_Source message sent to CPU	0x0

Note: Clearing an interrupt cause bit is done by writing '0' value to it. Writing '1' has no effect.

**Table 30: Interrupt Mask<sup>1</sup>, Offset: 0x1C**

Bits	Field Name	Function	Initial Value
23:1	MaskBits	Mask to the CPU interrupt line for the appropriate bits in the Interrupt Cause register. '0' value enables the interrupt for the corresponding Cause bit.	0x0

1. Note that all interrupts are UNmasked by default.

### 18.1.1 CPU Tx High Desc1 and 2, Offset: 0x20, 0x24

Tx High Desc1 and Tx High Desc2 registers contain one of the following two formats depending on DescType bit [30] in the CPU Tx High Desc2:

- CPU\_Tx\_High\_Desc2[30] = 0 - Packet descriptor
- CPU\_Tx\_High\_Desc2[30] = 1 - New\_Address message (can be New\_Address, Error\_Source address or an address that was removed due to automatic aging.

Note: These two registers are not burst read.

**Table 31: CPU Tx High Desc1 - Packet Descriptor, Offset 0x20**

Bits	Field Name	Function	Initial Value
10:0	SourBufAddr	Source Buffer Address (divided by 0x600)	0x0
21:11	Byte Count	Byte Count	0x0
25:22	SourcePort	Source Port number	0x0
26	Mult	Multicast Packet (1 - Multicast)	0x0
27	Unk	Unknown Packet (1 - Unknown)	0x0
28	Sniff	Sniffer Packet (1 - Sniffer)	0x0
29	IGMP	IGMP Packet (1 - IGMP)	0x0
30	Interv	Intervention Packet (1 - Intervention)	0x0
31	Reserved	Reserved	0x0

**Table 32: CPU Tx High Desc2 - Packet Descriptor, Offset 0x24**

Bits	Field Name	Function	Initial Value
13:0	EaseSamp	Ease Sampling for ports [13:0] 0 - EASE Sampling is enabled for port 1 - EASE Sampling is disabled for port	0x0
14	EaseSampCpu	Ease sample is an original packet to CPU 0 - EASE Sample is an original packet to CPU 1 - EASE Sample is not an original packet to CPU	0x0
29:15	-	Reserved	0x0
30	DescType	Descriptor Type 0 - Packet descriptor 1 - New_Address message	0x0
31	Valid	Valid bit. 0 - CPU TX High1 and 2 do not contain valid data 1 - CPU TX High1 and 2 contain valid data. Valid bit is cleared upon reading CPU Tx High Desc2.	0x0

**Table 33: CPU Tx High Desc1 - New Address, Offset 0x20**

Bits	Field Name	Function	Initial Value
31:0	MacAddr[16:47]	MAC Address[16:47]	0x0

**Table 34: CPU Tx High Desc2 - New Address, Offset 0x24**

Bits	Field Name	Function	Initial Value
15:0	MacAddr	MAC Address[0:15]	0x0
19:16	SourcePort	Source Port number	0x0
20	FCSErr	FCS error indication (only for Error source)	0x0
21	OverCount	Over Count frame indication (only for Error source)	0x0
22	Pd	Priority Destination	0x0
23	Ps	Priority Source	0x0
24	St	Static	0x0
25	Mult	Multiple	0x0
26	Id	Intervention Destination	0x0
27	Is	Intervention Source	0x0

**Table 34: CPU Tx High Desc2 - New Address, Offset 0x24 (Continued)**

Bits	Field Name	Function	Initial Value
29:28	NaType	New Address Type: 00: New Address 01: Error Source Address 10: Address that was removed from the Address Table due to aging 11: Reserved	
30	DescType	Descriptor Type 0 - Packet descriptor 1 - New_Address message	
31	Valid	Valid bit. 0 - CPU TX High1 and 2 do not contain valid data 1 - CPU TX High1 and 2 contain valid data. Valid bit is cleared upon reading CPU Tx High Desc2.	

**18.1.2 CPU Tx Low Desc1 and 2, Offset: 0x28, 0x2C**

Tx Low Desc1 and Tx Low Desc2 registers have the same format as the CPU Tx High Desc registers. The New\_Address messages are always entered into the high descriptor, therefore the CPU Tx Low Desc2 register contains only Packet\_Descriptor messages (CPU Tx Low Desc2[30] = 0). . . . .

**Table 35: CPU EL Free Req, Offset: 0x30**

Bits	Field Name	Function	Initial Value
10:0	SourBufAddr	Source Buffer Address (divided by 0x600)	0x0
14:11	SourcePort	Source Port number	0x0
18:15	TrgPort	Target Port number	0x0
19	Mult	Multicast packet (1 - Multicast) <b>Note:</b> The value of this bit must match the value of the Mult bit (bit 26) in the CPU Tx High Desc1 (offset 0x20).	0x0
30:20	Reserved	Reserved	Reserved
31	Busy	Busy bit 0 - Not Busy, the CPU can re-load this register 1 - Busy - The data in this register was not processed. The CPU should not re-load this register. When the CPU writes this register to clear a buffer, it should write this bit as '1'. Before the CPU proceeds to clear another buffer, it should poll this bit until it has a value of '0'.	0x0

**Table 36: CPU Empty Buffer, Offset: 0x34**

Bits	Field Name	Function	Initial Value
10:0	BuffAddr	Buffer Address (divided by 0x600)	0x0
30:11	Reserved	Reserved	Reserved
31	Valid	Valid bit. 0 - The address (pointer) in 10:0 is NOT VALID. 1 - This register contains a valid address (pointer) to an empty buffer in 10:0.  This bit is set to '0' upon reading this register.	0x0

**Table 37: CPU Enqueue1, Offset: 0x38**

Bits	Field Name	Function	Initial Value
10:0	TrgBufAddr	Target Buffer Address (divided by 0x600)	0x0
21:11	Byte Count	Byte Count. When GenCrc is set, Byte Count does not include the 4 bytes of CRC	0x0
22	Mult	Multicast Packet (1 - Multicast, 0 - Unicast) <b>Note:</b> Should comply with the limitations set forth in Section 7.3 "Forwarding a Packet to the CPU" on page 33 with respect to re-enqueuing of packets.	0x0
23	Pri	Priority 0 - Low Priority 1 - High Priority	0x0
24	GenCrc	CRC generation 0 - Do not generate CRC on Transmit 1 - Generate CRC on Transmit  NOTE: Bit 23, CRCEn, of the Global Control Register (0x04) must be set to '1' in order to Generate CRC on Transmit.	0x0
31:25	Reserved	Reserved	0x0

**Table 38: CPU Enqueue2, Offset: 0x3C**

Bits	Field Name	Function	Initial Value
13:0	TgtPortBitMap	Target Port Bitmap (one bit per port, this field is ignored when queuing a Unicast packet)	0x0
15:14	Reserved	Reserved	0x0
19:16	TgtPort	Target Port Number (this field is ignored when queuing a Multicast packet)	0x0

**Table 38: CPU Enqueue2, Offset: 0x3C**

Bits	Field Name	Function	Initial Value
23:20	SrcPort	Source Port Number (0xe if source port is CPU)	0x0
30:24	Reserved	Reserved	0x0
31	Busy	<p>Busy bit</p> <p>0 - Not Busy, the CPU can re-load this register</p> <p>1 - Busy - The data in this register was not processed. The CPU should not re-load this register.</p> <p>When the CPU writes this register to queue a transmit packet, it should write this bit as '1'. Before the CPU proceeds to enqueue another buffer, it should poll this bit until it has a value of '0'.</p> <p>Note: The Busy bit applies to both CPU Enqueue Registers at offsets 0x38 and 0x3c.</p>	0x0

**Table 39: CPU New Address1, Offset: 0x40**

Bits	Field Name	Function	Initial Value
0	NaType	<p>New Address Type</p> <p>0 - CPU is writing a New Address to the table or is updating a current entry in the Address Table</p> <p>1 - The CPU is querying the Address Table for a particular entry. When querying, the only relevant fields are MAC[47:20] and MAC[19:0]. The rest of the fields in NewAddress1, NewAddress2 and NewAddress3 are ignored.</p>	0x0
1	Sk	<p>Skip Entry</p> <p>0 - No not remove this entry from the Address Table</p> <p>1 - Remove this entry from the Address Table</p>	0x0
2	Aging	<p>Aging</p> <p>0 - New address</p> <p>1 - Not new address</p>	0x0
3	Reserved	Reserved	0x0
31:4	MAC[47:20]	MAC[47:20]	0x0

**Table 40: CPU New Address2, Offset: 0x44**

Bits	Field Name	Function	Initial Value
19:0	MAC[19:0]	MAC[19:0]	0x0
23:20	PortNu	Port Number	0x0

Table 40: CPU New Address2, Offset: 0x44 (Continued)

Bits	Field Name	Function	Initial Value
24	Pd	Priority Destination 0 - Low 1 - High	0x0
25	Ps	Priority Source 0 - Low 1 - High	0x0
26	St	Static 0 - Learn address every time it comes in on different port 1 - Do not learn address every time it comes in on different port	0x0
27	Mult	Multiple (only valid if Static is '1') 0 - Do not send to multiple ports on static address 1 - Send on multiple ports to static address	0x0
28	Id	Intervention Destination 0 - Intervention on Destination Address not set 1 - Intervention on Destination Address is set	0x0
29	Is	Intervention Source 0 - Intervention on Source Address is not set 1 - Intervention on Source Address is set	0x0
31:30	Reserved	Reserved	0x0

Table 41: CPU New Address3, Offset: 0x48

Bits	Field Name	Function	Initial Value
14:0	Forw	Forwarding bits for ports [13:0] and the CPU (#14)	0x0
30:15	Reserved	Reserved	0x0
31	Busy	Busy bit 0 - Not Busy, the CPU can re-load this register 1 - Busy - The data in this register was not processed.  The CPU should not re-load this register. Busy bit is set to 1 by the CPU and cleared when the GT-482xx takes the data. Note: The Busy bit applies to all three CPU New Address Registers at offsets 0x40, 0x44, and 0x48.	0x0

Table 42: CPU Query, Offset: 0x4C

Bits	Field Name	Function	Initial Value
0	Reserved	Reserved	0x0



Table 42: CPU Query, Offset: 0x4C

Bits	Field Name	Function	Initial Value
1	EntryValid	Entry Valid - This bit indicates if the address was found: 0 - was not found 1 - was found	0x0
2	Pd	Priority Destination 0 - Low priority 1 - High priority	0x0
3	Ps	Priority Source 0 - Low priority 1 - High priority	0x0
7:4	SourcePort	Source Port Number	0x0
8	Aging	Aging 0 - new address 1 - not new address & will be cleared	0x0
9	St	Static 0 - Learn address every time it comes in on different port 1 - Do not learn address every time it comes in on different port	0x0
10	Mult	Multiple (only valid if Static is '1') 0 - Do not send to multiple ports on static address 1 - Send on multiple ports to static address	0x0
11	Id	Intervention Destination 0 - Intervention on Destination Address not set 1 - Intervention on Destination Address is set	0x0
12	Is	Intervention Source 0 - Intervention on Source Address not set 1 - Intervention on Source Address is set	0x0
27:13	Forw	Forwarding bits for ports [13:0] and the CPU (#14)	0x0
30:28	Reserved	Reserved	Reserved
31	Valid	Valid bit. 0 - CPU Query does not contain valid data 1 - CPU Query contains valid data. Valid bit is cleared upon reading this register	0x0

Table 43: SMI Register, Offset: 0x50

Bits	Field Name	Function	Initial Value
15:0	Data	<b>For SMI READ operation:</b> Two CPU transactions are required: (1) CPU write to the SMI register with OpCode = 1, PhyAd, RegAd with the Data being any value. (2) CPU read from the SMI register. When reading back the SMI register, the Data is the addressed Phy register contents if the ReadValid bit (#27) is 1. The Data remains undefined as long as ReadValid is 0. <b>For SMI WRITE operation:</b> One CPU transaction is required: CPU write to the SMI register with OpCode = 0, PhyAd, RegAd with the Data to be written to the addressed Phy register.	N/A
20:16	PhyAd	PHY device address	0x0
25:21	RegAd	PHY device register address	0x0
26	OpCode	0 - Write 1 - Read	0x0
27	ReadValid	1 - indicates that the Read operation has been completed for the addressed RegAd register, and the data is valid on the Data field.	0x0
28	Busy	Busy bit 0 - Not Busy, the CPU can re-load this register 1 - Busy - The data in this register was not processed. The CPU should not re-load this register. Busy bit is set to 1 by the CPU and cleared when the data was written to the PHY registers.	0x0
31:29	N/A	This bits should be driven 0x0 during any write to the SMI register.	0x0

Table 44: 802.1Q Ethertype Register, Offset: 0x54

Bits	Field Name	Function	Initial Value
15:0	VLEtherType	VL EtherType value	0x0 - P0/P1/P2 0x8100 - P3/P4 <sup>1</sup>

1. Note: For compliance with IEEE standards 802.1P and 802.1Q this register should be programmed to 0x8100 for P3/P4.

Table 45: General Purpose Register1, Offset: 0x58

Bits	Field Name	Function	Initial Value
11:0	GP	General Purpose value - These pins are used to sample or to drive the ColE/GP[11:0] pins in 10Base-T or 10Base-FL modes.	0x0

**Table 46: General Purpose Register2, Offset: 0x5C**

Bits	Field Name	Function	Initial Value
11:0	GpDir	General Purpose direction. These bits determine the direction of the ColE/GP[11:0] pins (per bit basis) as follows: 1 - ColE/GP[11:0] are inputs. GP bits reflect the value on these pins. 0 - ColE/GP[11:0] are output. They carry the value of GP bits.	0xFFFF

**Table 47: Rx\_10 Threshold, Offset: 0x60**

Bits	Field Name	Function	Initial Value
7:0	RxBufThr	Receive Buffer Threshold for the 10Mbps ports. The threshold is: RxBufThr * 8.	0x4 @1M (32) 0x8 @4M (64) Earlier versions: 0x2 @1M (16) For GT-48212-P5: 0x3 (32)
15:8	XOffLimit	X-OFF Limit - When Flow control is enabled, the GT-482xx will send X-OFF packet when the number of buffers for this port reach (XOffLimit * 4) value	0x7 @1M (28) 0x8 @4M (32)
23:16	XOnLimit	X-ON Limit - When Flow control is enabled, the GT-482xx will send X-ON packet when the number of buffers for this port going below (XOnLimit * 4) value	0x6 @1M (24) 0x6 @4M (24)
31:24	HolLimit	HOL Limit - the GT-482xx discards packet when the Tx queue exceeds TxThr limit and the number of buffers allocated to this port exceed (HolLimit * 8)	0x3 @1M (24) 0x3 @4M (24)

**Table 48: Rx\_100 Threshold, Offset: 0x64**

Bits	Field Name	Function	Initial Value
7:0	RxBufThr	Receive Buffer Threshold for the 100Mbps ports. The threshold is: RxBufThr * 8.	0x8 @1M (64) 0x4A @4M (592) Earlier versions: 0x12 @1M (144) For GT-48212-P5: 0x8 (64)
15:8	XOffLimit	X-OFF Limit - When Flow control is enabled, the GT-482xx will send X-OFF packet when the number of buffers for this port reach (XOffLimit * 8) value	0x3 @1M (24) 0x46 @4M (560)

Table 48: Rx\_100 Threshold, Offset: 0x64 (Continued)

Bits	Field Name	Function	Initial Value
23:16	XOnLimit	X-ON Limit - When Flow control is enabled, the GT-482xx will send X-ON packet when the number of buffers for this port going below (XOnLimit * 8) value	0x3 @1M (24) 0x43 @4M (536)
31:24	HolLimit	HOL Limit - the GT-482xx discards packet when the Tx queue exceeds TxThr limit and the number of buffers allocated to this port exceed (HolLimit * 8)	0x7 @1M (56) 0x43 @4M (536)

Table 49: CPU Threshold, Offset: 0x68

Bits	Field Name	Function	Initial Value
7:0	CpuBufThr	Buffer Threshold for the CPU. The threshold is: CpuBufThr * 8.	0x2 @1M 0x8 @4M

Table 50: LED Override, Offset: 0x6C

Bits	Field Name	Function	Initial Value
13:0	LedData	LED Data bits	0x0
14	OverRide	Override - This bit selects the data to be driven on the Led-Data pin in LedMode1 0 - Bits #0 to #27 of LedData contain link_test_pass indication 1 - Bits #0 and #1 contain LedData[0] Bits #2 and #3 contain LedData[1] * * Bits #26 and #27 contain LedData[13]	0x0

Table 51: Flow Control Source Address Low, Offset: 0x70

Bits	Field Name	Function	Initial Value
31:0	SA[16:47]	Source Address - The most significant bits of the source address for all ports. This address is being used for Flow Control. Note: SA[47] is the I/G bit. The first bit to be received from the wire.	0x0

**Table 52: Flow Control Source Address High, Offset: 0x74**

Bits	Field Name	Function	Initial Value
11:0	SA[4:15]	Source Address - The most significant bits of the source address for all ports. This address is being used for Flow Control.	0x0

**Table 53: CPU Time Out Register, Offset: 0x7C**

Bits	Field Name	Function	Initial Value
15:0	CpuTimeOut	Cpu Time Out - These bits specify the initial value in Ack cycles of the CPU to DRAM access timer	0x0
16	TimeOutEn	TimeOut En - Enable the CPU to DRAM access timer. 0 - disable 1 - enable	0x0

**Table 54: DRAM Configuration, Offset: 0x1448**

Bits	Field Name	Function	Initial Value
13:0	RefCntInt	Refresh Counter Initial Value	0x200 (10 micro seconds @50Mhz)

**Table 55: DRAM Parameters, Offset: 0x144C**

Bits	Field Name	Function	Initial Value
1:0	CasLat	Cas Latency 00 - Reserved 01 - 2 cycles 10 - 3 cycles 11 - Reserved	0x01
3	RasPreTime	RAS Precharge Time 0 - 1 clocks 1 - 2 clocks	0x0
10	RasToCas	RAS to CAS cycles 0 - 2 clocks 1 - 3 clocks	0x0
12	BurstLimit	Burst Limit 0 - 8 words 1 - 4 words	0x0

Table 56: SDRAM Operation, Offset: 0x1474

Bits	Field Name	Function	Initial Value
2:0	SdramOp	SDRAM Operation. These bits select the SDRAM operation as follows: 000 - Normal DRAM mode 001 - NOP Command 010 - All banks precharge command 011 - Mode register command enable 100 - CBR cycle enable	0x0

Table 57: Address Decode, Offset: 0x147C

Bits	Field Name	Function	Initial Value
0		DRAM Size 0 - 1Mbyte 1 - 4Mbyte	DQM pin at reset

Table 58: Port Control (10M ports), Offset: 0x400-0x40C, 0x800 - 0x80C, 0xC00 - 0xC0C

Bits	Field Name	Function	Initial Value
0	PortEn	Port Enable. 0 - Port is disabled 1 - Port is enabled	0x1
1	FullDx	Half/full duplex. 0 - Port works in half-duplex mode 1 - Port works in full-duplex mode	TxDE (at RESET)
3:2	SerMode	Serial Mode. These bits indicate the serial mode. The logic order at reset is DData[1:0] for port 0, DData[3:2] for port 1, where port $n$ corresponds to DData [ $2n+1:2n$ ] . 00 - 10Base-T 01 - 10Base-FL 10 - AUI 11 - Reserved	Corresponding DData[] pins at RESET (see Table 16, "RESET Pin Strapping Options," on page 66)
4	LockPort	Locked Port 0 - port is in normal mode 1 - port is locked. The GT-482xx does not learn new addresses and send then ONLY to the CPU	0x0
5	ForBroadCPU	Forward Broadcast to the CPU. Meaningful only when Fil-Broad (bit 10) is clear. 0 - Forward Broadcast packets to all ports 1 - Forward Broadcast packets only to the CPU Note: For exact description see "Address Recognition" on page 28	0x0

**Table 58: Port Control (10M ports), Offset: 0x400-0x40C, 0x800 - 0x80C, 0xC00 - 0xC0C**  
**(Continued)**

Bits	Field Name	Function	Initial Value
6	BackPressureEn	Back Pressure Enable. Only valid when FullDx (bit 1) is set to '0'. 0 - disable 1 - enable	DAddr[11] at reset
7	FlowControlEn	Flow Control Enable. Only valid when FullDx (bit 1) is set to '1'. 0 - disable 1 - enable	DData[28] at reset
8	EaseEn	Enable EASE on specific port. 0 -enable 1- disable	0x0
9	Prio	Priority 0 - low priority 1 - high priority	0x0
10	FilBroad	Filter Broadcast. 0 - Broadcast packets are forwarded to all ports. 1 - the GT-482xx discards Broadcast packets. Note: For exact description see "Address Recognition" on page 28	0x0
11	ForwUnk	Enable Forwarding Unknown Unicast DA Packets to this transmit port. 0 - Unknown packets are forwarded. 1 - the GT-482xx does not forward Unknown packets to this port. Note: For exact description see "Address Recognition" on page 28	0x0
12	SpanEn	Spanning Tree Enable. Meaningful only when SpanEn bit in the Global Control register is set. 0 - All packets are accepted. 1 - the GT-482xx discards all incoming/outgoing packets except for BPDU packets.	0x0
13	VTagEn	Virtual LAN Tagging Packet Extension Enable 0 - Accept packets up to 1518 bytes in length 1 - Accept packets up to 1536 bytes in length	DAddr[8] at reset
14	ForcePri	Force Priority. 0 - Priority is set base on the Pd, Ps, Pri bit in the port control register or the 802.1Q quality of service feild. 1 - Priority is set ONLY based on Pri bit in the port control register	0x0
18:15	SA[0:3]	Source Address - The least significant bits of the source address for all ports. This address is being used for Flow Control.	port #

**Table 58: Port Control (10M ports), Offset: 0x400-0x40C, 0x800 - 0x80C, 0xC00 - 0xC0C**  
(Continued)

Bits	Field Name	Function	Initial Value
31:19	Reserved		

**Table 59: Port Control 12 (100M ports), Offset: 0x1000**

Bits	Field Name	Function	Initial Value
0	PortEn	Port Enable. 0 - Port is disabled 1 - Port is enabled	0x1
1	FullDx	Half/full duplex. 0 - Port works in half-duplex mode 1 - Port works in full-duplex mode	DAddr[24] for port 12 or DAddr[25] for port 13 at RESET
2	EaseEn Port12	Enable Ease on Port 12 0 - disabled 1 - enabled	0x0
3	EaseEn Port13	Enable Ease on Port 13 0 - disabled 1 - enabled	0x0
4	LockPort	Locked Port 0 - port is in normal mode 1 - port is locked. The GT-482xx does not learn new addresses and send then ONLY to the CPU	0x0
5	ForBroadCPU	Forward Broadcast to the CPU. Meaningful only when FilBroad is clear. 0 - Forward Broadcast packets to all ports 1 - Forward Broadcast packets only to the CPU Note: For exact description see "Address Recognition" on page 28	0x0
6	BackPressureEn	Back Pressure Enable 0 - disable 1 - enable	DAddr[11] at RESET
7	FlowControlEn	Flow Control Enable 0 - disable 1 - enable	DData[30] for Port 13, DData[29] for Port 12 at RESET



**Table 59: Port Control 12 (100M ports), Offset: 0x1000 (Continued)**

Bits	Field Name	Function	Initial Value
8	AutoNeg	Auto-Negotiation Enable 0 - Auto-Negotiation is disable 1 - Auto-Negotiation is enable	DData[27] for Port 13, DData[26] for Port 12 at RESET
9	Pri	Priority 0 - low priority. 1 - high priority	0x0
10	FilBroad	Filter Broadcast. 0 - Broadcast packets are forwarded to all ports. 1 - The the GT-482xx discards Broadcast packets. Note: For exact description see "Address Recognition" on page 28	0x0
11	ForwUnk	Enable Forwarding Unknown Unicast DA Packets to this transmit port. 0 - Unknown packets are forwarded. 1 - the GT-482xx does not forward Unknown packets to this port. Note: For exact description see "Address Recognition" on page 28	0x0
12	SpanEn	Spanning Tree Enable. Meaningful only when SpanEn bit in the Global Control register is set. 0 - All packets are accepted. 1 - The the GT-482xx discards all incoming/outgoing packets except for BPDU packets.	0x0
13	VTagEn	Virtual LAN Tagging Packet Extension Enable 0 - Accept packets up to 1518 bytes in length 1 - Accept packets up to 1536 bytes in length	DAddr[8] at reset
14	ForcePri	Force Priority. 0 - Priority is set base on the Pd, Ps, Pri bit in the port control register or the 802.1Q quality of service feild. 1 - Priority is set based on Pri bit in the port control register	0x0
18:15	SA[0:3]	Source Address - The least significant bits of the source address for all ports. This address is being used for Flow Control.	port #
31:19	reserved		

Table 60: Port Control 13 (100M ports), Offset: 0x1004

Bits	Field Name	Function	Initial Value
0	PortEn	Port Enable. 0 - Port is disabled 1 - Port is enabled	0x1
1	FullDx	Half/full duplex. 0 - Port works in half-duplex mode 1 - Port works in full-duplex mode	DAddr[24] for port 12 or DAddr[25] for port 13 at RESET
3:2	Port13 Expansion Priority	bit [2] Rx_Prio: 0 - disabled 1 - enabled bit [3] Tx_Prio: 0 - disabled 1 - enabled	Both bits value are sampled at reset as per the INVERTED value of Burst-Addr[1] configuration input
4	LockPort	Locked Port 0 - port is in normal mode 1 - port is locked. The GT-482xx does not learn new addresses and send then ONLY to the CPU	0x0
5	ForBroadCPU	Forward Broadcast to the CPU. Meaningful only when FilBroad is clear. 0 - Forward Broadcast packets to all ports 1 - Forward Broadcast packets only to the CPU Note: For exact description see "Address Recognition" on page 28	0x0
6	BackPressureEn	Back Pressure Enable 0 - disable 1 - enable	DAddr[11] at RESET
7	FlowControlEn	Flow Control Enable 0 - disable 1 - enable	DData[30] for Port 13, DData[29] for Port 12 at RESET
8	AutoNeg	Auto-Negotiation Enable 0 - Auto-Negotiation is disable 1 - Auto-Negotiation is enable	DData[27] for Port 13, DData[26] for Port 12 at RESET
9	Pri	Priority 0 - low priority. 1 - high priority	0x0

**Table 60: Port Control 13 (100M ports), Offset: 0x1004 (Continued)**

Bits	Field Name	Function	Initial Value
10	FilBroad	Filter Broadcast. 0 - Broadcast packets are forwarded to all ports. 1 - The the GT-482xx discards Broadcast packets. Note: For exact description see "Address Recognition" on page 28	0x0
11	ForwUnk	Enable Forwarding Unknown Unicast DA Packets to this transmit port. 0 - Unknown packets are forwarded. 1 - the GT-482xx does not forward Unknown packets to this port. Note: For exact description see "Address Recognition" on page 28	0x0
12	SpanEn	Spanning Tree Enable. Meaningful only when SpanEn bit in the Global Control register is set. 0 - All packets are accepted. 1 - The the GT-482xx discards all incoming/outgoing packets except for BPDU packets.	0x0
13	VTagEn	Virtual LAN Tagging Packet Extension Enable 0 - Accept packets up to 1518 bytes in length 1 - Accept packets up to 1536 bytes in length	DAddr[8] at reset
14	ForcePri	Force Priority. 0 - Priority is set base on the Pd, Ps, Pri bit in the port control register or the 802.1Q quality of service feild. 1 - Priority is set based on Pri bit in the port control register	0x0
18:15	SA[0:3]	Source Address - The least significant bits of the source address for all ports. This address is being used for Flow Control.	port #
31:19	Reserved		

**Table 61: EASE Register, Offset: 0x410-0x41C, 0x810-0x81C, 0xC10-0xC1C, 0x1008-0x100C**

Bits	Field Name	Function	Initial Value
19:0	Ease_register	Value loaded to the internal count-down counter of the ports	0x0
31:20	Reserved	Reserved	Reserved

**18.2 Port MIB Counters (14 Blocks), Offset (start): 0x600, 0xA00, 0xE00, 0x1200**

The CPU must read all of the MIB counters during initialization in order to reset the counters to '0'. All counters are 32-bits. The CPU must access the counters using single datum transactions (burst reads/writes are not allowed).

MIB counters can be cleared by reading, or left intact after a read based on the MIB Clear Mode (bit 21 in the Global Command Register.) During initialization, the CPU must read all of the MIB counters in order to reset the counters to '0'. The counters will only be reset to '0' if MIBClrMode (bit 21 of the Global Control Register) is set to '0' (default). If MIBClrMode bit is '1', reading the MIB counters will have no effect.

Table 62 lists the definitions for terms used in the counter descriptions.

**Table 62: Definitions Used in Counter Descriptions**

Term	Definition
Packet Data Section	All data bytes in the packet following the SFD until the end of the packet
Packet Data Length	The number of data bytes in the packet data section
Data Octet	A single byte from the packet data section
Nibble	4 bits (half byte) of a data octet
Misaligned Packet	A packet with an odd number of nibbles
Received Good Packet	A received packet which is not rejected and enters the switching core to be transmitted later
Transmitted Good Packet	Any transmitted packet from the GT-482xx
Collision Event	A collision has been detected until than 576 bit times into the transmitted packet after TxEn.
Late Collision Event	A collision has been detected later than 576 bit times into the transmitted packet after TxEn.
Rx Error Event	The input RX_ERR has been asserted
Dropped Packet	A received packet which is ignored due to lack of available receive buffers (port is in buffer_full state)
Local Packet	A received packet whose destination address is mapped to the receiving port
Rejected Packet	A received packet which is not forwarded due to error such as bad CRC, Rx Error Event, Invalid size (too short or too long).
MIBCtrMode	Bit 26, MIBCtrMode, of the Global Control Register (offset 0x140028)
VTagEn	Bit 9, VTagEn, of the Port Control Register (offset 0x040200-0x040204)
MAXFRAMESIZE	1518 for VTagEn = 0 (default) or 1536 for VTagEn = 1

Table 63: Port MIB Counters

Address for Port 0	Counter Name	Function	Initial Value
0x000600	Bytes Received	MIBCtrMode = 0: This counter is incremented once for every data octet of good packets (Unicast + Multicast + Broadcast) received. MIBCtrMode = 1: This counter is incremented once for every data octet of good packets (Unicast + Multicast + Broadcast packets) and for LOCAL and DROPPED packets.	-
0x000604	Bytes Sent	This counter is incremented once for every data octet of a transmitted good packet.	-
0x000608	Frames Received	MIBCtrMode = 0: This counter is incremented once for every good packet (Unicast + Multicast + Broadcast) received. MIBCtrMode = 1: This counter is incremented once for every good packet (Unicast + Multicast + Broadcast packets) and for LOCAL and DROPPED packets received.	-
0x00060c	Frames Sent	This counter is incremented once for every transmitted good packet.	-
0x000610	Total Bytes Received	This counter is incremented once for every data octet of all received packets. This includes data octets of rejected and local packets which are NOT forwarded to the switching core for transmission. This counter should reflect all the data octets received on the line. NOTE: A nibble is NOT counted as a whole byte.	-
0x000614	Total Frames Received	This counter is incremented once for every received packets. This includes rejected and local packets which are NOT forwarded to the switching core for transmission. This counter should reflect all packets received on the line.	-
0x000618	Broadcast Frames Received	MIBCtrMode = 0: This counter is incremented once for every good Broadcast packet received. MIBCtrMode = 1: This counter is incremented once for every good Broadcast packet received and for LOCAL and DROPPED Broadcast packets.	-
0x00061c	Multicast Frames Received	MIBCtrMode = 0: This counter is incremented once for every good Multicast packet received. MIBCtrMode = 1: This counter is incremented once for every good Multicast packet received and for LOCAL and DROPPED Broadcast packets. This counter does not include Broadcast packets.	-

Table 63: Port MIB Counters (Continued)

Address for Port 0	Counter Name	Function	Initial Value
0x000620	CRC Error	This counter is incremented once for every received packet which meets all the following conditions (i.e. logical AND of the following conditions): 1. Packet data length is between 64 and MAXFRAMESIZE bytes inclusive (i.e. valid packet data length per IEEE std). 2. Packet has invalid CRC (non-A also counted packets with odd number of nibbles). 3. Collision Event has not been detected. 4. Late Collision Event has not been detected. 5. Rx Error Event has not been detected.	-
0x000624	Oversize Frames	This counter is incremented once for every received packet which meets all the following conditions (i.e. logical AND of the following conditions): 1. Packet data length is greater than and MAXFRAMESIZE. 2. Packet has valid CRC. 3. Rx Error Event has not been detected.	-
0x000628	Fragments	This counter is incremented once for every received packet which meets all the following conditions (i.e. logical AND of the following conditions): 1. Packet data length is less than 64 bytes -OR- packet without SFD and is less than 64 bytes in length. 2. Collision Event has not been detected. 3. Late Collision Event has not been detected. 4. Rx Error Event has not been detected.	-
0x00062c	Jabber	This counter is incremented once for every received packet which meets all the following conditions (i.e. logical AND of the following conditions): 1. Packet data length is greater than MAXFRAMESIZE. 2. Packet has invalid CRC. 3. Rx Error Event has not been detected.	-
0x000630	Collision	This counter is incremented once for every received packet which meets both of the following conditions (i.e. logical AND of the following conditions): 1. Collision Event has been detected. 2. Rx Error Event has not been detected.	-
0x000634	Late Collision	This counter is incremented once for every received packet which meets both of the following conditions (i.e. logical AND of the following conditions): 1. Late Collision Event has been detected. 2. Rx Error Event has not been detected.	-
0x000638	Frames 64 Bytes	This counter is incremented once for every received and transmitted packet with size of 64 bytes. This counter includes local, dropped and transmitted packets.	-

Table 63: Port MIB Counters (Continued)

Address for Port 0	Counter Name	Function	Initial Value
0x00063c	Frames 65-127 Bytes	This counter is incremented once for every received and transmitted packet with size of 65 to 127 bytes. This counter includes local, dropped and transmitted packets.	-
0x000640	Frames 128-255 Bytes	This counter is incremented once for every received and transmitted packet with size of 128 to 255 bytes. This counter includes local, dropped and transmitted packets.	-
0x000644	Frames 256-511 Bytes	This counter is incremented once for every received and transmitted packet with size of 256-511 bytes. This counter includes local, dropped and transmitted packets.	-
0x000648	Frames 512-1023 Bytes	This counter is incremented once for every received and transmitted packet with size of 512-1023 bytes. This counter includes local, dropped and transmitted packets.	-
0x00064c	Frames 1024-1522 Bytes	This counter is incremented once for every received and transmitted packet with size of 1024 to MAXFRAMESIZE bytes. This counter includes local, dropped and transmitted packets.	-
0x000650	MAC Rx Error	This counter is incremented once for every received and transmitted packet where the Rx Error Event has been detected. The counters that will not be incremented when this counter is incremented include: CRC Error, Oversize Frames, Fragments, Jabber, Collision and Late Collision.  This counter is RESERVED for ports 0 thru 11.	-
0x000654	Dropped Frames	This counter is incremented once for every received packet that is dropped.	-

## 19. GT-482XX PINOUT DIFFERENCES

Currently all members of the Galaxy Switch Family are “supersocket compatible”. This means that a properly designed system can accept a GT-48207 (unmanaged 8+2), a GT-48208 (managed 8+2) or a GT-48212 (managed 12+2).

This section describes how to design such a system.

### 19.1 Pinout Differences between GT-48207, GT-48208, and GT-48212 Devices

The pinout differences between the three Galaxy devices are summarized in Table 64.

**Table 64: Pinout Differences**

Galaxy Device	Functions not implemented	Pins Deleted	Note
GT-48212	N/A	N/A	Baseline pinout
GT-48208	Ethernet ports 1, 5, 7 and 11	All pins relating to these ports	For a GT-48212 to be used in a GT-48208 socket you must properly disable the four unused Ethernet ports as described below.
GT-48207	Ethernet ports 1, 5, 7 and 11  Management CPU interface	All pins relating to the these Ethernet ports and the CPU interface	For a GT-48212 to be used in a GT-48207 socket you must properly disable the four unused Ethernet ports and disable the CPU interface.

### 19.2 Using a GT-48212 in a GT-48208/7 Socket: Disabling Unused Ethernet Ports

The GT-48208 and GT-48207 do not implement Ethernet port numbers 1, 5, 7 and 11 that are found on the GT-48212. In GT-48208/7 silicon these ports do not exist so no special “tying off” is required. However, if you need to use a GT-48212 in a GT-48208/7 socket (using only 8 of the 12 ports) then you **MUST** properly disable these ports. Simply following the attached pinout for the GT-48208/7 will disable these ports if a GT-48212 is placed in the same socket. The pins that are used for receive data for the ports not implemented on the GT-48207/8 (RxdE1,5,7,11) are shown as Vcc (pins 29, 31, 46, and 50). This disables these ports and prevents accidental clocking when using a GT-48212 in a GT-48208/7 design.

If you wish to build a system that can implement either 12 or 8 Ethernet ports in the same socket, be sure to pull these pins high through a resistor or jumper. That way you can make it a “stuffing option” to disable the 4 additional ports or to implement them when a GT-48212 is installed.

**Note:** The attached pinout for the GT-48208/7 is updated from previous documentation to reflect the “tying off” of RxdE1/5/7/11. This is the final pinout.

### 19.3 Using a GT-48212 or GT-48208 in a GT-48207 Socket: Disabling Unused CPU Interface

A similar situation exists with regards to the CPU interface when using a GT-48212/08 in a GT-48207 socket. Since the GT-48207 does not implement this interface, if you wish to use a GT-48208/212 in a GT-48207 socket then you **MUST** properly disable the CPU interface. Simply following the attached pinout for the GT-48207 will disable the CPU interface if a GT-48208/212 is placed in the GT-48207 socket.



If you wish to build a system that can accept either managed or unmanaged Galaxy devices in the same socket, be sure to pull the CPU interface pins high through a resistor or jumper. That way you can make it a “stuffing option” to disable the CPU interface or to implement it when a GT-48208/212 is installed.

#### 19.4 CClk in an Unmanaged System

The CPU clock (CClk) must have a clock source applied in any Galaxy system even when no CPU is present (this is also true for the GT-48207). This ensures a proper RESET condition throughout the device. In an unmanaged system, connect CClk to either the 25MHz 10/100 PHY clock or to the 20MHz 10Mbps PHY clock.

NOTE: Do not tie ACIk to CClk, this will result in synchronization failures. Tying ACIk to CClk was recommended in previous documentation, however, this is incorrect information.

## 20. GT-482XX PINOUT TABLES, 208-PQFP

Table 65: GT-48212 Pinout Table (Sorted by Pin Number)

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
1	LedClk	39	TxdE3	77	DData[31]	115	Vss
2	LedStb	40	RxdE9	78	DData[30]	116	Vcc
3	LedData	41	TxdE9	79	DData[29]	117	DAddr[7]
4	ColE0	42	RxdE4	80	DData[28]	118	DAddr[8]
5	Vcc	43	TxdE4	81	DData[27]	119	MDIO
6	Vss	44	RxdE10	82	DData[26]	120	MDC
7	ColE1	45	TxdE10	83	DData[25]	121	Crs0
8	ColE2	46	RxdE5	84	DData[24]	122	Col0
9	ColE3	47	Vss	85	Vss	123	Txd0[3]
10	ColE4	48	TxdE5	86	Vcc	124	Txd0[2]
11	ColE5	49	Vcc	87	DData[23]	125	Txd0[1]
12	ColE6	50	RxdE11	88	DData[22]	126	Txd0[0]
13	ColE7	51	TxdE11	89	DData[21]	127	TxEn0
14	ColE8	52	Scan*	90	DData[20]	128	RxDV0
15	Vss	53	Tristate*	91	DData[19]	129	RxEr0
16	SClk	54	EnDev*	92	Vcc	130	Vcc
17	Vcc	55	Vcc	93	Vss	131	TxClk0
18	ColE9	56	Vss	94	DData[18]	132	Vss
19	ColE10	57	Limit4*	95	DData[17]	133	RxClk0
20	ColE11	58	DData[8]	96	Vss	134	Rxd0[3]
21	SerLinkStat	59	DData[9]	97	DData[16]	135	Rxd0[2]
22	LinkStat0	60	DData[10]	98	Dqm	136	Rxd0[1]
23	TxEnE0	61	DData[11]	99	We*	137	Rxd0[0]
24	RxdE0	62	DData[12]	100	Cas*	138	Crs1
25	TxdE0	63	DData[13]	101	Ras*	139	Col1
26	RxdE6	64	DData[14]	102	Cs*	140	Txd1[0]
27	TxdE6	65	Vss	103	DAddr[11]	141	Txd1[1]
28	TxdE1	66	DData[15]	104	DAddr[10]	142	Txd1[2]
29	RxdE1	67	DData[7]	105	Vss	143	Txd1[3]
30	Vss	68	DData[6]	106	Vcc	144	TxEn1
31	RxdE7	69	DData[5]	107	DAddr[9]	145	RxDV1
32	TxdE7	70	DData[4]	108	DAddr[0]	146	RxEr1

Table 65: GT-48212 Pinout Table (Sorted by Pin Number) (Continued)

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
33	RxdE2	71	DData[3]	109	DAddr[1]	147	Vss
34	AClk	72	DData[2]	110	DAddr[2]	148	TxCk1
35	TxdE2	73	DData[1]	111	DAddr[3]	149	Vcc
36	RxdE8	74	Vss	112	DAddr[4]	150	RxCk1
37	TxdE8	75	Vcc	113	DAddr[5]	151	Rxd1[0]
38	RxdE3	76	DData[0]	114	DAddr[6]	152	Rxd1[1]
153	Rxd1[2]	167	AD[6]	181	AD[14]	195	AD[26]
154	Rxd1[3]	168	Vcc	182	AD[15]	196	AD[27]
155	Rst*	169	Vss	183	AD[16]	197	AD[28]
156	RdCEn*	170	AD[7]	184	AD[17]	198	AD[29]
157	BurstAddr[1]	171	Vss	185	AD[18]	199	Vss
158	BurstAddr[2]	172	AD[8]	186	AD[19]	200	Blast*
159	AD[0]	173	AD[9]	187	AD[20]	201	AD[30]
160	AD[1]	174	AD[10]	188	AD[21]	202	AD[31]
161	Vss	175	AD[11]	189	Vss	203	W/R*
162	Vcc	176	AD[12]	190	Vcc	204	Ads*
163	AD[2]	177	AD[13]	191	AD[22]	205	Vcc
164	AD[3]	178	Vss	192	AD[23]	206	Vss
165	AD[4]	179	Cclk	193	AD[24]	207	Ready*
166	AD[5]	180	Vcc	194	AD[25]	208	Int*

Table 66: GT-48208 Pinout Table (Sorted by Pin Number)

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
1	LedClk	39	TxdE3	77	DData[31]	115	Vss
2	LedStb	40	RxdE9	78	DData[30]	116	Vcc
3	LedData	41	TxdE9	79	DData[29]	117	DAddr[7]
4	ColE0	42	RxdE4	80	DData[28]	118	DAddr[8]
5	Vcc	43	TxdE4	81	DData[27]	119	MDIO
6	Vss	44	RxdE10	82	DData[26]	120	MDC
7	ColE1	45	TxdE10	83	DData[25]	121	Crs0
8	ColE2	46	Vcc <sup>2</sup>	84	DData[24]	122	Col0
9	ColE3	47	Vss	85	Vss	123	Txd0[3]

Table 66: GT-48208 Pinout Table (Sorted by Pin Number) (Continued)

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
10	ColE4	48	NC	86	Vcc	124	Txd0[2]
11	ColE5	49	Vcc	87	DData[23]	125	Txd0[1]
12	ColE6	50	Vcc <sup>2</sup>	88	DData[22]	126	Txd0[0]
13	ColE7	51	NC	89	DData[21]	127	TxEn0
14	ColE8	52	Scan*	90	DData[20]	128	RxDV0
15	Vss	53	Tristate*	91	DData[19]	129	RxEr0
16	Sclk	54	EnDev*	92	Vcc	130	Vcc
17	Vcc	55	Vcc	93	Vss	131	TxCk0
18	ColE9	56	Vss	94	DData[18]	132	Vss
19	ColE10	57	Limit4*	95	DData[17]	133	RxCk0
20	ColE11	58	DData[8]	96	Vss	134	Rxd0[3]
21	SerLinkStat	59	DData[9]	97	DData[16]	135	Rxd0[2]
22	LinkStat0	60	DData[10]	98	Dqm	136	Rxd0[1]
23	TxEnE0	61	DData[11]	99	We*	137	Rxd0[0]
24	RxdE0	62	DData[12]	100	Cas*	138	Crs1
25	TxdE0	63	DData[13]	101	Ras*	139	Col1
26	RxdE6	64	DData[14]	102	Cs*	140	Txd1[0]
27	TxdE6	65	Vss	103	DAddr[11]	141	Txd1[1]
28	NC <sup>1</sup>	66	DData[15]	104	DAddr[10]	142	Txd1[2]
29	Vcc <sup>2</sup>	67	DData[7]	105	Vss	143	Txd1[3]
30	Vss	68	DData[6]	106	Vcc	144	TxEn1
31	Vcc <sup>2</sup>	69	DData[5]	107	DAddr[9]	145	RxDV1
32	NC	70	DData[4]	108	DAddr[0]	146	RxEr1
33	RxdE2	71	DData[3]	109	DAddr[1]	147	Vss
34	Aclk	72	DData[2]	110	DAddr[2]	148	TxCk1
35	TxdE2	73	DData[1]	111	DAddr[3]	149	Vcc
36	RxdE8	74	Vss	112	DAddr[4]	150	RxCk1
37	TxdE8	75	Vcc	113	DAddr[5]	151	Rxd1[0]
38	RxdE3	76	DData[0]	114	DAddr[6]	152	Rxd1[1]
153	Rxd1[2]	167	AD[6]	181	AD[14]	195	AD[26]
154	Rxd1[3]	168	Vcc	182	AD[15]	196	AD[27]
155	Rst*	169	Vss	183	AD[16]	197	AD[28]
156	RdCEn*	170	AD[7]	184	AD[17]	198	AD[29]
157	BurstAddr[1]	171	Vss	185	AD[18]	199	Vss

**Table 66: GT-48208 Pinout Table (Sorted by Pin Number) (Continued)**

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
158	BurstAddr[2]	172	AD[8]	186	AD[19]	200	Blast*
159	AD[0]	173	AD[9]	187	AD[20]	201	AD[30]
160	AD[1]	174	AD[10]	188	AD[21]	202	AD[31]
161	Vss	175	AD[11]	189	Vss	203	W/R*
162	Vcc	176	AD[12]	190	Vcc	204	Ads*
163	AD[2]	177	AD[13]	191	AD[22]	205	Vcc
164	AD[3]	178	Vss	192	AD[23]	206	Vss
165	AD[4]	179	Cclk	193	AD[24]	207	Ready*
166	AD[5]	180	Vcc	194	AD[25]	208	Int*

1. NC = No connect. Do not connect this pin.

2. If users only want to design the board for the GT-48208, these pins can be tied directly to Vcc. If users want to design the board for both the GT-48212 and the GT-48208, these pins should be pulled up to Vcc through a resistor. Galileo recommends using a 4.7KOhm resistor value.

**Table 67: GT-48207 Pinout Table (Sorted by Pin Number)**

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
1	LedClk	39	TxdE3	77	DData[31]	115	Vss
2	LedStb	40	RxdE9	78	DData[30]	116	Vcc
3	LedData	41	TxdE9	79	DData[29]	117	DAddr[7]
4	ColE0	42	RxdE4	80	DData[28]	118	DAddr[8]
5	Vcc	43	TxdE4	81	DData[27]	119	MDIO
6	Vss	44	RxdE10	82	DData[26]	120	MDC
7	ColE1	45	TxdE10	83	DData[25]	121	Crs0
8	ColE2	46	Vcc <sup>2</sup>	84	DData[24]	122	Col0
9	ColE3	47	Vss	85	Vss	123	Txd0[3]
10	ColE4	48	NC	86	Vcc	124	Txd0[2]
11	ColE5	49	Vcc	87	DData[23]	125	Txd0[1]
12	ColE6	50	Vcc <sup>2</sup>	88	DData[22]	126	Txd0[0]
13	ColE7	51	NC	89	DData[21]	127	TxEn0
14	ColE8	52	Scan*	90	DData[20]	128	RxDV0
15	Vss	53	Tristate*	91	DData[19]	129	RxEr0
16	Sclk	54	EnDev*	92	Vcc	130	Vcc
17	Vcc	55	Vcc	93	Vss	131	TxCk0
18	ColE9	56	Vss	94	DData[18]	132	Vss

Table 67: GT-48207 Pinout Table (Sorted by Pin Number) (Continued)

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
19	ColE10	57	Limit4*	95	DData[17]	133	RxCk0
20	ColE11	58	DData[8]	96	Vss	134	Rxd0[3]
21	SerLinkStat	59	DData[9]	97	DData[16]	135	Rxd0[2]
22	LinkStat0	60	DData[10]	98	Dqm	136	Rxd0[1]
23	TxE0	61	DData[11]	99	We*	137	Rxd0[0]
24	RxE0	62	DData[12]	100	Cas*	138	Crs1
25	TxE0	63	DData[13]	101	Ras*	139	Col1
26	RxE6	64	DData[14]	102	Cs*	140	Txd1[0]
27	TxE6	65	Vss	103	DAddr[11]	141	Txd1[1]
28	NC <sup>1</sup>	66	DData[15]	104	DAddr[10]	142	Txd1[2]
29	Vcc <sup>2</sup>	67	DData[7]	105	Vss	143	Txd1[3]
30	Vss	68	DData[6]	106	Vcc	144	TxE1
31	Vcc <sup>2</sup>	69	DData[5]	107	DAddr[9]	145	RxDV1
32	NC	70	DData[4]	108	DAddr[0]	146	RxEr1
33	RxE2	71	DData[3]	109	DAddr[1]	147	Vss
34	Aclk	72	DData[2]	110	DAddr[2]	148	TxCk1
35	TxE2	73	DData[1]	111	DAddr[3]	149	Vcc
36	RxE8	74	Vss	112	DAddr[4]	150	RxCk1
37	TxE8	75	Vcc	113	DAddr[5]	151	Rxd1[0]
38	RxE3	76	DData[0]	114	DAddr[6]	152	Rxd1[1]
153	Rxd1[2]	167	Vcc <sup>2</sup>	181	Vcc <sup>2</sup>	195	Vcc <sup>2</sup>
154	Rxd1[3]	168	Vcc	182	Vcc <sup>2</sup>	196	Vcc <sup>2</sup>
155	Rst*	169	Vss	183	Vcc <sup>2</sup>	197	Vcc <sup>2</sup>
156	Vcc <sup>2</sup>	170	Vcc <sup>2</sup>	184	Vcc <sup>2</sup>	198	Vcc <sup>2</sup>
157	Vcc <sup>2</sup>	171	Vss	185	Vcc <sup>2</sup>	199	Vss
158	Vcc <sup>2</sup>	172	Vcc <sup>2</sup>	186	Vcc <sup>2</sup>	200	Vcc <sup>2</sup>
159	Vcc <sup>2</sup>	173	Vcc <sup>2</sup>	187	Vcc <sup>2</sup>	201	Vcc <sup>2</sup>
160	Vcc <sup>2</sup>	174	Vcc <sup>2</sup>	188	Vcc <sup>2</sup>	202	Vcc <sup>2</sup>
161	Vss	175	Vcc <sup>2</sup>	189	Vss	203	Vcc <sup>2</sup>
162	Vcc	176	Vcc <sup>2</sup>	190	Vcc	204	Vcc <sup>2</sup>
163	Vcc <sup>2</sup>	177	Vcc <sup>2</sup>	191	Vcc <sup>2</sup>	205	Vcc
164	Vcc <sup>2</sup>	178	Vss	192	Vcc <sup>2</sup>	206	Vss
165	Vcc <sup>2</sup>	179	Cclk <sup>3</sup>	193	Vcc <sup>2</sup>	207	NC
166	Vcc <sup>2</sup>	180	Vcc	194	Vcc <sup>2</sup>	208	NC

1. NC = No connect. Do not connect this pin. See text on "Building Systems to Accept GT-48212/208/207 Devices"
  2. If users only want to design the board for the GT-48207, these pins can be tied directly to Vcc. If users want to design the board for the GT-48207 and the GT-48208/212, these pins should be pulled up to Vcc through a resistor. Galileo recommends using a 4.7KOhm resistor value.
  3. For the GT-48207, Pin 179 (CClk) must be driven by a clock source slower than Aclk. The 25MHz clock for the 10/100 PHY or the 20MHz clock for the 10Mbps PHY is acceptable.
-

## 21. DC CHARACTERISTICS - *PRELIMINARY/SUBJECT TO CHANGE* )

**Table 68: Absolute Maximum Ratings**

Symbol	Parameter	Min.	Max.	Unit
Vcc	Supply Voltage	-0.3	4	V
Vi	Input Voltage	-0.3	5.5	V
Vo	Output Voltage	-0.3	5.5	V
Io	Output Current		24	mA
Iik	Input Protect Diode Current		+20	mA
Iok	Output Protect Diode Current		+20	mA
Tc	Operating Case Temperature	0	70	C
Tstg	Storage Temperature	-40	125	C
ESD			2000	V

**Table 69: Recommended Operating Conditions**

Symbol	Parameter	Min.	Typ.	Max.	Unit
Vcc	Supply Voltage	3.15	3.3	3.45	V
Vi	Input Voltage (all pins are 5V tolerant)	0		5.5	V
Vo	Output Voltage	0		Vcc	V
Tc	Case Operating Temperature	0		70	C
Cin	Input Capacitance		7.2		pF
Cout	Output Capacitance		7.2		pF

**Table 70: DC Electrical Characteristics Over Operating Range (Tc=0-70 C; Vcc=+3.3V, +/-5%)**

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Unit
Vih	Input HIGH level	Guaranteed Logic HIGH level	2.0		5.5	V
Vil	Input LOW level	Guaranteed Logic LOW level	-0.5		0.8	V
Voh	Output HIGH Voltage	IoH = 2 mA IoH = 4 mA IoH = 8 mA	2.4			V
Vol	Output LOW Voltage	IoL = 2 mA IoL = 4 mA IoL = 8 mA			0.4	V
Iih	Input HIGH Current				+1	uA
Iil	Input LOW Current				+1	uA



**Table 70: DC Electrical Characteristics Over Operating Range** ( $T_c=0-70\text{ }^{\circ}\text{C}$ ;  $V_{cc}=+3.3\text{V}$ ,  $\pm 5\%$ )

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Unit
I <sub>ozh</sub>	High Impedance Output Current				$\pm 1$	$\mu\text{A}$
I <sub>ozl</sub>	High Impedance Output Current				$\pm 1$	$\mu\text{A}$
V <sub>h</sub>	Input Hysteresis					
I <sub>cc</sub>	Operating Current	$V_{cc} = 3.45\text{V}$ $f=66\text{MHz}$		500		$\text{mA}$

**Note:** Pullup/Pulldown resistors are 45K $\Omega$  minimum, 65K $\Omega$  typical, 80K $\Omega$  maximum.

## 21.1 Thermal Data

Table 71 shows the package thermal data for the GT-482xx. Please contact Galileo Technology if you are in doubt as to thermal considerations for your system.

**Table 71: 208 PQFP Thermal Data**

Parameter	Definition	Value
$\theta_{ja}$	Thermal resistance: junction to ambient, 0 ft/s airflow	25C/W
$\theta_{jc}$	Thermal resistance: junction to case, 0 ft/s airflow	5C/W
T <sub>j</sub>	Operating Junction temperature	100C

## 22. AC TIMING - TARGET/SUBJECT TO CHANGE

**Table 72: CPU Interface Timings** (Tc= 0-70C; Vcc = +3.3V +/- 5%)

Symbol	Signals	Description	Min	Max	Unit
	CClk	CPU interface clock frequency	16	50 (lower than ACIk)	MHz
	CClk	CPU interface clock duty cycle	40	60	%
tsu	Ads*, Blast*, BurstAddr[1:0], AD[31:0], RdCEn*, Wr*	CPU interface input signal setup time (relative to CClk)	8		nS
thd	Ads*, Blast*, BurstAddr[1:0], AD[31:0], RdCEn*, Wr*	CPU interface input signal hold time (relative to CClk)	1		nS
toV	AD[31:0], Ready*, RdCEn*	CPU interface output (or I/O) output valid delay (relative to CClk)	2	10	nS
toZ	AD[31:0], Ready*, RdCEn*	CPU interface output (or I/O) output float delay from drive (relative to CClk)	2	9	nS
tzo	AD[31:0], Ready*, RdCEn*	CPU interface output (or I/O) output drive delay from float (relative to CClk)	2	10	nS

### Notes:

1. All Delays, Setup, and Hold times are referred to CClk rising edge, unless stated otherwise.
2. All outputs are specified for 50pF load

**Table 73: Switch Engine Interface Timings** (Tc= 0-70C; Vcc = +3.3V +/- 5%)

Symbol	Signals	Description	Min	Max	Unit
	ACIk	Switch engine clock frequency	40	66	MHz
	ACIk	Switch engine clock duty cycle	40	60	%
tsu	ColE11-0	Switch engine input signal setup time (relative to ACIk)	8		nS
thd	ColE11-0	Switch engine input signal hold time (relative to ACIk)	1		nS
tsu	DData[31:0]	SDRAM data setup time (relative to ACIk)	2		nS
thd	DData[31:0]	SDRAM data hold time (relative to ACIk)	1		nS
toV	ColE11-0, Cs*, DAddr[11:0], DData[31:0], Cas*, DramWr*) Ras*, Dqm	Switch engine output (or I/O) output valid delay (relative to ACIk)	3	8	nS

**Table 73: Switch Engine Interface Timings** (T<sub>c</sub> = 0-70°C; V<sub>cc</sub> = +3.3V +/- 5%) (Continued)

Symbol	Signals	Description	Min	Max	Unit
toz	DData[31:0], CoIE11-0	Switch engine output (or I/O) output float delay from drive (relative to ACIk)	2	9	nS
tzo	DData[31:0], CoIE11-0	Switch engine output (or I/O) output drive delay from float (relative to ACIk)	2	10	nS

**Notes:**

1. All Delays, Setup, and Hold times are referred to ACIk rising edge, unless stated otherwise.
2. All outputs are specified for 50pF load

**Table 74: MII, LED and MDC/MDIO Timings** (T<sub>c</sub> = 0-70°C; V<sub>cc</sub> = +3.3V +/- 5%)

Symbol	Signals	Description	Min	Max	Unit
	RxCk	MII receive clock frequency <b>Note:</b> For applications requiring even lower frequencies please consult your local Galileo FAE.	2.5	25 (lower than ACIk)	MHz
	TxCk	MII transmit clock frequency <b>Note:</b> For applications requiring even lower frequencies please consult your local Galileo FAE.	2.5	25 (lower than ACIk)	MHz
	RxCk	MII receive clock frequency in expansion mode See Note 2.		60 (lower than ACIk)	MHz
	TxCk	MII transmit clock frequency in expansion mode See Note 2.		60 (lower than ACIk)	MHz
tsu	RxDv, RxEr	MII signal setup time with respect to RxCk	10		nS
thd	RxDv, RxEr	MII signal hold time with respect to RxCk	3		nS
tsu	Rxd[3:0]	MII receive data setup time with respect to RxCk	6		nS
thd	Rxd[3:0]	MII receive data hold time with respect to RxCk	3		nS
toV	TxEn, Txd[3:0]	MII transmit output valid delay (relative to TxCk)	2	10	nS
tsu	LinkStat0, SerLink-Stat	Link status for Port 0 and serial link status setup with respect to LedCk	10		nS
thd	LinkStat0, SerLink-Stat	Link status for Port 0 and serial link status hold time with respect to LedCk	1		nS

**Table 74: MII, LED and MDC/MDIO Timings** ( $T_c = 0-70^\circ\text{C}$ ;  $V_{cc} = +3.3\text{V} \pm 5\%$ )

Symbol	Signals	Description	Min	Max	Unit
toV	LedData, LedStrobe	Output valid delay with respect to Led-Clk. <b>Note:</b> See Section 13.4 "LED Interface Description" on page 57 regarding the relation between LEDClk and these signals.	$3 * \text{AClk clock cycles} + 2\text{ns}$	$3 * \text{AClk clock cycles} + 10\text{ns}$ (or more for margin)	nS
tsu	MDIO	Setup time for MDIO relative to MDC	40		nS
thd	MDIO	Hold time for MDIO relative to MDC	10	1	nS
toV	MDIO	MDIO output valid from MDC	2	50	nS

**Notes:**

1. All outputs are specified for 50pF load.
2. The expansion MII may run up to 60MHz, but must be lower than AClk by 3MHz.

**Table 75: Serial Clock Timings** ( $T_c = 0-70^\circ\text{C}$ ;  $V_{cc} = +3.3\text{V} \pm 5\%$ )

Symbol	Signals	Description	Min	Max	Unit
t1	SClk	Rise Time		2	ns
t2	SClk	Fall Time		2	ns
	SClk	Frequency Stability		$\pm 50$	PPM

**Figure 28: Serial Clock Waveform (SClk)**

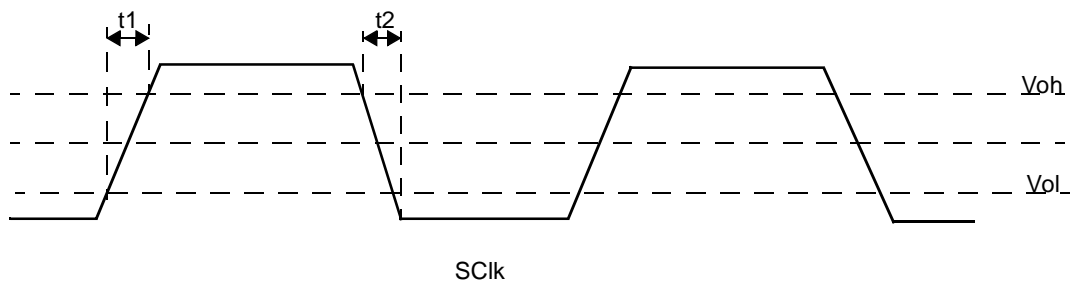


Figure 29: Output Delay from Rising Edge

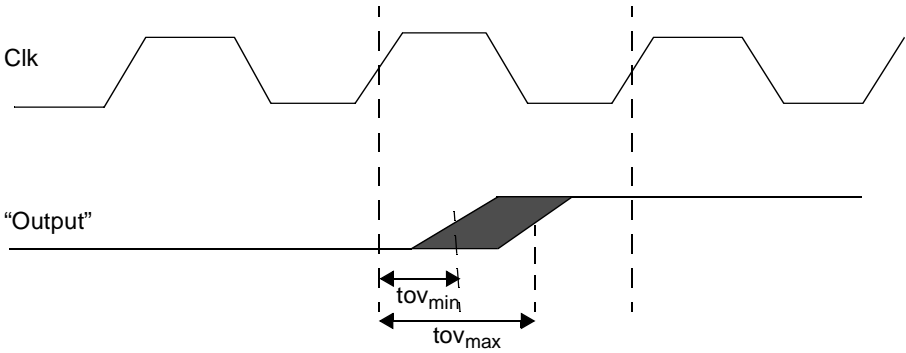


Figure 30: Input Setup and Hold

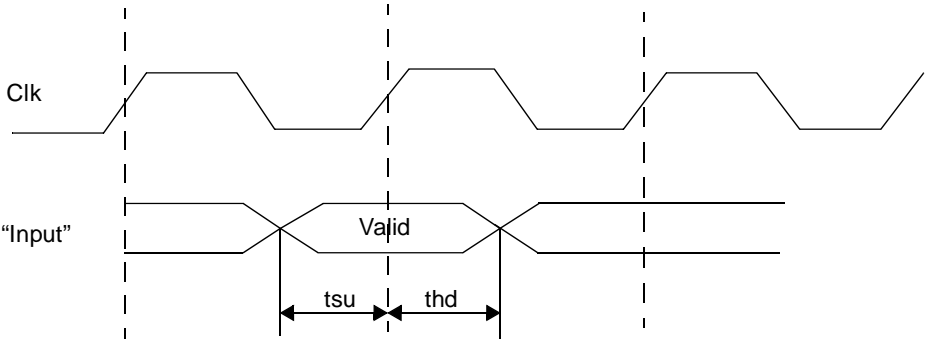
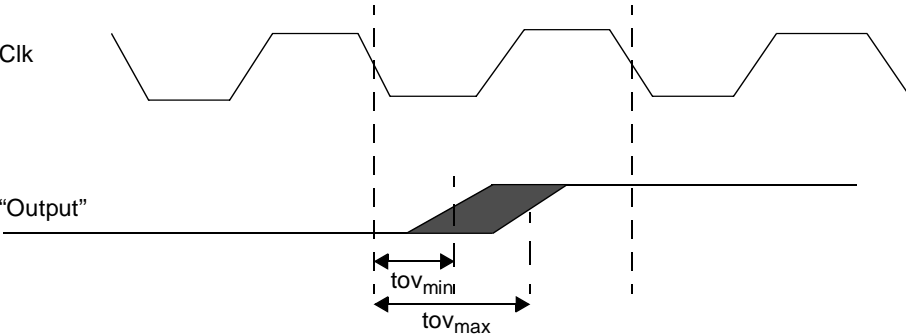
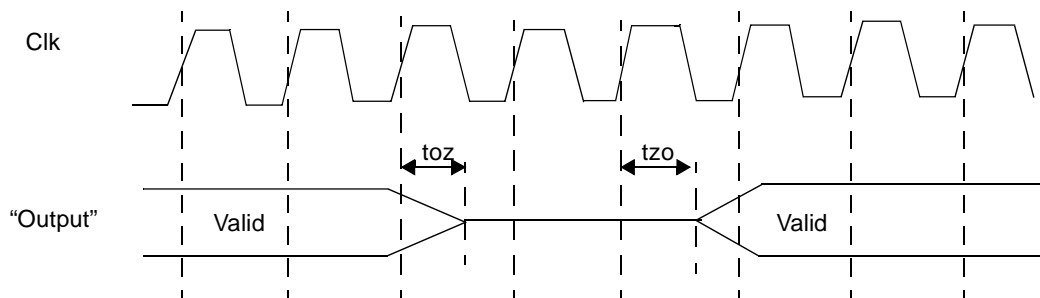


Figure 31: Output Delay from Clock

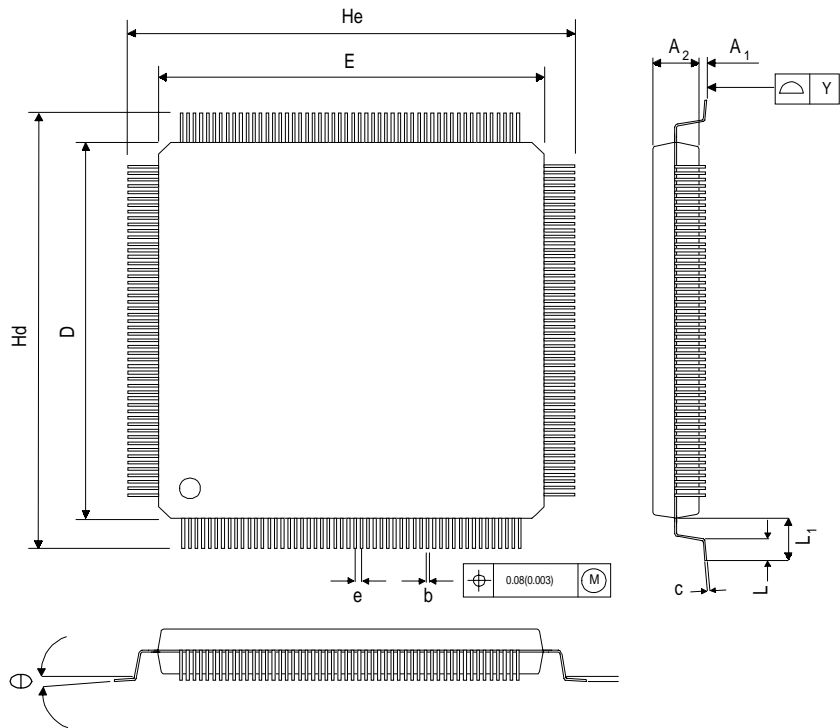


**Figure 32: Output Float and Drive Delay**



**23. PACKAGING**

**Figure 33: 208 Lead PQFP Package Outline**



**Table 76: 208 PQFP Package Dimensions**

SYMBOL	MILLIMETERS		
	MIN.	NOM.	MAX.
A <sub>1</sub>	0.05	0.25	0.50
A <sub>2</sub>	3.17	3.32	3.47
b	0.10	0.20	0.30
c	0.10	0.15	0.20
D	27.90	28.00	28.10
E	27.90	28.00	28.10
e		0.50	

Table 76: 208 PQFP Package Dimensions (Continued)

	MILLIMETERS		
Hd	30.35	30.60	30.85
He	30.35	30.60	30.85
L	0.45	0.60	0.75
L <sub>1</sub>		1.30	
Y			0.08
Q	0		7



## 24. DOCUMENT HISTORY

Table 77: Document History

Document Type	Rev. #	Date	Comments
PRELIMINARY	0.8	9/97	First revision derived from Rev 1.8 architectural specification. [MK]
PRELIMINARY	0.9	10/15/97	Last pre-silicon release. [MK]
PRELIMINARY	0.91ip	12/8/97	<p>First interim release after alpha silicon. Changes made:</p> <ol style="list-style-type: none"> <li>1. HOL pin strapping option (DData[31] removed. HOL is controlled by RecBufThr strapping option. NOTE: This was wrong and changed back in rev 0.913ip (see below).</li> <li>2. Direction bits in General Purpose Control Register2 had the sense inverted. This is corrected in 0.91ip.</li> <li>3. The 10M Port Control Register erroneously showed the FullDx bit as being sampled from TxEn. It is actually sampled from the TxD pin for the port (as shown elsewhere in the spec).</li> <li>4. The polarity of the Link Status bits in the Status Register is different for the 10M and 100M ports. This has been added.</li> <li>5. The Port 13 Control Register was shown in 0.9 as being at 0x100C instead of 0x1004 where it actually is.</li> <li>6. The AutoPol bit in the 10M Port Control registers is now RESERVED.</li> <li>7. The polarity of CPU address bit 22 (decoding CPU access to memory vs. registers) was inverted in 0.9. This is fixed in 0.9ip.</li> <li>8. The 10M Port Control register now shows that the serial mode is sampled from DData[23:0], <b>not</b> the TxDel/TxD pins as shown in 0.9. This was correct elsewhere in the spec.</li> <li>9. 100Mbit port control - 0x1000 - 0x100c. Correction: Bit[1] FDX - this is configured by boot strapping the DAddr[7:6], DAddr[7] for port13, Daddr[6] port12.</li> <li>10. Global Control Register - 0x04 Correction: Bit[4] HASH table mode - Default value is "1".</li> <li>11. SMI Register: 0x50 Correction: Bit[26]: OpCode - This bit is cleared at reset.</li> <li>12. Serial parameters 100 Register: 0x78 Correction: Bit[25:22]: Reserved - These bits are cleared at reset.</li> <li>13. CPU New Address: 0x40 and 0x44. Correction: Bit[31:4] of Address1 should be MAC[47:20], Bit[19:0] of Address2 should be MAC[19:0].</li> </ol>

Table 77: Document History (Continued)

Document Type	Rev. #	Date	Comments
			<ul style="list-style-type: none"> <li>14. Better description of memory array endian added in CPU section.</li> <li>15. Pin strappings deleted from pin descriptions and left only in RESET configuration section.</li> <li>16. Clocking restrictions added to AC Timings.</li> </ul>
PRELIMINARY	0.911ip	12/9/97	<ul style="list-style-type: none"> <li>1. Flow control strapping polarity was corrected in the RESET strapping options table on page 66.</li> </ul>
PRELIMINARY	12/12/97	0.912ip	<ul style="list-style-type: none"> <li>1. Fixed some broken cross references. No functional changes.</li> </ul>
PRELIMINARY	0.913IP	1/5/98	<ul style="list-style-type: none"> <li>1. Corrected HOL strapping option <i>back</i> to DData[31].</li> </ul>
PRELIMINARY	0.92	1/14/98	<ul style="list-style-type: none"> <li>1. Page 31: Corrected DData[27:26] definition for Auto-Negotiation control to agree with Page 58. Tie pins HIGH to enable, and LOW for disable.</li> <li>2. Page 83: Corrected FullDuplexMode select to agree with Page 57 definition. Should be DData[25:24] not DData[7:6].</li> <li>3. Page 63: Corrected PortCtrlReg[0:3] offset from 0x400-0x454 to 0x400-0x40C.</li> <li>4. Page 84: Corrected EASE register offset from 0x1010-0x1018 to 0x1008-0x1010.</li> <li>5. Page 83/84: Added definition of 100M PortCtrl-Reg[31:19] to table as RESERVED.</li> <li>6. Page 81/82: Added definition of 10M PortCtrl-Reg[31:19] to table as RESERVED.</li> </ul>
PRELIMINARY	0.93	N/A	<ul style="list-style-type: none"> <li>1. Changed all "Vccc" to "Vcc"; "Vssc" to "Vss" and "Vdd" to "Vcc".</li> <li>2. Page 27, Section 7.3: Operation of GT-48212 writing new descriptor to the CPU_Tx_Hi/Low_Desc register is clarified.</li> <li>3. Page 90. Vih (Max) corrected to read 5.5V from Vdd+0.5V.</li> <li>4. Register 0x54, VLAN EtherType: Added recommendation that it should be programmed to 0x8100 for compliance with IEEE specifications.</li> </ul>

Table 77: Document History (Continued)

Document Type	Rev. #	Date	Comments
PRELIMINARY	1.0	3/4/98	<ol style="list-style-type: none"> <li>Page 10: Corrected Limit4 to be active HIGH instead of active LOW. Corrected direction of Rdcen*/BurstAddr[0] from Input to Input/Output. Corrected direction of TxEn[1:0] from Input/Output to Output only.</li> <li>Page 11: Corrected description of Rst* signal to be asserted for at least 10 MII clocks, not 10 system clocks. Corrected AC1k description to a max of 66MHz, not 50MHz.</li> <li>Page 12: Removed Reset pin definitions from Addr[11:0] description and referred to Table 16, "RESET Pin Strapping Options," on page 66. Added reference to Table 19, "CPU Mode Selection," on page 70 for Ras*.</li> <li>Page 22: Corrected Aging bit definition.</li> <li>Page 57: Corrected polarity of DAddr[11] in RESET Pin Strapping Options for Back Pressure Enable.</li> <li>Chapter 18: Corrected offset addresses for Counters[3:0] and Counters[7:4] in the Register Map. General clean-up of register definitions. Provided polarity and default values for all signals.</li> <li>Page 93: Added <math>\theta_{ja}</math> and <math>\theta_{jc}</math> values to Table 71. Removed Tj maximum value from Table 71.</li> <li>Page 95: Corrected thd (MII hold time) values to be 3nS minimum, from 1nS maximum.</li> <li>Removed "Galileo Technology Confidential Document."</li> </ol>
PRELIMINARY	1.1	4/30/98	<ol style="list-style-type: none"> <li>Added GT-48208 and GT-48207 pinouts (page 115) and information to create a unified datasheet.</li> <li>Added RV32364 information (page 69).</li> <li>Added additional Aging information (page 30).</li> </ol>

GALILEO TECHNOLOGY CONFIDENTIAL -- DO NOT REPRODUCE

Table 77: Document History (Continued)

Document Type	Rev. #	Date	Comments
PRELIMINARY	1.2	1/27/99	<ol style="list-style-type: none"> <li>1. Applied new template and format.</li> <li>2. Added List of Tables and Figures.</li> <li>3. Added a block diagram illustrating a typical managed switch with two 100Mbit ports + 12 10 Mbit ports.</li> <li>4. Changed ACIk max frequency to 66MHz and CCik max frequency to 50MHz.</li> <li>5. CCik frequency must be higher than 25% of ACIk and lower than total ACIk.</li> <li>6. Changed RxClk and TxClk on the expansion port to max 60MHz frequency .</li> <li>7. Added Appendix A on page135, illustration of the IGMP packet formats.</li> <li>8. Section 1.1 "Fast Ethernet Ports" on page 9, added information about two Galaxy devices connected with an expansion port.</li> <li>9. Section 1.6 "Address Recognition" on page 10, changed the number of different Unicast MAC addresses and unlimited Multicast/Broadcast MAC addresses recognized by GT-482xx from 8000 to 8192, and (2000 changed to 2,024 in 1Mbyte configuration).</li> <li>10. The following changes were made to Table 2, "Pin Functions," on page 14: <ul style="list-style-type: none"> <li>- Symbol for DRAM Address changed from Addr[11:0] to DAddr[11:0]</li> <li>- In the description for the TxEN[0] symbol: Transmit Enable, added that packet is transmitted on Port 0</li> <li>- Corrected description "It should be driven at the falling edge of the clock" to "It should be driven at the falling edge of the LEDClk"</li> <li>- Corrected description for the MDIO symbol: Management Data Input/Output.</li> <li>- Corrected description for the LEDClk symbol</li> <li>- Added polarity information for Link Status and Serial Link Status Indication</li> </ul> </li> <li>11. Table 4, "GT-482xx DRAM Address Mapping," on page 24, changed 1Mbyte address for CPU descriptors.</li> <li>12. Section 6.1 "Forwarding Mask" on page 26, added "When the GT-482XX performs new address learning, the forw[14:0] bits are set to 0x7FFF (all 1's)".</li> <li>13. Section 1.6 "Address Recognition" on page 10, corrected note about the treatment of incoming PAUSE packets when flow control is disabled.</li> <li>14. Section 4. "Microarchitectural Overview" on page 21, corrected FIFO size in 10Mbit port and 100Mbit port units.</li> <li>15. Section 6.3 "Address Learning Process" on page 27, added information specific to two device configurations.</li> </ol>

Table 77: Document History (Continued)

Document Type	Rev. #	Date	Comments
PRELIMINARY	1.2	1/27/99	<p>16. Section 7.4 "Forwarding a Packet from the CPU to the GT-482xx" on page 34, corrected information in step 5).</p> <p>17. Section 7.5 "Intervention Mode" on page 35, Intervention Mode is also supported by multicast.</p> <p>18. Section 8.11.2 "Entering Partition State" on page 40, in third case added "Only for port configured to AU mode".</p> <p>19. Section 8.12 "Back Pressure" on page 41, added information about packet transmit during BP.</p> <p>20. Section 8.15 "MII Management Interface (SMI)" on page 42, added "Delay time between two consecutive SMI write transactions should be a least (4x64=256)MDC clock cycles."</p> <p>21. Section 9.6 "Link Integrity" on page 45, deleted Auto Polarity Detection information, this feature is not supported.</p> <p>22. Added Section 9.15 "Serial Link Status Indication" on page 47, with waveform example.</p> <p>23. Section 9.7 "Data Blinder" on page 45, changed the default value from 6uS to 32uS.</p> <p>24. Section 9.9.2 "Entering Partition State" on page 46, added "However, a more aggressive Backoff Algorithm can be set if Limit4 is activated" to 2nd bullet.</p> <p>25. Added Figure 9: Example of Serial Link Status Indicator of Port 1 Link Fail on page 47.</p> <p>26. Section 11.2 "Monitoring (Sniffer) Mode" on page 49, corrected information "In Monitoring mode, the GT-482xx sends all receive (including local traffic) and transmits packets to the CPU, or to <b>another port in the same</b> GT-482xx device which is assigned to be the Sniffer. Included information about multiple device configurations.</p> <p>27. Section 12.7 "Enabling/Disabling EASE Functionality" on page 52, added EASE enabling/disabling algorithm.</p> <p>28. Section 13.4 "LED Interface Description" on page 57, deleted "LEDClk frequency during Rst* is 33 MHz."</p> <p>29. Section 15.1 "Configuration Pins" on page 66, added recommendation to pull up the pins for unmanaged systems unless the user desires to set a specific feature.</p> <p>30. Table 16, "RESET Pin Strapping Options," on page 66, added Pin BurstAddr[1] and changed configuration function of DAddr[10].</p>

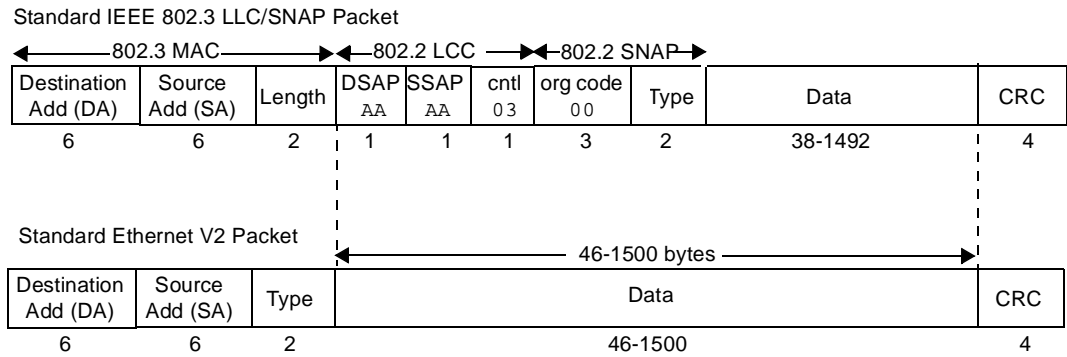
Table 77: Document History (Continued)

Document Type	Rev. #	Date	Comments
PRELIMINARY	1.2	1/27/99	<p>31. The following changes were made to Table 18, "CPU Interface Pin Mappings," on page 70:</p> <ul style="list-style-type: none"> <li>- Changed BurstAddr[1:0] in i960Jx to "pull up"</li> <li>- Changed BurstAddr[1:0] in ColdFire to "pull up"</li> <li>- Changed RdCEn in ColdFire to "pull up"</li> <li>- Changed RdCEn in RV32364 to "pull up"</li> <li>- Header "BurstAddr[1:0]" -&gt; "BurstAddr[2:1]"</li> <li>- Coldfire CPU mode (MCF5202) connected to DA*[0]</li> </ul> <p>32. Section 16.6 "CPU Interface Applications" on page 71, added waveform information</p> <p>33. Added Section 17.1 "DRAM Configuration" and Section 17.2 "DRAM Initialization" on page 80.</p> <p>34. Table 23, "Global Control, Offset: 0x04," on page 84, added that Initial Value should be 0x1 for P-0/1/2 and 0x0 for P-3/4.</p> <p>35. Table 24, "Status Register, Offset: 0x08," on page 87, Initial value [3] is sampled from DQM pin and not Daddr[5].</p> <p>36. Table 37, "CPU Enqueue1, Offset: 0x38," on page 94 added "0-Unicast" for Mult bit function.</p> <p>37. Table 47 and Table 48 on page 99, changed initial values.</p> <p>38. Table 58, "Port Control (10M ports), Offset: 0x400-0x40C, 0x800 - 0x80C, 0xC00 - 0xC0C," on page 102, changed bit 8 field from Reserved to EaseEn.</p> <p>39. Table 59, "Port Control 12 (100M ports), Offset: 0x1000," on page 104, replaced Reserved bit [3:2] with Priority for Expansion port.</p> <p>40. Added Table 60, "Port Control 13 (100M ports), Offset: 0x1004," on page 106</p> <p>Table 72, "CPU Interface Timings (Tc= 0-70C; Vcc = +3.3V +/- 5%)," on page 122, corrected min frequency to 1 thd symbol.</p> <p>41. Table 73, "Switch Engine Interface Timings (Tc= 0-70C; Vcc = +3.3V +/- 5%)," on page 122, corrected min frequency as 1 for thd symbol and changed ACIk max frequency to 66MHz.</p> <p>42. Updated Table 74, "MII, LED and MDC/MDIO Timings (Tc= 0-70C; Vcc = +3.3V +/- 5%)," on page 123</p> <p>43. Added Table 75, "Serial Clock Timings (Tc= 0-70C; Vcc = +3.3V +/- 5%)," on page 124.</p> <p>44. Added Figure 28: Serial Clock Waveform (SCIk) on page 124</p> <p>45. Deleted Section 22 "Functional Waveforms", and included CPU waveform information in Section 16.6 "CPU Interface Applications" on page 71</p>

APPENDIX A

Figure 34 illustrates the packet format of a standard Ethernet V2 packet and an IEEE 802.3 LLC/SNAP.

Figure 34: Ethernet Packet Format



**Note:** The above do not include the IEEE 802.1 Q/p tag. If desired, this 4-byte tag can be included in the format and is added immediately after the SA field. In this case, the maximum acceptable packet length is 1522 bytes (instead of 1518 bytes).