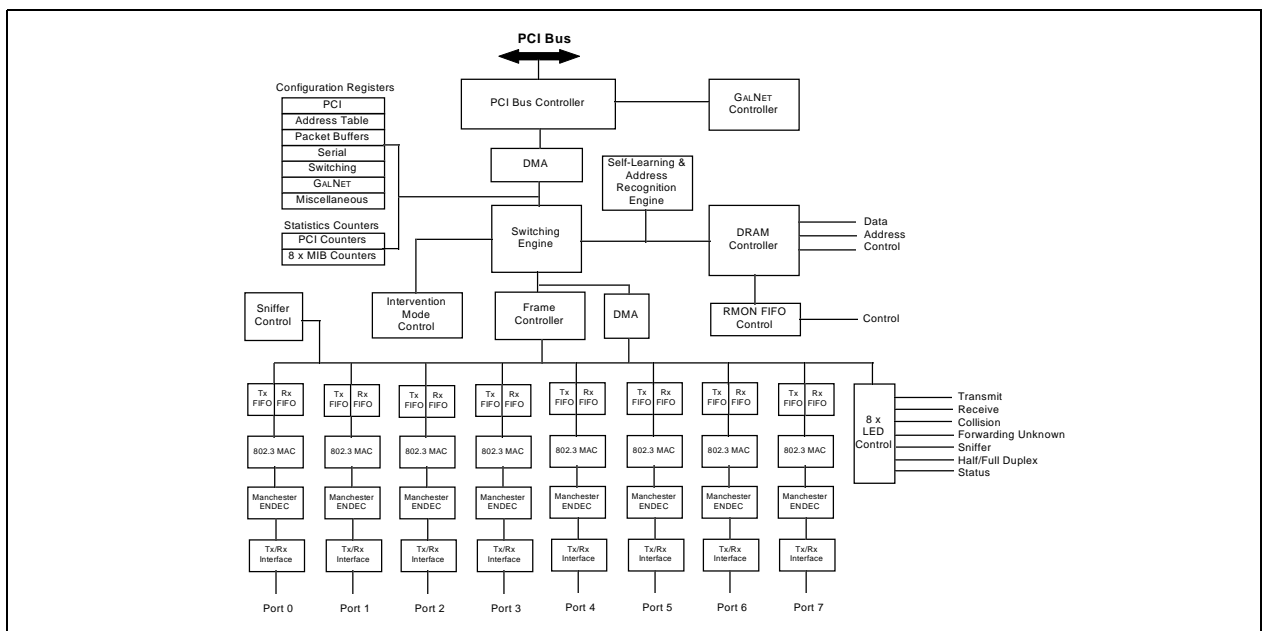


FEATURES

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

- Single-chip, low cost, Switched Ethernet Controller
 - Provides packet switching functions between 8 on-chip Ethernet ports and the PCI expansion port
 - Switch expansion via 1Gbps PCI bus
- GalNet Architecture Family Member
 - Advanced distributed switching architecture
 - Connects seamlessly to other GalNet Family Devices
 - GT-48002A 100BaseX and GT-48003 100VG-AnyLAN devices available
- Incorporates eight 802.3 compliant Ethernet ports
 - 10Mbps half-duplex or full-duplex 20Mbps Ethernet for each port
 - Serial mode selectable per port: 10Base-T, 10Base-FL, AUI, and NRZ Synchronous
- All digital logic on-chip for each port
 - Media Access Control (MAC)
 - Manchester encoder/decoder
 - Link integrity, Partition
 - Automatic polarity detection and correction
 - Dual 32-byte FIFOs for receive and transmit
 - 7 LEDs for Link Status, Receive, Transmit, Collision, Forward Unknown Packets, Port Sniffer, and Half/Full Duplex
 - CRC generation for CPU generated packets
- High-Performance Distributed Switching Engine
 - Performs forwarding and filtering at full wire speed
 - 14,880 packets/sec on each Ethernet port
 - Flexible software or hardware intervention in packet routing decisions
- Supports 'Store and Forward' switching approach
 - Low last-bit in to first-bit out delay
 - Allows bridging between higher/lower speed interfaces (Fast Ethernet, ATM, WAN)
- Advanced address recognition
 - 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
 - Broadcast storm rate filtering
- Direct support for packet buffering
 - Glueless interface to 1 or 2Mbyte of 60ns EDO DRAM
 - Up to 1K buffers, 1536-bytes each, dynamically allocated to the receive and PCI ports
- PCI Rev 2.1 interface for switch expansion and management CPU connection
 - Up to 10 GT-48001A devices per PCI bus without PCI-to-PCI bridging
 - Up to 32 GalNet devices in a single switch
 - Standard CPU connection for management
 - Simple interface to other networking interfaces (ATM, FDDI, etc.)
- Extensive network management support
 - Repeater MIB and PCI counters
 - Address aging support
 - Hardware assist for Spanning Tree algorithm
 - RMON Station-to-Station connectivity matrix
 - CPU access to Address Table
 - Ability to define static addresses
 - Monitoring (sniffer) mode
- HP-EASE Packet sampling management technology
 - Takes "snapshots" of packets at programmable intervals
 - Allows for the implementation of HP-EASE or sampled RMON with low-cost CPUs
- 208 pin PQFP package



Contents

1. Functional Overview	6
1.1 The GalNet Switching Architecture	6
1.2 Ethernet Ports	6
1.3 Address Recognition	7
1.4 CPU Packet Routing	7
1.5 Intervention Mode	7
1.6 Network Management Features	7
1.7 DRAM Interface	7
1.8 PCI Interface	7
2. Pin Information	8
2.1 Logic Symbol	8
2.2 Pin Functions and Assignment	9
3. Operational Overview	13
3.1 Enabling/Disabling the GT-48001A	13
3.2 Basic Operation	13
3.3 Address Learning	14
3.4 Packet Buffering	14
3.5 Packet Forwarding	14
3.6 The GalNet Protocol	14
3.7 Terminology	14
4. MAC Address Learning Process.....	16
4.1 Address Recognition	16
4.2 Recovery Process	16
4.3 Address Aging	17
4.4 Static Addresses	17
4.5 Address Recognition Failure	17
5. GT-48001A Buffers and Queues	18
5.1 Receive Buffer Threshold Programming	19
6. Packet Forwarding	20
6.1 Forwarding a Unicast Packet to a Local Port	20
6.2 Forwarding a Unicast Packet to a Port in a Different GalNet Device	20
6.3 Forwarding a Multicast Packet	21
6.3.1 Local Ports	21
6.3.2 Between GalNet Devices	21
6.3.2.1 CPU Disabled	21
6.3.2.2 CPU Enabled	21
6.4 Forwarding a Packet to the CPU Directly	22
6.5 Forwarding a Packet from the CPU to a GalNet Device	24
6.6 CRC Generation	25
6.7 Tx Watchdog Timer	25
7. Device Table Operation	26
7.1 Automatic Device Table Initialization	26
7.2 Manual Device Table Initialization	26
7.3 Programming Device Numbers	26

8. Unicast Intervention Mode	27
8.1 Unicast Intervention Mode Address Space	28
9. Address Table	29
10. GalNet Messaging Protocol	31
10.1 GalNet Protocol Region	31
10.2 GalNet Messages Between Devices	33
10.2.1 NEW_ADDRESS Message between GalNet devices	33
10.2.2 BUFFER_REQUEST Message between GalNet devices	34
10.2.3 START_OF_PACKET Message between GalNet devices	34
10.2.4 PACKET_TRANSFER Message between GalNet devices	35
10.2.5 END_OF_PACKET Message between GalNet devices	35
10.3 GalNet Messages Between a GalNet Device and a CPU	36
10.3.1 NEW_ADDRESS Message (GalNet to CPU)	36
10.3.2 NEW_ADDRESS Message (CPU to GalNet)	37
10.3.3 BUFFER_REQUEST Message (GalNet to CPU)	38
10.3.4 BUFFER_REQUEST Message (CPU to GalNet)	38
10.3.5 START_OF_PACKET Message (GalNet to CPU)	39
10.3.6 START_OF_PACKET Message (CPU to GalNet)	39
10.3.7 PACKET_TRANSFER Message (GalNet to CPU 16 Block Buffer)	40
10.3.8 PACKET_TRANSFER Message (GalNet to CPU in Unicast Intervention Mode)	41
10.3.9 PACKET_TRANSFER Message (CPU to GalNet)	41
10.3.10 END_OF_PACKET Message (GalNet to CPU 16 Block Buffer)	42
10.3.11 END_OF_PACKET Message (GalNet to CPU in Unicast Intervention Mode)	42
10.3.12 END_OF_PACKET Message (CPU to GalNet)	43
11. PCI Bus Operation	44
11.1 PCI Configuration Header Registers	44
11.2 Accessing DRAM and Internal Registers through the PCI Interface	44
11.3 PCI Bandwidth/Performance Issues	44
11.4 Plug-and-Play Considerations In PCI Systems	45
11.5 Unused PCI Bus in Stand-Alone Systems	45
11.6 PCI Bus Arbiter in Multiple GalNet Device Systems	45
12. Ethernet Interfaces	46
12.1 Media Access Control (MAC)	46
12.2 Illegal Frames	46
12.3 Selecting the Duplex Mode	46
12.3.0.1 Packet Transmission	46
12.4 Backoff Algorithm Options	46
12.5 Manchester Encoder/Decoder	46
12.6 Link Integrity and Auto Polarity Detector	47
12.7 Data Blinder	47
12.8 Inter-Packet Gap (IPG)	47
12.9 Partition Mode	47
12.9.1 Enabling Partition Mode	47
12.9.2 Entering Partition State	47
12.9.3 Exiting from Partition State	47
12.10 Back-pressure	48
12.11 VLAN Tagging Support	48
12.12 Serial Modes	48
12.12.1 Signal Polarity in Specific Serial Modes	48
12.12.2 10BaseT Mode	48

12.12.2.1	Generating the Required 10BaseT Signals	49
12.12.2.2	Pol Output/Auto-Polarity in 10BaseT	50
12.12.3	10BaseFL Mode	51
12.12.4	AUI Mode	52
12.12.4.1	TxD Pins	52
12.12.4.2	AUILinkUp	52
12.12.4.3	Setting DAddr[6] on Reset	52
12.12.5	Synchronous Mode	53
13.	Network Management Support	55
13.1	Repeater MIB and PCI Counters	55
13.2	Station-to-Station Connectivity Matrix	55
13.2.1	Data Structure Format	56
13.3	Monitoring (Sniffer) Mode	57
13.4	Spanning Tree Support	57
13.5	Broadcast Storm Filtering	57
14.	HP-EASE Packet Sampling Technology	58
14.1	HP EASE Technology Overview	58
14.2	EASE Functionality on the GT-48001A	59
14.3	Ease_Register	59
14.4	EASE Interrupts	59
14.5	Sampled Packet Indication	60
14.6	Error Source Indications	61
14.7	Enabling/Disabling EASE Functionality	62
14.8	Interaction With Other GT-48001A Features	62
15.	DRAM Interface and Usage	63
16.	LED Support	63
16.1	Led Indications Interface Description	63
16.2	LED Serial Interface Description	63
16.3	Detailed LED Signal Description	64
16.3.1	Primary Port Status LED	64
16.3.1.1	Primary Port Status LED (Mode 0): (LEDMode input is LOW)	64
16.3.1.2	Status LED blink timing (Mode 0)	64
16.3.1.3	Primary Port Status LED (Mode 1): (LEDMode input is HIGH)	65
16.3.2	Transmit data in progress	66
16.3.3	Receive data in progress	66
16.3.4	Collision active	66
16.3.5	Full/Half duplex	66
16.3.6	Receive Buffer Full	66
16.3.7	Forwarding of unknown packets enabled	66
16.3.8	The port is configured as Sniffer	66
16.3.9	Link Fail State	66
16.3.10	Partition State	66
16.4	LED Signals Timing Type	66
16.4.1	Static LED Signals	66
16.4.2	Dynamic Internal Signals:	66
16.4.3	Table of Internal Activities/Status Driven via the LED Interface	66
17.	Interrupts	69

18. RESET Configuration	69
18.1 Configuration Pins	69
18.2 Configuration Input Timings	69
19. Switch Expansion	70
20. Development Tools.....	71
20.1 Evaluation Platforms	71
20.2 Verilog Models	71
20.3 Reference Designs	71
20.4 Complimentary Products	71
21. Register Tables	72
21.1 Internal Control Registers	73
21.2 PCI Configuration Registers	86
21.3 Register Modification Restrictions	89
22. Pinout for 208 pin PQFP (sorted by number).....	90
22.1 Package/Pin Drawing	92
23. DC Characteristics - PRELIMINARY/SUBJECT TO CHANGE	93
23.1 Absolute Maximum Ratings	93
23.2 Recommended Operating Conditions	93
23.3 DC Electrical Characteristics Over Operating Range	93
23.4 Thermal Data	94
24. AC Timing - PRELIMINARY/SUBJECT TO CHANGE.....	95
25. Functional Waveforms	98
25.1 PCI Read/Write Cycle	98
26. Packaging.....	100
27. Document History	101

1. Functional Overview

The GT-48001A is a high-performance, low-cost, Switched Ethernet Controller that provides packet switching functions between eight on-chip 10/20Mbps Ethernet ports and the 1Gbps PCI backplane. The GT-48001A uses the innovative GalNet Switching Architecture to allow expansion to additional Ethernet, Fast Ethernet and 100VG-AnyLAN ports. The GT-48001A is backwards compatible with the GT-48001- the industry's first single-chip Ethernet switch.

1.1 The GalNet Switching Architecture

The GalNet Switching Architecture is based on a proprietary messaging protocol using the industry standard PCI bus as a medium. GalNet devices are designed to connect seamlessly allowing packets to be switched between devices without processor intervention (see Figure 1). Each GalNet device acts as an intelligent agent, sharing information between all other devices in the system. For example, when one GalNet device learns a new address, it automatically updates all other GalNet devices via the NEW_ADDRESS message. GalNet messages are defined as *write-only* (request/response) in order to achieve the maximum bandwidth from the PCI bus.

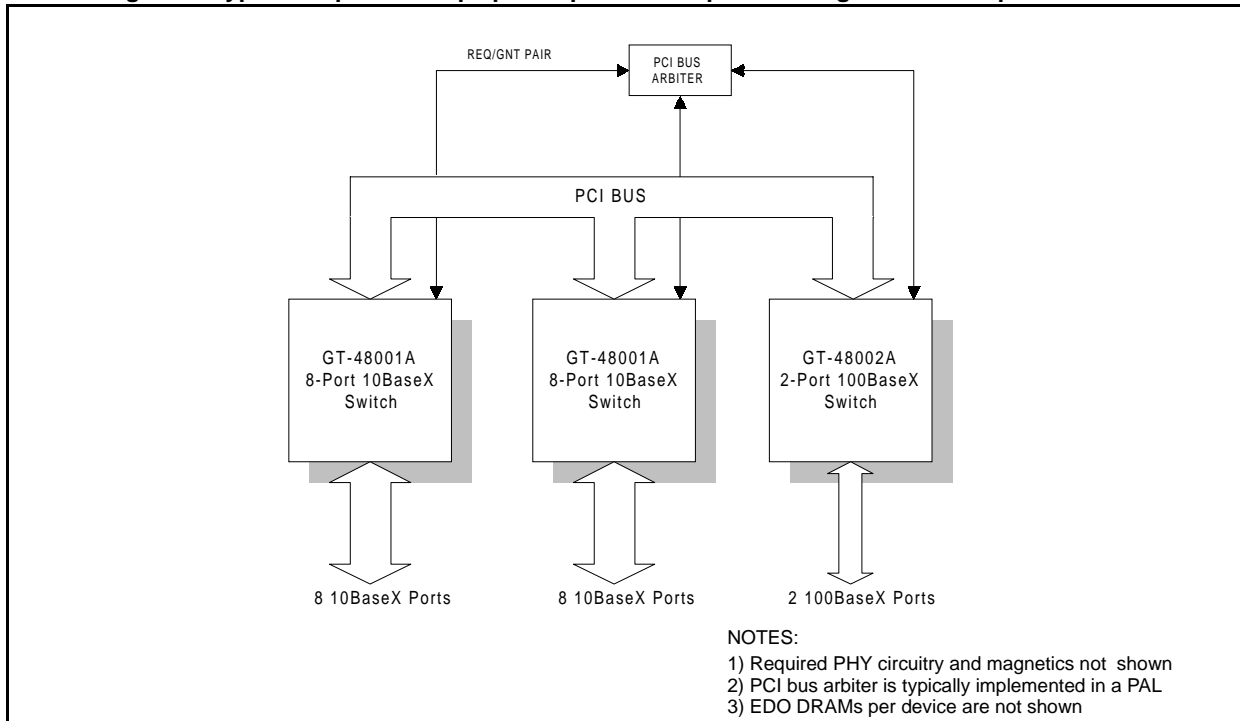
The GalNet Architecture Family currently consists of three products: the GT-48001A (eight ports of 10BaseX), the GT-48002A (two ports of 100BaseX), and the GT-48003 (two ports of 100VG-AnyLAN). In addition, Galileo Technology provides a number of other complementary PCI interface products for popular microprocessors.

1.2 Ethernet Ports

The GT-48001A integrates eight 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, auto polarity logic and LED interface.

The GT-48001A's Ethernet ports are compliant with both the 802.3 and Ethernet specifications. Physical media such as 10Base-T, 10Base-5, 10Base-2 and 10Base-FL are supported.

Figure 1: Typical 16-port 10-Mbps plus 2-port 100-Mbps Unmanaged Switch Implementation



1.3 Address Recognition

Each GT-48001A in a system can recognize up to 8,000 different Unicast MAC addresses and unlimited Multicast/Broadcast MAC addresses. An intelligent address recognition mechanism enables filtering and forwarding packets at full Ethernet wire speed. Hardware assist for address aging and static addresses is also included.

The GT-48001A provides an address self-learning mechanism. Each device has a private address table located in its DRAM array. As the GT-48001A learns new addresses, it updates all address tables in the system via the GalNet messaging protocol.

1.4 CPU Packet Routing

The GT-48001A has the capability to automatically forward certain packets to the CPU for routing including:

- Unicast packets with a destination address tagged for the CPU
- Multicast packets
- Unknown packets (packets with MAC addresses that have not been recognized)

This gives the system designer the flexibility to decide how to handle such packets in a managed system.

1.5 Intervention Mode

The GT-48001A incorporates an enhanced feature called *intervention mode*. This feature permits software or hardware intervention in the packet routing decision mechanisms for unicast packets. When intervention mode is selected for a MAC address, the GT-48001A sends the packet to the system CPU instead of switching it as usual. This capability can be used for many functions including: layer 3 routing, security, virtual LAN support, filtering and management.

1.6 Network Management Features

The GT-48001A provides comprehensive management capabilities enabling the switch OEM to implement a wide range of network management features. For OEMs offering RMON capability, the GT-48001A provides per-port statistics counters and PCI traffic counters. In addition, the GT-48001A provides station-to-station connectivity matrix information and the ability to select a port to work in monitoring (sniffer) mode.

Also included in the GT-48001A is 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, much less local processing is required over standard RMON implementations. Source addresses of every errored packet are also sent to the CPU allowing switch OEM to support error counters in RMON host and matrix groups.

Also included in the GT-48001A is hardware assistance for address aging and bridge spanning tree algorithms.

1.7 DRAM Interface

Each GalNet device in the system requires its own, separate EDO DRAM buffer space. The DRAM is used to store the incoming/outgoing packets as well as the address table and other device data structures. The interface to EDO DRAM is glueless; all signals needed to control EDO devices are included. For more information, see Section 15.

1.8 PCI Interface

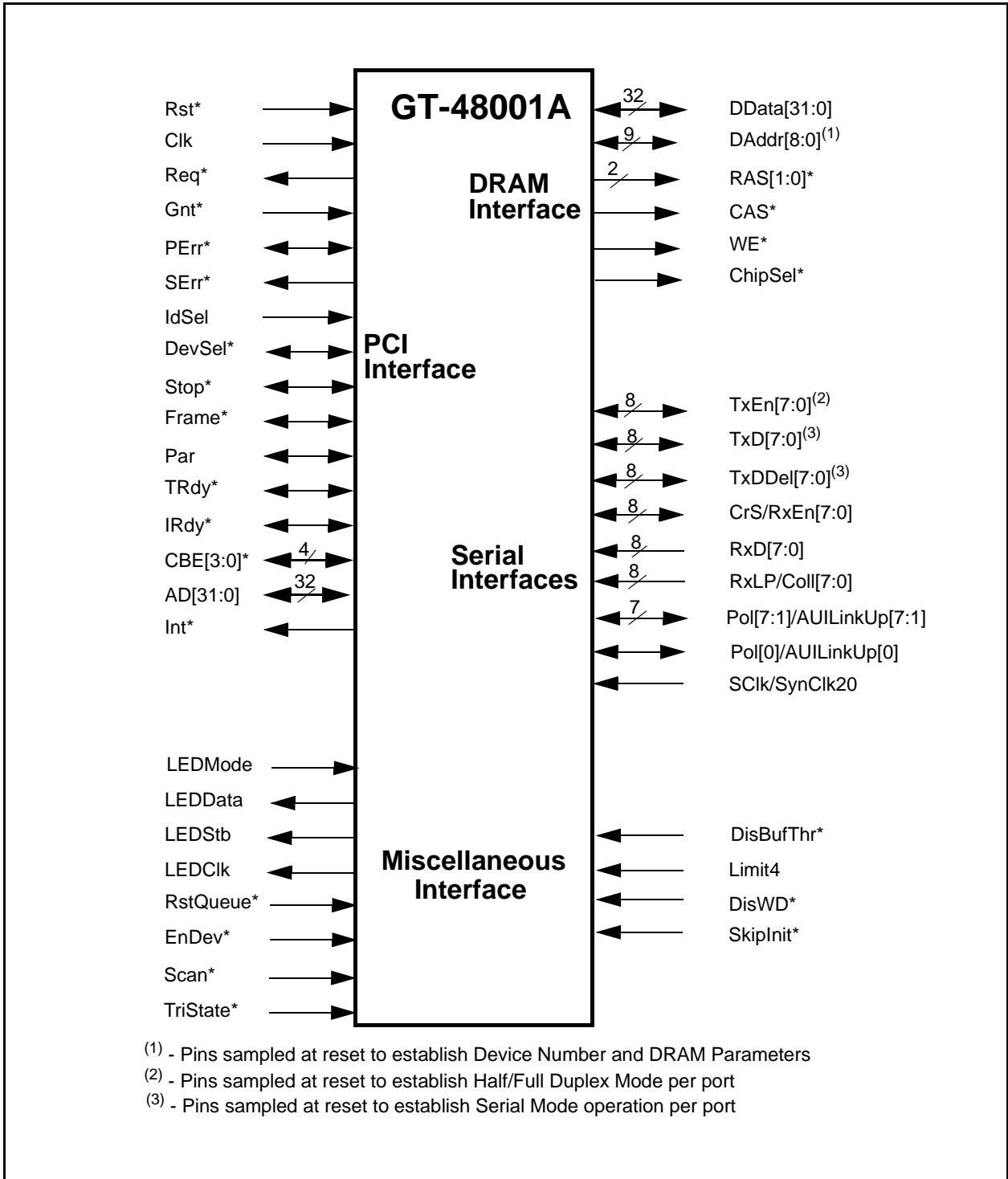
Switch expansion and access to internal management features is possible with the GT-48001A's PCI interface. The GT-48001A can be either a master initiating a PCI bus operation or a target responding to a PCI bus operation. Up to ten GT-48001A devices can reside on the same PCI bus, forwarding packets from one port to the other without CPU intervention.¹ By using PCI-to-PCI bridge devices, the switch can be expanded to up to 32 devices (256 ports).

The PCI bus may also be used to connect to an optional CPU for management, or to connect other high speed LAN adapters such as ATM and FDDI.

1. Ten devices is the physical limit of the PCI specification. Please refer to the PCI bus section of this data sheet for performance limitations.

2. Pin Information

2.1 Logic Symbol



2.2 Pin Functions and Assignment

Symbol	Type	Description
<i>PCI Bus Interface</i>		
Rst*	I	RESET: Active LOW. Rst* must be asserted for at least 10 PCI clock cycles. When in the reset state, all PCI output pins are tristated and all open drain signals are floated. Following Rst* deassertion, the GT-48001A clears the internal buffers and initializes the address table in the DRAM. The address table initialization takes 165,000 CLK cycles to complete. Any incoming packets during the address table initialization, are ignored.
Clk	I	Clock: Provides the timing for the GT-48001A internal units. All functional units except for the serial interfaces use this clock. Clk also provides timing for PCI bus transactions. The clock frequency is 30-33MHz.
Req*	O	Bus Request: Asserted by the GT-48001A to indicate to the PCI bus arbiter that this device requires mastership of the bus.
Gnt*	I	Bus Grant: Indicates to the GT-48001A that access to the PCI bus is granted.
PErr*	I/O	Parity Error: Asserted when a data parity error is detected on the PCI bus.
SErr*	O	System Error: Asserted by the GT-48001A when an address parity error is detected on the PCI bus. The GT-48001A asserts SErr* two cycles after the failing address. This output features an open-collector driver.
IDSel	I	Initialization Device Select: Asserted by a PCI bus master to gain access to the GT-48001A's configuration header during configuration read/write transactions.
DevSel*	I/O	Device Select: Asserted by the target of the current PCI access. When the GT-48001A is a bus master, it expects the target to assert DevSel* within 5 bus cycles, confirming the access. If the target does not assert DevSel* within the required bus cycles, the GT-48001A aborts the cycle. As a target, the GT-48001A asserts DevSel* as a "medium speed" PCI device (two cycles after the assertion of Frame*).
Stop*	I/O	Stop: Indicates that the current target is requesting the bus master to stop the current transaction. As a master, the GT-48001A responds to the assertion of Stop* by either disconnecting, retrying, or aborting. As a target, the GT-48001A asserts Stop* to force a retry.
Frame*	I/O	Cycle Frame: Asserted by the GT-48001A to indicate the beginning and duration of a master transaction. Frame* is asserted to indicate the beginning of the cycle. While Frame* is asserted, data transfer continues. Frame* is deasserted to indicate that the next data phase is the final data phase transaction. Frame* is monitored when the GT-48001A acts as a target, to detect a configuration or memory transaction.
Par	I/O	Parity: Calculated by the GT-48001A as an even parity bit for the AD[31:0] and CBE[3:0]* lines.
TRdy*	I/O	Target Ready: Indicates the target agent's ability to complete the current data phase of the transaction. A data phase is completed on any clock when both TRdy* and IRdy* are asserted. Wait cycles are inserted until both IRdy* and TRdy* are asserted together.
IRdy*	I/O	Initiator Ready: Indicates the bus master's ability to complete the current data phase of the transaction. A data phase is completed on any clock when both TRdy* and IRdy* are asserted. Wait cycles are inserted until both IRdy* and TRdy* are asserted together.

Symbol	Type	Description
AD[31:0]	I/O	Address/Data: 32-bit multiplexed PCI address and data lines. During the first clock of the transaction, AD[31:0] contains a physical byte address (32 bits). During subsequent clock cycles, AD[31:0] contains data.
CBE[3:0]*	I/O	Bus Command/Byte Enable: During the address phase of the PCI transaction, CBE[3:0]* provide the Bus Command. During the data phase, CBE[3:0]* provide Byte Enables, which determine which bytes carry valid data.
Int*	O	Interrupt Request Line: Int* is asserted by the GT-48001A when one (or more) of the bits in the Interrupt Cause register is set. This output features an open-collector driver.
DRAM Interface		
DData[31:0]	I/O	DRAM Data: 32-bit EDO DRAM data bus. These signals connect directly to the data input/output pins of the DRAM devices.
DAddr[8:0]	I/O	DRAM Multiplexed Address Bus: In normal operation, DAddr[8:0] contain the DRAM multiplexed row/column address. During RESET, these multiplexed pins are sampled by the GT-48001A to indicate the Device Number and the DRAM Parameters (see RESET Configuration section). Values are determined by connecting pull-up/pull-down resistors. The Device Number and the DRAM Size are read by the CPU from the Status register.
RAS[1:0]*	O	Row Address Strobes: DRAM row address strobes. RAS[0]* is used for Bank 0. RAS[1]* is used for Bank 1.
CAS*	O	Column Address Strobe: DRAM column address strobe. The GT-48001A always accesses 32-bit values and does not require a separate CAS* for each byte.
WE*	O	Write Enable: DRAM write enable.
ChipSel*	O	FIFO Chip Select: ChipSel* asserted by the GT-48001A when a packet's Destination Port(s), Byte Count, Destination Address and Source Address are read from the DRAM. This information can be stored in an external FIFO and accessed by the management CPU for station-to-station connectivity matrix information.
Ethernet Interfaces		
TxEn[7:0] (Fdx)	I/O	Transmit Enable: TxEn envelopes the transmitted packet. During reset, this multiplexed input indicates the port mode operation: pull-up for full duplex, pull-down for half duplex (see RESET Configuration section).
TxD[7:0] (SerMode0)	I/O	Transmit Data: In normal operation, TxD drives the transmitted data. During reset, this multiplexed input pin behaves as SerMode0, which together with TxDDel (SerMode1) indicate the port mode of operation based on pull-up/pull-down resistors connected to them (see RESET Configuration section).
TxDDel[7:0] (SerMode1)	I/O	Transmit Data Delayed: In normal operation, TxDDel outputs the transmitted data delayed. TxDDel follows the TxD signal by 50nsec. During reset, this multiplexed input pin behaves as SerMode1, which together with TxD (SerMode0) indicate the port mode of operation based on pull-up/pull-down resistors connected to them (see RESET Configuration section).
CrS/RxEn[7:0]	I/O	Carrier Sense/Receive Enable: In 10Base-T, 10Base-FL and AUI, this output pin indicates the Carrier Sense. In Sync mode, this input pin envelopes the receive packet.
RxD[7:0]	I	Receive Data: RxD is the received serial bitstream from the Ethernet wire.

Symbol	Type	Description
RxLP/Coll[7:0]	I	Receive Link Pulses/Collision: This multiplexed pin carries the Receive Link Pulses in 10Base-T. In AUI mode, it indicates Collision.
Pol[7:1]/ AUILinkUp[7:1]	I/O	Polarity/AUI Link Status: Indicates the line polarity of Ports 7:1 in 10Base-T mode (output). In AUI mode, The AUILinkUp[7:1] pins of the GT-48001A are used as inputs to indicate the link status which can be driven by the PHY. When the respective AUILinkUp[7:1] pin is a '1', the port has established link. When the respective AUILinkUp[7:1] pin is a '0', the link has failed on that port.
Pol[0]/AUILinkUp[0]/ SynClk10	I/O	Polarity/AUI Link Status/Synchronous Clock 10: Indicates the polarity of Port 0 in 10Base-T mode (output). In AUI mode, The AUILinkUp[0] pin of the GT-48001A are used as an input to indicate the link status which can be driven by the PHY. When AUILinkUp[0] pin is a '1', port 0 has established link. When AUILinkUp[0] pin is a '0', the link has failed on port 0. In Synchronous mode, this input pin carries the 10MHz synchronous clock.
SClk/SynClk20	I	80MHz Serial Clock/Synchronous Clock 20: Used to recover the receive clock and to generate the the internal transmit clocks. In Synchronous mode, this pin carries the 20MHz synchronous clock.
Miscellaneous Interface Pins		
LEDData	O	LED Data: LED indicators (Link Status, Receive, Transmit, Collision, Unknown, Port Sniffer, and Half/Full Duplex) of each port. The data is shifted out in 128 bit long frames using the LEDClk and LEDStb pins.
LEDStb	O	LED Strobe: Indicates the beginning of valid data frame on the LEDData pin.
LEDClk	O	LED Clock: LEDClk frequency is 1 MHz. During Rst* assertion, LEDClk frequency is 10 MHz. This pin is used to clock the serial data out.
LEDMode	I	LED Mode: When the LEDMode pin is LOW, the LED mode is compatible with the GT-48001. When HIGH, the GT-48001A enters LED Mode 1. See the LED interface section for a detailed description.
RstQueue*	I	Reset Transmit Queues: When asserted, all internal transmit and receive queues are cleared. All GT-48001A state machines are moved to their initial state.
EnDev*	I	Enable Device: Enables serial and PCI ports. When asserted LOW, all serial ports and the PCI port are active. When deasserted, the ports and the PCI are disabled (see Section 3.1).

Symbol	Type	Description															
Scan*	I	<p>Scan: This pin together with TriState* indicate the GT-48001A mode of operation as follows:</p> <table border="1" data-bbox="575 291 1228 550"> <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-48001A 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-48001A 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-48001A drives all outputs and I/O pins to high impedance.															
0	0	Factory test mode (reserved)															
TriState*	I	<p>Tri-State: This pin together with Scan* indicate the GT-48001A mode of operation as described above.</p>															
DisWD*	I	<p>Disable Watchdog Timer: Active LOW. DisWD* controls the enabling (HIGH) or disabling (LOW) of the Tx Watchdog Timer on all ports.</p>															
DisBufThr*	I	<p>Buffer Threshold: Active LOW. This pin externally enables or disables the buffer threshold. When HIGH, the buffers allocated to the ports and the PCI are limited to the number written in the Rx Buffer Threshold Register. When LOW, the buffers are dynamically allocated to the ports and the PCI bus (i.e. there is no limitation on the buffers' allocation.)</p>															
Limit4	I	<p>Backoff Algorithm: This pin selects the number of retransmit attempts after a collision will occur before the back-off algorithm is restarted. When LOW, 16 retransmit attempts after a collision must occur before the back-off algorithm is restarted (802.3 standard). When HIGH, 4 retransmit attempts after a collision must occur before the back-off algorithm is restarted (more aggressive.)</p>															
EnEIScrub*		<p>Enable Empty List Scrubbing: For testing purposes only. Must be pulled HIGH.</p>															
SkipInit*	I	<p>Skip Initialization Stage: Active LOW. This pin controls the initialization stage of the GT-48001A. When asserted, the GT-48001A skips the initialization stage (clearing the address table which is stored in the DRAM), upon the deassertion of Rst*. This pin is typically used for testing and the default state is to pull this pin HIGH.</p>															

3. Operational Overview

The GalNet Architecture Family of switching devices has been defined as an extensible, scalable architecture for the switching of packetized data. The GalNet Architecture Family currently supports Ethernet (GT-48001A), Fast Ethernet (GT-48002A), and 100VG-AnyLAN (GT-48003.) Galileo Technology will extend the innovative GalNet family to include other communications standards including WAN in the near future.

All GalNet Architecture Family devices act as distributed intelligent agents within a switching system. Each GalNet device makes switching decisions independent of other devices in the system, and can communicate information regarding the network to all other agents. This distributed processing approach is a significant performance improvement over switching architectures that rely on centralized switching “engines” or single-point address recognition devices. Unlike centralized resource approaches, GalNet designs can actually add packet processing capability as additional ports are added.

The GalNet Architecture Family uses a “store-and-forward” switching approach. Store-and-forward was chosen for the following reasons:

- Store-and-forward switches allow switching between differing speed media (e.g. 10BaseX and 100BaseX.) Such switches require the large elastic buffers that are provided by the EDO DRAM arrays.
- Store-and-forward switches improve overall network performance by acting as a “network cache”, effectively buffering packets during times 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.

A typical unmanaged GalNet Architecture system is extremely simple to implement as shown in Figure 1 on page 6. No CPU is needed as each GalNet device is intelligent and capable of sharing network information and packet data autonomously. A CPU may be added to provide network management capability.

3.1 Enabling/Disabling the GT-48001A

Ports of the GT-48001A can be enabled and disabled depending on the combination of:

- An external hardware pin, EnDev* (LOW - device enabled, HIGH - device disabled)
- EnableDevice, bit 27 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 will be received or transmitted from the serial ports or the PCI bus. Even though ports are disabled, another PCI master can read from and write to the GT-48001A's registers. See Table 1 for enabling or disabling ports of the GT-48001A.

Table 1: Enabling/Disabling Ports of the GT-48001A

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

3.2 Basic Operation

The basic operation of the GT-48001A is quite simple. The GT-48001A 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 be either be local (one of the GT-48001A's ports) or in a different GT-48001A device that

resides on the same PCI bus. If the destination address is not found, the GT-48001A treats the packet as a multicast packet and forward the packet to all ports of all devices in the system specified to forward unknown packets.

The GT-48001A 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-48001A's Address Table, the device adds it to the table (with an indication of on which port the address resides). The GT-48001A then notifies other GalNet devices in the system of the new address via a NEW_ADDRESS message.

3.3 Address Learning

The GT-48001A can learn up to 8K unique MAC addresses. Addresses are stored in the Address Table located in DRAM. The Address Table is managed automatically by the GT-48001A (i.e. a new address is automatically added to the Address Table). The GT-48001A's address learning process is outlined in Section 4.

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

3.4 Packet Buffering

Incoming packets are buffered in the DRAM array. These buffers provide elastic storage for transferring data between low-speed and high-speed segments. The packet buffers are managed automatically by the GT-48001A.

3.5 Packet Forwarding

Once an address has been learned, and the packet is buffered, it must be forwarded. The packet forwarding mechanism for the GT-48001A is handled automatically based on the destination address. Optionally, the CPU can be involved in unicast packet forwarding decisions by using *intervention mode*. If a CPU is utilized for system management functions, multicast packets will be forwarded to the CPU for forwarding decisions.

3.6 The GalNet Protocol

The GT-48001A uses a proprietary inter-chip communication protocol on the PCI bus known as the GalNet Protocol messages. The protocol consists of five groups of messages: NEW_ADDRESS, BUFFER_REQUEST, START_OF_PACKET, PACKET_TRANSFER, and END_OF_PACKET.

All GalNet messages are *write-only*. For example, a GT-48001A may request a buffer location in another GalNet device by writing a BUFFER_REQUEST message to the target device. The target device responds by writing a START_OF_PACKET message to the requesting GT-48001A. Read transactions are strictly avoided since they tend to stall the PCI bus, thereby wasting precious bandwidth.

3.7 Terminology

It is important to understand the basic terminology used to describe the GalNet Architecture Family before getting into the detailed description. Table 2 explains the terms used throughout this document.

Table 2: Terminology

Term	Definition
Address Table	The Address Table is a data structure in the GT-48001A'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 was sent.

Table 2: Terminology

Term	Definition
Device Number	Each GalNet device in a system (including the CPU, if any) has a specific Device Number. There are 32 possible Device Numbers.
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. MAC Address Learning Process

The GT-48001A has a self-learning mechanism for learning the MAC addresses of attached Ethernet devices in real time. The GT-48001A searches for the Source Address (SA) of an incoming packet in the Address Table and acts as follows:

If the SA was not found in the Address Table (a new address), the GT-48001A waits until the end of the packet (Non errored packet) and updates the Address Table. It also notifies the other GT-48001A devices and the CPU by sending a separate NEW_ADDRESS message to each GalNet device on the PCI bus. If a CPU is enabled in the system (bit 10 of the Global Control Register, 0x140028), the NEW_ADDRESS message can optionally be forwarded to the CPU (bit 7 in the Global Control Register, 0x140028). This message contains the new MAC address, the Device Number and the Port Number.

The NEW_ADDRESS message format sent from one GalNet device to another is described in Section 10.2.1. The NEW_ADDRESS message that is sent to the CPU is of a different format and is described in Section 10.3.1. In addition, the GT-48001A asserts Int* to notify the CPU that the Address Table was modified.

1. If the SA was found, the GT-48001A compares all fields of the NEW_ADDRESS message to the entry in the Address Table. If any fields differ (and the 'Static' bit in the Address Table is '0'¹), the GT-48001A updates the entry with the new information in the NEW_ADDRESS message and notifies the other GT-48001A devices and the CPU. If the device and port numbers are equal, the packet is not switched (i.e. this packet was destined for a device on the same network segment and does not require forwarding.)
2. If the SA was found in the Address Table, the Aging bit is set. This is done to indicate to the aging software that this address was accessed recently.

4.1 Address Recognition

The GT-48001A forwards the incoming packets to the appropriate port(s) according to Destination Address (DA) as follows:

1. If the DA is a Unicast address and the address was found in the Address Table, the GT-48001A acts as follows:
 - If the Port Number and the Device Number are equal to the Port/Device on which the packet was received, the packet is discarded.
 - If the Port Number is different, but the Device Number is equal, the packet is forwarded to the appropriate local port.
 - If the Device Number is different, the packet is forwarded to the appropriate GT-48001A device via the PCI bus.
2. If the DA is a Unicast address and the address was not found (Unknown), the GT-48001A acts as if the unknown packet is a Multicast packet and forwards it to ports and the devices that have been programmed to receive unknown packets (bit 4 in the Command register).
3. If the DA is a Multicast address, the packet is forwarded to all the local ports (except for the port in which the packet was received). It is also forwarded to all the other devices via the PCI bus. This procedure is outlined in Section 6.3.

4.2 Recovery Process

The purpose of the Recovery Process is to guarantee that Address Tables entries in all the devices correlate. When the packet is Unknown, the source GT-48001A sends a NEW_ADDRESS message to all the devices. Each device searches its own Address Table for the new address. More than one device can find the address, but only one device "owns" this address (the Device Number written in the Address Table is equal to its own Device Number). This particular device updates the source GT-48001A's Address Table with the new address (by in turn sending it a NEW_ADDRESS message).

1. Static Address Table entries cannot be modified, as described in Section 4.4.

4.3 Address Aging

The GT-48001A includes hardware support for address aging, which requires the use of a CPU to be implemented. The GT-48001A gives an indication to the CPU of the relative “age” of an address by setting the Aging bit in the address table when it receives a packet. The CPU reads the Aging bit each aging period (the default from the 802.12d specification is 300 seconds) and clears the bit. If in the next period, the bit remains clear, the CPU knows that this station didn't transmit any packet in this period of time and the address can be removed from the table.

Note that the aging bit is set only in the GT-48001A device that received the packet; other GT-48001A's in the system are not notified since, by definition, the receiving device “owns” the address.

The only time the GT-48001A will delete an entry by itself is when a user changes the location of a station. The GT-48001A will then automatically learn the new location of the station the next time the station sends a packet.

4.4 Static Addresses

The GT-48001A includes 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 timeout mechanism”. This means that when an address is selected to be static, it will not be removed from the address table during aging.

During normal address recognition, if an address is static, the GT-48001A will not update its address table parameters and will not send a NEW_ADDRESS message over the PCI (if the address has changed ports.)

4.5 Address Recognition Failure

It is possible that an address recognition cycle will fail when more than 8K addresses have already been entered into the address table. In the case of an address recognition failure the packet will be treated as unknown and forwarded to all ports. An interrupt is also generated to the CPU (if any.)

Address recognition failures are not fatal and do not need to be handled (e.g. designers of unmanaged systems need not worry about them.) Managed systems may want to “clean” the address table of old addresses when such a failure occurs (see Section 4.4).

5. GT-48001A Buffers and Queues

The GT-48001A incorporates nine transmit queues for the 8 Ethernet ports and the PCI bus port, and one common receive buffer area (see Figure 2.) The receive buffers as well as the transmit queues are located in the DRAM along with the Address Table. DRAM address mapping in the GT-48001A is shown in Table 3. The GT-48001A data structure components are the following:

- **Receive Buffer** - A common Rx buffer area for all ports. The buffer is divided into 304 or 992 blocks (depending on the DRAM size) of 1.5KBytes (1536 bytes) each. Each block contains the entire packet.
- **Rx Empty List** - A list of 304 or 992 bits. Each bit contains the status of its appropriate receive block in the DRAM (empty or occupied).
- **Tx Descriptors** - A set of 9 transmit descriptor rings. Each ring contains 1024 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** - 9 pairs of pointers to the transmit descriptors.

Figure 2: GT-48001A Buffers and Queues

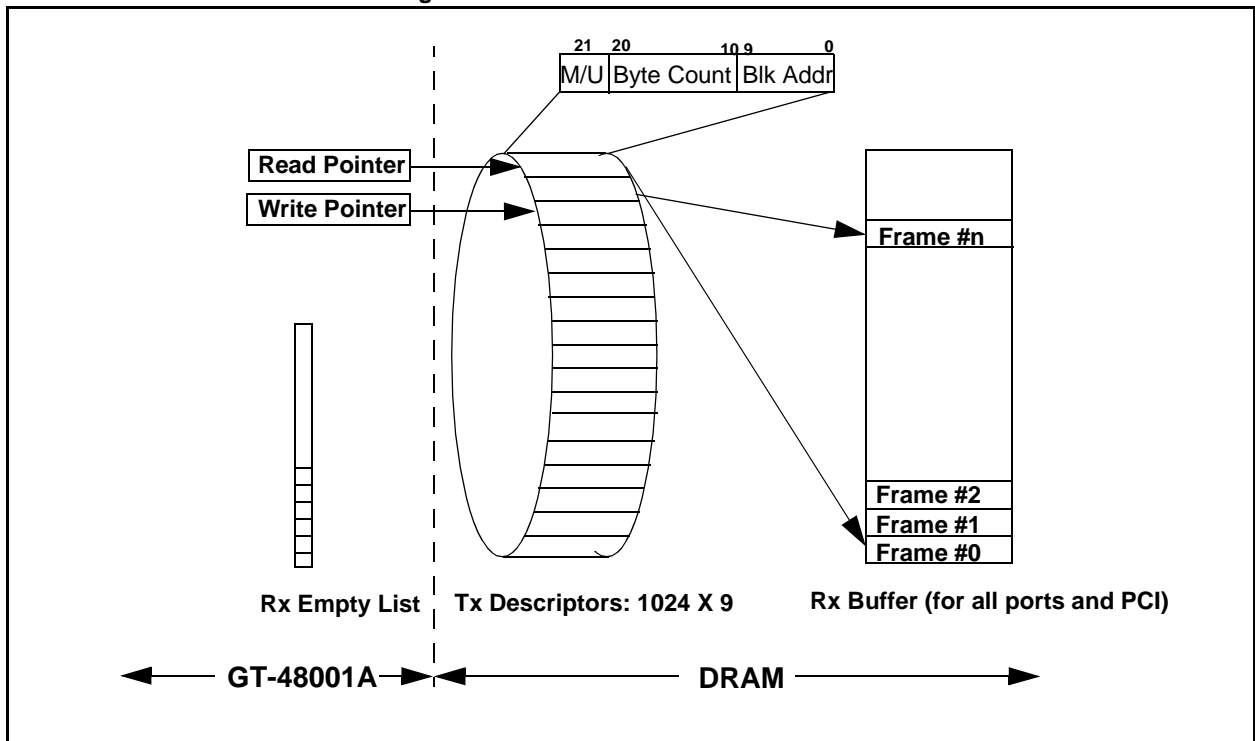


Table 3: GT-48001A DRAM Address Mapping

Memory	Description	1Mbyte	2Mbyte
Rx Buffers	304 Buffers for 1Mbyte 992 Buffers for 2 Mbyte	0x00000 - 0x71fff	0x00000 - 0x171fff
PCI Tx Descriptor		0x77000 - 0x088fff	0x177000 - 0x188fff
PCI Rx Descriptor		0x89000 - 0x94fff	0x189000 - 0x194fff
Reserved	4KBytes	0x95000 - 0x095fff	0x195000 - 0x195fff
Tx Descriptor		0x096000 - 0x09ffff	0x196000 - 0x19ffff
Address Table		0xa0000 - 0xfffff	0x1a0000 - 0x1fffff

5.1 Receive Buffer Threshold Programming

The number of receive buffers allocated to each port of the GT-48001A is controlled by the RxBufThr field in the Rx Buffer Threshold register and the DisBufThr* pin. The default value is 29 buffers per port for 1MB DRAM and 79 buffers per port for 2MB DRAM. If the buffer threshold is disabled (by clearing the BufThrEn bit in the Global Control register or the DisBufThr* pin is held LOW) the GT-48001A dynamically allocates the buffers to the 8 ethernet ports and the PCI bus port. In other words, there are no limits on each buffers' allocation. The RX Buffer Threshold value can be used to tune performance during development. See Table 4 for Rx Buffer Threshold settings. If a received packet overflows the Rx buffer allowance, then that packet will be discarded and the Dropped Packets counter will be incremented.

The overflow of the Rx buffer threshold is also indicated by the "Receive Buffer Full" LED in the Serial and Parallel LED Interfaces (Rx Buffer Threshold must be enabled).

Table 4: Setting the Rx Buffer Threshold

DisBufThr* Pin	BufThrEn Bit	Result
HIGH	1	Receive Buffer Size is limited to the value of RxBufThr
HIGH	0	Dynamic Receive Buffer Allocation
LOW	1	Dynamic Receive Buffer Allocation
LOW	0	Dynamic Receive Buffer Allocation

6. Packet Forwarding

The following sections describe the procedures for forwarding packets in the following situations:

- A unicast packet to a local port in the same GalNet device (Section 6.1)
- A unicast packet between GalNet devices (Section 6.2)
- A multicast packet to local ports in the same GalNet device (Section 6.3.1)
- A multicast packet between GalNet devices in a system without a CPU (Section 6.3.2.1)
- A multicast packet between GalNet devices in a system with a CPU (Section 6.3.2.2)
- A packet destined for the CPU, multicast, or unicast packet from a GalNet device to the CPU (Section 6.4)
- A unicast/multicast packet from the CPU to a GalNet device (Section 6.5)

6.1 Forwarding a Unicast Packet to a Local Port

The sequence for forwarding a unicast packet to a local port (port in the same GT-48001A) is as follows:

1. The incoming packet is fed to the Rx FIFO (there is an 8x32-bit Rx FIFO per port) 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-48001A 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 target GalNet device transmits whenever the Write Pointer is not equal to the Read Pointer.
5. At the end of the packet transmit process, the target GalNet device increments the Read Pointer and clears the appropriate bit in the Empty List.

6.2 Forwarding a Unicast Packet to a Port in a Different GalNet Device

The sequence for forwarding a Unicast packet to a port in a different GalNet device located on the PCI bus is as follows:

1. The incoming packet is fed to the Rx FIFO 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-48001A uses the DA's corresponding Port Number and Device Number to queue the packet to the appropriate GalNet device and port.
3. At the end of an error-free packet transfer, packet information is written to the PCI's transmit descriptor. This information includes the byte count and Receive Buffer Block Address which is pointed to by the Write Pointer. When the PCI's transmit descriptor's Write Pointer is not equal to the Read Pointer, the source GalNet device sends a BUFFER_REQUEST message (Section 10.2.2) to the appropriate target GalNet device indicating that there is a packet for transmission across the PCI bus.
4. The target GalNet device receives this message and allocates a buffer in its DRAM. This target device then sends a START_OF_PACKET message (Section 10.2.3) back to the source GT-48001A indicating it is ready to receive the packet.
5. The source GT-48001A transfers the packet with the PACKET_TRANSFER message (Section 10.2.4) using PCI master operations in multiple eight 32-bit bursts. The packet is buffered in the Receive Buffer area of the target device's DRAM. After the entire packet has been transmitted, the source GT-48001A performs an additional write transaction by sending the END_OF_PACKET message (Section 10.2.5) indicating completion of the packet transfer. This message contains the Byte Count, the target Port Number, the Rx Block address, and the Packet

Type. The source GT-48001A also clears the appropriate bit in its Empty List.

6. Some packet information included in the END_OF_PACKET message is written to the appropriate transmit descriptor in the target device. This information includes the Byte Count and the Receive Buffer address which is pointed to by the Write Pointer.
7. The Write Pointer of the outgoing port's transmit descriptor is incremented. The target GalNet device transmits whenever the Write Pointer is not equal to the Read Pointer.
8. At the end of the packet transmit process, the target GalNet device increments the Read Pointer and clears the appropriate bit in the Empty List.

6.3 Forwarding a Multicast Packet

The GT-48001A forwards Multicast packets to all local ports and devices using the same mechanism as described for Unicast packets. The GT-48001A has the ability to forward multicast packets to a management CPU for intervention routing, if desired.

6.3.1 Local Ports

For local ports in the same device, the packet is queued to all transmit ports except for the port which the packet arrived and the packet is transferred with the same procedure outlined in Section 6.1 to each port.

6.3.2 Between GalNet Devices

Forwarding multicast packets to other GalNet devices is handled differently depending if a CPU is disabled or enabled in the system or not.

6.3.2.1 CPU Disabled

Systems which do not utilize a CPU (bit 10 of the Global Control Register, 0x140028, is not set) will automatically forward multicast packets to all of the local ports and devices with the following procedure:

1. The incoming packet is fed to the Rx FIFO and is transferred to an empty block in the Receive Buffer area of DRAM.
2. In parallel, an address recognition cycle is performed for the SA. The DA marks this packet as a multicast packet. At the end of a good packet transfer, packet is forwarded to all of the local ports according to Section 6.3.1. This multicast packet is also forwarded to the other GalNet devices in the system with the same procedure as forwarding a unicast packet from one device to another. This procedure is outlined in Section 6.2. There is a single, separate, multicast packet transfer from the source GT-48001A to each of the GalNet devices in the system. Bit 21 of Data 0 of the BUFFER_REQUEST message (Section 10.2.2) will be set to indicate this is a multicast packet.

6.3.2.2 CPU Enabled

Systems which utilize a CPU (bit 10 of the Global Control Register, 0x140028, is set), will **always** have multicast packets forwarded to the CPU. This allows the CPU to intervene, if necessary, and redirect or update the multicast packet before forwarding. Control of forwarding multicast packets to all of the ports is set by bit 22 of the Global Control Register. The default setting is to forward all multicast packets to all of the ports in the system as well as the CPU. If this bit is set, multicast packets will go *only* to the CPU.

Assuming bit 22 is not set and all multicast packets are forwarded to the CPU as well as all of the ports, the procedure for handling multicast packets is as follows:

1. The incoming packet is fed to the Rx FIFO 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 DA marks this packet as a multicast packet. At the end of a good packet transfer, packet is forwarded to all of the local ports according to Section 6.3.1. This multicast packet is also forwarded to the other GalNet devices in the system with the same procedure as forwarding a unicast packet from one device to another. This procedure is outlined in Section 6.2. There is a single, separate, multicast packet transfer from the source GT-48001A to each of the GalNet devices in the system. Bit 21 of Data 0 of the BUFFER_REQUEST message (Section 10.2.2) will be set to indicate this is a multicast packet.
3. Packet information is also written to the PCI's transmit descriptor which instructs the GT-48001A to send this multicast packet to the CPU. This multicast packet is then forwarded directly to the CPU with the procedure outlined in Section 6.4.

Again, if bit 22 is set, all multicast packets will *only* be forwarded to the CPU and not to the local ports, nor other ports of other GalNet devices. The CPU can then decide to what ports the multicast packet should be sent to. Only one packet needs to be sent to each GalNet device and each device will automatically forward the packet only to the ports that the CPU tagged for that specific Multicast packet. These ports are tagged in bits [29:22] of the END_OF_PACKET message.

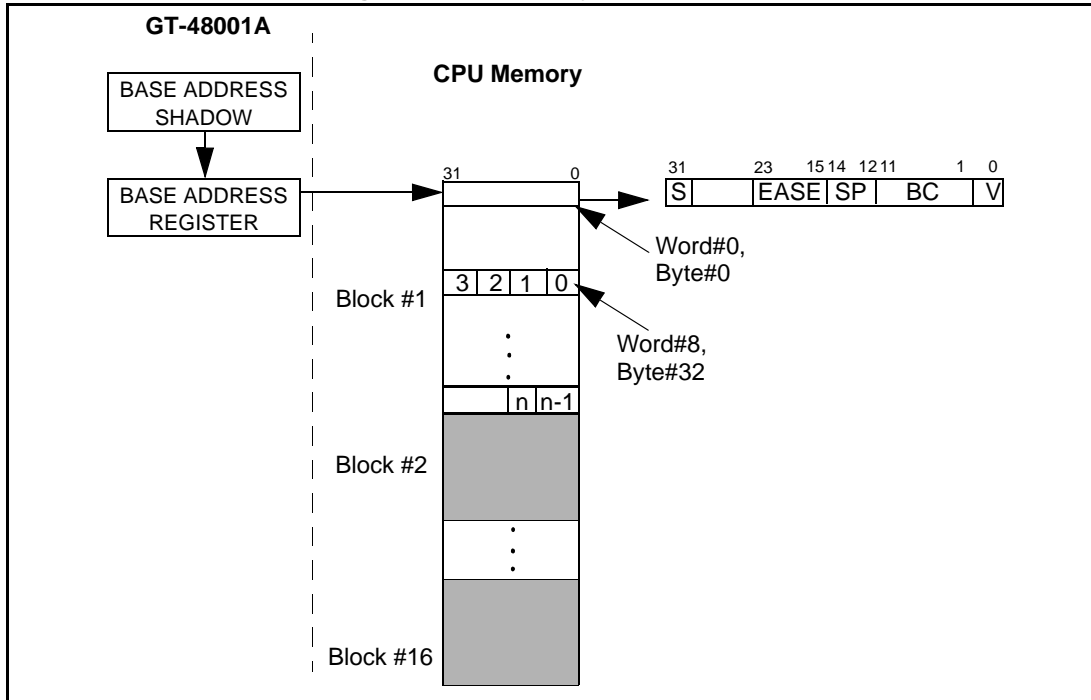
6.4 Forwarding a Packet to the CPU Directly

Systems which utilize a CPU (bit 10 of the Global Control Register, 0x140028, is set) will forward certain packets directly to the management CPU's memory. These packets include:

- Unicast packets destined for the CPU (Device Number in the address table is equal to the CPU number)
- Multicast packets
- Unknown packets (if set by bit 6 in the Global Control Register, 0x140028)
- BPDU messages
- Sniffer packets when the CPU is the target sniffer
- EASE packets

The GT-48001A contains two pointers to a sixteen block buffer area in the CPU's memory space. The registers are called the CPU Base Address (CBA) and CPU Base Address Shadow (CBAS.) These registers are physically located at the same address in the GT-48001A (0x140034). The first write to this register updates the CBA register, the second write updates the shadow register (CBAS). Figure 3 shows the data structure in the CPU memory.

Figure 3: CPU Memory Data Structure



The data structure components are the following:

- **CPU Base Address Register (CBA)**¹ - A register that points to the beginning of a sixteen block area in CPU memory.
- **CPU Base Address Shadow Register (CBAS)** - A second register that holds a pointer to a second sixteen block area in the CPU main memory. The value in the Shadow register propagates into the Base Address register after sixteen packets are transferred to the main memory.
- **Buffer Area** - The CPU buffer area consists of 16 blocks of 2Kbytes each. The first word (Word #0) of each block contains the Sniffer Indication [31], EASE indication [23:15], Source Port Number [14:12], Byte Count (bits [11:1]), and the Valid bit (bit 0). These bits are written as the END_OF_PACKET (Section 10.3.10) message transferred from the source GT-48001A to the CPU at the end of a valid packet transfer. Words 1 to 7 are left empty for user purposes.

The communication between the GT-48001A and the CPU follows this sequence:

1. Upon initialization, CPU updates the CBA (1st write to 0x140034).
2. Following the initial write to the CBA, the CPU updates the CBAS (2nd write to 0x140034).
3. GT-48001A transfers 16 packets to the CPU main memory and asserts the Int* at the end of each packet transfer.
4. The CPU must count sixteen interrupts and then update the Shadow register in the GT-48001A (write to 0x140034). Also, the BufWrap interrupt can be checked instead of counting sixteen interrupts. In other words, the CPU must update 0x140034 ONCE after 16 packets have been transferred from the GT-48001A to the CPU. This single word write after sixteen packets have been transferred to the CPU are to update the Shadow register only.

1. Assuming CBA and CBAS have already been written to, any writes to 0x140034 will update the CBAS register ONLY. In other words, CBA can only be updated on the first write, and after 16 packets have been written to the CPU where the CBA will take the value of the CBAS.

Steps 3-4 are repeated. The packet transfer to the CPU is done as follows:

1. The incoming packet is fed to the Rx FIFO 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. If the packet is resolved to be a packet of the type listed at the beginning of this section, packet information is written to the PCI's transmit descriptor. This information includes the byte count and Receive Buffer Block Address which is pointed to by the Write Pointer. When the PCI's transmit descriptor's Write Pointer is not equal to the Read Pointer, the source GT-48001A sends the packet DIRECTLY to the appropriate block in the CPU main memory with the PACKET_TRANSFER (Section 10.3.7) message using PCI master operations in multiple eight 32-bit bursts. The data is entered starting at the 8th word (33rd byte) of the next free block. Words 1 to 7 are left empty for user purposes. The address of this packet transfer is based upon the CBA.
3. At the end of the packet transfer, the source GT-48001A sends the END_OF_PACKET message (Section 10.3.10) to the first word of the block (Word #0). It also sends an interrupt via Int* to the CPU, increments the Read Pointer, and clears the appropriate bit in its Empty List.

The CPU now has the packet buffered in its buffer area. This gives the CPU the ability to intervene in the packet's routing or to modify the contents of the packet.

6.5 Forwarding a Packet from the CPU to a GalNet Device

The sequence for forwarding a packet from the CPU to a port in a GalNet device is the same for unicast and multicast packets. The procedure is as follows:

1. The CPU sends a BUFFER_REQUEST message (Section 10.3.4) to the appropriate target GalNet device indicating that there is a packet ready for transmission across the PCI bus.
2. The target GalNet device receives this message and allocates a buffer in its DRAM. This target device then sends a START_OF_PACKET message (Section 10.3.5) back to the CPU indicating it is ready to receive the packet. This START_OF_PACKET message is sent to a CPU buffer location indicated by the Start Packet Base Address register (0x140038). This address points to a CPU buffer area that can hold up to 32 START_OF_PACKET messages. The CPU must poll this structure to know when a target GT-48001A is ready to receive a packet, and to what address the packet should be sent. If the Byte Count field in the START_OF_PACKET message is 0, then the CPU should not write the PACKET_TRANSFER message nor the END_OF_PACKET message to the device. This indicates that either the buffers are full in the target GalNet device, or that the link is down on the target port.
3. The CPU transfers the packet with the PACKET_TRANSFER message (Section 10.3.9) using PCI master operations in multiple eight 32-bit bursts. The packet is buffered in the Receive Buffer area of the target device's DRAM. After the entire packet has been transmitted, the CPU performs an additional write transaction by sending the END_OF_PACKET message (Section 10.3.12) indicating completion of the packet transfer. This message contains the Byte Count, the target Port Number, the Rx Block address, and the Packet Type. It also includes a bit which commands the GT-48001A to generate CRC for this out-going packet or not. If the packet is a multicast packet, the packet will only be forwarded to the ports tagged in bits [29:22] of the END_OF_PACKET message (The CPU should NEVER write '1' to bits 29:22 that had '0' in in the same bits of the START_OF_PACKET message.)
4. Some packet information included in the END_OF_PACKET message is written to the appropriate transmit descriptor in the target device. This information includes the Byte Count and the Receive Buffer address which is pointed to by the Write Pointer.
5. The Write Pointer of the outgoing port's transmit descriptor is incremented. The target GalNet device transmits whenever the Write Pointer is not equal to the Read Pointer.
6. At the end of the packet transmit process, the target GalNet device increments the Read Pointer and clears the appropriate bit in the Empty List.

6.6 CRC Generation

As mentioned in Section 6.5, the GT-48001A also includes a CRC generator for packets sent out by the CPU. The CRC generator enhances system performance by implementing the CPU intensive packet CRC calculation in hardware. Note that CRC generation is not required for packets transferred between GalNet 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 (bit 29). In addition, the CPU must set the GenCRC bit in the END_OF_PACKET (Section 10.3.12) message sent to the GT-48001A at the completion of forwarding the packet.

6.7 Tx Watchdog Timer

The GT-48001A includes a transmit watchdog timer for each transmit queue. The default value of the timer is 630 msec; the range is 105 to 1575 msec in steps of 105 msec. The timer measures the time between the transmission of two consecutive outgoing packets. When the timer expires, the GT-48001A 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 enabled (DisWD* HIGH.)

7. Device Table Operation

The GalNet Architecture supports a maximum of 32 devices per system. Each device needs to know of the existence of all other GalNet devices in the system in order to communicate information and packet data. The Device Table is a 32-bit register that uses a single bit for each possible Device Number to indicate its presence/absence in the system.

7.1 Automatic Device Table Initialization

Upon RESET, each GT-48001A in the system sets all of its Device Table bits to '1'. This is essentially a "guess" by the GT-48001A that *all* Device Numbers are in use. In a system with any PCI device that is NOT a GalNet device (CPU, graphics card, etc.), it is recommended that the CPU initialize the device table before the GalNet device is enabled (or before packet transmission starts).

Specific Device Table bits are cleared, and the corresponding Device Numbers are logically eliminated from the system, by:

- The receipt of a PCI Master Abort when the GT-48001A attempts to access the corresponding Device Number
- Management CPU programming

Device Table "cleaning" is handled automatically by each device upon receipt of the first unknown packet following a RESET. This process discovers all other GalNet devices in the system without processor intervention:

1. The GT-48001A receives the first good packet from any Ethernet port and places it in the buffer area in DRAM.
2. Since the Address Table is cleared (immediately following RESET) the packet must be forwarded to all ports, including those located on other GalNet devices.
3. The GT-48001A attempts to allocate a buffer in all 31 possible additional GalNet devices in the system. This is done by issuing 31 BUFFER_REQUEST messages- one to every possible device in the system.
4. Each BUFFER_REQUEST message that fails due to a PCI Master Abort results in the corresponding Device Number bit being cleared in the Device Table. A PCI Master Abort only occurs when there is no target device at the requested address (i.e. the target device doesn't exist.)

7.2 Manual Device Table Initialization

Alternatively, the CPU can notify each GT-48001A of the existence of the GalNet devices. This is done by setting the by setting bit 2 in the global control register (DevTabMod.) In this mode, the CPU is responsible for notifying the GT-48001A of the existence of all GT-4800x devices in the system. The CPU does this by writing the appropriate values to each device table.

"Manual" mode must be used when there are multiple PCI buses in the system separated via PCI-to-PCI bridges.

7.3 Programming Device Numbers

The Device Number for each GalNet device is set after RESET by the values on the DAddr[4:0] pins. The device number can be changed by writing the new device number to [26:22] of the DRAM/Internal Register Base Address Register at 0x010. Typically, once the number of a device has been changed, the device table of all devices in the system will also need to be updated.

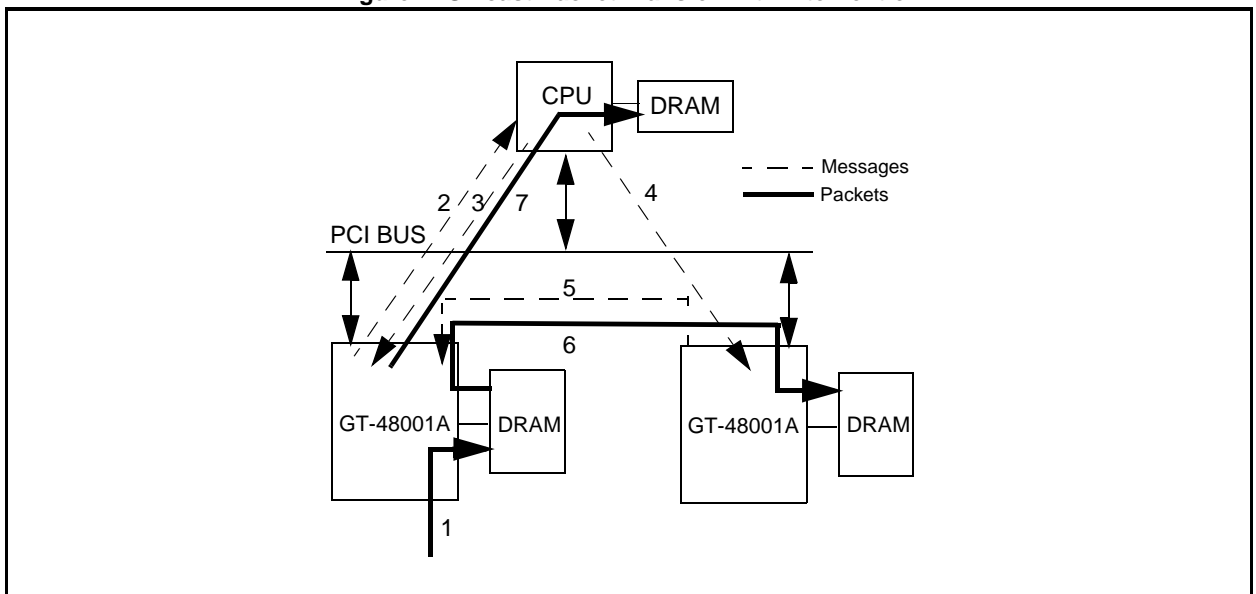
8. Unicast Intervention Mode

The GT-48001A supports a powerful feature called Intervention Mode for unicast packets. Intervention in Unicast traffic is optional per MAC address. Each entry in the Address Table includes an Intervention bit for the destination address (bit 62) and the source address. When an Intervention mode bit is set, the GT-48001A will *not* forward the packet automatically to the destination device. Instead, the GalNet device will send a BUFFER_REQUEST message (Section 10.3.3) to CPU memory. The BUFFER_REQUEST messages will be sent to the buffer area in the CPU main memory which contains 256 entries of dual 32-bit words. The location of this CPU memory is specified by the CPU Intervention Base Address at 0x140048. The buffer area is pointed to by this Base Address register and a Shadow register at this same offset. The BUFFER_REQUEST message includes information about the routing of the packet (Source and Target port/device numbers). The CPU then has the option to:

- discard the packet
- forward the packet to a specific device
- request the packet be transferred to CPU buffer memory

Figure 4 shows a Unicast packet transfer using Intervention mode.

Figure 4: Unicast Packet Transfer With Intervention



The sequence illustrated is as follows:

1. The incoming packet is fed to the Rx FIFO and transferred to an empty block in the Receive Buffer area of DRAM.
2. If a corresponding Intervention bit is set, the device sends a BUFFER_REQUEST message (Section 10.3.3) to the CPU's Intervention Base Address. The BUFFER_REQUEST includes the source port and the source buffer address

The CPU then has the following options:

1. Discard the packet.
 - The CPU sends a START_OF_PACKET message (Section 10.3.6) to the source device with the Byte Count field cleared with all 0s (arrow #3).
2. Forward the Packet to a port in the same device

- The CPU sends a BUFFER_REQUEST (Section 10.2.2) to back to the same device where the target device number is equal to the source device number. The correct port number is also specified. This BUFFER_REQUEST is of the same format that is used between GalNet devices (arrow #2).
 - The packet is then sent out on a port of same source device as specified by the CPU.
3. Forward the packet to a different device.
- The CPU sends a BUFFER_REQUEST (Section 10.2.2) to the destination device. This BUFFER_REQUEST is of the same format that is used between GalNet devices and uses the source device number as the Source Device instead of the CPU device number (arrow #4).
 - The target device allocates a buffer and sends a START_OF_PACKET message (Section 10.2.3) to the source device (arrow #5).
 - The source device transfers the packet with the PACKET_TRANSFER message (Section 10.2.4) followed by an END_OF_PACKET message (Section 10.2.5) to the target device (arrow #6).
4. Take the packet.
- The CPU sends a START_OF_PACKET message (Section 10.3.6) back to the source GalNet device. The target device in this message is the CPU number. The source GT-48001A device transfers the packet with the PACKET_TRANSFER message (Section 10.3.8) using PCI master operations in multiple eight 32-bit bursts. The first data word of the packet will be written to the second word of the buffer. The first word is left empty for the END_OF_PACKET message. NOTE: Unicast packets tagged for intervention mode are **not** forwarded to the same buffer space that is specified by the CPU Buffer Base Address (CBA) at offset 0x140034).
 - At the end of the packet transfer, the source GT-48001A sends the END_OF_PACKET message (Section 10.3.11) to the first word of the buffer. It also sends an interrupt via Int* to the CPU, and clears the appropriate bit in its Empty List.

The CPU buffer now contains the packet whose destination address was marked for intervention. The CPU can perform a number of functions with the packet including layer 3 routing, security, virtual LAN support, filtering and management. After the CPU has modified the packet, the packet can be transferred to the appropriate target GalNet device with the procedure outlined in Section 6.5.

8.1 Unicast Intervention Mode Address Space

Prior to the CPU receiving unicast packets tagged for intervention mode, an address space for GalNet devices to forward packets to must be allocated. This address space should reside in the GalNet protocol region (see Section 10.1). Therefore, this buffer region will have a similar address to a GalNet device. The buffer address format is shown in Table 5.

Table 5: Buffer Address for Unicast Intervention Mode Packets

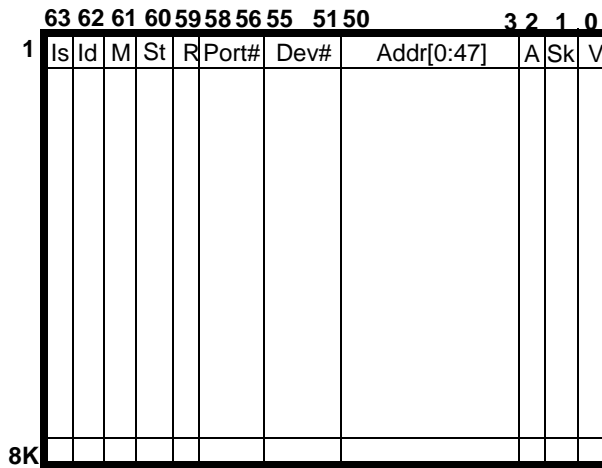
PCI Bits	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	CPU Device Number (bits [12:8] in 0x140030)
[21]	'1' (DRAM)
[20:0]	DRAM location

9. Address Table

The GT-48001A's Address Table resides in the local DRAM array. Figure 5 shows the Address Table structure. The Address Table structure occupies approximately 320Kbytes and is entirely controlled and initialized by the GT-48001A. Following RESET, the GT-48001A initializes the Address Table by setting all Valid Bits in the table to '0'. Every new address that is learned has its entry's Valid Bit set to '1'. In order to remove an address entry from the table, ONLY the Skip Bit should be set to '1' (the Valid Bit should not be modified).

Modifications to the Address Table are normally made through GalNet protocol message requests by the CPU. It is possible to access the Address Table directly, however, this mode is not recommended. Please contact Galileo if you feel your application requires direct Address Table access.

Figure 5: Address Table Format



Address Table Bits Description

Bit	Description
V	Valid: Indicates this entry is taken or previously was taken 0 - Not valid 1 - Valid
Sk	Skip - Indicates if this entry is currently being used 0 - Active entry, do not skip 1 - Inactive entry, overwriting this entry is allowed
A	Aging - This bit is used for the Aging process 0 - Cleared by the CPU 1 - Set by the GT-48001A upon receiving a packet from the station corresponding to this entry
Addr	48 bit MAC address
Dev#	Device Number - 5 bit GalNet device number indicating which of a maximum of 32 devices in the system is associated with this address.
Port#	Port Number - 3 bit Port number indicating which of the eight ports in a GT-48001A is associated with this address.
R	Reserved
St	Static - Indicates whether an entry can be modified or not 0 - The entry can be modified 1 - The entry is static, the Dev# and Port# cannot be modified
M	Multiple - Meaningful when bit St is set 0 - Forward this packet only to the destination port 1 - Forward this packet to all ports (as Unknown)
Id	Intervention for Destination Addresses 0 - Don't activate Intervention mode for this address as a destination 1 - Activate Intervention mode
Is	Intervention for Source Addresses 0 - Don't activate Intervention mode for this address as a source 1 - Activate Intervention mode

10. GalNet Messaging Protocol

The GalNet Messaging Protocol is comprised of the messages passed over the PCI bus between

- GalNet family members
- a GalNet family member and a CPU

.The messages are encoded in both the address and the data phases of the PCI transfers. Five groups of messages that are currently defined: NEW_ADDRESS, BUFFER_REQUEST, START_OF_PACKET, PACKET_TRANSFER, and END_OF_PACKET.

GalNet messages are write-only (request/response). For example, a GalNet device that needs to transfer data to another GalNet chip starts the transfer by requesting a buffer (BUFFER_REQUEST) from the target. The target responds with an address to which the packet should be transferred to (START_OF_PACKET).

The GalNet messaging protocol allows messages to be interleaved. For example, a device may send out multiple BUFFER_REQUEST messages to other devices before receiving a START_OF_PACKET message reply.

Write only messaging was used since it better utilizes PCI bus bandwidth. Reads on the PCI bus tend to stall the bus, and the reading device thereby degrading overall system performance.

Please note that there are subtle differences in the GalNet message format when transferring messages between devices (Section 10.2) and between a device and a CPU (Section 10.3).

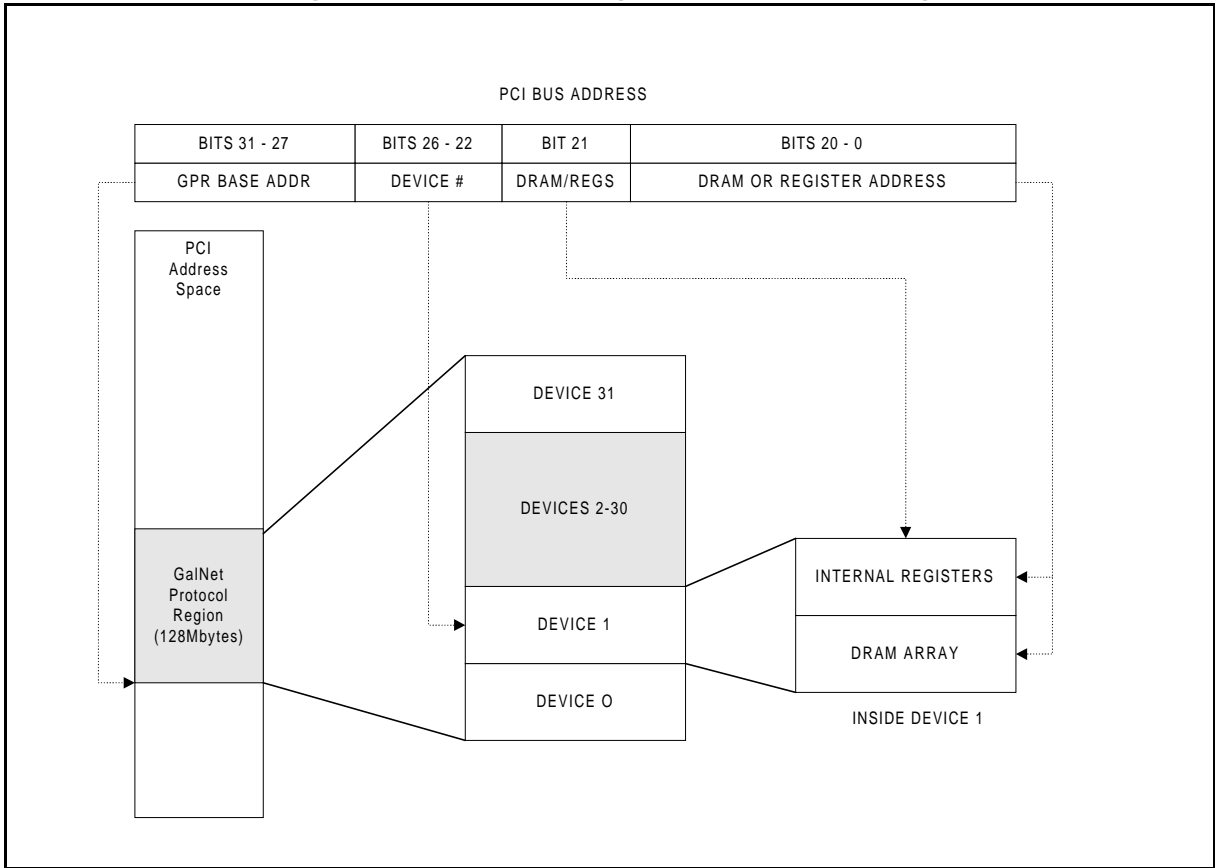
10.1 GalNet Protocol Region

All GalNet devices in a system must reside within a single 128Mbyte region in the PCI memory address space known as the GalNet Protocol Region (GPR). The base address of the GPR is set in bits 31:27 of the DRAM/Internal Base Address Register at offset 0x10 in the GT-48001A's PCI configuration header.¹ All GalNet devices in a system MUST have the same value in this field. GalNet devices default to a value of '00001' in the GPR bits following RESET.

Each GalNet device in the system occupies a separate 4Mbyte "slice" of the GalNet Protocol Region. This "slice" is decoded by the Device Number field in the DRAM/Internal Base Address Register (bits 26:22). The individual device's address space is further divided by the DRAM/Register bit (bit 21). This bit determines whether an access to a target device is directed to the DRAM array or to the registers.

1. Offset 0x10 in the PCI configuration register is known as Base Address Register 0 for standard PCI devices.

Figure 6: GalNet Protocol Region and Address Decoding



10.2 GalNet Messages Between Devices

The five groups of GalNet messages as described in the sections below are sent from one GalNet device to another. The data format for all GalNet messages is *little-endian*, which is the PCI standard.

10.2.1 NEW_ADDRESS Message between GalNet devices

A message sent from one GalNet device to another to alert of a new address.

Table 6: NEW_ADDRESS Message between GalNet devices

PCI Bits	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	Target Device Number
[21]	'0' (Internal registers)
[20:18]	'100' (NEW_ADDRESS message)
[17:0]	'0'
Data 0	
[31:3]	MAC address [19:47]
[2]	Aging
[1]	Skip
[0]	Valid
Data 1	
[31]	'1'
[30]	Static
[29]	Address Unknown/New address
[28]	Multiple
[27]	'0'
[26:24]	Port Number
[23:19]	Device Number
[18:0]	MAC address [0:18]
Data 2	
[24]	Intervention mode for Destination Address
[25]	Intervention mode for Source Address

The NEW_ADDRESS message may also be posted as a 'question' to other GalNet devices during the recovery process (see Section 4.2 on page 16). The indication is via the Address Unknown/New Address bit (bit 29). If bit 29 is clear, then the NEW_ADDRESS message is a 'question'. When bit 29 is set, the message indicates a 'real' address to put into the target devices' address tables.

The NEW_ADDRESS message must be transferred in a burst of three words over the PCI.

10.2.2 BUFFER_REQUEST Message between GalNet devices

A message sent from a source GalNet device to a target GalNet device to request a buffer.

Table 7: BUFFER_REQUEST Message between GalNet devices

PCI Bits	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	Target Device Number
[21]	'0' (Internal registers)
[20:18]	'101' (BUFFER_REQUEST message)
[17:13]	Source Device Number
[12:0]	'0'
Data 0	
[31]	Sniffer (0-Sniffer type message)
[30]	Unknown (0-Unknown message)
[29:28]	'0'
[27:25]	Source Port Number
[24:22]	Target Port Number
[21]	Multicast/Unicast (0-Unicast)
[20:10]	Byte Count
[9:0]	Source Buffer Address (divided by 0x600)

10.2.3 START_OF_PACKET Message between GalNet devices

A message from a target GalNet device to the source GalNet device which contains the Empty Receive Buffer address.

Table 8: START_OF_PACKET Message between GalNet devices

PCI Line	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	Source Device Number
[21]	'0' (Internal registers)
[20:18]	'110' (START_OF_PACKET message)
[17:0]	'0'
Data 0	
[31]	Sniffer (0 - Sniffer type message)
[30]	'0'
[29:22]	Target Port Number (1bit per each port; bit 22 - Port 0, bit 23 - Port 1 etc.)
[21]	Multicast/Unicast (0-Unicast)
[20:10]	Byte Count
[9:0]	Target Buffer Address (divided by 0x600)
Data 1	
[31:18]	'0'
[17:15]	Source Port Number
[14:5]	Source Buffer Address (divided by 0x600)
[4:0]	Target Device Number

The START_OF_PACKET message must be transferred in a burst of two words over the PCI.

10.2.4 PACKET_TRANSFER Message between GalNet devices

A burst of up to 8 32-bit words from the source GalNet device to the target GalNet device which contains the packet.

Table 9: PACKET_TRANSFER Message between GalNet devices

PCI Line	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	Target Device Number
[21]	'1' (DRAM)
[20:0]	DRAM location
Data 0	
[31:0]	Data 0
----	----
----	----
Data 7	
[31:0]	Data 7

The PACKET_TRANSFER message can be transferred using any burst size over the PCI. The GalNet device uses a burst of 8 words.

10.2.5 END_OF_PACKET Message between GalNet devices

A message from the source GalNet device to the target GalNet device which indicates the end of the packet.

Table 10: END_OF_PACKET Message between GalNet devices

PCI Line	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	Target Device Number
[21]	'0' (Internal registers)
[20:18]	'111' (END_OF_PACKET message)
[17:0]	'0'
Data 0	
[31]	CRC (0 - Do not append, 1 - Append) Always '0', GalNet never appends CRC
[30]	'0'
[29:22]	Target Port Number (1bit per each port; bit 22 - Port 0, bit 23 - Port 1 etc.)
[21]	Multicast/Unicast (0-Unicast)
[20:10]	Byte Count
[9:0]	Target Buffer Address (divided by 0x600)

10.3 GalNet Messages Between a GalNet Device and a CPU

When a management CPU is used in a GalNet switch system, bit 10 (CPUEn) of the Global Control Register, 0x140028, must be set.

The five groups of GalNet messages described in the sections below are sent from one GalNet device to a CPU and from a CPU to a GalNet device.

10.3.1 NEW_ADDRESS Message (GalNet to CPU)

A message from a GalNet device to the CPU that contains information about a new MAC Address. Bit 7 (ForwNewAdd) of the Global Control Register (0x140028) must be set.

Table 11: NEW_ADDRESS Message (GalNet to CPU)

PCI Bits	Description
Address	
[31:8]	NEW_ADDRESS Base Address (Offset: 0x14003c)
[7:3]	Offset Counter (32 total entries)
[2:0]	'000'
Data 0	
[31:3]	MAC address [19:47]
[2]	'1'
[1]	'0'
[0]	'0'
Data 1	
[31]	'1'
[30]	Reserved
[29]	Address Unknown/New address (0 - Unknown, 1 - New Address Message)
[28]	Reserved
[27]	Reserved
[26:24]	Port Number
[23:19]	Device Number
[18:0]	MAC address [0:18]

The NEW_ADDRESS message must be transferred in a burst of two words over the PCI from a GalNet device to the CPU.

10.3.2 NEW_ADDRESS Message (CPU to GalNet)

A message from the CPU to a GalNet device that contains information about a new MAC Address.

Table 12: NEW_ADDRESS Message (CPU to GalNet)

PCI Bits	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	Target Device Number
[21]	'0' (Internal registers)
[20:18]	'100' (NEW_ADDRESS message)
[17:0]	'0'
Data 0	
[31:3]	MAC address [19:47]
[2]	Aging
[1]	Skip
[0]	Valid (should always be '1')
Data 1	
[31]	'1'
[30]	Static
[29]	Address Unknown/New address (0 - Unknown, 1 - New Address Message)
[28]	Multiple
[27]	'0'
[26:24]	Port Number
[23:19]	Device Number
[18:0]	MAC address [0:18]
Data 2	
[24]	Intervention Mode for Destination Address
[25]	Intervention Mode for Source Address

The NEW_ADDRESS message must be transferred in a burst of three words over the PCI.

Data 2 must be sent when sending the NEW_ADDRESS message. After every NEW_ADDRESS message sent by the CPU, a dummy NEW_ADDRESS message (with a non existing MAC address) should be sent with Data2 cleared (all 0s). The dummy MAC address can be all zeros or another MAC address that doesn't exist in the network.

10.3.3 BUFFER_REQUEST Message (GalNet to CPU)

A message from the source GalNet device to the target CPU to request a buffer. This message is only used when a unicast packet's destination address is marked for Intervention Mode and will be forwarded to the CPU buffer.

Table 13: BUFFER_REQUEST Message (GalNet to CPU)

PCI Bits	Description
Address	
[31:11]	CPU Intervention Base Address (Offset: 0x140048)
[10:3]	Offset Counter (256 total entries)
[2:0]	'000'
Data 0	
[31]	Sniffer (0-Sniffer type message)
[30]	Unknown (0-Unknown destination address)
[29:28]	'0'
[27:25]	Source Port Number
[24:22]	Target Port Number
[21]	Multicast/Unicast (0-Unicast)
[20:10]	Byte Count
[9:0]	Source Buffer Address (divided by 0x600)
Data 1	
[31:29]	'0'
[28:24]	Target Device Number
[23:0]	'0'

The BUFFER_REQUEST message must be transferred in a burst of two words over the PCI.

10.3.4 BUFFER_REQUEST Message (CPU to GalNet)

A message from the source CPU to the target GalNet device to request a buffer.

Table 14: BUFFER_REQUEST Message (CPU to GalNet)

PCI Bits	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	Target Device Number
[21]	'0' (Internal registers)
[20:18]	'101' (BUFFER_REQUEST message)
[17:13]	Source CPU Device Number (bits [12:8] in 0x140030)
[12:0]	'0'
Data 0	
[31]	Sniffer (0-Sniffer type message)
[30]	Unknown (0-Unknown destination address)
[29:28]	'0'
[27:25]	Reserved
[24:22]	Target Port Number
[21]	Multicast/Unicast (0-Unicast)
[20:10]	Byte Count
[9:0]	Reserved

10.3.5 START_OF_PACKET Message (GalNet to CPU)

A message from a target GalNet device to the CPU which contains the target Buffer address.

Table 15: START_OF_PACKET Message (GalNet to CPU)

PCI Bits	Description
Address	
[31:8]	START_OF_PACKET Base Address (0x140038)
[7:3]	Offset Counter (32 total entries)
[2:0]	'000'
Data 0	
[31]	Sniffer (0 - Sniffer type message)
[30]	'0'
[29:22]	Target Port Number (1bit per each port; bit 22 - Port 0, bit 23 - Port 1 etc.)
[21]	Multicast/Unicast (0-Unicast)
[20:10]	Byte Count
[9:0]	Target Buffer Address (divided by 0x600)
Data 1	
[31:0]	'0'

The START_OF_PACKET message must be transferred in a burst of two words over the PCI.

10.3.6 START_OF_PACKET Message (CPU to GalNet)

A message from the target CPU to the source GalNet device which contains the Empty Buffer address. This message will only be sent when the GalNet devices needs to forward a unicast packet to the CPU via Intervention mode.

Table 16: START_OF_PACKET Message (CPU to GalNet)

PCI Bits	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	Source Device Number
[21]	'0' (Internal registers)
[20:18]	'110' (START_OF_PACKET message)
[17:0]	'0'
Data 0	
[31]	Sniffer (0 - Sniffer type message)
[30]	'0'
[29:22]	Reserved
[21]	Multicast/Unicast (0-Unicast)
[20:10]	Byte Count
[9:0]	Target Buffer Address (divided by 0x600)
Data 1	
[31:18]	'0'
[17:15]	Source Port Number
[14:5]	Source Buffer Address (divided by 0x600)
[4:0]	Target CPU Device Number (bits [12:8] in 0x140030)

The START_OF_PACKET message must be transferred in a burst of two words over the PCI.

10.3.7 PACKET_TRANSFER Message (GalNet to CPU 16 Block Buffer)

A burst of up to 8 32-bit words from the source GalNet device to the target CPU buffer which contains the packet. This message is used when the GalNet device forwards one of the following packets directly to the CPU 16 block buffer:

- Unicast packets destined for the CPU (Device Number in the address table is equal to the CPU number)
- Multicast packets
- Unknown packets (if set by bit 6 in the Global Control Register, 0x140028)
- BPDU messages
- Sniffer packets when the CPU is the target sniffer
- EASE packets

The data is entered starting at the 8th word (33rd byte) of the next free block. Words 1 to 7 are left empty for user purposes. The address of this packet transfer is based upon the CPU Buffer Base Address.

Table 17: PACKET_TRANSFER Message (GalNet to CPU)

PCI Bits	Description
Address	
[31:15]	CPU Buffer Base Address (0x140034)
[14:11]	Offset Counter (16 total blocks)
[10:2]	Address within the buffer
[1:0]	'0'
Data 0	
[31:0]	Data 0
----	----
----	----
Data 7	
[31:0]	Data 7

The PACKET_TRANSFER message can be transferred using any burst size over the PCI. The GalNet device uses a burst of 8 words.

10.3.8 PACKET_TRANSFER Message (GalNet to CPU in Unicast Intervention Mode)

A burst of up to 8 32-bit words from the source GalNet device to the target CPU buffer which contains the unicast packet whose address is tagged for intervention mode. The first data word of the packet will be written to second word of the buffer. The first word is left empty for the END_OF_PACKET message.

Table 18: PACKET_TRANSFER Message (GalNet to CPU)

PCI Bits	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	CPU Device Number (bits [12:8] in 0x140030)
[21]	'1' (DRAM)
[20:0]	DRAM location
Data 0	
[31:0]	Data 0
----	----
----	----
Data 7	
[31:0]	Data 7

The PACKET_TRANSFER message can be transferred using any burst size over the PCI. The GalNet device uses a burst of 8 words.

10.3.9 PACKET_TRANSFER Message (CPU to GalNet)

A burst of up to 8 32-bit words from the source CPU Buffer to the target GalNet device which contains the packet.

Table 19: PACKET_TRANSFER Message (CPU to GalNet)

PCI Bits	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	Target Device Number
[21]	'1' (DRAM)
[20:0]	DRAM location
Data 0	
[31:0]	Data 0
----	----
----	----
Data 7	
[31:0]	Data 7

The PACKET_TRANSFER message can be transferred using any burst size over the PCI. It is recommended to use a burst of 8 words.

10.3.10 END_OF_PACKET Message (GalNet to CPU 16 Block Buffer)

A message from the source GalNet device to the CPU which indicates the end of the packet. This message is used when the GalNet device forwards one of the following packets directly to the CPU 16 block buffer:

- Unicast packets destined for the CPU (Device Number in the address table is equal to the CPU number)
- Multicast packets
- Unknown packets (if set by bit 6 in the Global Control Register, 0x140028)
- BPDU messages
- Sniffer packets when the CPU is the target sniffer
- EASE packets

The END_OF_PACKET message is written to the first word of the buffer.

Table 20: END_OF_PACKET Message (GalNet to CPU 16 Block Buffer)

PCI Bits	Description
Address	
[31:15]	CPU Buffer Base Address (0x140038)
[14:11]	Offset Counter (16 total blocks)
[10:0]	'0' (Always to the First word)
Data 0	
[31]	Sniffer Packet
[30:24]	Reserved
[23:16]	EASE sample (port0, bit 16; port7, bit 23)
[15]	EASE sample is an original packet to CPU
[14:12]	Source Channel Number
[11:1]	Byte Count
[0]	Valid Bit

10.3.11 END_OF_PACKET Message (GalNet to CPU in Unicast Intervention Mode)

A message from the source GalNet device to the CPU which indicates the end of a unicast packet transfer to the CPU in intervention mode. The END_OF_PACKET message is written to the first word of the buffer.

Table 21: END_OF_PACKET Message (GalNet to CPU in Unicast Intervention Mode)

PCI Bits	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	CPU Device Number (bits [12:8] in 0x140030)
[21]	'1' (DRAM)
[20:0]	'0' (Always to the First word)
Data 0	
[31]	Sniffer Packet
[30:24]	Reserved
[23:16]	EASE sample (port0, bit 16; port7, bit 23)
[15]	EASE sample is an original packet to CPU
[14:12]	Source Channel Number
[11:1]	Byte Count
[0]	Valid Bit

10.3.12 END_OF_PACKET Message (CPU to GalNet)

A message from the source CPU to the target GalNet device which indicates the end of the packet.

Table 22: END_OF_PACKET Message (CPU to GalNet)

PCI Bits	Description
Address	
[31:27]	GalNet Protocol Region
[26:22]	Target Device Number
[21]	'0' (Internal registers)
[20:18]	'111' (END_OF_PACKET message)
[17:0]	'0'
Data 0	
[31]	CRC (0 - Do not append, 1 - Append)
[30]	'0'
[29:22]	Target Port Number (1bit per each port; bit 22 - Port 0, bit 23 - Port 1 etc.)
[21]	Multicast/Unicast
[20:10]	Byte Count
[9:0]	Target Buffer Address (divided by 0x600)

11. PCI Bus Operation

The GT-48001A can act as either a master initiating a PCI bus operation, or as a target responding to a PCI bus operation. The GT-48001A is enabled to both master the PCI bus and respond to memory accesses after RESET.

The GT-48001A has two sets of internal registers that are accessible through the PCI interface. Internal control registers for the GT-48001A are memory mapped and accessible through standard PCI read/write operations. PCI control registers are accessible through PCI configuration cycles.

11.1 PCI Configuration Header Registers

The GT-48001A's PCI configuration registers are located in the standard PCI configuration header locations. Access to these registers is through PCI configuration reads/writes with the IdSel signal asserted. PCI configuration registers control PCI functions and return PCI information (device/vendor ID, etc.)

11.2 Accessing DRAM and Internal Registers through the PCI Interface

All GT-48001A internal control registers and the local DRAM are mapped into PCI memory space. The GT-48001A looks at PCI address bits 31:22 to determine if it is the target for the current cycle. This results in a decode region of 4Mbytes per GT-48001A. Bit 21 of the address is used by the GT-48001A to subdecode on-chip between accesses to DRAM and accesses to the internal registers. Bits 20:0 are used to select a specific register or memory location. Register addresses given in this document refer to the value of bits 20:0 in the PCI address. The GT-48001A expects data in little-endian format, as is required by the PCI specification.

The GT-48001A acts as a “medium speed” decide device, returning DevSel* in 2 clocks.

11.3 PCI Bandwidth/Performance Issues

The PCI bus has an ideal maximum bandwidth of just under 132Mbytes/sec (about 1 gigabit per second.) In systems with many GalNet devices the bandwidth available on the PCI bus can become a performance limitation.

There are several factors that reduce the maximum achievable bus bandwidth:

- Aggressiveness of the PCI arbiter. For example, the use of hidden arbitration can save cycles between adjacent access and improve overall bandwidth. Simpler arbiters will degrade bandwidth as cycles that could be used for data transfer are used for arbitration.
- The GalNet protocol does involve some overhead when compared to raw data transfer.

While performance in any given system is impossible to estimate due to design differences, Galileo Technology has determined the theoretical maximum PCI bus loading for each 10-Mbps port (see Table 23.) The bandwidth numbers shown in this table take into account overhead for the GalNet protocol, as well as overhead for the PCI bus itself (arbitration, clocks/transfer, etc.)

Table 23: PCI Bandwidth Estimates

Packet Length	Maximum PCI Bandwidth Required per 10-Mbps Port	Maximum Number of 10-Mbps Ports per PCI Bus ¹
64	2.08 Mbytes/s	63
128	1.70 Mbytes/s	78
256	1.50 Mbytes/s	88
512	1.41 Mbytes/s	94
1024	1.36 Mbytes/s	97

1. Maximum number of ports to guarantee 0% packet loss at worst case loading conditions.

Table 23 makes the following worst-case assumptions:

- Packets are being received and transmitted at full wire speed (14,880 pps for 64-byte packets) in full-duplex
- All ports are fully loaded
- All packets must be routed across the PCI bus (not switched between ports in the same device.)

11.4 Plug-and-Play Considerations In PCI Systems

In plug-and-play systems, when the BIOS writes 0xFFFF.FFFF to the Internal/DRAM Base Address register during initialization, the GT-48001A will return 0xFFC0.0000 when read back indicating the GT-48001A is requesting 4 MBytes of PCI address space. Designers who wish to use more than one GT-48001A in a plug and play environment must relocate the multiple GT-48001As into the same 128MByte region (the GalNet Protocol Region.) An alternative solution is to isolate the GT-48001A's on a secondary PCI bus through a PCI-to-PCI bridge device with address remapping capability.

11.5 Unused PCI Bus in Stand-Alone Systems

Single-chip stand-alone applications do not require use of the PCI bus. The PCI bus pins must be connected as shown in Table 24 to insure proper operation when not using the PCI bus. All pull-up and pull-down resistors should have a value of 4.7K Ω . Be sure to have your own reset and 33.0 MHz clock on-board. All other PCI bus signals can remain unconnected.

Table 24: Pin Strapping Requirements for Unused PCI Signals

Pin Name	Strapping
DevSel*	Pulled up to Vcc
Stop*	Pulled up to Vcc
Par*	No Connect
PErr*	Pulled up to Vcc
Frame*	Pulled up to Vcc
IRdy*	Pulled up to Vcc
TRdy*	Pulled up to Vcc
Gnt*	Pulled down to GND
IdSel*	Pulled down to GND
SErr*	No Connect
Req*	No Connect
Int*	No Connect
AD[31:0]	No Connect
CBE[3:0]	No Connect

11.6 PCI Bus Arbiter in Multiple GalNet Device Systems

All GalNet systems that use more than one device will require a PCI bus arbiter. The arbiter examines the bus request signals from each device and determines which device is granted the bus. Galileo provides reference designs for PCI bus arbiters (implemented in inexpensive PALs) on our website.

It is important to note that individual systems may have different arbiter design requirements. For example, if your system has several GT-48001A devices and a single GT-48002A (100-Mbps switch) it may make sense to give higher priority to the higher-speed 100BaseX interfaces.

12. Ethernet Interfaces

The GT-48001A incorporates all the required digital circuitry to interface to 10Base-T, 10Base-5, 10Base-2, 10Base-FL, and NRZ Synchronous media. Eight Ethernet ports are integrated in the GT-48001A and only a small amount of external logic is needed to implement the standard physical interfaces.

12.1 Media Access Control (MAC)

The GT-48001A operates in half-duplex or full-duplex modes. In half-duplex mode, the GT-48001A checks that there is no competitor for the network bus before transmission. In addition to listening for a clear line before transmitting, the GT-48001A 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-48001A listens while it is transmitting, and it can detect a collision. If a collision is detected, the GT-48001A 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-48001A transmits unconditionally.

12.2 Illegal Frames

The GT-48001A will discard all illegal frames and increment the appropriate error MIB counters. Examples include: runts (less than 64 bytes), oversize (greater than 1518 or 1522 bytes), and bad FCS (bad CRC.)

12.3 Selecting the Duplex Mode

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

12.3.0.1 Packet Transmission

If the Col (Collision) signal is not asserted, there is a major difference between Full and Half duplex mode when transmitting packets. In Half duplex mode, the port complies with the "Carrier Sense" part of the CSMA/CD protocol, which means that a pending transmission will always start only after the programmed interframe gap has expired following the last of the following two events:

1. End of receive activity -and-
2. End of previous transmission

In Full duplex mode, a pending transmission will always start after the programmed interframe gap has expired following the end of previous transmission. Any receive activity is ignored.

12.4 Backoff Algorithm Options

The GT-48001A 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. Limit4 controls the number of consecutive packet collisions that will occur before the consecutive collision counter is reset. When Limit4 is LOW, the GT-48001A resets the collision counter after 16 consecutive retransmit trials, restarts the backoff algorithm, and continues to try and retransmit the frame. A packet which is endlessly colliding on re-transmits will continue to be re-transmitted forever, only changing backoff intervals. The GT-48001A supports port partitioning on consecutive collisions, a mode which must be enabled by the CPU. The retransmission is done from the data already stored in the DRAM. In the case of a successful transmission, the GT-48001A 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-48001A will reset its collision counter and restarts the backoff algorithm after 4 consecutive transmit trials. This results in the GT-48001A being more aggressive in acquiring the media following a collision. This will result in better overall switch throughput (less packet loss) in standardized tests. Limit4 can be toggled during switch operation.

12.5 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.

12.6 Link Integrity and Auto Polarity Detector

The GT-48001A implements the Link Integrity test as specified in the IEEE 802.3 10Base-T and 10Base-FL supplements. The GT-48001A generates the link integrity signals on ports configured to either 10Base-T and 10Base-FL modes. In addition, the GT-48001A provides an Auto Polarity method for 10Base-T, to switch the polarity of the data going into the MAC layer accordingly. The default, value is Auto-Polarity enabled.

12.7 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 6.4us.

12.8 Inter-Packet Gap (IPG)

IPG is the idle time between any two successive packets from the same port. The default (from the standard) is 9.6us for 10Mbps Ethernet and 960nsec for 100-Mbps Fast Ethernet. Note that the IPG can be made smaller or larger than the Ethernet standards by programming. Making the IPG smaller can improve test scores at the cost of Ethernet compatibility (a trick used by many vendors during head-to-head magazine tests.) We do not recommend this mode of operation, however, as it violates IEEE standards.

IPG is programmable in the Serial Parameters register.

12.9 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 it will not receive. The PaEn bit in the corresponding Port Control 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.

12.9.1 Enabling Partition Mode

Partition is enabled (for all ports) by setting the enable bit in the GT-48001A 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 strapping option.

12.9.2 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-48001A will issue an interrupt to the CPU upon entering Partition state, and will set the partition bit of that port in the Interrupt Cause register.
- The port will continue to transmit its pending packet, regardless of the collision detection, and will not follow the usual Backoff Algorithm. Additional packets pending for transmission, will be 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 vs. the cost of halting the 8-port 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).

12.9.3 Exiting from Partition State

The port will exit from Partition state, following the end of a successful packet transmission. A successful packet transmission will be declared, if no collisions were detected on the first 512 bits of the transmission. If the interrupt is not masked, the GT-48001A will issue an interrupt to the CPU upon exiting from Partition state, and will clear the partition bit of that port in the Interrupt Cause register.

12.10 Back-pressure

Back-pressure is not supported by the GT-48001A. Back-pressure can greatly deteriorate real network throughput in favor of better standardized test scores. Back-pressure works is by "jamming" an entire network segment when the switch cannot accept new transmissions. This blocks *all* traffic, including traffic between nodes on the same network segment (like blocking local phone calls when the long distance circuits are busy.) Simply dropping packets is a better solution for overall network throughput, although this is not reflected in some simplistic standardized tests. (Dropped packets are detected by higher-layer protocols and regenerated.)

12.11 VLAN Tagging Support

The GT-48001A will receive and transmit Ethernet frames up to 1522 bytes in length, thereby accommodating the standard VLAN tagging bytes (four extra bytes.) Bytes longer than 1522 are discarded as over-size frames. VLAN Tagging Support is disabled by default.

12.12 Serial Modes

Each Ethernet port can operate in one of four serial modes: 10BaseT, 10BaseFL, AUI, or Synchronous. The serial mode is set after RESET by the state of the TxDDel/TxD pins for each port. These values can be overridden in the Port Control Register.

12.12.1 Signal Polarity in Specific Serial Modes

The polarity (active HIGH or LOW) of the TxEn (output), RxEn (input/output) and Coll (input) signals depend on the serial mode for which the GT-48001A is configured (see Table 25.)

Table 25: Signal Polarity by Serial Mode

Signal	Serial Mode					
	10BaseT	I/O	10BaseF	I/O	AUI	I/O
TxEN	HIGH	O	HIGH	O	HIGH	O
CrS	HIGH	O	HIGH	O	HIGH	O
RxLP/Coll	Link Pulse	I	Connect '1'	I	Coll-10 MHz	I

12.12.2 10BaseT Mode

Table 26: 10BaseT Ethernet Interface Pin Descriptions

Pin Name	I/O	10BaseT Mode	Functionality
SClk	I	80 MHz	80 MHz Clock.
RxD	I	RxD	Receive Data: Manchester encoded data.
RxLP	I	RxD_LP	Link Pulses - This pin carries the link pulses.
CrS	O	CrS	Carrier Sense: Indicates presence on received packet. (pre-amble to SOI)
Pol	O	Pol	Indicates the Polarity. 1 - Positive polarity; 0 - Negative polarity. When auto-polarity is enabled, the Pol output reflects the polarity of the SOI symbol at the end of a valid incoming packet. Auto-polarity is enabled by default for the GT-48001A. Auto-polarity is disabled by default for the GT-48001.

Table 26: 10BaseT Ethernet Interface Pin Descriptions

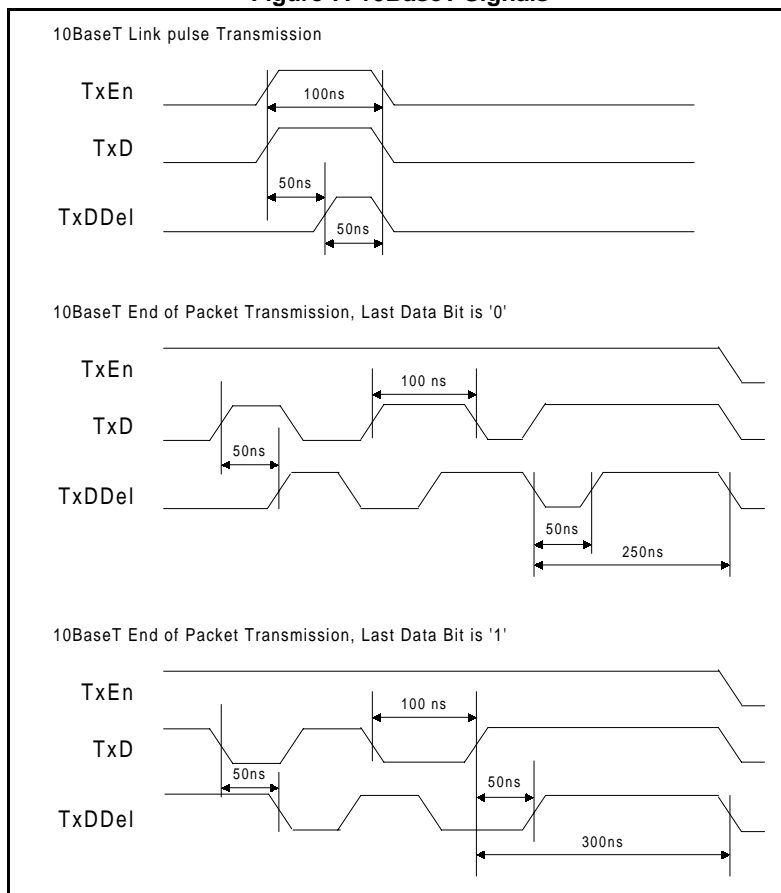
Pin Name	I/O	10BaseT Mode	Functionality
TxD	O	TxD	Transmit Data: Manchester encoded data. Whenever data is not being transmitted, the idle signal, TP_IDL, is transmitted via the TxD. The TP_IDL is a 100 ns wide positive pulse, once every 13msec. Following a packet, the first TP_IDL pulse is transmitted within 13msec to 26msec from the SOI of the packet.
TxDDeI	O	TxDDeI	Transmit Data Delayed: TXD Delayed by 50ns.
TxEn	O	TxEn	Transmit Enable. Indicates that the data is present on TxD and TxD_DL lines.

12.12.2.1 Generating the Required 10BaseT Signals

The GT-48001/1A drives TxD, TxDDeI and TxEn outputs (TTL levels). These signals are used to generate the required 10BaseT pre-distortion transmit signals.

TxDDeI reflects the value of TxD, delayed by 50ns. There are two cases which differ according to the last transmitted bit for the End of Packet Transmission as shown in Figure 7.

Figure 7: 10BaseT Signals



The pre-distortion signal is generated using 4 exclusive-OR (XOR) gates (74AC86) as non-inverting and inverting buffers and a set of standard summing resistors typically used in 10BT filters/transformers.

TxD, TxDDel and TxEn are connected to the XOR gates as follows:

- TXD is connected to XOR1/in1 and XOR2/in1
- TXDDEL is connected to XOR3/in1 and XOR4/in1
- TXEN is connected to XOR2/in2 and XOR3/in2
- XOR2/in2 and XOR4/in2 are connected to ground

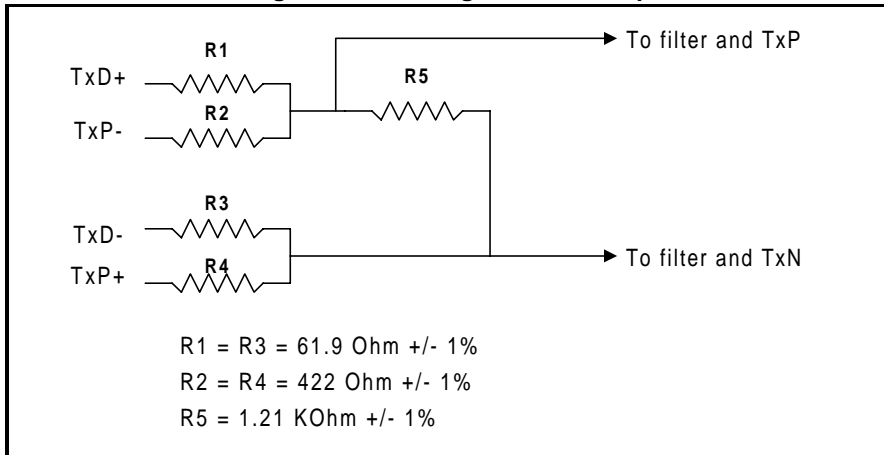
The XOR gates outputs provide the signals shown in Table 25.

Table 27: XOR Gate Outputs

Signal Name	XOR Gate Output	Value when TxEn = 0 (no Transmit)	Value when TxEn = 1 (Transmit)	Buffer Type
TxD+	XOR1	TxD	TxD	non-inverting
TxP-	XOR3	TxDDel	inverted TxDDel	inverting
TxD-	XOR2	TxD	inverted TxD	inverting
TxP+	XOR4	TxDDel	TxDDel	non-inverting

The connection to the summing resistors is shown in Figure 8.

Figure 8: Summing Resistor Setup



This connection is described in the GT-48001/1A Serial Interface Application Note located in the library section of the www site (<http://www.galileoT.com>).

12.12.2.2 Pol Output/Auto-Polarity in 10BaseT

The default value for the GT-48001A is to have the Auto-Polarity function enabled. The default value for the GT-48001(non-A), is to have the Auto-Polarity function disabled. The Pol output, in a 10Base-T configured port, should be connected in a manner that prevents the assertion of link pulses at the GT-48001/1A RxLP input when the detected polarity (indicated by the Pol output) does not match the correct polarity of the RxD pair.

Pol = 1 matches a correct RxD pair connection; Pol = 0 matches a crossed RxD pair. Such connection is specified in Galileo's 10Base-T application note. A crossed RxD pair will prevent the GT-48001/1A from seeing the link pulses which in turn cause the port to enter link-test-fail state. As a result, the Pol output will start to toggle, remaining at 0 or 1 for ~0.5 Seconds. Once the Pol reaches a state in which link-pulses are decoded and asserted at the GT-48001/1A RxLP input, the port will exit from link-test-fail and the Pol output will stabilize. If the user does not connect the Pol out-

put as described above, it is probable that after exiting from link-test-fail, the Pol state will not match the actual RxD pair wiring e.g. the RxD pair is correct but Pol = 0. Since the received packet is inverted internally when Pol = 0, the received packets will be rejected due to bad CRC. The incoming packets may correct the polarity, thus resolving this temporary problem, as described below.

When the Auto-Polarity function is enabled, every received packet, which is of valid length (512 data bits or more), and did not experience a collision, is used to update the internal polarity indication (which is driven externally through Pol output). This is done by sampling the Start of TP_IDL symbol found at the end of the packet. (Start of TP_IDL is defined in Fig 14-10 in the 802.3 std). If a positive polarity minimum size packet is received (e.g. a 64 byte packet through correct RxD pair connection), and Pol = 0, then the Start Frame Delimiter (SFD) which follows the Preamble will not be detected since the data is internally inverted. It is probable that a bit string matching the SFD pattern will be found as part of the packet data content, but this packet will be regarded as too short and will not update the polarity state (as noted earlier). Longer packets are more probable to correct the polarity. Once a packet corrects the polarity to Pol = 1, any further packets will be received correctly.

12.12.3 10BaseFL Mode

Table 28: 10BaseFL Ethernet Interface Pin Descriptions

Pin Name	I/O	10BaseF Mode	Functionality
SClk	I	80 MHz	80 MHz Clock.
RxD	I	RxD	Receive Data: Manchester encoded data. Link Integrity Test is performed on this input whenever there is no incoming packet. The Link Test function monitors the existence of a valid OPT_IDL (1 MHz periodic pulse). At the absence of the OPT_IDL, the port will enter Link Test Fail (LTF) state.
RxLP	I	unused	Should be connected to logic HIGH
CrS	O	CrS	Carrier Sense: Indicates presence on received packet (preamble to SOI).
Pol	O	NC	
TxD	O	TxD	Transmit Data: Manchester encoded data. Whenever data is not being transmitted, the idle signal, OPT_IDL, is transmitted via the TxD. The OPT_IDL is a periodic pulse waveform of frequency 1 MHz. Following a packet and the SOI, the OPT_IDL starts with a logic HIGH.
TxDDel	O	TxDDel	Transmit Data Delayed: TXD Delayed by 50ns.
TxEn	O	TxEn	Transmit Enable. Always High

12.12.4 AUI Mode

Table 29: AUI Interface Pin Descriptions

Pin Name	I/O	AUI Mode	Functionality
SClk	I	80 MHz	80 MHz Clock.
RxD	I	RxD	Receive Data: Manchester encoded data.
RxLP	I	Coll	Collision Detect: Active HIGH. A collision is indicated by the CSO signal (10 MHz signal).
CrS	O	CrS	Carrier Sense: Indicates presence on received packet (pre-amble to SOI).
Pol[0]	I	AUILinkUp[0]	This pin carries the link status driven from the external PHY.
Pol[7:1]	I	AUILinkUp[7:1]	These pins carries the link status driven from the external PHY.
TxD	O	TxD	Transmit Data.
TxDDeI	O	NC	Do not drive this pin.
TxEn	O	TxEn	Transmit Enable: Indicates that the data is present on TxD and TxD_DL lines.

12.12.4.1 TxD Pins

When the GT-48001A is in AUI mode, the default value of TxD pins in an idle state (i.e. when the GT-48001A is not transmitting) is '1'. The default value of the TxD pins of the GT-48001 in a idle state is '0'.

12.12.4.2 AUILinkUp

In AUI mode, The AUILinkUp[7:0] pins of the GT-48001A are used as inputs to indicate the link status which can be driven by the PHY. When the respective AUILinkUp[7:0] pin is a '1', the port has established link. When the respective AUILinkUp[7:0] pin is a '0', the link has failed on that port. The port will enter a link-test-fail state in which it performs similarly to a 10BaseT port in the link-test-fail state. This includes:

1. the port will ignore incoming packets
2. the port will not transmit packets
3. packets will not be forwarded to the port from the PCI nor other ports on the same GT-48001A
4. the LED interface will indicate link-test-fail state for the port
5. since the transmission are halted, the MIB counters for the port will reflect the true wire activity

12.12.4.3 Setting DAddr[6] on Reset

A: The IEEE Ethernet Spec defines the beginning of the interframe-gap after both transmission and carrier-sense have stopped. To comply with the Ethernet MAC specification, the GT-48001 and GT-48001A, start measuring the interframe-gap after both transmission and carrier-sense have stopped. The external tranceiver loopback function (loop back of TxD transmission back to the RxD input), defines the period of time from the end of GT-48001/GT-48001A transmission, and the end of the carrier-sense (which is deasserted following the end of loop-back on RxD). Effectively, the interframe-gap is increased relative to the programmed/default value IPG due to the loop-back. This prevents the GT-48001A from meeting full-wire speed transmissions, unless DAddr[6] is LOW. When DAddr[6] is LOW, the interpacket gap will restart at the end of TxEn, ignoring any loopback of TxD back to RxD. This will enable the GT-48001A port to meet full-wire speed transmissions. When DAddr[6] is HIGH the inter-packet gap will restart when there is no transmit or receive activity (GT-48001 compatible).

12.12.5 Synchronous Mode

Synchronous mode allows an external device to drive or receive an NRZ encoded bitstream directly to/from the GT-48001A. The interface is similar to the MII interface defined for 100Mbps Fast Ethernet. This mode is particularly useful when interfacing to Motorola embedded controllers such as the MPC860 and 68360 family.

NOTICE: To use Sync Mode, all ports must be configured for Sync Mode. See Section 18.

Table 30: Synchronous Ethernet Interface Pin Descriptions

Pin Name	I/O	Sync Mode	Functionality
SClk	I	SynClk20	20 MHz Clock.
RxD	I	RxD*	Receive Data: Active LOW. The decoded NRZ data input. Synchronous to SynClk10's rising edge.
RxLP	I	Coll*	Collision Detect: Active LOW. Synchronous to SynClk20's rising edge. This pin must be pulled HIGH for full duplex operation.
CrS	I	RxDV*	Receive Data Valid: Active LOW. Indicates that the data present on the RxD* is valid. Synchronous to SynClk10's rising edge.
Pol[0]	I	SynClk10	10 MHz transmit/receive clock. This clock serves as the receive and transmit clock for all 8 ports. There is no separate clock for each port.
Pol[7:1]	O	NC	Do not drive this pin.
TxD	O	TxD*	Transmit Data: Active LOW. The NRZ transmitted data output. Driven from SynClk10's falling edge.
TxDDel	O	NC	Do not drive this pin.
TxEn	O	TxEn*	Transmit Enable: Active LOW. Indicates that the data present on the TxD* is valid. Driven from SynClk10's falling edge.

Figure 9 illustrates the relationship that must be observed for the SynClk20 and SynClk10 clock inputs when the GT-48001A is in Synchronous Mode. Notice that the rising edge of SynClk10 should match the falling edge of SynClk20.

Figure 9: SynClk20 Relationship to SynClk10

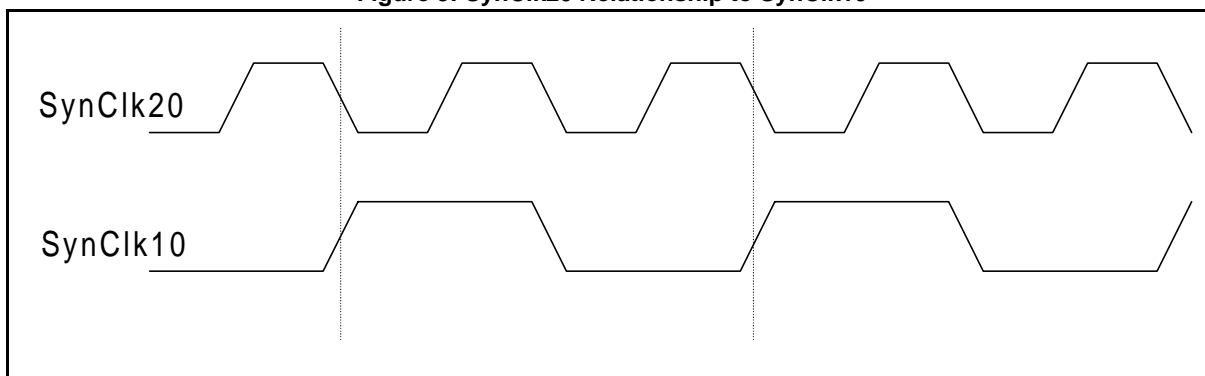


Table 31: Synchronous Mode Transmit Timings

Pin	Min Delay	Max Delay	Clock Edge
TxD*	5nS	25nS	SynClk10 falling
TxEn*	5nS	25nS	SynClk10 falling

Figure 10: Synchronous Mode Transmit Waveforms

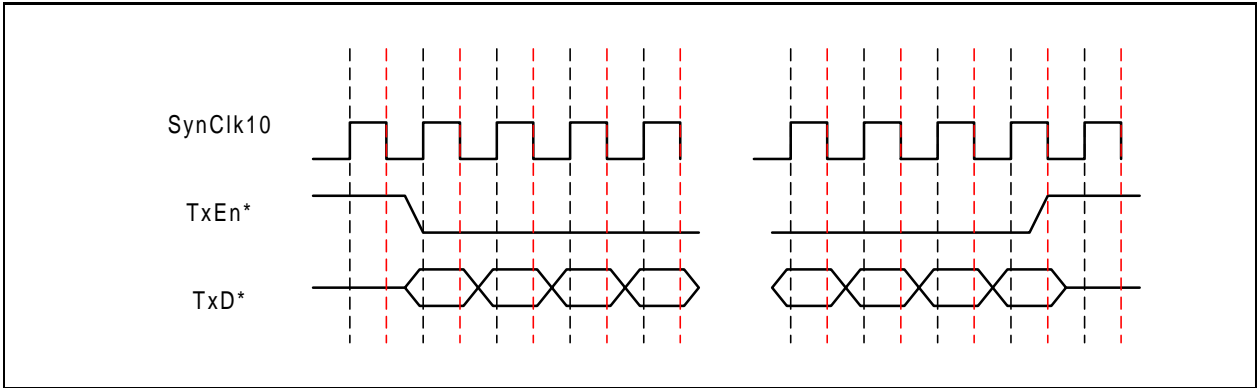
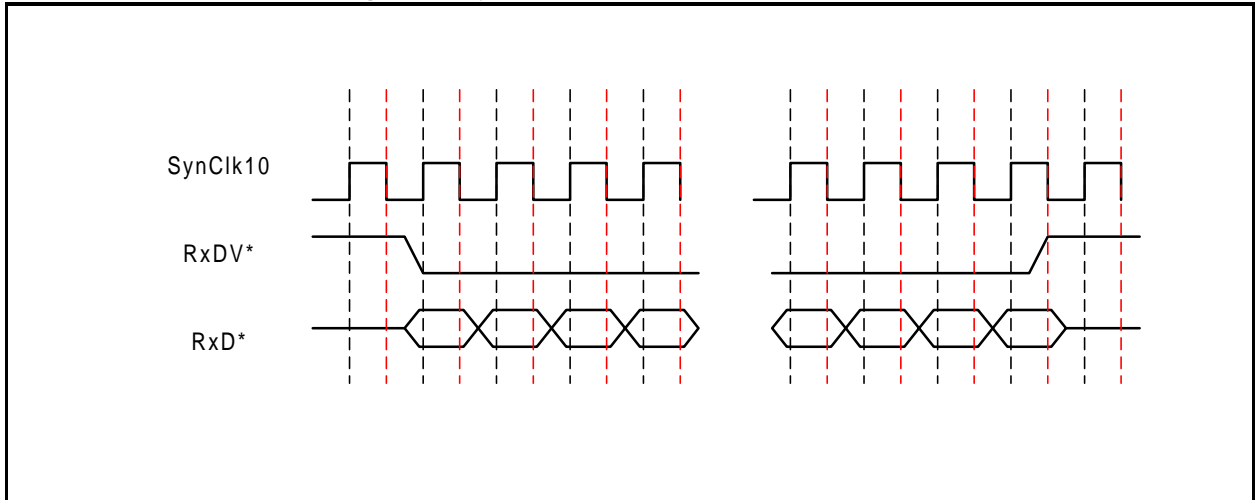


Table 32: Synchronous Mode Receive Timings

Pin	Min Setup	Min Hold	Clock Edge
RxD*	5nS	5nS	SynClk10 rising
RxDV*	5nS	5nS	SynClk10 rising
Coll*	5nS	5nS	SynClk20 rising

Figure 11: Synchronous Mode Receive Waveforms



13. Network Management Support

The GT-48001A supports the following management features:

- HP-EASE packet sampling technology
- Repeater MIB and PCI counters
- Station-to-Station connectivity matrix
- Port monitoring (sniffer) mode

This section describes the MIB counter, connectivity matrix and port mirroring functions. The HP-EASE functions are described in the following section.

13.1 Repeater MIB and PCI Counters

The GT-48001A incorporates a full set of Repeater MIB counters for each Ethernet port, as well as counters for activity on the PCI interface. Counters are accessed by the management CPU through the PCI interface.

The Repeater MIB counters include the following:

- Bytes Received
- Multicast Bytes Received
- Broadcast Bytes Received
- Bytes Sent
- Frames Received
- Multicast Frames Received
- Broadcast Frames Received
- Frames Sent
- Collision
- Late Collision
- CRC and Alignment Error
- Jabber
- Frame Too Short
- Frame Too Long
- Bad Bytes Received (CRC Error, Frame Too Long)

The global PCI counters are:

- PCI Frames Received
- PCI Frames Sent

Please see the register description section for more information on the Repeater MIB and PCI Counters.

13.2 Station-to-Station Connectivity Matrix

The GT-48001A provides a mechanism to record the Destination Port(s), Destination MAC Address, Source MAC Address and the Byte Count of all the forwarding packets in an external FIFO for RMON Station-to-Station (STS) Connectivity Matrix support. The FIFO is connected to the DRAM's data lines and controlled directly by the GT-48001A. The GT-48001A asserts the ChipSel* pin, and reads the packet routing, Byte count, the Destination Address and Source Address. Figure 12 is a timing diagram which shows the relationship between the assertion of ChipSel*, DRAM control signals and DRAM data lines for a one word burst read of RMON data. Figure 10 shows a multiple word burst read of RMON data. An application note giving detailed information about hooking up a FIFO to the 48001A for station-to-station connectivity support is located on Galileo's website (<http://www.galileoT.com>).

Figure 12: ChipSel* Timing, 1 Word Burst Read

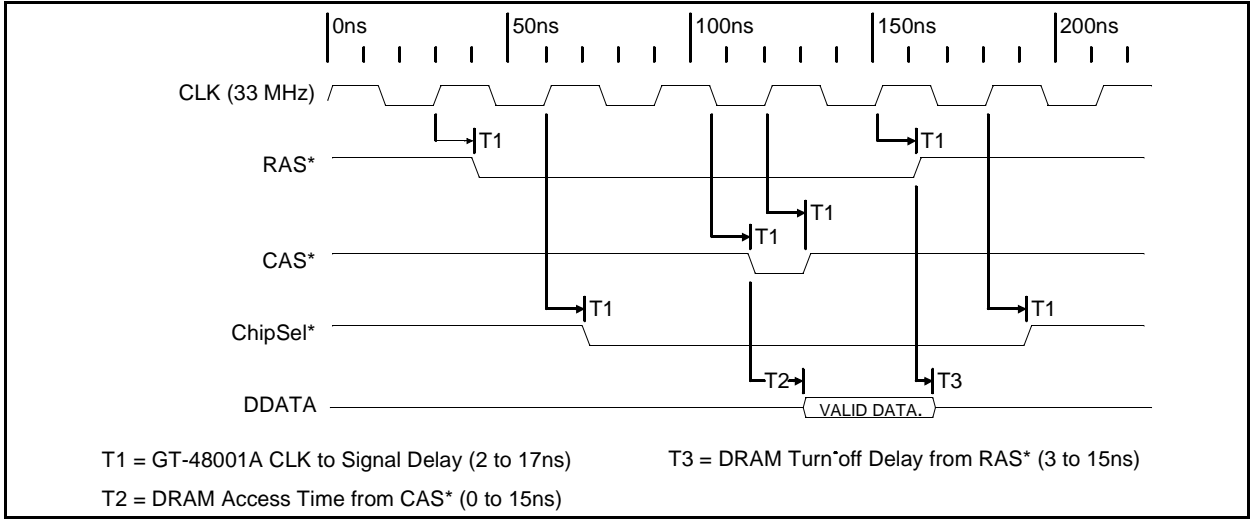
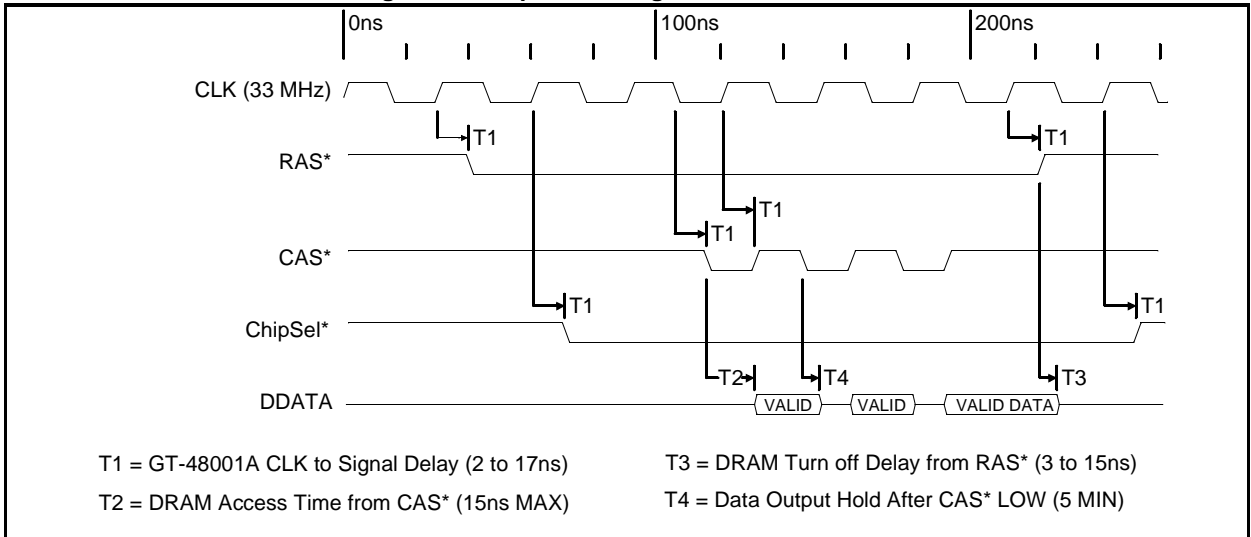


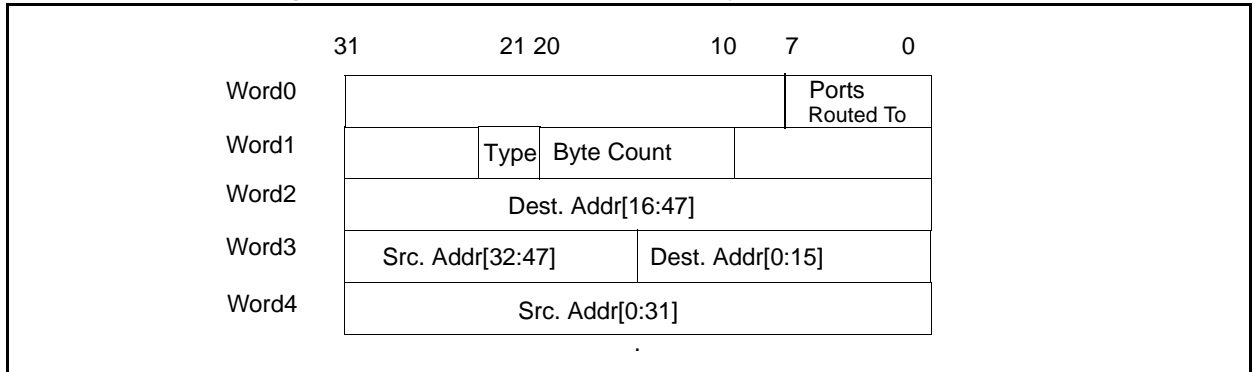
Figure 13: ChipSel* Timing, 3 Word Burst Read



13.2.1 Data Structure Format

Figure 14 shows the format of the STS data structure. The first word includes information about the destination port(s) for this packet indicated by the first 8 bits. This word is useful for multicast packets only and is read back by the GT-48001A by a one word burst read. The next word is the byte count and the type of packet, indicated by bit 21 (1 = multicast, 0 = unicast), also read back by the GT-48001A with a one word burst read. The third fourth and fifth word includes the Destination Address and the Source Address read back by the GT-48001A with a three word burst read. The assertion of this data is indicated by ChipSel*, a dedicated output pin. Please note that the assertion of the ChipSel* does not necessarily happen in consecutive cycles. In other words, the RMON data is not read back by the GT-48001A in three consecutive bursts.

Figure 14: Station-to-Station Connectivity Matrix Data Structure



13.3 Monitoring (Sniffer) Mode

The CPU can program the GT-48001A to work in Monitoring mode for one of the eight Ethernet ports. In Monitoring mode, the GT-48001A sends all receive (including local traffic) and transmit packets to the CPU or to a port in one of the GT-48001A devices which was assigned to be the Sniffer. The packets that are forwarded to the Sniffer are not necessarily in a linear time order.

Monitoring Mode is enabled by setting bit 4 in the Port Control register. The target sniffer is written into the CPU and Sniffer Numbers register. Only one port in each GT-48001A device can work in monitoring mode at a time.

13.4 Spanning Tree Support

The GT-48001A provides the hardware assistance for Bridge Spanning Tree Algorithm implementation. The Spanning Tree algorithm itself is performed by a management CPU.

The GT-48001A includes a SpanEn bit in the Global Control register and additional SpanEn bits in each of the 8 Port Control registers. Table 33 summarizes the hardware assistance for the Spanning Tree algorithm.

Table 33: Spanning Tree Enable Bit Definition

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

Note 1: The GT-48001A does not learn MAC addresses during the Spanning Tree 'Learning' stage (it is 'learning' the bridge topology while in this mode.) The GT-48001A only learns MAC addresses in the Forward mode.

Note 2: The CPU can send BPDU messages to a port of the GT-48001A which is disabled. The mechanism to send BPDUs from the CPU to a locked port is to send a BUFFER_REQUEST message like the format shown in Section 10.3.4, but with the LSB bits of the address as 0x58, instead of 0x0. This BUFFER_REQUEST message will cause the GT-48001A to allocate a buffer regardless of the state of that port.

13.5 Broadcast Storm Filtering

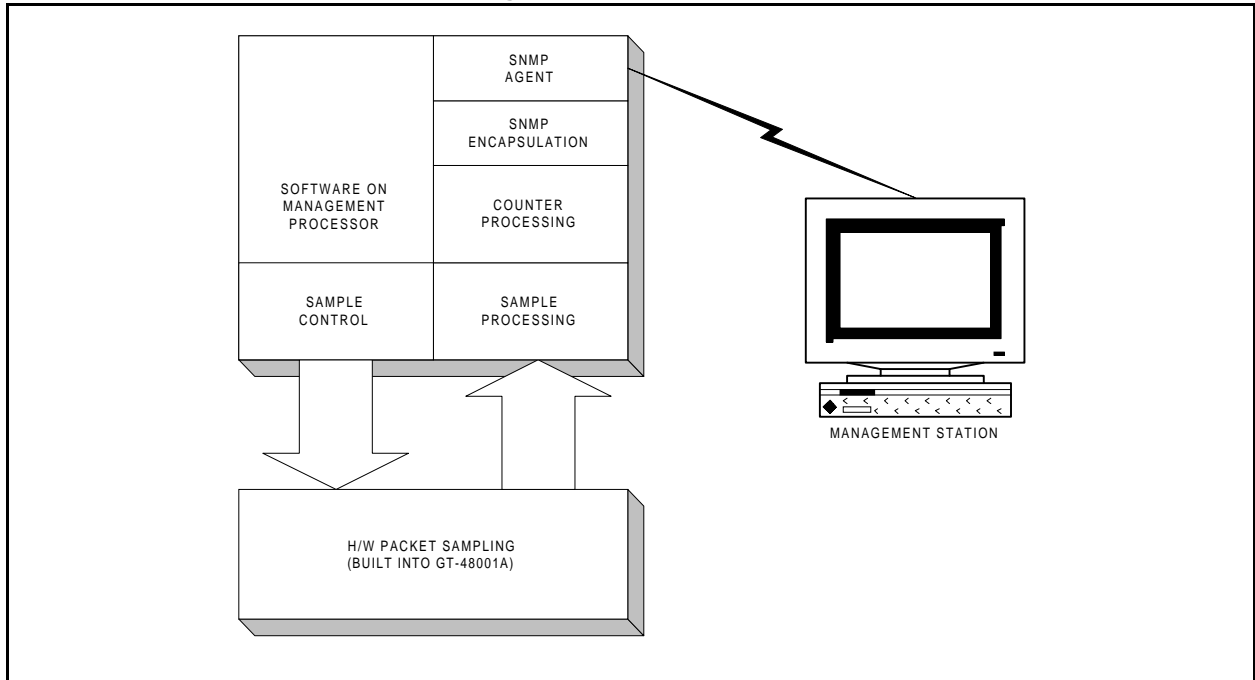
Excessive broadcast packets (broadcast "storms") can be filtered in a managed switch by setting the FilBroad bit in the port control register. Broadcast packets can be re-enabled once the loops causing the broadcast storm are eliminated via the spanning tree algorithm.

14. HP-EASE Packet Sampling Technology

Hewlett-Packard's Embedded Advanced Sampling Environment (HP-EASE) is supported directly by custom hardware in the GT-48001A device. HP-EASE provides many of the management functions that RMON provides, but at a greatly reduced implementation cost. With the GT-48001A device, the switch OEM has the freedom to implement RMON, sampled RMON, EASE, or any of the above.

The HP-EASE functionality is best understood using the familiar stack model, as shown in Figure 15. An HP-EASE agent running on the management CPU, combines samples of packets passing through the switch, with snapshots of the MIB counters and forwards these to a management console through SNMP "trap" messages. This approach is different from RMON, as RMON keeps all data within the switch until polled by the management console. The GT-48001A provides the hardware packet sampling portion of the HP-EASE protocol. This capability can be used to implement a "true" HP-EASE compliant product, or as a method to implement sampled management such as statistical RMON.

Figure 15: HP-EASE Stack



A system implementing HP-EASE provides network management functions similar to RMON. However, a network device (or CPU) implementing HP-EASE is not required to have the extensive CPU or memory resources needed to fully implement all groups of RMON. EASE significantly reduces the requirement for these resources by statistically sampling data on network segments while RMON requires the processing of every network event. The philosophy of EASE is to off-load the intelligence and processing power required for network monitoring to the network management station rather than the network device.

14.1 HP EASE Technology Overview

A system with HP EASE provides network management functions similar to RMON. However, a network device (or CPU) implementing HP EASE is not required to have the extensive CPU or memory resources needed to fully implement all groups of RMON. EASE significantly reduces the requirement for these resources by statistically sampling data on network segments. RMON requires the processing of every network event. The philosophy of EASE is to off-load the intelligence and processing power required for network monitoring to the network management station rather than the network device.

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.

14.2 EASE Functionality on the GT-48001A

Support for EASE sampling is directly integrated in the GT-48001A 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-48001A chip supports eight network segments (one per each port) as well as a PCI system bus. Sampling will occur only on the network segments, and sampled packets will be sent to the CPU via the PCI system bus. Sampling is not performed on the PCI bus. It may, however, be performed on packets received from the PCI bus, but only as a function of the counters for the destination ports (i.e. packets entering the GT-48001A via the PCI bus and being transmitted through one or more ports). There is no counter for the PCI 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-48001A chip in order to support EASE functionality, may be accessed by the CPU from the PCI system bus.

14.3 Ease_Register

A register is defined for each external port supported on the GT-48001A 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 15 bits in length and is used to actually determine when a sample is to be made. The GT-48001A 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-48001A will simply wait, indefinitely, for the CPU to write a value into EASE_register. In this situation, EASE is effectively disabled on that port.

EASE (8 Registers), Offset: 0x040228 - 0x040244

Bits	Field Name	Function	Initial Value
14:0	Ease_Register	value loaded to the internal count-down counter of port 0	0x0
31:15	-	Reserved.	-

14.4 EASE Interrupts

A status bit indicating the full/empty status of the Ease_Register for each external port supported on the GT-48001A, 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-48001A 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-48001A.

EASE Interrupt Cause Register, Offset: 0x050

Bits	Field Name	Function	Initial Value
0	EaseReg0Empty	Ease_Register of port 0 is empty	0x0
1	EaseReg1Empty	Ease_Register of port 1 is empty	0x0
2	EaseReg2Empty	Ease_Register of port 2 is empty	0x0
3	EaseReg3Empty	Ease_Register of port 3 is empty	0x0
4	EaseReg4Empty	Ease_Register of port 4 is empty	0x0
5	EaseReg5Empty	Ease_Register of port 5 is empty	0x0
6	EaseReg6Empty	Ease_Register of port 6 is empty	0x0
7	EaseReg7Empty	Ease_Register of port 7 is empty	0x0
8	ErrorSASent	Error_Source message sent to CPU	0x0
[31:9]	n/a	Always write with 0x0	0x0

EASE Interrupt Mask Register, Offset: 0x054

Bits	Field Name	Function	Initial Value
8:0	MaskBits	Mask the CPU interrupt line for the appropriate bits in the Interrupt Cause register.	0x0
[31:9]	n/a	Always write with 0x0	0x0

14.5 Sampled Packet Indication

Sampled packets are copied into the CPU's receive buffers using the same mechanism as normal receive packets. The only difference, from the CPU's point of view, is that the GT-48001A will put an indication in the first word of the receive buffer which identifies the packet as a sample. The sample indication bits specify which ports on the particular GT-48001A the 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-48001A device operates independently, so it is possible for the CPU to receive the same sample from different GT-48001A devices. For example, a broadcast packet flooded to all ports in the system may be sampled by several GT-48001As at the same time. Each sample will result in a separate copy of the packet being sent to the CPU. It is also possible to sample a packet which would normally be received by the CPU. In this case, only a single copy of the packet can be sent to the CPU. The CPU should be responsible for determining if a sampled packet should also be accepted as a normal receive packet. In the case where a normally received packet is also a sample from multiple GT-48001A

devices (e.g. a broadcast packet), the GT-48001A must provide an indication which allows the CPU to avoid processing duplicate packets. This indication is provided by the GT-48001A which actually received the packet from an external link.

An additional bit in the packet header indicates that the sample packet was originally received from an external link to the CPU as opposed to the PCI system bus. Other GT-48001A devices which sampled the flooded packet only because it was received from the PCI interface and is being transmitted on a port whose internal counter was decremented to zero will not have this indication. These samples are “pure” samples and the CPU will know that it should not process the packet as a normally received packet.

The first word in each 2K block holding the packet to CPU contains the following bits:

- Sniffer (bit 31) (active HIGH)
- n/a (bits [30:24])
- EASE sample for port 7 (bit 23) (active high)
- EASE sample for port 6 (bit 22) (active high)
- EASE sample for port 5 (bit 21) (active high)
- EASE sample for port 4 (bit 20) (active high)
- EASE sample for port 3 (bit 19) (active high)
- EASE sample for port 2 (bit 18) (active high)
- EASE sample for port 1 (bit 17) (active high)
- EASE sample for port 0 (bit 16) (active high)
- EASE sample is an original packet to CPU (bit 15) (active high)
- Source Channel number (bit [14:12], bit 14 is MSB)
- Byte Count (bits [11:1], bit 11 is MSB)
- Valid bit (bit 0, active HIGH)

These parameters are written at the end of the packet transfer.

14.6 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-48001A informs the CPU of error source conditions by writing the Error_Source message to a new Error_Source buffer area in CPU memory. Operations in the Error_Source buffer area are similar to those in the NEW_ADDRESS, START_OF_PACKET and Intervention buffer areas. There is an Error_Source Base Address Register in the GT-48001A in which the CPU writes a pointer to the Error_Source buffer area. The Error_Source buffer area is able to hold 32 entries. Two types of errors are defined for this procedure: FCS error and frames too long. When the GT-48001A receives a packet with any of the above conditions, it will generate and write an Error_Source message to the CPU's buffer area. 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. The CPU may poll the Error_Source buffer area for new messages. However, the GT-48001A includes a separate bit in the Interrupt Cause register which indicates that the GT-48001A has written an Error_Source message into the CPU's memory. An appropriate mask bit is defined in the Interrupt Mask register.

CPU Error Source Base Address, Offset: 0x140050

Bits	Field Name	Function	Initial Value
31:8	ErrorSourceBaseAdd	Contains a pointer to the CPU 'Error_Source' area. The area includes 32 entries (2 32-bit words each) for the GT-48001A's 'Error_Source' messages.	0x0
7:0	-	Reserved. Must be 0x0 when written.	-

'Error_Source': The data written by the GT-48001A device to the 'Error_Source' messages buffer area that contains

information about an Error Source Address. The data format is as follows:

PCI Bits	Description
Address	
[31:8] [7:3] [2:0]	ErrorSourceBaseAdd offset pointer to entry '000'
Data 0	
[31:3] [2] [1] [0]	MAC address [19:47] 1 - Late Collision Error 1 - FCS Error 1 - Over Count Error
Data 1	
[31:27] [26:24] [23:19] [18:0]	reserved Port # (bit 26 is MSB) Device# (bit 23 is MSB) MAC address [0:18]

14.7 Enabling/Disabling EASE Functionality

An explicit HP EASE enable/disable bit is provided in the Global Control register for the GT-48001A 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.

14.8 Interaction With Other GT-48001A Features

Some GT-48001A features are incompatible with HP EASE, and will require that HP EASE be disabled. In most cases, it can be the responsibility of software to assure that HP EASE is disabled when used with other GT-48001A features.

- **Broadcast Intervention Mode:** HP EASE is independent of broadcast intervention mode.
- **Unicast Intervention Mode:** HP EASE should be disabled when using this mode.
- **Sniffer Mode:** HP EASE should be disabled on the GT-48001A device which one of its ports has been configured to work in monitoring mode (e.g. that port's RX and TX traffic are sent to the Sniffer).
- **RMON Station-to-Station Matrix:** HP EASE is independent of the Station-to-Station Matrix feature.
- **Spanning Tree Support:** HP EASE may only be enabled on GT-48001A which has its ports configured in the forwarding state. If the global Spanning Tree enable bit is set and the port is blocked, listening or learning, HP EASE should not be enabled. If the global Spanning Tree enable bit is clear or the port is in the forwarding state, HP EASE can be enabled. It can be the responsibility of software to assure that EASE is enabled correctly when used with Spanning Tree features.
- **Address Learning:** HP EASE in no way impacts the learning process of the GT-48001A device.
- **LED Serial Interface:** EASE packet sample indications are accessible via the serial LED interface mainly for debug purpose. LedData bit # 125: EASE sample indication for port 0. LedData bit # 126: EASE sample indication for port 1. (LedData bit#1 is the bit on which LedStb is HIGH). The LED circuit implements a "monostable" stretching function to enable viewing these dynamic signals.

15. DRAM Interface and Usage

The GT-48001A includes direct support for EDO DRAMs. The performance of EDO satisfies the required bandwidth for data transfer, address recognition and Tx descriptor fetch/update. The DRAM interface is entirely glueless. All accesses are performed as 32-bits. The DRAM interface is designed for 60ns EDO DRAMs and all timings are guaranteed to work with these devices. Refresh is performed automatically by the GT-48001A. Please refer to the Galileo-6Q evaluation platform schematics for an example of EDO DRAM design with the GT-48001A.

The GT-48001A requires about 300Kbytes of the DRAM for the address table and other private data structures. The remainder is used for packet buffers. Following power-up or system RESET, the GT-48001A device creates the MAC Address Table in DRAM, and initializes all locations in the table to indicate that invalid entries exist in all locations.

Galileo recommends using DRAM with 256K x 16 configuration. When using this configuration, 2 DRAM chips are required for 1 MByte, and 4 DRAM chips are required for 2 MBytes. If 1 MByte is selected, RAS0* should be connected to 2 DRAM chips while RAS1* should be left unconnected.

If 2 MBytes is selected, RAS0* will control the first 1MB bank, while RAS1* will activate the second 1MB bank. DData[31:0], DAddr[8:0], CAS*, and WE* should be connected to both banks.

Using 1 or 2 MBytes of DRAM is entirely up to the architect. 2MBytes increases the size of the Rx Buffer space as shown in Table 3. This performance advantage must be weighed against the cost of additional memory.

16. LED Support

The GT-48001A supplies a serial bit stream designed to drive status LEDs for each port, as well as for overall system information. Over 80 internal signals are available through the 3-wire LED port. Galileo also provides reference designs and example PAL equations in the LED interface application note available on our website.

16.1 Led Indications Interface Description

Table 34 shows the data accessible on the LED Indications Serial Interface for each of the GT-48001A ports.

Table 34: 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
Collision active	collision	dynamic
Full/Half duplex	full_duplex	static
Receive Buffer Full	rx_buffer_full	dynamic
Forwarding of unknown packets enabled	unknown_enable	static
The port is configured as Sniffer	port_is_sniffer	static
Link Fail State	link_test_fail	static
Partition State	partition	static

16.2 LED Serial Interface Description

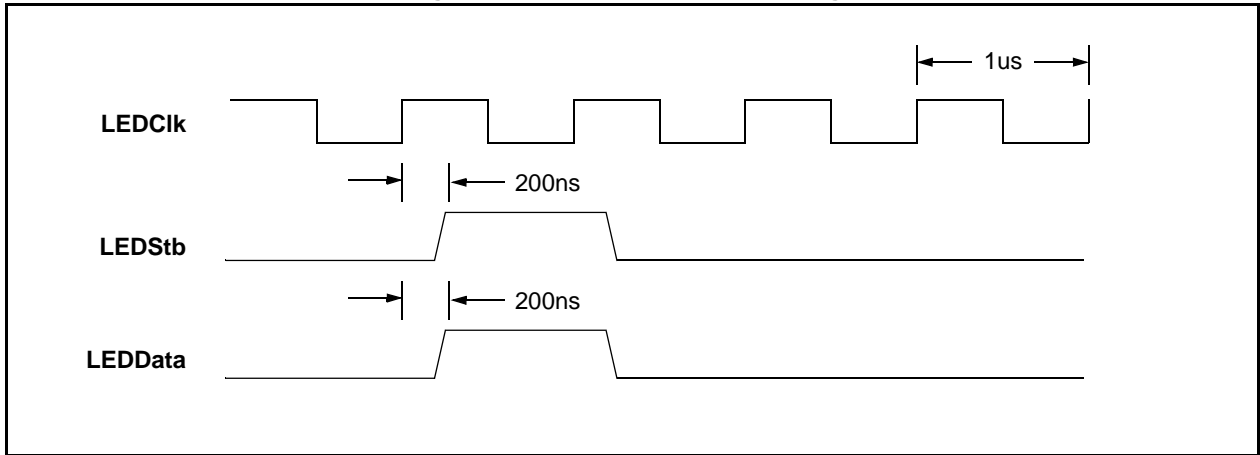
The LED serial interface consists of three outputs:

- **LEDClk**: LEDClk is the primary timebase of the LED Indications Interface. It is a 50% duty cycle free running clock at a fixed frequency of 1 MHz. LEDClk is active when Rst* is asserted. LEDClk frequency during Rst* is 10 MHz.
- **LEDStb**: LEDStb (active HIGH) indicates the beginning of the data frame. LEDStb is activated for a duration of one LEDClk cycle once every 128 LEDClk cycles, starting from Rst* deactivation. This signal marks the beginning of the 128 bit long LED data frame. LEDStb transitions occur 200 ns after LEDClk rising edge.

- LEDData:** The internal signals are multiplexed on the LEDData output for every data frame. LEDStb activation signals the presence of data bit #1 (out of 128 bits) on the LEDData output. LEDData transitions occur 200 ns after LEDCIk 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 16, below.

Figure 16: LED Serial Interface Timings



16.3 Detailed LED Signal Description

16.3.1 Primary Port Status LED

Port Status LED indicates the port status in two operation modes selectable via the LEDMode input pin.

When a port is disabled (via the Port Control Register Port Enable bit or via the EnDev* input), the link detection circuits are stopped. This means that the link test LED and the internal indications of the link (such as those in the status register) Once the port is enabled, the link detection circuits are enabled. In the case of no link pulses, the link test fail indication will be indicated after the timeout period of 150ms. This means that the earliest time any valid link indication (via registers or LED interface) is at least 150ms.

16.3.1.1 Primary Port Status LED (Mode 0): (LEDMode input is LOW)

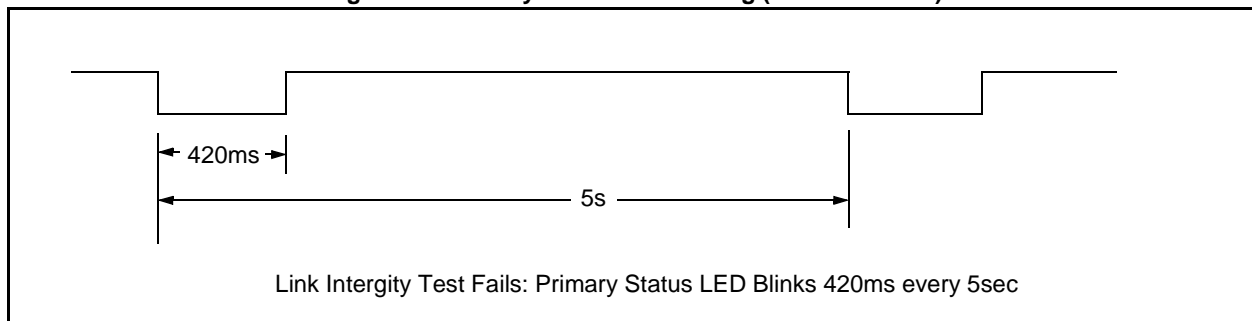
In this mode, the Port Status LED provides the following information

- 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)

16.3.1.2 Status LED blink timing (Mode 0)

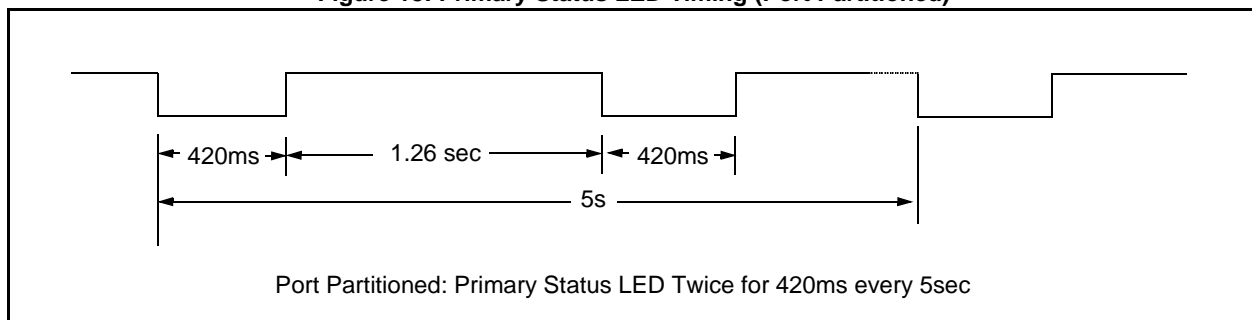
Link Integrity test failed - Status LED blinks once every 5 seconds. Primary status bit is active for 420 ms every 5 seconds

Figure 17: Primary Status LED Timing (Link Test Fails)



Partition, Status LED blinks twice * Primary status bit is activated twice every 5 seconds for 420 ms each time, with a period of 1.26 seconds between two consecutive activations.

Figure 18: Primary Status LED Timing (Port Partitioned)



16.3.1.3 Primary Port Status LED (Mode 1): (LEDMode input is HIGH)

In this mode, the Port Status LED initially displays the port link status information (indication type a) and then switches to reflect the port traffic (Transmit or Receive) activity (indication type b). The switching between the two types of indications is as follows:

Indication type a:

The Port Status LED indicates the port link status for a period of about 3 to 3.5 seconds (active - link test passes, inactive - link test fails) following any of the events below:

- 1) Rst* deassertion
- 2) transition between link test fail to link test pass

Following this time period, the Port Status LED will switch to indication type b. In the case that the link test fails, the Port Status LED will remain inactive and will not switch to indication type b.

Indication type b:

The Port Status LED indicates the port traffic (Transmit or Receive) activity which is a logical OR of the transmit and receive dynamic signals. The "monostable" function is applied to this indication type so the LED can be viewed for a period of about 200 ms per each traffic activity. The Port Status LED will switch to indication type a on the following cases:

- 1) Rst* assertion
- 2) transition from link test pass to link test fail

16.3.2 Transmit data in progress

This signal indicates the port is transmitting data.

16.3.3 Receive data in progress

This signal indicates port receive activity.

16.3.4 Collision active

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

16.3.5 Full/Half duplex

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

16.3.6 Receive Buffer Full

In order for this LED to be active, Rx Buffer Threshold must be enabled (see Table 4). This signal indicates the port receive buffer status: active - the buffer exceeds its programmed threshold, inactive otherwise.

16.3.7 Forwarding of unknown packets enabled

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

16.3.8 The port is configured as Sniffer

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

16.3.9 Link Fail State

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

16.3.10 Partition State

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

16.4 LED Signals Timing Type

16.4.1 Static LED Signals

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

16.4.2 Dynamic Internal Signals:

These signals are typically active for short time periods. In order to be visible through the LED Indication Interfaces, the GT-48001A 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 200 ms.

16.4.3 Table of Internal Activities/Status Driven via the LED Interface

The following table defines a bit by bit description of the internal signals driven through the LED Indications Serial Interface. The bit number (bit#) refers to the activation of LEDStb. LEDStb is active for bit# 1. RESERVED bits contents is not defined (i.e. can be either HIGH or LOW).

Table 35: LED Signals

Bit Number	Signal	Bit Number	Signal
0	primary_port_status[0] (as per LEDMode input)	32	transmit[3]
1	primary_port_status[1] (as per LEDMode input)	33	receive[3]
2	primary_port_status[2] (as per LEDMode input)	34	collision[3]
3	primary_port_status[3] (as per LEDMode input)	35	rx_buffer_full[3]
4	primary_port_status[4] (as per LEDMode input)	36	unknown_enable[3]
5	primary_port_status[5] (as per LEDMode input)	37	port_is_sniffer[3]
6	primary_port_status[6] (as per LEDMode input)	38	full_duplex[3]
7	primary_port_status[7] (as per LEDMode input)	39	-reserved-
8	transmit[0]	40	transmit[4]
9	receive[0]	41	receive[4]
10	collision[0]	42	collision[4]
11	rx_buffer_full[0]	43	rx_buffer_full[4]
12	unknown_enable[0]	44	unknown_enable[4]
13	port_is_sniffer[0]	45	port_is_sniffer[4]
14	full_duplex[0]	46	full_duplex[4]
15	-reserved-	47	-reserved-
16	transmit[1]	48	transmit[5]
17	receive[1]	49	receive[5]
18	collision[1]	50	collision[5]
19	rx_buffer_full[1]	51	rx_buffer_full[5]
20	unknown_enable[1]	52	unknown_enable[5]
21	port_is_sniffer[1]	53	port_is_sniffer[5]
22	full_duplex[1]	54	full_duplex[5]
23	-reserved-	55	-reserved-
24	transmit[2]	56	transmit[6]
25	receive[2]	57	receive[6]
26	collision[2]	58	collision[6]
27	rx_buffer_full[2]	59	rx_buffer_full[6]
28	unknown_enable[2]	60	unknown_enable[6]
29	port_is_sniffer[2]	61	port_is_sniffer[6]
30	full_duplex[2]	62	full_duplex[6]
31	-reserved-	63	-reserved-

Table 35: LED Signals

Bit Number	Signal	Bit Number	Signal
64	transmit[7]	80	partition[0]
65	receive[7]	81	partition[1]
66	collision[7]	82	partition[2]
67	rx_buffer_full[7]	83	partition[3]
68	unknown_enable[7]	84	partition[4]
69	port_is_sniffer[7]	85	partition[5]
70	full_duplex[7]	86	partition[6]
71	-reserved-	87	partition[7]
72	link_test_fail[0]	88-127	-reserved-
73	link_test_fail[1]		
74	link_test_fail[2]		
75	link_test_fail[3]		
76	link_test_fail[4]		
77	link_test_fail[5]		
78	link_test_fail[6]		
79	link_test_fail[7]		

17. Interrupts

The GT-48001A signals interrupts to a management CPU via the PCI INTA# pin. Interrupts are maskable through the Interrupt Mask register and the interrupt source is determined through the Interrupt Cause register. The Interrupt Mask register defaults to masking all interrupts. A '0' in the appropriate bit means that particular interrupt will be masked. A '1' in the appropriate bit means that particular interrupt will not be masked. The default is that all interrupts are masked.

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.

18. RESET Configuration

The GT-48001A 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 usual function.

18.1 Configuration Pins

Configuration pins must be pulled up or down externally at reset to select the desired operational parameter. The recommended value of the pull-up/down resistors is 4.7K ohms. Table 36 shows the configuration pins for the GT-48001A.

Table 36: Reset Pin Strapping Options

Pin	Configuration Function
DAddr[4:0]	Device Number
DAddr[5]	DRAM Size
0-	2Mbyte
1-	1Mbyte
DAddr[6]	AUI Mode
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.
DAddr[8]	DRAM Type
0-	Reserved
1-	EDO
TxEn/FDx[7:0]	Half/Full Duplex Mode Per Port
0-	Half Duplex
1-	Full Duplex
TxDDeI[7:0],TxD[7:0]	Serial Mode Per Port (Note that the logic order is TxDDeI/TxD)
00-	10Base-T
01-	10Base-FL
10-	AUI
11-	Sync (All ports must be configured as '11' to use Sync Mode)

18.2 Configuration Input Timings

The configuration inputs have two timing requirements:

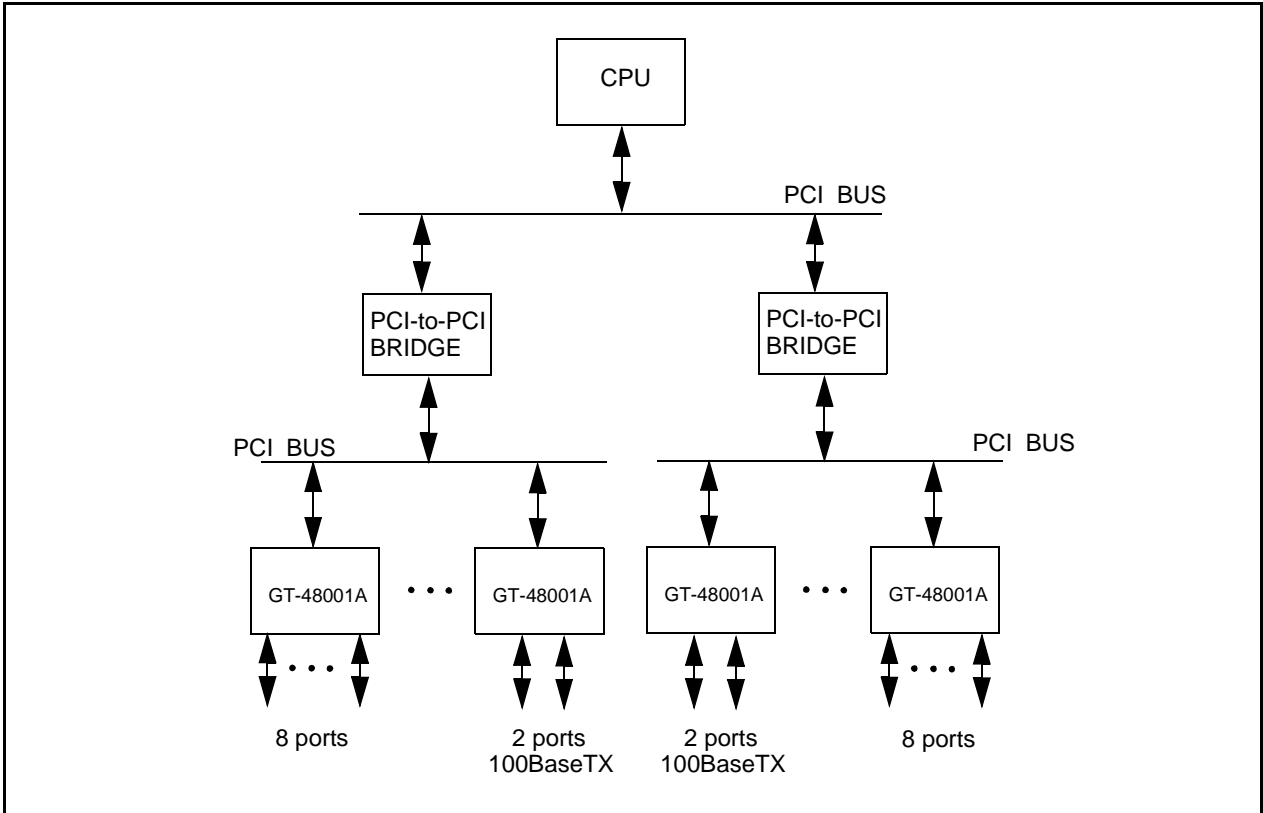
- setup/hold time to clock (as any synchronous input)
- setup of at least 10 clock cycles before RESET de-assertion (rising edge).

You can guarantee these parameters by using resistors to strap the configuration pins and delaying RESET de-assertion until at least 10 clock cycles after the clock is stable.

19. Switch Expansion

Figure 19 shows how a GT-4800x switch may be expanded beyond the PCI physical limit of 10 loads. Multiple GT-48001A devices are connected to a local PCI bus. The PCI buses are connected via PCI-to-PCI bridges. These devices are being used to filter the traffic between multiple PCI buses, and therefore to increase the effective aggregate bandwidth in the switch. The local traffic between GT-48001A devices in one PCI bus is not forwarded to the other GT-48001A devices. Only the traffic between GT-48001A devices in different PCI buses crosses the bridges. Such an implementation requires a CPU to initialize the bridges.

Figure 19: Switch Expansion via PCI-to-PCI Bridges



20. Development Tools

Galileo Technology and a number of third party vendors offer development tools for the GT-48001A. Vendors wishing to join Galileo's third party developers program should contact Galileo's Ethernet marketing group (ethernet@galileoT.com.)

20.1 Evaluation Platforms

Galileo provides the Galileo-6Q Evaluation platform for the GT-48001A. Galileo-6Q is a full sized PCI card that includes a single GT-48001A with 8 10BaseT ports. Galileo-6 can be used stand-alone, or plugged into a PCI backplane, including the Galileo-4PB passive backplane. The Galileo-6Q can be easily combined with the Galileo-7 100BaseT evaluation board for the GT-48002A, allowing mixed 10/100-Mbps switched to be prototyped rapidly.

20.2 Verilog Models

Galileo provides Verilog models for all components, including the GT-48001A. Please contact your local sales representative for pricing information.

20.3 Reference Designs

Several reference designs are available for the GT-48001A. Full schematics and design information are available for the Galileo-6 board on our website. Other reference designs are available (including a 24-port i960 processor based switch designed by Intel). Check our website periodically for updates.

20.4 Complimentary Products

Galileo manufactures a line of CPU core logic chips for MIPs processors that incorporate PCI interfaces. The GT-64010A provides all core logic functions for 64-bit bus MIPs processors: R4600, R4650, R4700 and R5000. For the 32-bit bus R4640, Galileo offers the GT-64011. Information on both devices is available on our website.

21. Register Tables

The GT-48001A incorporates the required PCI Configuration registers, Command registers and various counters for management purposes. The GT-48001A can work in stand-alone mode, in which 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 actual address of an internal register is the sum of the GT-48001A base address and the particular register's offset. The CPU defines the base address by writing to the Internal Registers Base Address register in the PCI Configuration area.

Register Map

Description	Offset
Internal Control Registers	
Global Control	0x140028
Port Control 0	0x040200
Port Control 1	0x040204
Port Control 2	0x040208
Port Control 3	0x04020c
Port Control 4	0x040210
Port Control 5	0x040214
Port Control 6	0x040218
Port Control 7	0x04021c
Status	0x14002c
CPU and Sniffer Numbers	0x140030
Interrupt Cause	0x044
Interrupt Mask	0x048
EASE Interrupt Cause Register	0x50
EASE Interrupt Mask Register	0x54
Serial Parameters	0x040220
Rx Buffers Threshold	0x040224
CPU Buffer Base Address	0x140034
CPU Start Of Packet Base Address	0x140038
CPU New Address Base Address	0x14003c
CPU Intervention Base Address	0x140048
Device Table	0x40
Timeout Counter	0x04c
Port 0 Counter Block	0x040000 - 0x040038
Port 1 Counter Block	0x040040 - 0x040088
Port 2 Counter Block	0x040080 - 0x0400b8
Port 3 Counter Block	0x0400c0 - 0x0400f8
Port 4 Counter Block	0x040100 - 0x040138
Port 5 Counter Block	0x040140 - 0x040178
Port 6 Counter Block	0x040180 - 0x0401b8
Port 7 Counter Block	0x0401c0 - 0x0401f8

Description	Offset
PCI Global Counters	0x140040 -0x140044
PCI Configuration	
Device and Vendor ID	0x000
Status and Command	0x004
Class Code and Revision ID	0x008
Header Type, Latency Timer, Cache Line	0x00c
Internal/DRAM Base Address Register	0x010
Interrupt Pin and Line	0x03c

21.1 Internal Control Registers

Global Control, Offset: 0x140028

Bits	Field Name	Function	Initial Value
0	DisLearnPro	Disable Learning Process. 0 - Learning process is enabled 1 - Learning process is disabled. The GT-48001A will not learn any new addresses from the Ethernet ports	0x0
1	RMONEEn	RMON Enable. 0 - RMON disabled 1 - The GT-48001A enters the RMON mode (Station-to-Station connectivity matrix) and asserts the ChipSel* pin when it reads the packet's Byte Count and Source and Destination Addresses from the DRAM	0x0
2	DevTabMod	Device Table Mode. 0 - The GT-48001A updates the appropriate bits in the table upon master abort 1 - The CPU updates the Device Table	0x0
3	-	Reserved.	0x1
4	DRAMArbPri	DRAM Arbiter Priority. This bit indicates the DRAM arbitration scheme of the four GT-48001A internal units as follows: 0 - 1) Frame Control unit, 2) Switching Core unit, 3) PCI and InterPCI Control units in round robin scheme. 1 - 1) Frame Control unit, 2) Switching Core unit, 3) InterPCI Control unit, 4) PCI Control unit	0x0
5	DescArbPri	Descriptor Control Arbiter Priority. This bit indicates the descriptor control arbitration scheme as follows: 0 -Round robin between the PCI side and the 8 Ethernet ports. The 8 ports have equal priority. 1 - PCI has higher priority than the 8 ports. The priority is: PCI, Port 0; PCI, Port 1;.....PCI, Port 7; PCI, Port 0...	0x1

Bits	Field Name	Function	Initial Value
6	ForwUnk	Forward Unknown Packets. It defines whether the GT-48001A will forward Unknown packets to the CPU or not. 0 - Do not forward 1 - Forward	0x0
7	ForwNewAdd	Forward New Address. It defines whether the GT-48001A will forward new address messages to the CPU or not. 0 - Do not forward 1 - Forward	0x0
8	RecEn	Recovery Enable. It defines whether the recovery process is enabled or not. 0 - Disabled 1 - Enabled	0x1
9	SnifTyp	Sniffer Type. This bit indicates the Sniffer type. The Sniffer can be a CPU or a dedicated port in one of the GT-48001A devices which was assigned to be the Sniffer. 0 - CPU type 1 - GT-48001A type	0x1
10	CPUEn	CPU Enable. This bit indicates that there is a CPU in the system. 0 - CPU does not exist 1 - CPU exists	0x0
11	RMONTToPCI	RMON to PCI Enable. Meaningful only when RMONEEn bit is set. 0 - The GT-48001A reads the DA/SA of the packet that is forwarded only to the local ports. 1 - The GT-48001A reads the DA/SA of the packet that is forwarded to the local ports and the PCI.	0x0
19:12	-	Reserved.	0x0
20	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 PCI are limited to the number which is written in the Rx Buffers Threshold register. This bit is meaningful only when DisBufThr* pin is disabled.	0x1
21	-	Reserved.	0x0
22	ForwMulti	Forward Multicast. 0 - The GT-48001A forwards Multicast packets to all the ports. 1 - Multicast packets forwarded only to the CPU.	0x0

Bits	Field Name	Function	Initial Value
23	ParEn	Partition Enable. When more than 32 collisions occur while transmitting, the GT-48001A 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 - Normal mode 1 - Partition mode	0x0
24	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-48001A forwards BPDU packets only to the CPU.	0x0
25	EnEASE	EASE sampling enable/disable. 0 - EASE sampling disabled. 1 - EASE sampling enabled.	0x0
26	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
27	EnableDevice	Used to enable or disable the GT-48001A. This bit along with EnDev* (external hardware pin) as well as EnPort (bit 0 of the Port Control Register) enables or disables the port/device. See Table 1, "Enabling/Disabling Ports of the GT-48001A," on page 13.	0x0
28	RstQueues	Reset Queues. Used to reset the transmit and receive queues via software. 0 - The Tx and Rx queues of the GT-48001A are reset according to RstQueue*. 1 - Reset the Tx and Rx queues immediately.	0x0
29	CRCGenEn	Used to enable or disable the CRC generation during transmit of CPU generated packets to the GT-48001A. 0 - CRC generation is disabled 1 - CRC generation is enabled .	0x0
30	MIBClrMode	MIB Counter Clear-on-Read mode 0 - MIB counters will be cleared after they have been read. 1 - MIB counters will not cleared after they have been read.	0x0
31	Reserved	This bit is reserved and must be written to 0.	0x0

Port Control (8 Registers), Offset: 0x040200 - 0x04021c

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	TxEn (at RESET)
3:2	SerMode	Serial Mode. These bits indicate the serial mode. The logic order at reset is TxDDel/TxD=[3:2]. 00 - 10Base-T 01 - 10Base-FL 10 - AUI 11 - Synchronous	TxDDel/TxD (at RESET)
4	MonMode	Monitoring Mode. 0 - Port works in normal mode 1 - Port works in monitoring mode; all Rx and Tx packets are sent to the Sniffer	0x0
6:5	Reserved	Reserved.	0x0
7	AutoPol	Auto-Polarity Disable. Enable/disable Auto-Polarity detection function. 0 - Disable 1 - Enable	GT-48001A 0x1 GT-48001 0x0
9:8	Reserved	Reserved.	0x0
10	FilBroad	Filter Broadcast. 0 - Broadcast packets are forwarded to all ports. 1 - The GT-48001A discards Broadcast packets.	0x0
11	ForwUnk	Forward Unknown. 0 - Unknown packets are forwarded. 1 - The GT-48001A does not forward Unknown packets to this port.	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-48001A 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 1522 bytes in length	0x0

Note: Modifying bits [13:1] of any Port Control Register must be preceded by disabling that port via bit[1], the EnDev* input or bit[27] of the Global Control Register (EnableDevice).

Status, Offset: 0x14002c (Read-Only)

Bits	Field Name	Function	Initial Value
4:0	DevNum	Device Number. Indicates the GT-48001A number chosen by the designer.	DAddr[4:0] (at RESET)
5	DRAMSize	DRAM Size. Indicates the DRAM size. 0- 2Mbyte 1- 1Mbyte	DAddr[5] (at RESET)
6	Port0Par	Port0 Partition. This bit indicates the port 0 Partition status. 0 - No Partition (Normal mode) 1 - Partition	0x0
7	Port0LTF	Port0 Link Test Fail. This bit indicates the port 0 Link Test status. 0 - Link Test Pass 1 - Link Test Fail	0x0
8	Port1Par	Port1 Partition. This bit indicates the port 1 Partition status. 0 - No Partition (Normal mode) 1 - Partition	0x0
9	Port1LTF	Port1 Link Test Fail. This bit indicates the port 1 Link Test status. 0 - Link Test Pass 1 - Link Test Fail	0x0
10	Port2Par	Port2 Partition. This bit indicates the port 2 Partition status. 0 - No Partition (Normal mode) 1 - Partition	0x0
11	Port2LTF	Port2 Link Test Fail. This bit indicates the port 2 Link Test status. 0 - Link Test Pass 1 - Link Test Fail	0x0
12	Port3Par	Port3 Partition. This bit indicates the port 3 Partition status. 0 - No Partition (Normal mode) 1 - Partition	0x0
13	Port3LTF	Port3 Link Test Fail. This bit indicates the port 33 Link Test status. 0 - Link Test Pass 1 - Link Test Fail	0x0
14	Port4Par	Port4 Partition. This bit indicates the port 4 Partition status. 0 - No Partition (Normal mode) 1 - Partition	0x0
15	Port4LTF	Port4 Link Test Fail. This bit indicates the port 4 Link Test status. 0 - Link Test Pass 1 - Link Test Fail	0x0

Bits	Field Name	Function	Initial Value
16	Port5Par	Port5 Partition. This bit indicates the port 5 Partition status. 0 - No Partition (Normal mode) 1 - Partition	0x0
17	Port5LTF	Port5 Link Test Fail. This bit indicates the port 5 Link Test status. 0 - Link Test Pass 1 - Link Test Fail	0x0
18	Port6Par	Port6 Partition. This bit indicates the port 6 Partition status. 0 - No Partition (Normal mode) 1 - Partition	0x0
19	Port6LTF	Port6 Link Test Fail. This bit indicates the port 6 Link Test status. 0 - Link Test Pass 1 - Link Test Fail	0x0
20	Port7Par	Port7 Partition. This bit indicates the port 7 Partition status. 0 - No Partition (Normal mode) 1 - Partition	0x0
21	Port7LTF	Port7 Link Test Fail. This bit indicates the port 7 Link Test status. 0 - Link Test Pass 1 - Link Test Fail	0x0

CPU and Sniffer Numbers, Offset: 0x140030

Bits	Field Name	Function	Initial Value
4:0	SnifDevNum	Sniffer Device Number. These bits specify the Sniffer Device Number chosen by the designer. The same value should be programmed to all GT-48001A/GT-48001A devices sharing the same PCI bus.	0x0
7:5	SnifPortNum	Sniffer Port Number. These bits specify the Sniffer Port Number chosen by the designer. The same value should be programmed to all GT-48001A/GT-48002A devices sharing the same PCI bus.	0x0
12:8	CPUDevNum	CPU Device Number. These bits specify the Device Number of the CPU as chosen by the designer. This number must not be the same as any other GalNet device is the system. The same value should be programmed to all GT-48001A/GT-48002A devices sharing the same PCI bus. The corresponding bit for the CPUDevNum value should be set in the Device table (offset 0x40) for all GT-48001A/GT-48002A devices sharing the same PCI bus.	0x0

Interrupt Cause, Offset: 0x044

Bits	Field Name	Function	Initial Value
0	Reserved	Reserved	0x0
1	NewAdd	New Address. This bit is set by the GT-48001A when a new address is received.	0x0
2	TxEnd	Tx End of Packet. This bit is set by the GT-48001A upon transferring a END_OF_PACKET message to the CPU main memory.	0x0
3	RxStart	Rx Start of Packet. This bit is set by the GT-48001A upon sending a START_OF_PACKET message to the CPU.	0x0
4	AddrRecF	Address Recognition Failed. This bit is set by the GT-48001A when the address recognition cycle fails (due to a large number of MAC addresses).	0x0
5	FlushTxQ	Flush Tx Queue. This bit is set by the GT-48001A when one of the Tx queues is flushed due to the Watchdog Timer.	0x0
6	MastRdPar	Master Read Parity. This bit is set by the GT-48001A upon master read parity error on the PCI.	0x0
7	MastWrPar	Master Write Parity. This bit is set by the GT-48001A upon master write error on the PCI.	0x0
8	AddPar	Address Parity. This bit is set by the GT-48001A upon address parity error on the PCI.	0x0
9	MastAbort	Master Abort. This bit is set by the GT-48001A upon master abort on PCI.	0x0
10	TarAbort	Target Abort. This bit is set by the GT-48001A upon target abort on the PCI.	0x0
11	LinkChange	Link State Change. This bit is set by the GT-48001A upon a change in the link state (down->up, up->down) for any pin.	0x0
12	Part	Partition. This bit is set by the GT-48001A upon entering Partition state in one of the ports.	0x0
13	BufWrap	Buffer Wrap-Around. This bit is set by the GT-48001A upon transferring 16 packets to the CPU main memory.	0x0

Bits	Field Name	Function	Initial Value
14	Interv	Intervention. This bit is set by the GT-48001A upon transferring a BUFFER_REQUEST message in Intervention mode.	0x0
15	IntervWrap	Intervention Wrap-Around. This bit is set by the GT-48001A upon transferring 256 BUFFER_REQUEST messages in Intervention mode.	0x0

Interrupt Mask, Offset: 0x048

Bits	Field Name	Function	Initial Value
15:0	MaskBits	Mask to the CPU interrupt line for the appropriate bits in the Interrupt Cause register. The default is to mask all interrupts. 0 - Mask Interrupt 1 - Do not mask Interrupt	0x0000

Serial Parameters, Offset: 0x040220

Bits	Field Name	Function	Initial Value
17:0	-	Reserved.	0x1e104
24:18	IPGData	Inter-Packet Gap (IPG) Data. The step is 100ns (1 bit-time). The default value is 96 decimal (9.6 μ 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-48001A 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 μ s).	064d

Rx Buffers Threshold, Offset: 0x040224

Bits	Field Name	Function	Initial Value
3:0	TxWatTim	Tx Watchdog Timer. The default is 630msec; the range is 105msec to 1575msec, in steps of 105msec. (0x0 has no meaning, illegal condition.)	0x6 (630 ms)
8:4	Reserve	Reserved.	0x04

Bits	Field Name	Function	Initial Value
12:9	RxBufThr	Receive Buffer Threshold. These bits determine the threshold of the receive buffers. Meaningful only when the BufThrEn bit in the Global Control register is set. The step is 10. The number of buffers is programmed value * 10 + 9. The value varies between 9 (0x0) and 159 (0xf). The default is 29 (0x2) for 1MB DRAM and 79 (0x7) for 2MB DRAM.	GT-48001A 0x2 (29) w/ 1M DRAM 0x7 (79) w/ 2M DRAM GT-48001 0x3 (39) w/ 1M DRAM 0x7 (79) w/ 2M DRAM
18:13	-	Reserved. During writes to this register, bits 18:13 must be written to their default values.	0x1 (1MB DRAM) 0x13 (2MB DRAM)

CPU Buffer Base Address, Offset: 0x140034

Bits	Field Name	Function	Initial Value
31:15	BaseAdd	Contains a pointer to the CPU buffer area	0x0
14:0	Reserve	Reserved. Must be 0x0 when written.	-

Note: When reading the CPU Buffer Base Address Register (0x140034), the address will be read out 1 bit shifted left.

CPU START_OF_PACKET Base Address, Offset: 0x140038

Bits	Field Name	Function	Initial Value
31:8	StBaseAdd	Contains a pointer to the CPU START_OF_PACKET area. The area includes 32 entries (2 32-bit words each) for the GT-48001A's START_OF_PACKET messages.	0x0
7:0	Reserve	Reserved. Must be 0x0 when written.	-

CPU NEW_ADDRESS Base Address, Offset: 0x14003c

Bits	Field Name	Function	Initial Value
31:8	NABaseAdd	Contains a pointer to the CPU NEW_ADDRESS area. The area includes 32 entries (2 32-bit words each) for the GT-48001A's NEW_ADDRESS messages.	0x0
7:0	Reserve	Reserved. Must be 0x0 when written.	-

CPU Intervention Base Address, Offset: 0x140048

Bits	Field Name	Function	Initial Value
31:11	IntBaseAdd	Contains a pointer to the CPU BUFFER_REQUEST area for unicast packets that are marked for intervention. The area includes 256 entries (2 32-bit words each) for the GT-48001A's BUFFER_REQUEST messages.	0x0
10:0	Reserve	Reserved. Must be 0x0 when written.	-

Device Table, Offset: 0x40

Bits	Field Name	Function	Initial Value
31:0	DevTab	GT-48001A Device Table. Each bit represents a GT-48001A device in the system.	0xffffffff

Time Out Counter, Offset: 0x4c

Bits	Field Name	Function	Initial Value
7:0	TimeOut0	Specifies in PCI clock units the number of clocks the GT-48001A holds the PCI bus before the generation of Retry termination. Used for the first data transfer.	0x0f (16 clocks)
15:8	TimeOut1	Specifies in PCI clock units the number of clocks the GT-48001A holds the PCI bus before the generation of Retry termination. Used for data transfers following the first data.	0x07 (8 clocks)
23:16	RetryCounter	Specifies the number of retries the GT-48001A attempts to do in the PCI before aborting the transaction. Value of 0x0 disable this counter (unlimited retries).	0xFF (256 times)

Port MIB Counters (8 Blocks), Offset: 0x040000 - 0x0401fc

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 counters will only be reset to '0' if MIBClrMode (bit 30 of the Global Control Register) is set to '0' (default). If MIBClrMode bit is '1', reading the MIB counters will have no effect. The CPU must access the counters using single datum transactions (burst reads/writes are not allowed.)

Table 37 lists definitions for terms used in the counter descriptions.

Table 37: 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
Received Good Packet	A received packet which is not rejected and enters the switching core to be transmitted later
Received Good Unicast Packet	A received packet which is not rejected and enters the switching core to be transmitted later, with destination address which is a unicast address
Received Good Multicast Packet	A received packet which is not rejected and enters the switching core to be transmitted later, with destination address which is a multicast address
Received Good Broadcast Packet	A received packet which is not rejected and enters the switching core to be transmitted later, with destination address which is a broadcast address
Transmitted Good Packet	Any transmitted packet from the GT-48001A
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 13, VTagEn, of the Port Control Register (offset 0x040200-0x04021c)
MAXFRAMESIZE	1518 for VTagEn = 0 (default) or 1522 for VTagEn = 1

Table 38: Port MIB Counters

Address for Port 0	Counter Name	Function	Initial Value
0x040000	BytRec	MIBCtrMode = 0: The total number of data octets for ALL received Good Unicast Packets which are forwarded to the switching core for transmission. MIBCtrMode = 1: The total number of data octets for ALL received Good Unicast Packets which are forwarded to the switching core for transmission and for LOCAL and DROPPED packets.	-
0x040004	MulBytRec	MIBCtrMode = 0: The total number of data octets for ALL received Good Multicast Packets which are forwarded to the switching core for transmission. MIBCtrMode = 1: The total number of data octets for ALL received Good Multicast Packets which are forwarded to the switching core for transmission and for LOCAL and DROPPED packets.	-
0x040008	BroadBytRec	MIBCtrMode = 0: The total number of data octets for ALL received Good Broadcast Packets which are forwarded to the switching core for transmission. MIBCtrMode = 1: The total number of data octets for ALL received Good Broadcast Packets which are forwarded to the switching core for transmission and for LOCAL and DROPPED packets.	-
0x04000c	BytSent	The total number of data octets for ALL transmitted Good Packets.	-
0x040010	FraRec	MIBCtrMode = 0: The total number of ALL received Good Unicast Packets which are forwarded to the switching core for transmission. MIBCtrMode = 1: The total number of ALL received Good Unicast Packets which are forwarded to the switching core for transmission and for LOCAL and DROPPED packets.	-
0x040014	MulFraRec	MIBCtrMode = 0: The total number of ALL received Good Multicast Packets which are forwarded to the switching core for transmission. MIBCtrMode = 1: The total number of ALL received Good Multicast Packets which are forwarded to the switching core for transmission and for LOCAL and DROPPED packets.	-
0x040018	BroadFraRec	MIBCtrMode = 0: The total number of ALL received Good Broadcast Packets which are forwarded to the switching core for transmission. MIBCtrMode = 1: The total number of ALL received Good Broadcast Packets which are forwarded to the switching core for transmission and for LOCAL and DROPPED packets.	-
0x04001c	FraSent	The total number of ALL transmitted Good Packets.	-
0x040020	Coll	The total number of collisions that have been detected up to 512 bits of the transmitted/received packet data length.	-
0x040024	LateColl	The total number of collisions has been detected after transmitted/received packet data length exceeded 512 bits.	-

Table 38: Port MIB Counters

Address for Port 0	Counter Name	Function	Initial Value
0x040028	CRCAligErr	The total number of received packets with valid packet data length between 64 and MAXFRAMESIZE bytes inclusive that had bad CRC and which a collision did not occur.	-
0x04002c	Jabber	The total number of received packets with packet data length greater than MAXFRAMESIZE bytes that had bad CRC, with or without occurring collisions.	-
0x040030	FraShort	The total number of received packets (not including packets with collision) that: - packet data length is less than 64 bytes -or- - packet without SFD and were less than 552 bits in length.	-
0x040034	FraLong	The total number of received Packets with valid CRC and packet data length greater than MAXFRAMESIZE.	-
0x040038	BadBytRec	The total number of data octets in all received packets with bad CRC and Frame Too Long conditions (Jabber and Over-size Packets).	-
0x04003c	DroppedPkts	This counter should be incremented once for every received packet which which is not transferred to the switching core to due lack of a available receive buffer.	-
0x040040		Beginning of the MIB counter structure for port 1. The above structure repeats every 0x40 bytes for the remaining ports.	

PCI Global Counters, Offset: 0x140040 - 0x140044

The CPU must read all counters during initialization in order to reset the counters to '0'. All counters are 32-bits.

Table 39: PCI MIB Counters

Address	Counter Name	Function	Initial Value
0x140040	PCIFraRec	Good Frames Received from the PCI	0x0
0x140044	PCIFraSent	Good Frames Sent to the PCI	0x0

21.2 PCI Configuration Registers

The GT- 48001A contains the required PCI configuration registers. These registers are accessed from the PCI.

Device and Vendor ID, PCI Offset: 0x0 (Read Only)

Bits	Field Name	Function	Initial Value
15:0	VenId	Vendor ID. Provides the manufacturer of the PCI device. For the GT-48001A this is Galileo's PCI vendor ID (0x11ab.)	0x11ab
31:16	DevId	Provides the unique GT- 48001A device ID number	0x4801

Status and Command, PCI Offset: 0x4

Bits	Field Name	Function	Initial Value
1	MemEn	Memory Enable. Controls the GT-48001A's response to memory accesses. 0 - Disable, the GT-48001A will ignore all memory accesses on the PCI bus. 1 - Enable, the GT-48001A will respond to memory accesses on the PCI bus.	0x1
2	MasEn	Master Enable. Controls the GT-48001A's ability to act as a master on the PCI bus. 0 - Disable 1 - Enable	0x1
4	MemWrInv	Memory Write and Invalidate. Controls the GT-48001A's ability to generate Memory Write and Invalidate commands on the PCI bus. 0 - Disable 1 - Enable	0x0
5	Reserved	Reserved. Read only.	0x0
6	ParEn	Parity Enable. Controls the GT-48001A's ability to respond to parity errors on the PCI. 0 - Disable 1 - Enable	0x0

Bits	Field Name	Function	Initial Value
8	SysErrEn	System Error Enable. Controls the GT-48001A's ability to assert the SErr* pin. 0 - Disable 1 - Enable	0x0
22:9	Reserve	Reserved. Read only.	0x0
23	TarFastBB	Target Fast Back-to-Back. This indicates that the GT-48001A is capable of accepting Fast Back-to-Back transactions on the PCI bus. Read-only bit.	0x1
24	DataParDet	Data Parity Detected. This bit is set by the GT-48001A when it detects a Data Parity Error during a master operation. This bit is cleared by writing '1' to it.	0x0
26:25	DevSelTim	Device Select Timing. These bits indicate the GT-48001A's DevSel* timing. The GT-48001A's DevSel* timing is always set at medium (01), as defined in the PCI specification. Read only.	0x1
28	TarAbort	Target Abort. This bit is set upon Target Abort. This bit is cleared by writing '1' to it.	0x0
29	MastAbort	Master Abort. This pin is set upon Master Abort. This bit is cleared by writing '1' to it.	0x0
30	SysError	System Error. This pin is set upon System Error. This bit is cleared by writing '1' to it.	0x0
31	DetParErr	Detected Parity Error. This pin is set upon detection of Parity Error (in both master and slave operations). This bit is cleared by writing '1' to it.	0x0

Class Code and Revision ID, Offset: 0x008 (Read-Only)

Bits	Field Name	Function	Initial Value
7:0	RevID	Revision ID. Indicates the GT-48001A revision number.	GT-48001A 0x10 GT-48001 0x00
23:16	SubClass	SubClass. Indicates the GT-48001A Subclass (0x0 - Ethernet).	0x0
31:24	BaseClass	Base Class. Indicates the GT-48001A Base Class (0x2 - Network Device).	0x2

Header Type, Latency Timer, Cache Line, Offset: 0x00C

Bits	Field Name	Function	Initial Value
7:0	CacheLine	Cache Line. Specifies the GT-48001A's cache line size (size=8). Read only.	0x7
15:8	LatTimer	Latency Timer. Specifies in units of PCI bus clocks the value of the latency timer of the GT-48001A. Default is 256 cycles (0xff).	0xff
23:16	HeadType	Header Type. Specifies the layout of bytes 10 hex through 3f hex.	0x0

For more information on these fields, please refer to the PCI specification.

Internal Register/DRAM Base Address Register, Offset: 0x010

Bits	Field Name	Function	Initial Value
21:0	-	These bits are cleared.	0x0
21	DramAccess	0 - Access internal registers 1 - Access DRAM array	0x0
26:22	DevNum	Device Number. These bits specify the GalNet Device Number.	DAddr[4:0] at RESET
31:27	BaseAdd	Galnet Protocol Region (GPR) Base Address. These bits set the base address for the GalNet Protocol Region. All GalNet devices in a system must have the same GPR base address (see text.)	0x1

Note: This register has special behavior in a plug and play environment (see Section 11.4 on page 45). The device number can be changed in this register after RESET (see Section 7.3 on page 26).

Interrupt Pin and Line, Offset: 0x03c

Bits	Field Name	Function	Initial Value
7:0	IntLine	Interrupt Line. Provides interrupt line routing information.	0x0
15:8	IntPin	Interrupt Pin. Indicates which interrupt pin the GT-48001A uses. The GT-48001A uses INTA in the PCI slot.	0x1

21.3 Register Modification Restrictions

Below are two registers for which the port should be disabled before bits are modified by the CPU.

Note that all ports must be disabled before modifying the bits in the Serial Parameters Register.

Table 40: Port Control Register Bit Modification Restrictions (Offset 0x040200 - 0x04021C)

Bit #	Bit Name	Must Disable Port?
0	PortEn	n/a
1	FullDx	No
3:2	SerMode	Yes
4	MonMode	No
6:5	Reserved- do not modify	n/a
7	AutoPol	No
9:8	Reserved- do not modify	n/a
10	FilBroad	No
11	ForwUnk	No
12	SpanEn	No
13	VTagEn	Yes

Table 41: Serial Parameters Register Bit Modification Restrictions (Offset 0x040220)

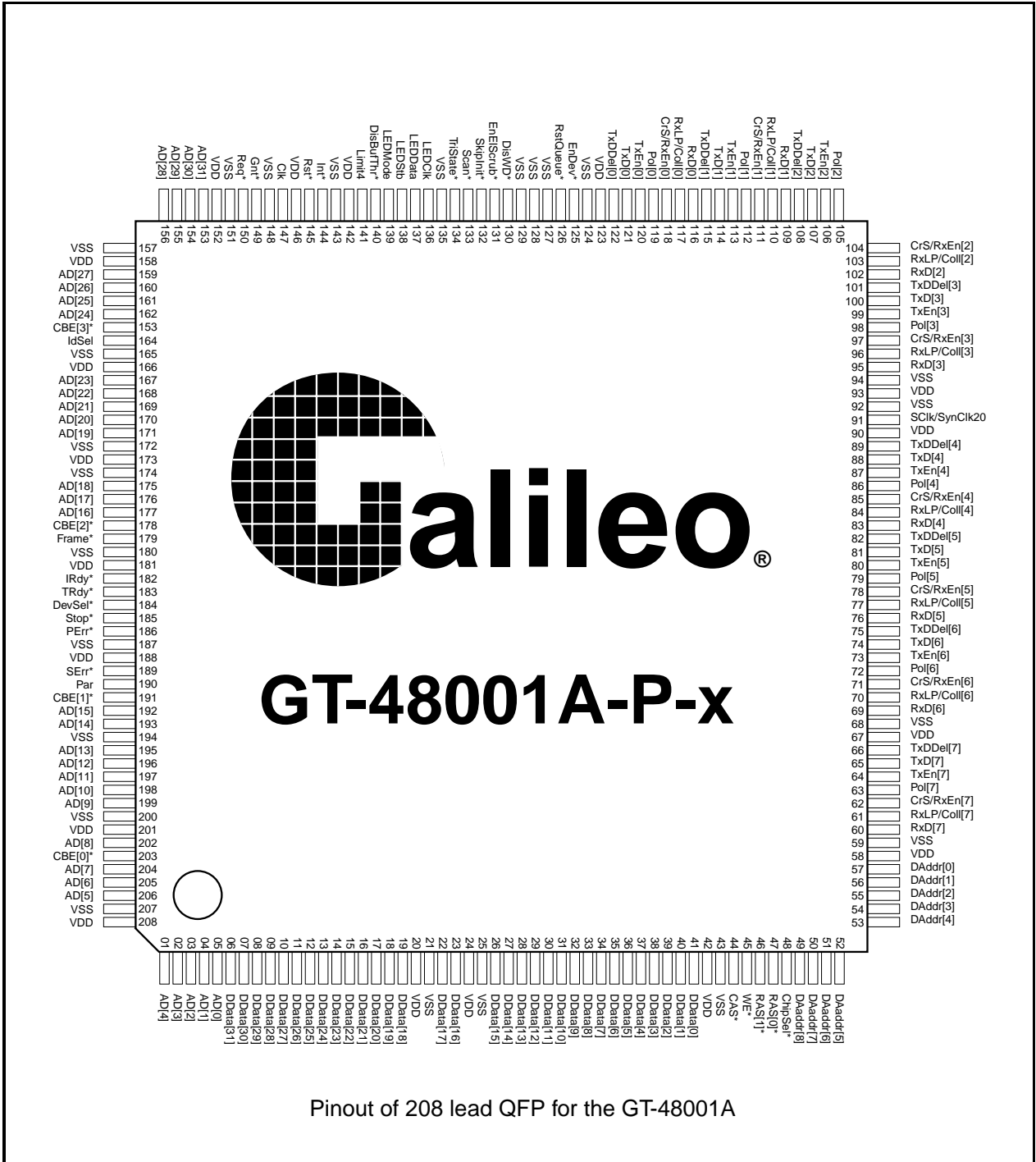
Bit #	Bit Name	Must Disable all Ports?
17:0	Reserved- do not modify	n/a
24:18	IPGData	Yes
31:25	DataBlind	Yes

22. Pinout for 208 pin PQFP (sorted by number)

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
1	AD[4]	36	DData[5]	71	CrS/RxEn[6]
2	AD[3]	37	DData[4]	72	Pol[6]/AUILinkUp[6]
3	AD[2]	38	DData[3]	73	TxEEn[6]
4	AD[1]	39	DData[2]	74	TxD[6]
5	AD[0]	40	DData[1]	75	TxDDeI[6]
6	DData[31]	41	DData[0]	76	RxD[5]
7	DData[30]	42	VDD	77	RxLP/Coll[5]
8	DData[29]	43	VSS	78	CrS/RxEn[5]
9	DData[28]	44	CAS*	79	Pol[5]/AUILinkUp[5]
10	DData[27]	45	WE*	80	TxEEn[5]
11	DData[26]	46	RAS[1]*	81	TxD[5]
12	DData[25]	47	RAS[0]*	82	TxDDeI[5]
13	DData[24]	48	ChipSel*	83	RxD[4]
14	DData[23]	49	DAddr[8]	84	RxLP/Coll[4]
15	DData[22]	50	DAddr[7]	85	CrS/RxEn[4]
16	DData[21]	51	DAddr[6]	86	Pol[4]/AUILinkUp[4]
17	DData[20]	52	DAddr[5]	87	TxEEn[4]
18	DData[19]	53	DAddr[4]	88	TxD[4]
19	DData[18]	54	DAddr[3]	89	TxDDeI[4]
20	VDD	55	DAddr[2]	90	VDD
21	VSS	56	DAddr[1]	91	SClk/SynClk20
22	DData[17]	57	DAddr[0]	92	VSS
23	DData[16]	58	VDD	93	VDD
24	VDD	59	VSS	94	VSS
25	VSS	60	RxD[7]	95	RxD[3]
26	DData[15]	61	RxLP/Coll[7]	96	RxLP/Coll[3]
27	DData[14]	62	CrS/RxEn[7]	97	CrS/RxEn[3]
28	DData[13]	63	Pol[7]/AUILinkUp[7]	98	Pol[3]/AUILinkUp[3]
29	DData[12]	64	TxEEn[7]	99	TxEEn[3]
30	DData[11]	65	TxD[7]	100	TxD[3]
31	DData[10]	66	TxDDeI[7]	101	TxDDeI[3]
32	DData[9]	67	VDD	102	RxD[2]
33	DData[8]	68	VSS	103	RxLP/Coll[2]
34	DData[7]	69	RxD[6]	104	CrS/RxEn[2]
35	DData[6]	70	RxLP/Coll[6]	105	Pol[2]/AUILinkUp[2]

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
106	TxEEn[2]	141	Limit4	176	AD[17]
107	TxD[2]	142	VDD	177	AD[16]
108	TxDDel[2]	143	VSS	178	CBE[2]*
109	RxD[1]	144	Int*	179	Frame*
110	RxLP/Coll[1]	145	Rst*	180	VSS
111	CrS/RxEEn[1]	146	VDD	181	VDD
112	Pol[1]/AUILinkUp[1]	147	Clk	182	IRdy*
113	TxEEn[1]	148	VSS	183	TRdy*
114	TxD[1]	149	Gnt*	184	DevSel*
115	TxDDel[1]	150	Req*	185	Stop*
116	RxD[0]	151	VSS	186	PErr*
117	RxLP/Coll[0]	152	VDD	187	VSS
118	CrS/RxEEn[0]	153	AD[31]	188	VDD
119	Pol[0]/AUILinkUp[0]/ SynClk10	154	AD[30]	189	SErr*
120	TxEEn[0]	155	AD[29]	190	Par
121	TxD[0]	156	AD[28]	191	CBE[1]*
122	TxDDel[0]	157	VSS	192	AD[15]
123	VDD	158	VDD	193	AD[14]
124	VSS	159	AD[27]	194	VSS
125	EnDev*	160	AD[26]	195	AD[13]
126	RstQueue*	161	AD[25]	196	AD[12]
127	VSS	162	AD[24]	197	AD[11]
128	VSS	163	CBE[3]*	198	AD[10]
129	VSS	164	IdSel	199	AD[9]
130	DisWD*	165	VSS	200	VSS
131	EnELScrub*	166	VDD	201	VDD
132	SkipInIt*	167	AD[23]	202	AD[8]
133	Scan*	168	AD[22]	203	CBE[0]*
134	TriState*	169	AD[21]	204	AD[7]
135	VSS	170	AD[20]	205	AD[6]
136	LEDClk	171	AD[19]	206	AD[5]
137	LEDData	172	VSS	207	VSS
138	LEDStb	173	VDD	208	VDD
139	LEDMode	174	VSS		
140	DisBufThr*	175	AD[18]		

22.1 Package/Pin Drawing



23. DC Characteristics - *PRELIMINARY/SUBJECT TO CHANGE*

23.1 Absolute Maximum Ratings

Symbol	Parameter	Min.	Max.	Unit
Vdd	Supply Voltage	-0.3	6.5	V
Vi	Input Voltage	-0.3	Vdd+0.3	V
Vo	Output Voltage	-0.3	Vdd+0.3	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

23.2 Recommended Operating Conditions

Symbol	Parameter	Min.	Typ.	Max.	Unit
Vdd	Supply Voltage	4.75		5.25	V
Vi	Input Voltage	0		Vdd	V
Vo	Output Voltage	0		Vdd	V
Tc	Case Operating Temperature	0		70	C
Cin	Input Capacitance		7.2		pF
Cout	Output Capacitance		7.2		pF

23.3 DC Electrical Characteristics Over Operating Range

(Tc=0-70 C; Vdd=+5V, +/-5%)

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Unit
Vih	Input HIGH level	Guaranteed Logic HIGH level	2.0		Vdd + 0.5V	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 IoH = 12 mA IoH = 16 mA IoH = 24 mA	2.4			V
Vol	Output LOW Voltage	IoL = 2 mA IoL = 4 mA IoL = 8 mA IoL = 12 mA IoL = 16 mA IoL = 24 mA			0.4	V

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Unit
I _{IH}	Input HIGH Current				+1	μA
I _{IL}	Input LOW Current				+1	μA
I _{OZH}	High Impedance Output Current				+1	μA
I _{OZL}	High Impedance Output Current				+1	μA
V _H	Input Hysteresis	V _{DD} = 4.5V V _{DD} = 5.0V V _{DD} = 5.5V	0.52 0.54 0.56		0.60 0.61 0.62	mV
I _{CC}	Operating Current	V _{DD} = 5.25V f=33MHz			600	mA

NOTE: Pullup/Pulldown resistors are 45KΩ minimum, 65KΩ typical, 80KΩ maximum.

23.4 Thermal Data

Table 42 shows the package thermal data for the GT-48001A. Galileo recommends the use of heatsink for most systems, especially those with little or no airflow. Please check with Galileo if you are in doubt as to thermal considerations for your system.

Table 42: 208 PQFP Thermal Data

Parameter	Definition	Value
θ _{JA}	Thermal resistance: junction to ambient, 0 ft/s airflow	22.5 C/W
θ _{JC}	Thermal resistance: junction to case, 0 ft/s airflow	5 C/W
T _J	Operating Junction temperature	100C
T _J	Maximum Junction temperature ¹	125C

1. Operating the device for extended periods of time at the maximum junction temperature may cause permanent damage to this device, resulting in functional or reliability type failures.

24. AC Timing - PRELIMINARY/SUBJECT TO CHANGE

(Tc= 0-70°C; VDD= +5V, +/- 5%)

Symbol	Signals	Description	Min	Max	Unit
	Clk	System Clock	30	33	MHz
	Clk	Rise/Fall Time (PCI Specification Rev. 2.1)	1	4	V/ns
	Rst*, Frame*, IRdy*, TRdy*, DevSel*, Stop*, PErr*, Par, Int* AD[31:0], CBE[3:0]*, Gnt*, IdSel, Req*, SErr*	See PCI Specification Rev. 2.1.			
t1	SClk	Rise Time		2	ns
t2	SClk	Fall Time		2	ns
	SClk	Frequency Stability		+/- 50	PPM
t3	DAddr[8:0], DData[31:0], CAS*, RAS*, WE*, ChipSel*	Delay from Clock Rising or Falling Edge	2	17	ns
t4	DData[31:0], Rst- Queue*, EnDev*	Setup	10		ns
t5	DData[31:0], Rst- Queue*, EnDev*	Hold	1		ns
t6	DData[31:0]	Float Delay	2	18	ns
t7	DData[31:0]	Drive Delay	2	12	ns

Notes:

1. All Delays, Setup, and Hold times are referred to Clk rising edge, unless stated otherwise.
2. All outputs are specified for 50pf load.
3. "All Inputs" and "All Outputs" also refer to I/O signal behavior.

Figure 20: Serial Clock Waveform (SClk)

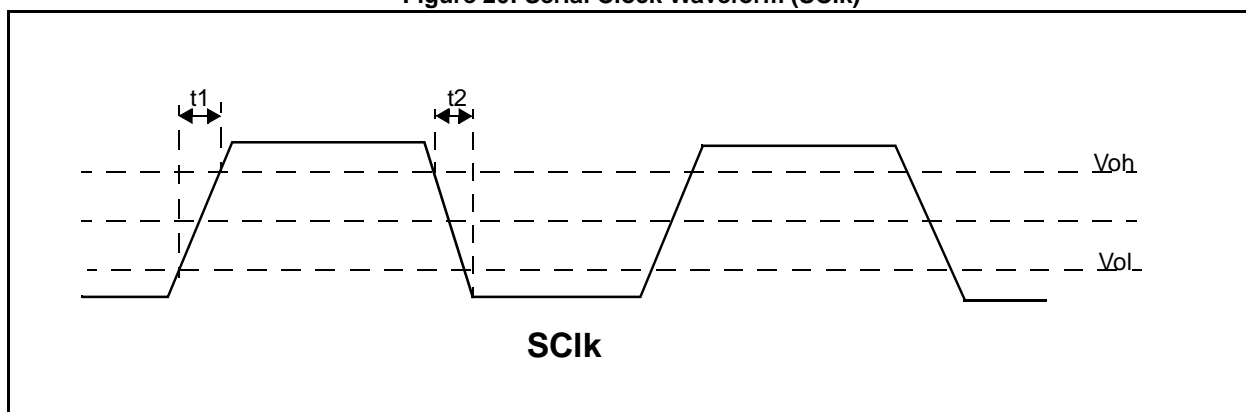


Figure 21: Output Delay from Rising Edge

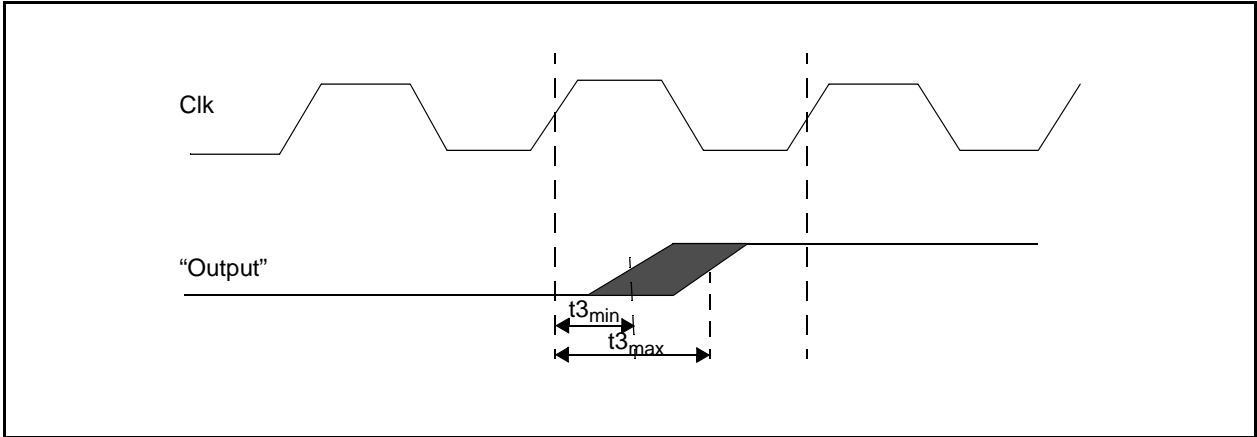


Figure 22: Input Setup and Hold

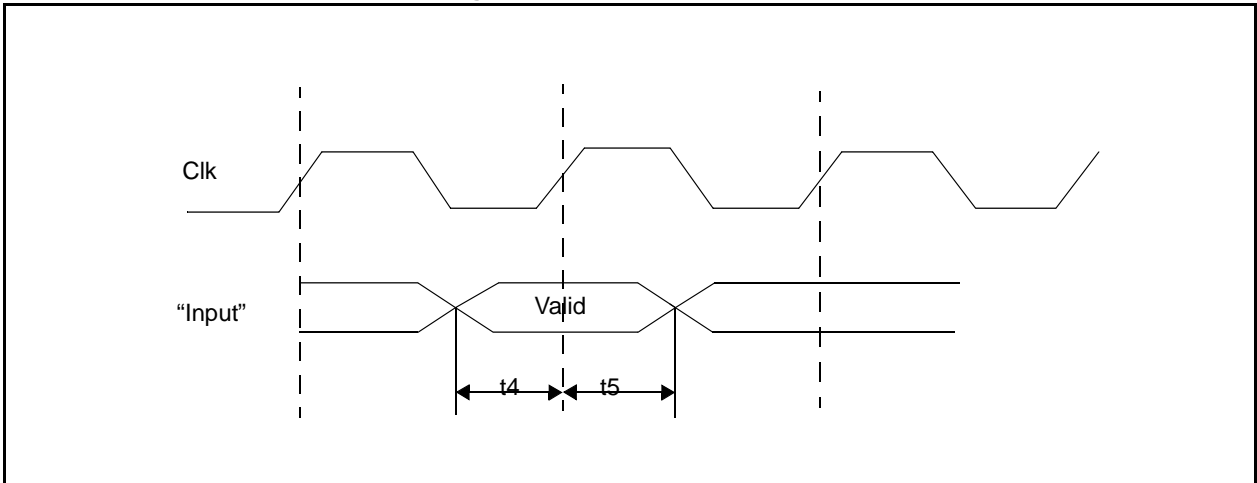


Figure 23: Output Delay from Clock

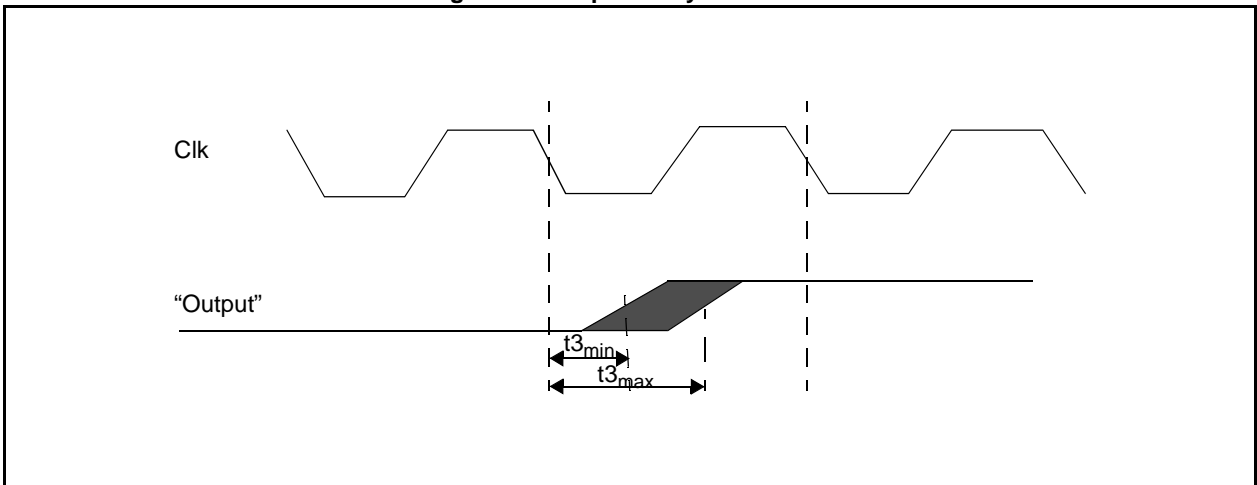
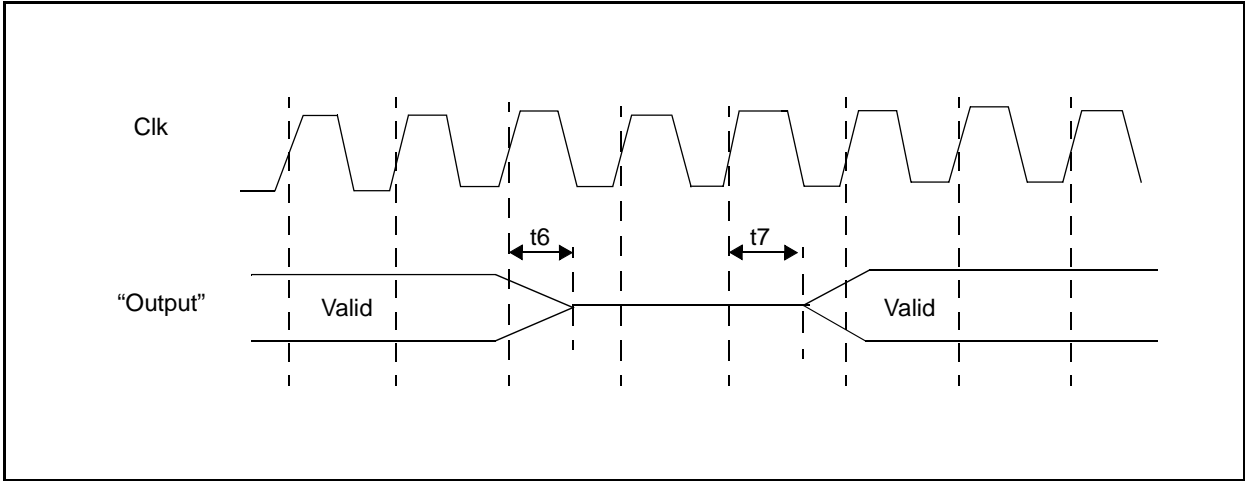


Figure 24: Output Float and Drive Delay



25. Functional Waveforms

Figure 25: EDO DRAM Read

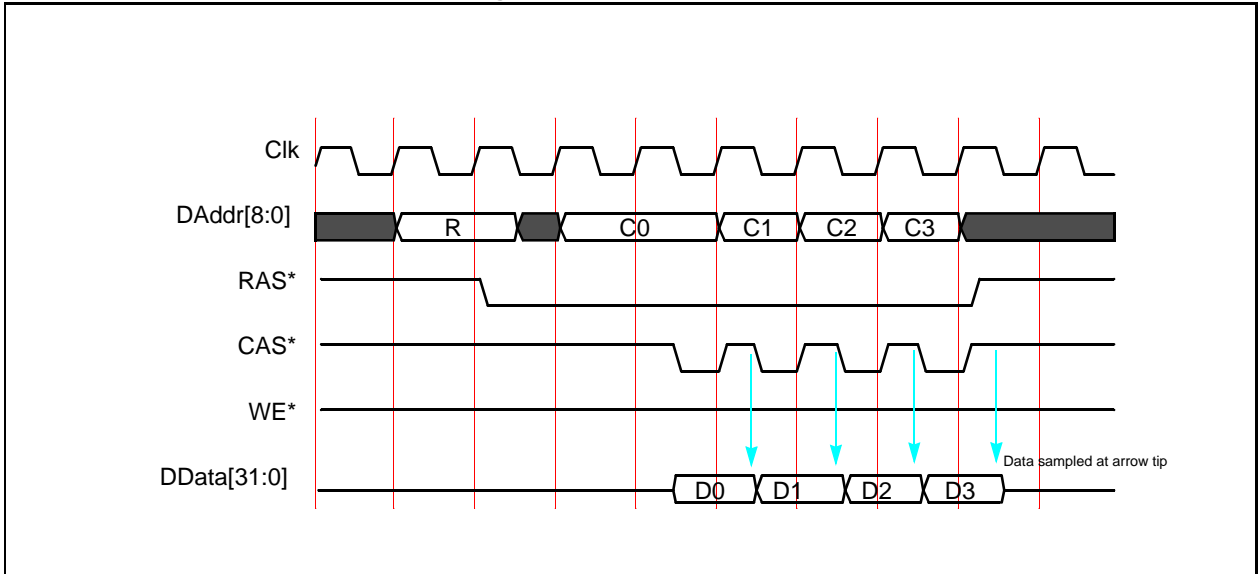
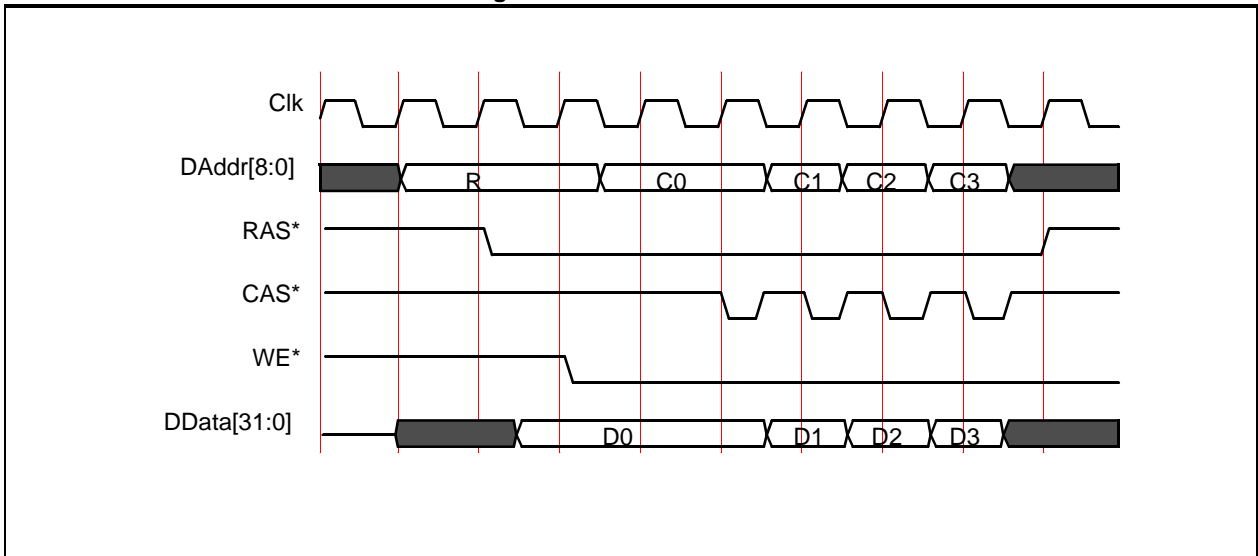


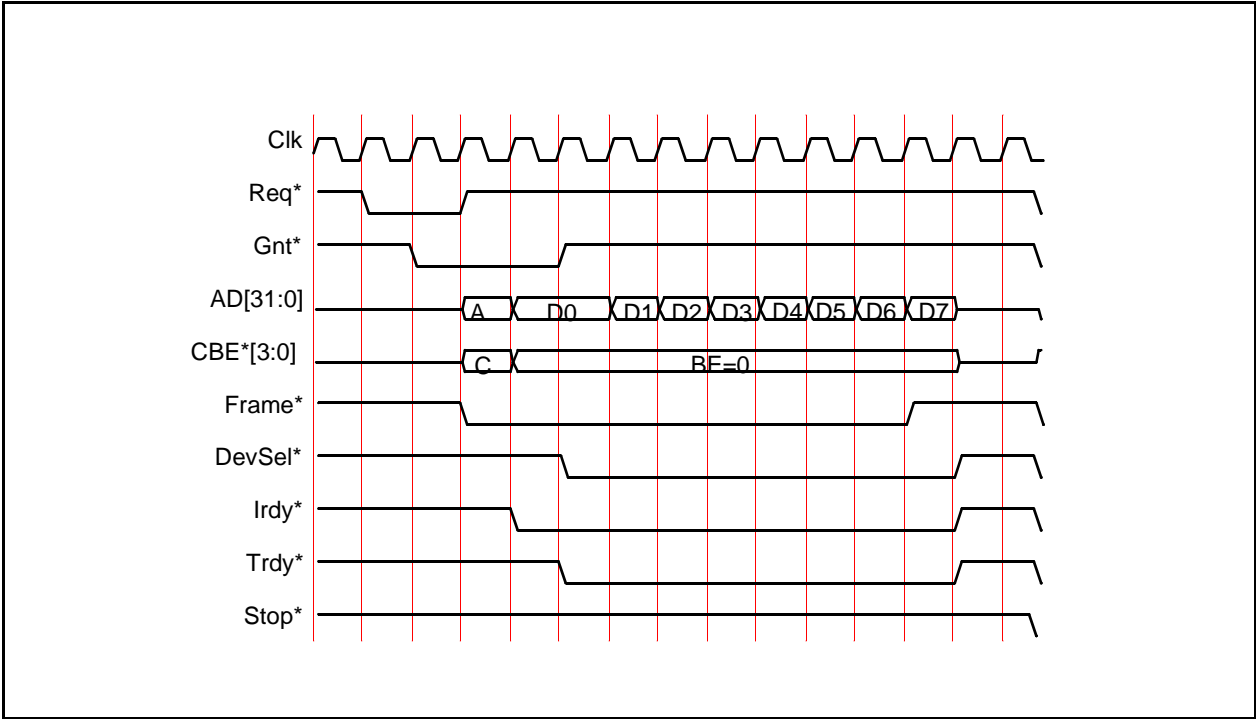
Figure 26: EDO DRAM Write



25.1 PCI Read/Write Cycle

The GT-48001A uses standard PCI read and write cycles. The longest burst that is executed by at GT-48001A device is eight words.

Figure 27: PCI Cycle Waveform



26. Packaging

Figure 28: 208 Lead PQFP Package Outline

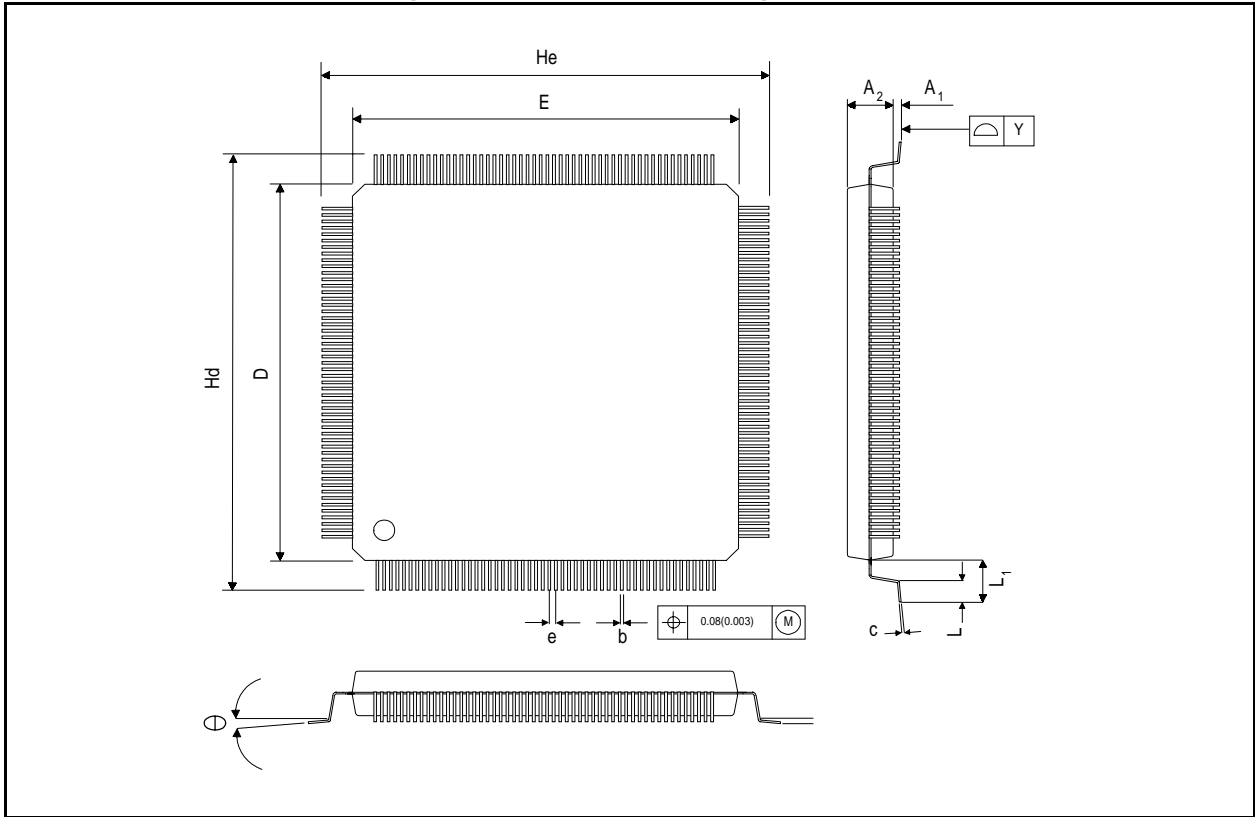


Table 43: 208 PQFP Package Dimensions

MILLIMETERS			
SYMBOL	MIN.	NOM.	MAX.
A ₁	0.05	0.25	0.50
A ₂	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	
Hd	30.35	30.60	30.85
He	30.35	30.60	30.85
L	0.45	0.60	0.75
L ₁		1.30	
Y			0.08
Q	0		7

27. Document History

Table 44: Document History

Document Type	Rev. Number	Date	Comments
PRELIMINARY REV.	1.0	1/30/97	First revision of PRELIMINARY REVISION for general distribution.
PRELIMINARY REV.	1.1	2/18/97	The following were revised: Message Forwarding, GalNet Messages, Unicast Intervention Mode Description
PRELIMINARY REV.	1.2	4/4/97	Added/Revised Sections: Enabling/Disabling Device, DRAM, Serial Interface, Interrupts, Spanning Tree, HP EASE Registers, Rx Buffer Threshold, MIB Counter descriptions, Theta Ja and Theta Jc values
PRELIMINARY REV.	1.3	4/28/97	Revised Sections: Serial Interface Removed: Synchronous Mode
PRELIMINARY REV.	1.4	7/17/97	Revised Sections: Enabling/Disabling Device, Theta Ja and Theta Jc values, Default Register Values, CPU to GalNet NEW_ADDRESS message Re-Added: Synchronous Mode
PRELIMINARY REV.	1.5	8/19/97	Revised Sections: RxBuffer Threshold Enable LED, Forwarding Packets direct to the CPU
PRELIMINARY REV.	1.6	12/29/97	Revised Sections: Corrected plug and play memory size response in section 11.4 to 0xFFC0.0000 (4 megabytes), added synchronous mode timings and additional info, Coll* input for synchronous mode is referenced to SynClk20, not SynClk10 as shown in the 1.5 datasheet, added table of bits that cannot be modified without disabling the port, phone number changed (we moved!).