TEXAS INSTRUMENTS

TSB12LV32, TSB12LV32I

IEEE 1394-1995 and P1394a Compliant General-Purpose Link-Layer Controller

Data Manual

April 2000

Mixed-Signal Products

SLLS336A

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its semiconductor products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Customers are responsible for their applications using TI components.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty or endorsement thereof.

Contents

List of Illustrations

List of Tables

1 Overview

1.1 TSB12LV32 Description

The TSB12LV32 (GP2Lynx) is a high-performance general-purpose IEEE P1394a link-layer controller (LLC) with the capability of transferring data between a host controller, the 1394 Phy-link interface, and external devices connected to the data mover port (local bus interface). The 1394 Phy-link interface provides the connection to the 1394 physical layer device and is supported by the LLC. The LLC provides the control for transmitting and receiving 1394 packet data between the microcontroller interface and the Phy-link interface via internal 2K byte FIFOs at rates up to 400 Mbit/s. The TSB12LV32 transmits and receives correctly formatted 1394 packets, generates and detects the 1394 cycle start packets, communicates transaction layer transmit requests to the Phy, and generates and inspects the 32-bit cyclic redundancy check (CRC). The TSB12LV32 is capable of being cycle master (CM), isochronous resource manager (IRM), bus manager, and supports reception of isochronous data on two channels.

The TSB12LV32 supports a direct interface to many microprocessors/microcontrollers including programmable endian swapping. TSB12LV32 has a generic 16/8-bit host bus interface which includes support for the ColdFire™ microcontroller mode at rates up to 60 MHz. The microinterface may operate in byte or word (16 bit) accesses. The data mover block in GP2Lynx is meant to handle an external memory interface of large data blocks. The port can be configured to either transmit or receive data packets. The packets can be either asynchronous, isochronous, or streaming data packets. Asynchronous or isochronous receive packets will be routed to the DM port or the GRF via the receiver routing control logic.

The internal FIFO is separated into a transmit FIFO and a receive FIFO each of 517 quadlets (2 Kbytes). Asynchronous packets may be transmitted from the DM port or the internal FIFO. If there is contention the FIFO has priority and will be transmitted first.

The LLC also provides the capability to receive status information from the physical layer device and to access the physical layer control and status registers by the application software.

1.2 TSB12LV32 Features

- Compliant With IEEE 1394-1995 Standards and P1394a Supplement for High Performance Serial Bus†
- Supports Transfer Rates of 400, 200, or 100 Mbit/s
- Compatible With Texas Instruments Physical Layer Controllers (Phys)
- Supports the Texas Instruments Bus Holder Galvanic Isolation Barrier
- Glueless Interface to 68000 and ColdFire Microcontrollers/Microprocessors
- Supports ColdFire Burst Transfers
- 2K-Byte General Receive FIFO (GRF) Accessed Through Microcontroller Interface Supports Asynchronous and Isochronous Receive
- 2K-Byte Asynchronous Transmit FIFO (ATF) Accessed Through Microcontroller Interface Supports Asynchronous Transmissions
- Programmable Microcontroller Interface With 8-Bit or 16-Bit Data Bus, Multiple Modes of Operation Including Burst Mode, and Clock Frequency to 60 MHz.
- 8-Bit or 16-Bit Data Mover Port (DM Port) Supports Isochronous, Asynchronous, and Streaming Transmit/Receive From an Unbuffered Port at a Clock Frequency of 25 MHz.
- Backward Compatible With All TSB12LV31(GPLynx) Microcontroller and Data Mover Functionality in Hardware.
- Four-Channel Support for Isochronous Transmit From Unbufferred 8/16 Bit Data Mover Port.
- Single 3.3-V Supply Operation With 5-V Tolerance Using 5-V Bias Terminals.
- High Performance 100-Pin PZ Package

† Implements technology covered by one or more patents of Apple Computer, Incorporated and SGS Thomson, Limited.
ColdFire is a trademark of Motorola, Inc.

1.3 Functional Block Diagram

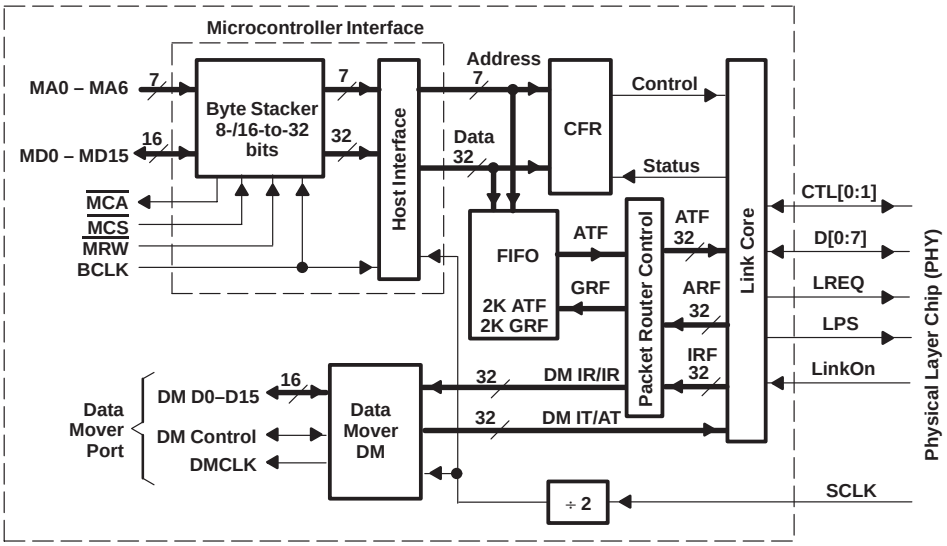
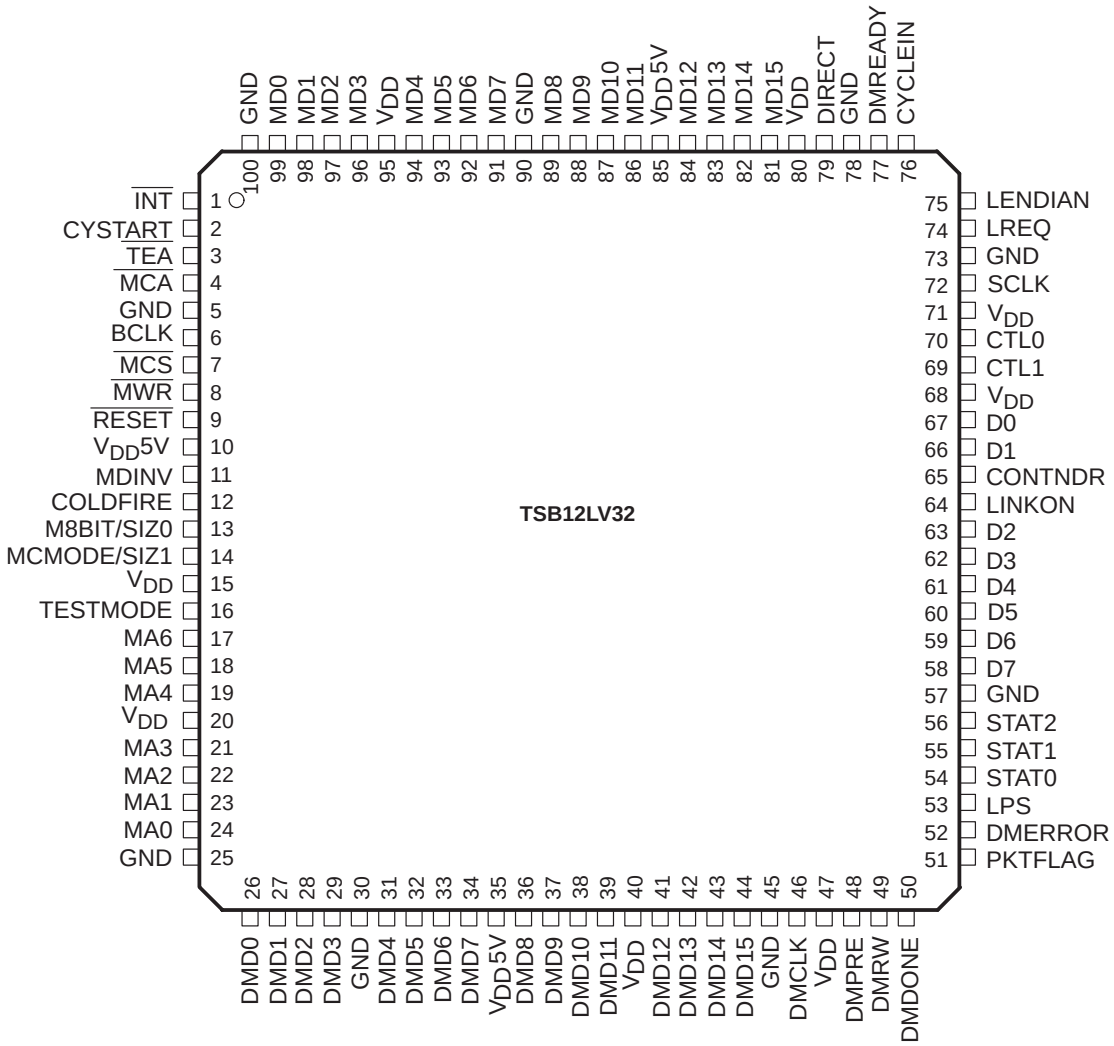


Figure 1–1. TSB12LV32 Functional Block Diagram

1.4 Terminal Assignments

PZ PACKAGE
(TOP VIEW)



1.5 Terminal Functions

The terminal functions are described in Table 1–1.

Table 1–1. Terminal Functions

TERMINAL NAME	TERMINAL NO.	I/O	DESCRIPTION
Microcontroller/Microprocessor Interface			
BCLK	6	I	Microinterface clock. Maximum frequency is 60 MHz. In the ColdFire mode, BCLK is the same as CLK, which is the clock-input signal to the ColdFire.
COLDFIRE	12	I	ColdFire mode. To operate in this mode, COLDFIRE must be asserted high.
LENDIAN	75	I	Little-endian mode for the microinterface. When this terminal is pulled up, the data on MD0–MD15 will be byte-swapped to little endian byte format before it is written to the CFR or FIFO and after it is read from the CFR or FIFO.
MA0 – MA6	24 – 21 19 – 17	I	Microcontroller address bus. MA0 is the most significant bit (MSB) of these 7 bits.
M8BIT/SIZ0	13	I	Configuration bit for microinterface. If the microinterface is 8 bits wide, this terminal must be pulled up to the supply voltage. In ColdFire mode, this terminal represents burst SIZ0.
MCMODE/SIZ1	14	I	Mode bit for microinterface. If the microinterface wants to communicate in a handshake manner this terminal must be pulled up to the supply voltage. When the ColdFire mode terminal (12) is high, this terminal represents burst SIZ1.
$\overline{\text{MCA}}$	4	O	Microinterface cycle acknowledge. When asserted low, $\overline{\text{MCA}}$ signals an acknowledge of the microcontroller cycle from the TSB12LV32.
$\overline{\text{MCS}}$	7	I	Microinterface cycle start. When asserted low, $\overline{\text{MCS}}$ signals the beginning of a microcontroller operation to the TSB12LV32.
MDINV	11	I	Microinterface data invariant mode. This terminal is meaningful only when LENDIAN (75) is high. When asserted high, the microinterface operates in the data invariant mode. When low, the microinterface operates in address invariant mode.
MD0 – MD15	99 – 96 94 – 91 89 – 86 84 – 81	I/O	Microinterface bidirectional data bus. MD0 is the most significant bit. However, byte significance is dependent on the state of the LENDIAN and MDINV terminals.
$\overline{\text{MWR}}$	8	I	Microcontroller read/write indicator. When asserted high, $\overline{\text{MWR}}$ indicates a read access from the TSB12LV32. When asserted low, $\overline{\text{MWR}}$ indicates a write access to the TSB12LV32.
$\overline{\text{TEA}}$	3	O	Transfer error acknowledge. This active-low signal is asserted low for one BCLK cycle whenever there is an illegal transfer request by the microcontroller (i.e., requested data transfer size is unsupported or $\overline{\text{MCS}}$ is asserted low for more than one BCLK cycle in ColdFire mode).

Table 1–1. Terminal Functions (Continued)

TERMINAL NAME	TERMINAL NO.	I/O	DESCRIPTION
Data-Mover Port Interface			
DMD0–DMD15	26 – 29 31 – 34 36 – 39 41 – 44	I/O	Data mover (DM) bidirectional data port. DMD0 is the MSB of these 16 bits.
DMCLK	46	O	Data mover clock at (SCLK/2) MHz
DMDONE	50	O	Data mover done. For transmit, this will be activated when the packet per block counter in the CFR counts down to zero. For receive, this terminal will pulse for one DMCLK prior to the first byte/word available to the DM interface.
DMERROR	52	O	Data mover error. DMERROR is asserted high when there is an error in the received packet or an illegal transmit speed was attempted.
DMPRE	48	O	Data mover predata indicator. In transmit mode, DMPRE pulses for one DMCLK prior to sending the first quadlet. In isochronous receive mode, DMPRE will pulse for one DMCLK when the sync bit in the header matches a bit set in the isochronous register. DMPRE is not used in asynchronous receive mode.
DMREADY	77	I	Data mover ready. Must be asserted high by the external logic controlling the DM interface when it is ready to supply data for transmit. DMREADY must be set low when the data mover is in receive mode.
DMRW	49	O	Data mover read/write indicator. When data is being moved from 1394 to the DM port (receive) this signal will go active high to indicate data is available on DMD[0:15]. When data is being moved from DM to 1394 bus (transmit) this signal will go active high to indicate that data must be supplied to the DMD[0:15] port for transmission.
PKTFLAG	51	O	Packet flag. When set, PKTFLAG is asserted high to indicate the first (header) or last (trailer) quadlet of a received packet on the DM interface. PKTFLAG is not valid in transmit mode.
Phy/Link Interface			
CTL0, CTL1	70, 69	I/O	Phy-link interface control lines.
D0–D7	67, 66, 63–58	I/O	Phy-link interface data lines. Data is only expected on D0 and D1 at 100 Mbit/s, D0–D3 at 200 Mbit/s, and D0–D7 at 400 Mbit/s. D0 is the MSB bit.
LINKON	64	I	Link-on from the Phy is a 4 MHz – 8 MHz clock. This signal will be activated when the link is inactive and the Phy has detected a link-on packet or a Phy interrupt. This clock will persist for no more than 500 ns. When the link detects this terminal as active, it will turn on and drive LPS.
LPS	53	O	Link power status. LPS is used to drive the LPS input to the Phy. It indicates to the Phy that the link is powered up and active. LPS toggles at a rate = 1/16 of BCLK.
LREQ	74	O	Link request to Phy. LREQ makes bus requests and register access requests to the Phy.
SCLK	72	I	System clock. SCLK is a 49.152 MHz clock supplied by the Phy. DMCLK is generated from SCLK.

Table 1–1. Terminal Functions (Continued)

TERMINAL NAME	TERMINAL NO.	I/O	DESCRIPTION
Miscellaneous Functions			
CONTNDR	65	I/O	Contender. When asserted high, this terminal tells the link that this node is a contender for isochronous resource manager (IRM) or bus manager functions. The state of the CONTNDR must match the state of the Phy contender terminal for 1394-1995 compliant Phys, and the Phy register bit for 1394.A compliant Phys. This terminal defaults to being an input on power up. After power up, the value of this terminal may be driven internally by the CTNDRSTAT bit (bit#12 at 08h)
CYCLEIN	76	I	Cycle in. This input is an optional external 8-kHz clock used as the isochronous cycle clock. It should only be used if attached to the cycle-master node. It is enabled by the cycle source bit and should be tied high when not used.
CYSTART	2	O	Isochronous cycle start indicator. CYSTART signals the beginning an isochronous cycle by pulsing for one DMCLK period.
DIRECT	79	I	Isolation terminal. When this terminal is asserted high, no isolation is present between the TSB12LV32 and the Phy. When low, bus holder isolation becomes active.
GND	5, 25, 30, 45, 57, 73, 78, 90, 100		Ground reference
$\overline{\text{INT}}$	1	O	Interrupt. NOR of all internal interrupts.
$\overline{\text{RESET}}$	9	I	System reset. This active-low signal is asynchronous to the TSB12LV32.
STAT0–STAT2	54 – 56	O	General status outputs. STATn is the output signal selected with the CFR at address 20h.
TESTMODE	16	I	This terminal is used to place the TSB12LV32 in the test mode. In normal operation, this terminal must be tied to ground.
$V_{DD}5V$	10, 35, 85		5 V (± 0.5V) supply voltage for 5-V tolerant inputs. Only the Phy/link interface side of the TSB12LV32 is **not** 5-V tolerant. Tie this terminal to the 3.3-V supply voltage if the TSB12LV32 is not connected to any devices driving 5-V signals. Tie this terminal to the 5-V supply voltage if the TSB12LV32 is connected to any devices driving 5-V signals. This terminal is only used to make inputs 5-V tolerant, it is not used for any outputs.
V_{DD}	15, 20, 40, 47, 68, 71, 80, 95		3.3 V (± 0.3 V) supply voltage

1.5.1 STAT0, STAT1, and STAT2 Programming

STAT0, STAT1 and STAT2 terminals can be independently programmed to show one of fourteen possible internal hardware status. The controls for the STAT terminals are in the *Diagnostic* register at address 20h of the CFR register. STAT0 is controlled by STATSEL0(bits 16–19), STAT1 is controlled by bits STATSEL1(bits 20–23), and STAT2 is controlled by STATSEL2 (bits 24–27). Refer to Table 1–2 for programming the STAT terminals.

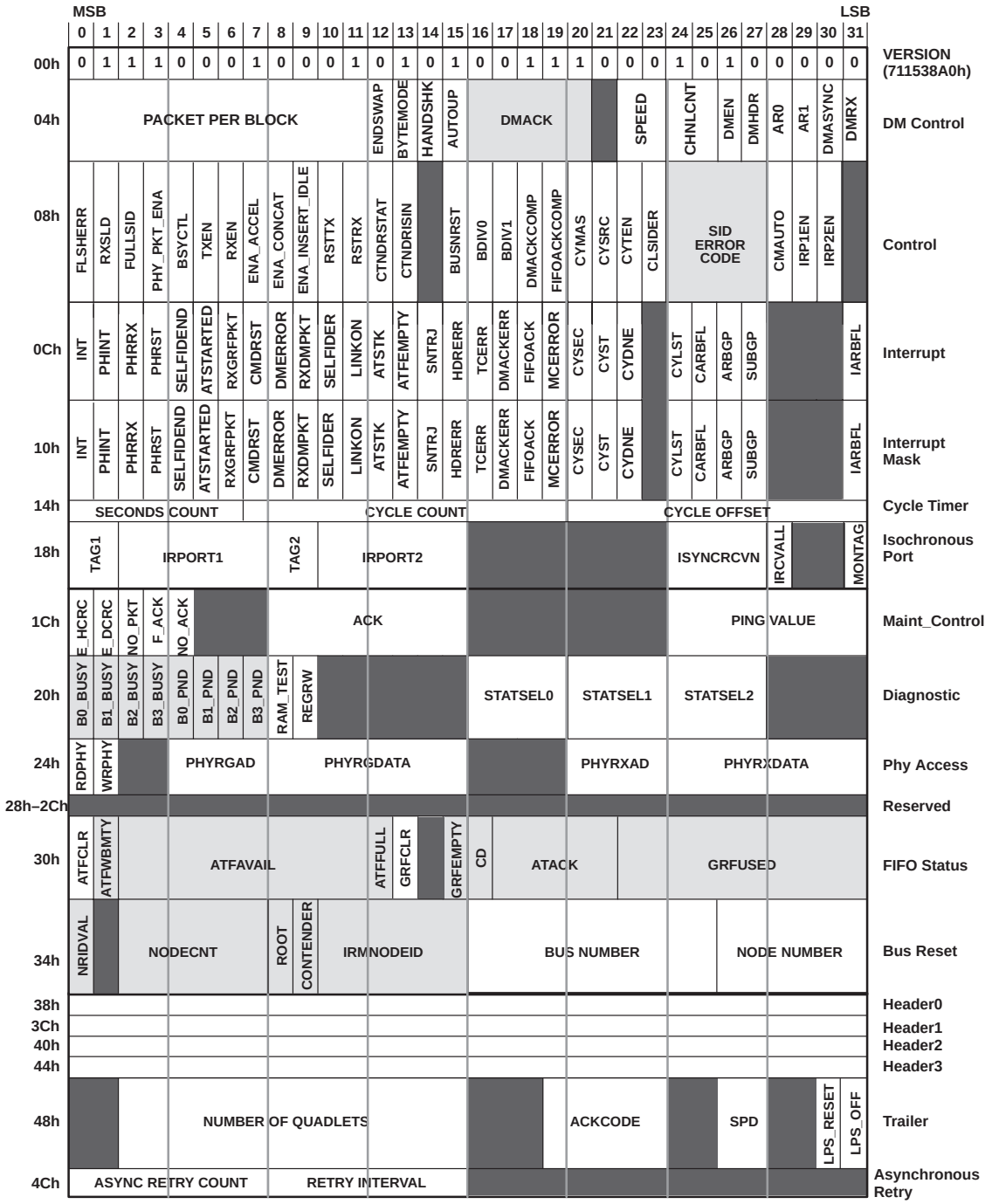
Table 1–2. STAT Terminals Programming

STATSEL0, STATSEL1, or STATSEL2				STAT0/STAT1/STAT2	DESCRIPTION
0	0	0	0	Reserved	Reserved
0	0	0	1	ATFFULL	ATF is full. Bit 12 in CFR at 30h.
0	0	1	0	Bus Reset	1394 Bus reset. Bit 3 in CFR at 0Ch
0	0	1	1	Arbitration reset gap	Bit 26 in CFR at 0Ch
0	1	0	0	CYCLEOUT	Cycle out. This is the link's cycle clock. It is based on the timer controls and the received cycle-start messages.
0	1	0	1	RXDMPKT	Packet received to DM interrupt. Activated at the end of a received packet. Bit 9 in CFR at 0Ch
0	1	1	0	RXGRFPKT	Packet received to GRF interrupt. Activated at the end of a received packet. . Bit 6 of CFR at 0Ch
0	1	1	1	BX_BUSY	Byte busy. This represents the OR of bits 0 – 3 of CFR at 20h
1	0	0	0	SUBGP	Subaction gap. Activated upon detection of a subaction gap. Bit 27 in CFR at 0Ch
1	0	0	1	CYCLE_DONE	Cycle done. Indicates the end of the isochronous period. This happens when a subaction gap has been detected.
1	0	1	0	ATSTARTED (default setting for STAT1)	Activated when an asynchronous packet transfer has started from the ATF. Bit 5 in CFR at 0Ch
1	0	1	1	DMACKERR	DM acknowledge was not Complete. Bit 17 in CFR at 0Ch
1	1	0	0	DMEN	DM enable. Bit 26 in CFR at 04h
1	1	0	1	GRFEMPTY (default setting for STAT2)	GRF is empty. Bit 15 in CFR at 30h.
1	1	1	0	Reserved	Reserved
1	1	1	1	Reserved	Reserved

2 Internal Registers

2.1 TSB12LV32 Configuration Registers

Table 2–1. Configuration Register (CFR) Map



NOTES: A. All dark gray areas (bits) are reserved bits.
B. All light gray areas are read-only bits. All remaining are read/write bits.

Table 2–2. Header Usage for CFRs 38h–44h

DIRECTION OF DM DATA TRANSFER	PACKET TYPE	AUTO HEADER INSERT/ EXTRACT	HEADER REGISTER
TRANSMIT (to 1394 Bus)	Isochronous	YES	Header 0 CFR formatted for isochronous transmission. Header1 – Header3 are used for additional channels.
		NO	Isochronous header supplied by DM interface. Header0 CFR is automatically written with extracted (from transmitted packet) isochronous header.
	Asynchronous/ asynchronous streaming	YES	Header0–Header3 CFRs formatted for asynchronous transmission.
		NO	Asynchronous header supplied by DM interface. Header0 – Header3 CFRs are automatically written with extracted (from transmitted packet) header.
RECEIVE (from 1394 Bus)	Isochronous/ asynchronous streaming	YES	Header0 – Header3 are always automatically updated. The isochronous header is streamed through the DM port. The trailer quadlet is always appended to the data stream.
		NO	Header0 – Header3 are always automatically updated. The isochronous header is streamed through the DM port along with the payload data. The trailer quadlet is always appended to the data stream.
	Asynchronous	YES	Header0 – Header3 are always automatically updated. Asynchronous headers are not streamed through the DM port. The trailer quadlet is always appended to the data stream.
		NO	Header0 – Header3 are always automatically updated. Asynchronous headers are streamed through the DM port along with data. The trailer quadlet is always appended to the data stream.

2.2 Configuration Register Definitions

2.2.1 Version Register at 00h

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	1	1	1	0	0	0	1	0	0	0	1	0	1	0	1	0	0	1	1	1	0	0	0	1	0	1	0	0	0	0	0

This register uniquely identifies this device to the software. The value is fixed at **7115_38A0'h**. This register is read only.

2.2.2 Data Mover Control Register at 04h

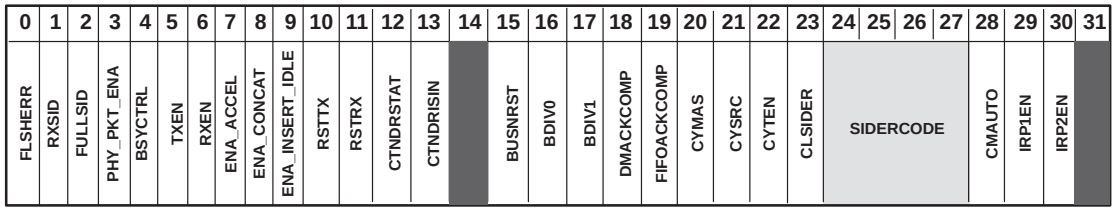
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
PACKET PER BLOCK												ENDSWAP	BYTEMODE	HANDSHK	AUTOUP	DMACK						SPEED		CHNLCNT		DMEN	DMHDR	AR0	AR1	DMASYNC	DMRX

This register controls the Data Mover port and must be set up before using the port. The power-up reset value of this register = **0000_0000'h**

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0–11	PACKET PER-BLOCK	Packets per Block	R/W	Number of packets per block. A packet is the size of the data payload and is specified as part of the header. The data mover logic uses this value to deactivate DMDONE. This field is only used in transmit mode.
12	ENDSWAP	Endian Swap	R/W	Swap endian. When this bit is set, the quadlet formed by stacking the DM data will be byte reversed, (i.e. the quadlet formed by fetching doublet AB01 then 'CD02' will be 02CD–01AB instead of AB01CD02). In byte mode the quadlet formed by fetching AB, 01, CD, 0 will be 02CD01AB instead of AB01CD02.
13	BYTEMODE	Byte Mode	R/W	Byte mode. When this bit is set the DM port will only look at DM0–DM7. DM8–DM15 will be ignored for transmit and will not be driven on receive. In this mode, the maximum speed allowed is 200 Mbps.
14	HANDSHK	Handshake Mode (CPLynx Mode)	R/W	Handshake. When this bit is 1 DMREADY and DMDONE are in strict handshake mode (i.e., TSB12LV31 compatible mode). DMREADY must not be deactivated until DMDONE activates. When this bit is set to 0, DMREADY may be deactivated before DMDONE activates.
15	AUTOUP	Automatic Address Update	R/W	Automatic update offset address. Valid only for asynchronous transmit using header insert mode (bit 27 DMHDR set to 1). For write request asynchronous packets, header quadlet 2 contains the destination offset low address for the write. When this bit is set, header quadlet 2 will be updated by the value of the payload size (rounded up to the nearest quadlet boundary).
16–20	DMACK	DM Acknowledge	R	DM acknowledge. This is the ack received from the receiving node. This is updated only when the transfer is from the DM port.
21	RESERVED			RESERVED
22–23	SPEED	DM Speed Code	R/W	Speed code. This is valid for isochronous transmit and asynchronous transmit through the DM port. The DM logic uses this field to specify to the Phy the speed of the isochronous transfer.

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
24–25	CHNLCNT	Channel Count	R/W	Channel count. This field is valid only in isochronous transmit. This field allows the node to transmit multiple packets during a single isochronous period. Each packet must have a different channel number, however, hardware does not check this. When the isochronous transmit header is supplied by the DM interface or automatically inserted by the hardware, a maximum of four different channels may be accessed in one isochronous period. In isochronous transmit with automatic header insert, Header0–Header3 CFRs are used as the isochronous header registers.
26	DMEN	DM Enable	R/W	DMEN controls the transmission of packets from the DM port. If this bit is 0, transmission through from the DM port is inhibited. This is used for asynchronous flow control. In normal operation, if an asynchronous packet transmitted from the DM port receives an acknowledge from the receiving node other than *ack complete*, this bit will be set to 0 and DMERROR is asserted high. Software will need to set this bit to allow further transmission of asynchronous packets from the DM port. The default and power-up value is 0.
27	DMHDR	DM Header Insert Control	R/W	DM header insert bit. When set to 0, the hardware will automatically insert the header(s) into the DM transmit data. In receive, setting this bit to 0 will strip off the header(s) before routing packet to the DM. Header(s) are always written to the CFR header registers regardless of the value of DMHDR.
28–29	AR0, AR1	Receive Control Routing	R/W	Receive packet routing control encoded bits. These bits in conjunction with DMASYNC and DMRX bits in the DM control register controls the routing of the received packet to either the data mover port or to the GRF. Refer to Table 4–1.
30	DMASYNC	DM Asynchronous	R/W	If this bit is set to 1 the DM port is configured for asynchronous traffic only. The DM port can not accept both asynchronous and isochronous traffic. It must be configured for asynchronous (DMASYNC = 1) or isochronous (DMASYNC = 0).
31	DMRX	DM Receive	R/W	If this bit is set to 1 the DM port is configured to receive. The DM port cannot both transmit and receive data at the same time, it must be configured for either transmit or receive.

2.2.3 Control Register at 08h



The control register dictates the basic operation of the TSB12LV32. The power-up reset value of this register equals **E004_0200'h**

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0	FLSHERR	Flush GRF on error	R/W	This bit controls the flushing of the GRF when a packet with a data CRC error is detected. The power–up value is 1, which means flush the GRF when a data CRC error is detected.
1	RXSID	Received Self-ID packets	R/W	If set, the self-identification (SID) packets generated by Phy devices during the bus initialization are received and placed into the GRF as a single packet. The default setting of this bit is 1. When set to 0, the SIDs are not placed into the GRF.
2	FULLSID	Save full Self-ID Packet in GRF	R/W	Save the full self-ID packets.When this bit is 1 the self-ID data quadlet and its inverse quadlet are saved in the GRF. When this bit is 0 only the self-ID data quadlet is saved in the GRF.
3	PHY_PKT_ENA	Phy Packets Receive Enable	R/W	Phy packet enable allows reception of all Phy packets. If this bit is reset to 0, all Phy packets, except for self-IDs, will be rejected and interrupt HDERR (if not masked) will be generated. One HDERR interrupt will be generated for every Phy packet received.
4	BSYCTRL	Busy Control	R/W	BSYCTRL controls which busy status the chip returns to incoming packets. When this bit is 0 the chip follows normal busy/retry protocol, only send busy when necessary. When this bit is 1 the chip sends a busy acknowledge to all incoming packets following the normal busy/retry protocol.
5	TXEN	Transmit Enable	R/W	When TXEN is cleared, the transmitter does not arbitrate or send packets. TXEN bit is cleared following a bus reset, and all traffic through the DM port will be interrupted. TXEN must be set before packet transmit can resume. Power-on reset value of TXEN is 0
6	RXEN	Receive Enable	R/W	When RXEN is cleared, the receiver does not receive any packets. This bit is not affected by a bus reset and is set to 0 after a power-on reset.
7	ENA_ACCEL	Acceleration Enable	R/W	Enable acceleration. When this bit is set, fly-by acceleration and accelerated arbitration are enabled. This bit cannot be set while TXEN and RXEN are set. This bit must only be used with a 1394a capable Phy.
8	ENA_CONCAT	Concatenation Enable	R/W	Enable concatenation. When this bit is set it allows the link to concatenate multiple isochronous or asynchronous packets. This bit must only be used with a 1394a capable Phy.

<table>
<tr><th>BIT NUMBER</th><th>BIT NAME</th><th>FUNCTION</th><th>DIR</th><th>DESCRIPTION</th></tr>
<tr><td>9</td><td>ENA_ INSERT_IDLE</td><td>Insert Idle Enable</td><td>R/W</td><td>Per P1394a, the link is required to insert an idle state on the control lines after the Phy grants the link control of the Phy/link interface. If using a P1394a Phy, this bit should be set to 1 in order for the link to drive an idle state following the grant state from the Phy. For 1394-1995 Phys this bit must remain low.</td></tr>
<tr><td>10</td><td>RSTTX</td><td>Transmitter Reset</td><td>R/W</td><td>When RSTTX is set, the entire transmitter resets synchronously. This bit clears itself.</td></tr>
<tr><td>11</td><td>RSTRX</td><td>Receiver Reset</td><td>R/W</td><td>When RSTRX is set, the entire receiver resets synchronously. This bit clears itself.</td></tr>
<tr><td>12</td><td>CTNDRSTAT</td><td>Contenter status</td><td>R/W</td><td>Contender status. On power up, this bit reflects the status of the CONTNDR pin. When bit 13, CTNDRISIN, is 0 this bit will be driven out to the CONTNDR pin. If CTNDRISIN is 1 this bit is not used. (Only use on 1394–1995 Phys, or P1394a Phys when using hardware reset, otherwise, use the 1394a Phy registers to set the nodes contender status).</td></tr>
<tr><td>13</td><td>CTNDRISIN</td><td>Contender Driver Enable</td><td>R/W</td><td>Driver enable for the CONTNDR pin. On power up this bit is set to 1 which disables the driver and allows reading of the state of the CONTNDR pin. Writing a 0 to this bit will enable the driver and will drive bit 12, CTNDRSTAT, to the CONTNDR pin.</td></tr>
<tr><td>14</td><td>RESERVED</td><td></td><td></td><td>Reserved</td></tr>
<tr><td>15</td><td>BUSNRST</td><td>Bus number reset enable</td><td>R/W</td><td>When this enable is set to high, the bus number field clears to 3FFh when a local bus reset is received.</td></tr>
<tr><td>16–17</td><td>BDIV0, BDIV1</td><td>BCLK divisor encode bits</td><td>R/W</td><td>BCLK divisors encode bits. Used to divide down the BCLK to generate the link power status (LPS) clock to the Phy.
<table>
<tr><th>BDIV0</th><th>BDIV1</th><th>DESCRIPTION</th></tr>
<tr><td>0</td><td>0</td><td>Divide by 16. Default power on value. Recommended for BCLK frequencies in the range of 8 – 88 MHz.</td></tr>
<tr><td>0</td><td>1</td><td>Divide by 2. Recommended for BCLK frequencies in the range of 1 – 11 MHz.</td></tr>
<tr><td>1</td><td>0</td><td>Divide by 4. Recommended for BCLK frequencies in the range of 2 – 22 MHz.</td></tr>
<tr><td>1</td><td>1</td><td>Divide by 32. Recommended for BCLK frequencies in the range of 16 – 176 MHz</td></tr>
</table></td></tr>
<tr><td>18</td><td>DMACKCOMP</td><td>Data Mover Acknowledge Complete</td><td>R/W</td><td>Data mover acknowledge complete. This bit controls the acknowledge response to an asynchronous packet received and routed to the DM port. The default and power on value is 0 which means to respond with ack pending. A 1 means to respond with an ack complete for write request packets.</td></tr>
<tr><td>19</td><td>FIFOACKCOMP</td><td>FIFO Acknowledge Complete</td><td>R/W</td><td>FIFO acknowledge complete. This bit controls the acknowledge response to an asynchronous packet received and routed to the GRF. The default and power on value is 0 which means to respond with ack pending. A 1 means to respond with ack complete.</td></tr>
</table>

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
20	CYMAS	Cycle Master	R/W	When CYMAS is set and the TSB12LV32 is attached to the root Phy, the cyclemaster function is enabled. When the cycle_count field of the cycle timer register increments, the transmitter sends a cycle-start packet.
21	CYSRC	Cycle Source	R/W	When CYSRC is set, the cycle_count field increments and the cycle_offset field resets for each positive transition of CYCLEIN. When CYSRC is cleared, the cycle_count field increments when the cycle_offset field rolls over.
22	CYTEN	Cycle timer enable	R/W	When CYTEN is set, the cycle_offset field increments.
23	CLRSIDER	Self-ID error-code clear	W	When CLRSIDER is set, the SIDERCODE field (bits 24–27) is cleared.This bit clears itself.
24–27	SIDERCODE	Self-ID error code	R	SIDERCODE contains the error code of the first Self-ID Error. The error code is as follows:
				0000 — No error
				0001 — Last self-ID received was not all child ports
				0010 — Received Phy ID in self-ID not as expected
				0011 — Quadlet not inverted (phase error)
				0100 — Phy ID sequence error (two or more gaps in IDs)
				0101 — Phy ID sequence error (large gap in IDs)
				0110 — Phy ID error within packet
				0111 — Quadlet not the inversion of the prior quadlet
				1000 — Reserved
28	CMAUTO	Auto set cycle master	R/W	When CMAUTO is high, the TSB12LV32 automatically enables CYMAS when the this node becomes the root following a bus reset.
29	IRP1EN	IR port 1 enable	R/W	When IRP1EN is set, the receiver accepts isochronous packets when the channel number matches the value in the IR port1 field at18h
30	IRP2EN	IR port 2 enable	R/W	When IRP2EN is set, the receiver accepts isochronous packets when the channel number matches the value in the IR Port2 field at18h
31	RESERVED			Reserved

2.2.4 Interrupt/Interrupt Mask Register at 0Ch and 10h

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
INT	PHINT	PHRRX	PHRST	SELFIDEND	ATSTARTED	RXGRFPKT	CMDRST	DMERROR	RXDMPKT	SELFIDER	LINKON	ATSTK	ATFEMPTY	SNTRJ	HDRERR	TCERR	DMACKERR	FIFOACK	MCERROR	CYSEC	CYST	CYDNE		CYLST	CARBFL	ARBGP	SUBGP				IARBFL

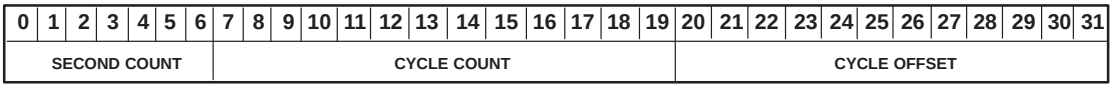
The interrupt and interrupt mask register work in tandem to inform the host bus interface when the state of the TSB12LV32 changes. The interrupt register is at 0Ch, the interrupt mask register is at 10h. The interrupt register powers up all 0s, however, the interrupt mask register powers up with the INT and the MCERROR bit set, i.e. 8000_1000h. The mask bits allows individual control for each interrupt. A 1 in the mask bit field allows the corresponding interrupt in the interrupt register to be generated. Once an interrupt is generated it must be cleared by writing a 1 to the bit in the interrupt register. For testing, each interrupt bit can be set manually. This is done by first setting the REGRW bit at20h and then setting the individual interrupt bit. This is also true for bit 0 at0Ch. In this test mode, the interrupt mask register is not used and has no effect.

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0	INT	Interrupt	R/W	INT contains the value of all interrupt and interrupt mask bits ORed together
1	PHINT	Phy chip interrupt	R/W	When PHINT is set, the Phy has signalled an interrupt through the Phy interface
2	PHRRX	Phy register information received	R/W	When PHRRX is set, a register value has been transferred to the Phy access register (offset 24h) from the Phy interface
3	PHRST	Phy reset started	R/W	When PHRST is set, a Phy-LLC reconfiguration has started (1394 bus reset)
4	SELFIDEND	Self-ID validated	R/W	Self-ID end. This bit is set at the end of the self-ID reporting process. When this bit is set, the contentF of the bus reset CFR at34h is valid.
5	ATSTARTED	Asynchronous transfer started	R/W	Asynchronous transfer started. Activated when the bus has been granted and the first quadlet from the FIFO is about to be popped from the ATF.
6	RXGRFPKT	GRF packet received	R/W	Receive packet to GRF. This bit is set whenever a complete packet has been confirmed into the GRF (asynchronous or isochronous).
7	CMDRST	CSR register reset request	R/W	If CMDRST is set, the receiver has been sent a quadlet write request to the Reset_Start CSR register(target address is FFFF_F000_000Ch)
8	DMERROR	Data Mover error	R/W	DM error. This bit will be set whenever there is an error in the DM stream. For transmit, if the DM port is configured for byte access and the speed code in the DM control register or the asynchronous header register is set for 400 Mbps then this bit will be set. Under this condition DMEN will be reset to 0 preventing further transmit. For receive this bit will be set if there is a header or data CRC error or if the DM port is configured for byte access and the data is received at 400 Mbps.
9	RXDMPKT	Data Mover packet receive	R/W	Receive packet to DM. This bit is set whenever a packet is received to the DM port.
10	SELFIDER	Self-ID packet error	R/W	Set if an error in the self-ID quadlet/packet has been detected.

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
11	LINKON	Link-ON detect	R/W	Set if a link-on pulse is detected on the LINKON input terminal. This bit should be used by software to reactivate the LPS output to the Phy.
12	ATSTK	Transmitter is stuck (AT)	R/W	When ATSTK is set, the transmitter has detected invalid data at the asynchronous transmit-FIFO interface. If the first quadlet of a packet is not written to the ATF_First or ATF_First&Update, the underflow of the ATF also causes an ATStuck interrupt. When this state is entered, no asynchronous packets can be sent until the ATF is cleared by way of the CLR ATF control bit. Isochronous packets can be sent while in this state.
13	ATFEMPTY	ATF empty interrupt	R/W	ATFEMPTY. This bit is set to 1 when the ATF is empty.
14	SNTRJ	Busy acknowledge sent by receiver	R/W	When SNTRJ is set, the receiver is forced to send a busy acknowledge to a packet addressed to this node because the GRF overflowed.
15	HDRERR	Header error	R/W	When HDRERR is set, the receiver detected a header CRC error on an incoming packet that may have been addressed to this node.
16	TCERR	Transaction code error	R/W	When TCERR is set, the transmitter detected an invalid transaction code in the data at the transmit-FIFO interface.
17	DMACKERR	Data Mover acknowledge error	R/W	DM acknowledge error. Set to 1 when the acknowledge received is not *ack complete*. When this occurs, DMEN(bit 26) of the DM Control CFR at04h will be reset to 0 and no more asynchronous transmit from the DM port will be allowed to take place until DMEN is set to 1.
18	FIFOACK	FIFO acknowledge interrupt	R/W	FIFO ack interrupt. This bit will be set when an acknowledge from a previous ATF transmit has been received.
19	MCERROR	Micro-interface error	R/W	Micro-interface error. Set whenever the microcontroller write protocol is violated.
20	CYSEC	Cycle second incremented	R/W	When CYSEC is set, the cycle-second field in the cycle timer register has incremented. This occurs about every second when the cycle timer is enabled.
21	CYST	Cycle started	R/W	When CYST is set, the transmitter has sent or the receiver has received a cycle-start packet.
22	CYDNE	Cycle done	R/W	When CYDNE is set, an arbitration gap has been detected on the bus after the transmission or reception of a cycle-start packet. This indicates that the isochronous cycle is over.
23	RESERVED			RESERVED
24	CYLST	Cycle lost	R/W	When CYLST is set, the cycle timer has rolled over twice without the reception of a cycle-start packet. This occurs only when this node is not the cycle master. All isochronous traffic stop once CYLST is set. However, asynchronous and asynchronous streaming traffic will not be affected.
25	CARBFL	Cycle arbitration failed	R/W	When CARBFL is set, the arbitration to send a cycle-start packet has failed.
26	ARBGP	Arbitration gap	R/W	When ARBGP is set, the serial bus has been idle for an arbitration reset gap.

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
27	SUBGP	Subaction gap	R/W	When SUBGP is set, the serial bus has been idle for a subaction gap time (fair-gap). This bit can be set only when the REGRW bit has been set in the diagnostics register at 20h.
28–30	RESERVED			RESERVED
31	IARBFL	Isochronous arbitration failed	R/W	When IARFL is set, the arbitration to send an isochronous packet has failed.

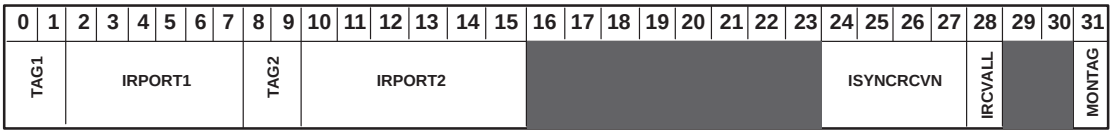
2.2.5 Cycle Timer Register at 14h



This register must be written to as a quadlet. The power-up reset value of this register = **0000_0000'h**

BIT NUMBER	BIT NAME	FUNCTION NAME	DIR	DESCRIPTION
0–6	Seconds_count	Seconds count	R/W	1-Hz cycle timer counter
7–19	Cycle_count	Cycle count	R/W	8,000-Hz cycle timer counter
20–31	Cycle_offset	Cycle offset	R/W	24.576-MHz cycle timer counter

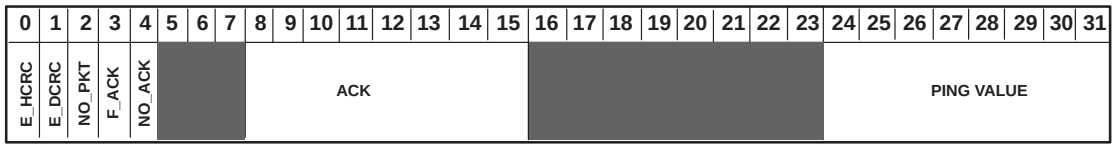
2.2.6 Isochronous Port Register at 18h



The power-up reset value of this register = 0000_0000h

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0–1	TAG1	Tag Field 1	R/W	The TAG1 field can further qualify the isochronous reception for isochronous Receive PORT1 when the MONTAG bit is set.
2–7	IRPORT1	Isochronous receive port 1 channel number	R/W	IR port1 contains the channel number of the isochronous packets that the receiver accepts. The receiver accepts isochronous packets with this channel number when the IRP1EN is set.
8–9	TAG2	Tag Field 2	R/W	The TAG2 field can further qualify the isochronous reception for isochronous Receive PORT2 when the MONTAG bit is set.
10–15	IRPORT2	Isochronous receive port 2 channel number	R/W	IR port2 contains the channel number of the isochronous packets that the receiver accepts. The receiver accepts isochronous packets with this channel number when the IRP2EN is set.
16–23	RESERVED			Reserved
24–27	ISYNCRCVN	Synchronous Enable	R/W	In isochronous receive mode to the DM port, when the ISYNCRCVN enable bits are high, the DMPRE terminal pulses when an isochronous packet is received whose SYNC bit field in its header matches the bit pattern in this field. The default is 0000b.
28	IRCVALL	Receive all isochronous packets	R/W	When the IRCVALL bit is set high, the TSB12LV32 receives all isochronous packets regardless of the channel number or tag number. The default is off.
29–30	RESERVED			Reserved
31	MONTAG	Match on tag	R/W	MONTAG is set when the user wants to only accept isochronous packets that match both the tag field and the channel number field. When set, MONTAG indicates that isochronous receive data is accepted. The default is off.

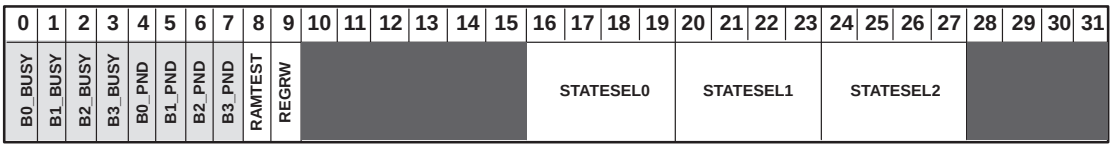
2.2.7 Maint_Control Register at 1Ch



This register is used to generate test conditions. The control bits in this register allow errors to be inserted into various places in the packets generated by this node. After the completion of error insertion, enabled error-insertion controls are disabled. The power-up reset value of this register = **0000_0000'h**

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0	E_HCRC	Header CRC Error	R/W	If E_HCRC is set, the packet header CRC component of the next primary packet generated by this node shall be in error or shall be invalid; otherwise, this bit has no effect. After the next packet for this node is generated, this bit will be cleared.
1	E_DCRC	Data CRC Error	R/W	If E_DCRC is set, the packet data CRC component of the next primary packet generated by this node shall be in error or shall be invalid; otherwise, this bit has no effect. After the next packet for this node is generated, this bit will be cleared to zero immediately upon transmission of the erroneous CRC.
2	NO_PKT	No Packet	R/W	If NO_PKT is set, the next primary packet to be generated by this node shall be discarded. This bit will be cleared to zero immediately after the next packet for this node is discarded.
3	F_ACK	Ack Field	R/W	If F_ACK is set, the ack field shall be used within the next acknowledge packet generated by this node. This bit will be cleared to zero immediately after the next acknowledge packet for this node is generated.
4	NO_ACK		R/W	If NO_ACK is set, the next acknowledge packet (that would normally have been generated by this node) is not sent. This bit will be immediately cleared to zero when the next acknowledge packet for this node is discarded.
5–7	RESERVED			Reserved
8–15	ACK		R/W	The 8-bit ACK field contains the 8-bit acknowledge packet (ack_code and ack_parity) to be supplied when the F_ACK bit indicates a modified acknowledge packet is to be generated.
16–23	RESERVED			Reserved
24–31	PINGVALUE	Ping timer value	R/W	Ping timer value. This value reflects the time it takes a node to respond to a ping packet. The granularity of this timer is 40 ns.

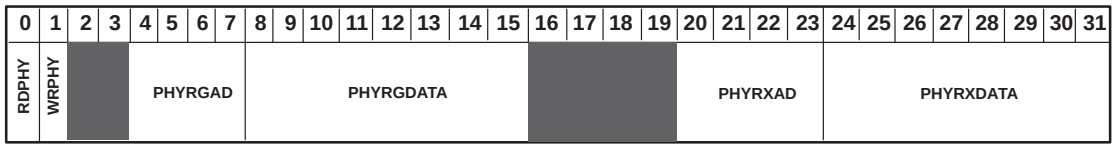
2.2.8 Diagnostic Register at 20h



The power-up reset value of this register = **0000_4AD0'h**

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0	B0_BUSY	Byte 0 busy	R	Byte 0 busy. When this bit is set, no microinterface write to byte 0 of any CFRs is allowed. The microinterface must first poll this bit before writing to byte 0.
1	B1_BUSY	Byte 1 busy	R	Byte 1 busy. When this bit is set, no microinterface write to byte 1 of any CFRs is allowed. The microinterface must first poll this bit before writing to byte 1.
2	B2_BUSY	Byte 2 busy	R	Byte 2 busy. When this bit is set, no microinterface write to byte 2 of any CFRs is allowed. The microinterface must first poll this bit before writing to byte 2.
3	B3_BUSY	Byte 3 busy	R	Byte 3 busy. When this bit is set, no microinterface write to byte 3 of any CFRs is allowed. The microinterface must first poll this bit before writing to byte 3.
4	B0_PND	Byte 0 pending	R	Byte 0 pending. When this bit is set, it indicates that byte 0 of a word or quadlet write has been accepted and the hardware is waiting for the remaining bytes to be written. When the full write is complete, this bit will be cleared.
5	B1_PND	Byte 1 pending	R	Byte 1 pending. When this bit is set, it indicates that byte 1 of a word or quadlet write has been accepted and the hardware is waiting for the remaining bytes to be written. When the full write is complete, this bit will be cleared.
6	B2_PND	Byte 2 pending	R	Byte 2 pending. When this bit is set, it indicates that byte 2 of a word or quadlet write has been accepted and the hardware is waiting for the remaining bytes to be written. When the full write is complete, this bit will be cleared.
7	B3_PND	Byte 3 pending	R	Byte 3 pending. When this bit is set, it indicates that byte 3 of a word or quadlet write has been accepted and the hardware is waiting for the remaining bytes to be written. When the full write is complete this bit will be cleared.
8	RAM_TEST		R/W	This bit can be set only when TESTMODE is high. When this bit is set, the built in self test(BIST) for the FIFOs (transmit and receive) will be run. On completion of the test hardware will reset this bit to 0 and simultaneously set bit 30 and 31.
9	REGRW	Register read/write access	R/W	When REGRW is set, write-protected bits in various registers can be written to.
10–15	RESERVED			Reserved
16–19	STATSEL0	State0 select	R/W	Status output select bits. Used to program the output of STAT0 terminal. See table in *Operation* section.
20–23	STATSEL1	State1 select	R/W	Status output select bits. Used to program the output of STAT1 terminal. See table in *Operation* section.
24–27	STATSEL2	State2 select	R/W	Status output select bits. Used to program the output of STAT2 terminal. See table in Operation section.
28–31	RESERVED			Reserved

2.2.9 Phy Access Register at 24h



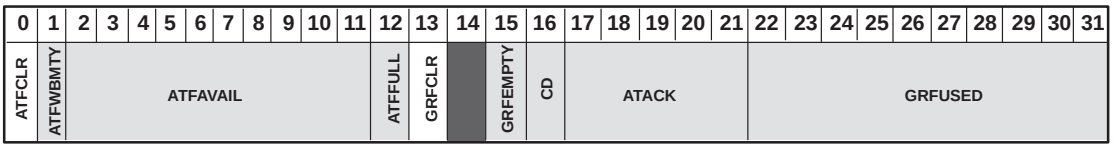
The Phy access register allows access to the registers in the attached Phy. The most significant 16 bits send read and write requests to the Phy registers. The least significant 16 bits are for the Phy to respond to a read request sent by the TSB12LV32. The Phy access register also allows the Phy interface to send information back to the TSB12LV32. When the Phy interface sends new information to the TSB12LV32, the Phy register-information-receive (PhyRRx) interrupt is set. The Phy acess register is at address 24h and is a read/write register. The power-up reset value of this register = **0000_0000'h**.

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0	RDPHY	Read Phy register	R/W	When RDPHY is set, the TSB12LV32 sends a read register request with the address equal to the PHYRGAD field to the Phy interface. This bit is cleared when the request is sent.
1	WRPHY	Write Phy register	R/W	When WRPHY is set, the TSB12LV32 sends a write register request with the address equal to the PHYRGAD field to the Phy interface. This bit is cleared when the request is sent.
2–3	RESERVED			RESERVED
4–7	PHYRGAD	Phy-register address	R/W	PHYRGAD is the address of the Phy register that is to be accessed.
8–15	PHYRGDATA	Phy-register data	R/W	PHYRGDATA is the data to be written to the Phy register indicated in PHYRGAD.
16–19	RESERVED			RESERVED
20–23	PHYRXAD	Phy-register received address	R/W	PHYRXAD is the address of the register from which PHYRXDATA came. For testing, these bits can be set only when the REGRW bit has been set in the diagnostics register at20h.
24–31	PHYRXDATA	Phy-register received data	R/W	PHYRXDATA contains the data from the register addressed by PHYRXAD. For testing, these bits can be set only when the REGRW bit has been set in the diagnostics register at20h.

2.2.10 Reserved Registers at 28h – 2Ch

These registers are reserved for future use.

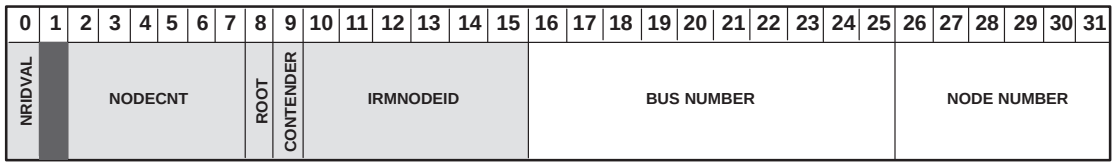
2.2.11 FIFO Status Register at 30h



The power-up reset value of this register = **6083_0000'h**

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0	ATFCLR	ATF clear	R/W	ATF clear. When set to 1 will clear the ATF. This bit clears itself.
1	ATFWBMTY	ATF write buffer empty	R	ATF write buffer empty. Set when the 4-quadlet FIFO write buffer is empty.
2–11	ATFAVAIL	ATF space available	R	Size of ATF available, in quadlets. The power-on value of this field is 1000001000 (520 quadlets)
12	ATFFULL	ATF full bit	R	When the ATF is full, this bit will be set.
13	GRFCLR	GRF clear bit	R/W	GRF clear bit. Set to 1 to clear the contents of the GRF.
14	RESERVED			Reserved
15	GRFEMPTY	GRF empty	R	GRF empty. GRFEMPTY is set when the four-quadlet GRF read buffer is empty.
16	CD	Check 33rd bit (GRF read)	R	This bit is set to 1 when the quadlet pointed to by the GRF read pointer is the first quadlet or the packet trailer.
17–21	ATACK	ATF acknowledge	R	The acknowledge received in response to a packet sent via the ATF.
22–31	GRFUSED	GRF space used	R	GRF space used, in quadlets. This value is the amount of used up quadlets in the GRF.

2.2.12 Bus Reset Register at 34h



The power-up reset value of this register = **81BF_FFC0'h**.

NOTE:

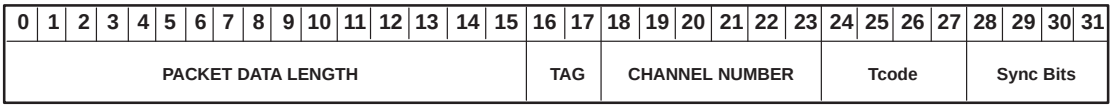
The power-up reset value shown above assumes one node on the bus only. A P1394a compliant Phy is assumed to be attached to the TSB12LV32. If a 1394-1995 Phy is attached to the TSB12LV32 link, the NODECNT filed will be 0. This is due to the fact that a 1394-1995 compliant Phy does not report its self-ID packet back to the local link.

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0	NRIDVAL	Valid	R	When set, NRIDVAL indicates that the Node ID, IRM Node ID, Node Count, and Root information are valid. This bit is read only.
1	RESERVED			Reserved
2–7	NODECNT	Node count	R	NODECNT contains the number of nodes detected in the system. This field is loaded with 1 following a power-on reset. The NODECNT field is read only.
8	ROOT	Root	R	Root is set when the current node is the root node. This bit is read only.
9	CONTENDER	Contender	R	Contender contains the status of the TSB12LV32 CONTNDR terminal. This bit is read only.
10–15	IRMNODEID	IRM node identification	R	IRMNODEID is the isochronous resource manager node identification. If there is no IRM node present on the bus, these bits will be equal to 3Fh. These bits are read only.
16–25	BUSNUMBER	Bus number	R/W	BUSNUMBER is the 10-bit IEEE-1212 bus number. These bits are set to 3FFh when RBUSNUM is set and there is a bus reset.
26–31	NODENUMBER	Node number	R/W	NODENUMBER is the node number of the current node. These bits are automatically updated following a bus reset. To change the node number of this node (spoofing), the TESTMODE terminal must be set high.

2.2.13 Header0 Register at 38h

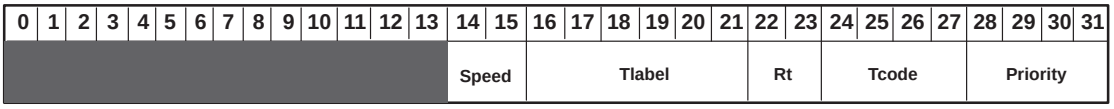
Header0 register must contain the isochronous header or the first quadlet of an asynchronous header if in header insert mode. If not in header insert mode or if in receive mode, this register will be updated with the received header. This register is write protected such that it cannot be written to unless automatic header insert mode is enabled and DM is in transmit mode(i.e.,DMHDR=0 and DMRX=0). The power-up reset value of this register = **0000_0000'h**

ISOCHRONOUS HEADER FOR QUADLET 0



BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0–15	PacketData-Length	Packet data length	R/W	Packet data length in bytes.
16–17	TAG	Tag field	R/W	The tag field provides a high-level label for the format of the data carried by the isochronous packet.
18–23	ChannelNumber	Channel number	R/W	Channel number field
24–27	Tcode	Transmission code	R/W	Packet transaction code
28–31	Syncbits	Synchronization code	R/W	An application-specific control field

ASYNCHRONOUS HEADER FOR QUADLET 0



BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0–13	RESERVED			RESERVED
14–15	Speed	Speed	R/W	Speed at which the Phy is to transmit a packet: 00 => S100 01 => S200 10 => S400
16–21	Tlabel	Transaction label	R/W	Transaction label
22–23	Rt	Retry code	R/W	The retry code specifies whether this packet is a retry attempt and the retry protocol to be followed by the destination node.
24–27	Tcode	Transaction code	R/W	The transaction code specifies the packet format and the type of transaction that is to be performed.
28–31	Priority	Priority field	R/W	Priority code (applies only to the backplane Phy)

2.2.14 Header1 Register at 3Ch

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Destination ID																															

Header1 register must contain the isochronous header or the second quadlet of an asynchronous header if in header insert mode. If not in header insert mode or if in receive mode, this register will be updated with the received header. This register powers up with all bits reset to 0. For multiple isochronous packets (within the same isochronous cycle), this register would contain the isochronous header of the second isochronous packet in the same format as the header0 register, if in header insert mode. This register is write protected such that it cannot be written to unless automatic header insert mode is enabled and DM is in transmit mode (i.e., DMHDR=0 and DMRX=0).

BIT NUMBER	BIT NAME	FUNCTION	DIR	DESCRIPTION
0–15	DestinationID	Destination ID	R/W	For asynchronous packets this field contains the destination nodes ID.
16–31	RESERVED			Reserved

2.2.15 Header2 Register at 40h

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
HEADER2																															

Header2 register must contain the isochronous header or the third quadlet of a asynchronous header if in header insert mode. If not in header insert mode or if in receive mode, this register will be updated with the received header. This register powers up with all bits reset to 0. For multiple isochronous packets (within the same isochronous cycle), this register would contain the isochronous header of the third isochronous packet in the same format as the header0 register, if in header insert mode. This register is write protected such that it cannot be written to unless automatic header insert mode is enabled and in transmit mode (i.e., DMHDR=0 and DMRX=0).

BIT NUMBER	BIT NAME	DIR	DESCRIPTION
0–31	Header2	R/W	Header quadlet for asynchronous or isochronous packet.

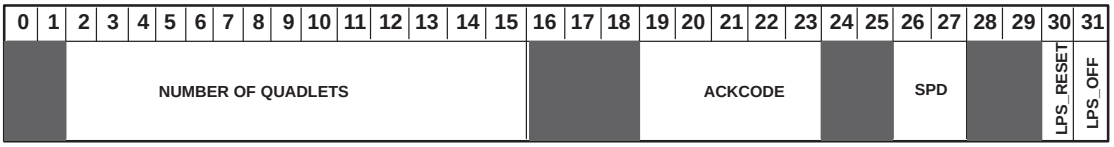
2.2.16 Header3 Register at 44h

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
HEADER3																															

Header3 register must contain the isochronous header or the fourth quadlet of an asynchronous header if in header insert mode. If not in header insert mode or if in receive mode, this register will be updated with the received header. This register powers up with all bits reset to 0. For multiple isochronous packets (within the same isochronous cycle), this register would contain the isochronous header of the fourth isochronous packet in the same format as the header0 register, if in header insert mode. This register is write protected such that it cannot be written to unless automatic header insert mode is enabled and DM is in transmit mode (i.e., DMHDR=0 and DMRX=0).

BIT NUMBER	BIT NAME	DIR	DESCRIPTION
0–31	Header3	R/W	Header quadlet for asynchronous or isochronous packet.

2.2.17 Trailer Register at 48h



The power-up reset value of this register = **0000_0000'h**.

BIT NUMBER	BIT NAME	FUNCTION NAME	DIR	DESCRIPTION
0–1	RESERVED			RESERVED
2–15	numofQuadlets	Number of Quadlets	R/W	Total number of quadlets in the current packet (data payload and header quadlets only)
16–18	RESERVED			RESERVED
19–23	AckCode	Acknowledge Code	R/W	This 5-bit field holds the acknowledge code sent by the receiver for the current packet (see Note following the table):

Ack Code	Name	
00000	Reserved	
00001	Ack_complete	
00010	Ack_pending	
00011	Reserved	
00100	Ack_busy_X	
00101	Ack_busy_A	
00110	Ack_busy_B	
00111 – 01100	Reserved	
01101	Ack_data_error	
01110	Ack_type_error	
01111	Reserved	
10000	No ack received	These codes are added by the link layer and are not part of the IEEE 1394–1995 specification.
10001	Ack too long	
10010	Ack too short	
10011 – 11111	Reserved	

24–25	RESERVED			RESERVED
26–27	SPD	Speed Code	R/W	The spd field indicates the speed at which the current packet was sent. 00 => 100 Mbit/s, 10 = > 400 Mbits/s, 01 => 200 Mbit/s, 11 is undefined.
28 – 29	RESERVED			RESERVED

NOTE:

The acknowledge code specified by the IEEE 1394-1995 specification is a 4-bit field. The AckCode field in this register is a 5-bit field. The TSB12LV32 logic core is able to provide (specifiy) three additional ack codes, which are not part of the original specification. The ack codes are 10000, 10001, and 10010.

BIT NUMBER	BIT NAME	FUNCTION NAME	DIR	DESCRIPTION	
30	LPS_RESET	LPS Reset	R/W	Link power status reset. This bit is set by software and is reset by hardware. When this bit is set, hardware will deactivate LPS for a fixed period to ensure that the Phy has reset the interface. It will then reactivate LPS. When this bit is cleared by hardware, a PHRST interrupt will also be generated.	
31	LPS_OFF	LPS Off	R/W	Link power status off. If set to 1, this bit will turn off the LPS pulsed output to the Phy. This bit can also be turned off from the Phy. Upon detection of the LINKON pulsed input signal, this bit will be turned off allowing LPS to be driven to the Phy which will, in turn, activate SCLK and power up the link.	

2.2.18 Asynchronous Retry Register at 4Ch

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
ASYNC RETRY COUNT								RETRY INTERVAL																							

2.2.19 Asynchronous Retry Register at 4Ch

The power-up reset value of this register = **0000_0000'h**.

BIT NUMBER	BIT NAME	FUNCTION NAME	DIR	DESCRIPTION
0–7	ASYNC RETRY COUNT	Retry count	R/W	Asynchronous retry count, specifies the number of times to automatically retry sending asynchronous packets from the ATF before giving up. After the retry count is exhausted FIFOACK interrupt will be generated and the ATACK field in CFR 30h will be updated to reflect the timeout.
8–15	RETRY INTERVAL	Retry interval	R/W	Asynchronous retry interval, is the time in increments of isochronous cycles, between asynchronous retries.
16–31	RESERVED			RESERVED

3 Microcontroller Interface

The microcontroller interface allows the local microcontroller/microprocessor to communicate with the internal control and configuration registers (CFR), asynchronous transfer FIFO (ATF) and general receive FIFO (GRF). All microcontroller reads/writes are initiated by the microcontroller. The micro interface supports read transactions from the CFR or GRF, and write transactions to the CFR or ATF.

The micro interface can operate in byte (8 bits) or word (16 bits) accesses. Each CFR, with the exception of the cycle timer register at 14h and the Phy access register at 24h, can be addressed on byte or word boundaries. The possible configurations for the interface are shown in Table 3–1. The TSB12LV32 can also be directly connected to the Motorola 68000 and ColdFire™ line of MC/MP. Table 3–2 defines the mapping of the micro interface pins between the TSB12LV32, the Motorola 68000 and the ColdFire microprocessor.

Table 3–1. Microcontroller Interface Modes of Operation

TSB12LV32 Mode-CONFIGURATION TERMINALS			MODE OF OPERATION
COLDFIRE	M8BIT_SIZ0	MCMODE_SIZ1	
0	0	0	16-bit fixed timing mode.
0	0	1	16-bit MCS–MCA handshake mode.
0	1	0	8-bit fixed timing mode.
0	1	1	8-bit MCS–MCA handshake mode
1	0	0	ColdFire 4-byte (2-word) burst mode
1	0	1	ColdFire 2-byte (1-word) mode
1	1	0	ColdFire 1-byte mode (not supported)
1	1	1	ColdFire 16-byte (8-word) burst mode

Table 3–2. TSB12LV32 MP/MC Interface Terminal Function Matrix

TSB12LV32		Motorola 68000/ColdFire MICROCONTROLLER	
TERMINAL NAME	USAGE	TERMINAL NAME	USAGE
MA0 – MA6	Input	A[6:0]	Output
MD0 – MD15	I/O	D[31:16]	I/O
$\overline{\text{MCA}}$	Output	TAZ	Input
$\overline{\text{MCS}}$	Input	TSZ	Output
$\overline{\text{MWR}}$	Input	R/WZ	Output
MCMODE/SIZ1, M8BIT/SIZ0	Input	SIZ1, SIZ0	Output
$\overline{\text{TEA}}$	Output	TEAZ	Input
BCLK	Input	SCLK / CLK	Input

The Byte stacker allows the TSB12LV32 to be easily connected to most processors. The byte stacker consists of a programmable 8-/16-bit data bus and a 7-bit address bus. The TSB12LV32 uses cycle-start and cycle-acknowledge handshake signals to allow the local bus clock and the 1394 clock to be asynchronous to one another. The TSB12LV32 is an interrupt driver to reduce cycling. All bus signal labeling on the TSB12LV32 microcontroller interface use bit #0 to denote the most significant bit (MSB).

ColdFire is a trademark of Motorola, Inc.

3.1 Microcontroller Byte Stack (Write) Operation

The microcontroller byte stack (write) protocol is shown in Figure 3–1.

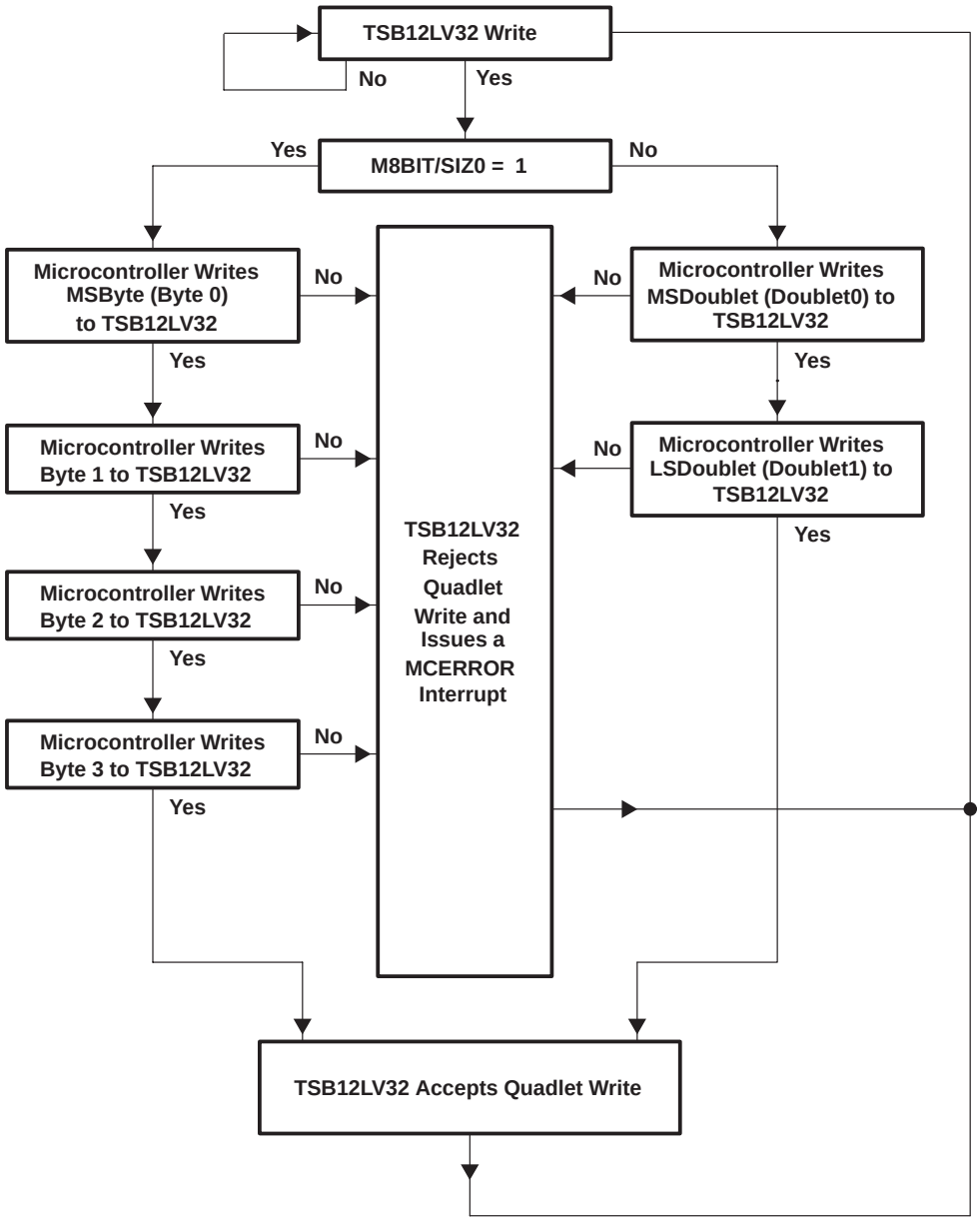


Figure 3–1. Microcontroller Byte Stack Operation (Write)

3.2 Microcontroller Byte Unstack (Read)

The microcontroller byte unstack (read) protocol is shown in Figure 3–2.

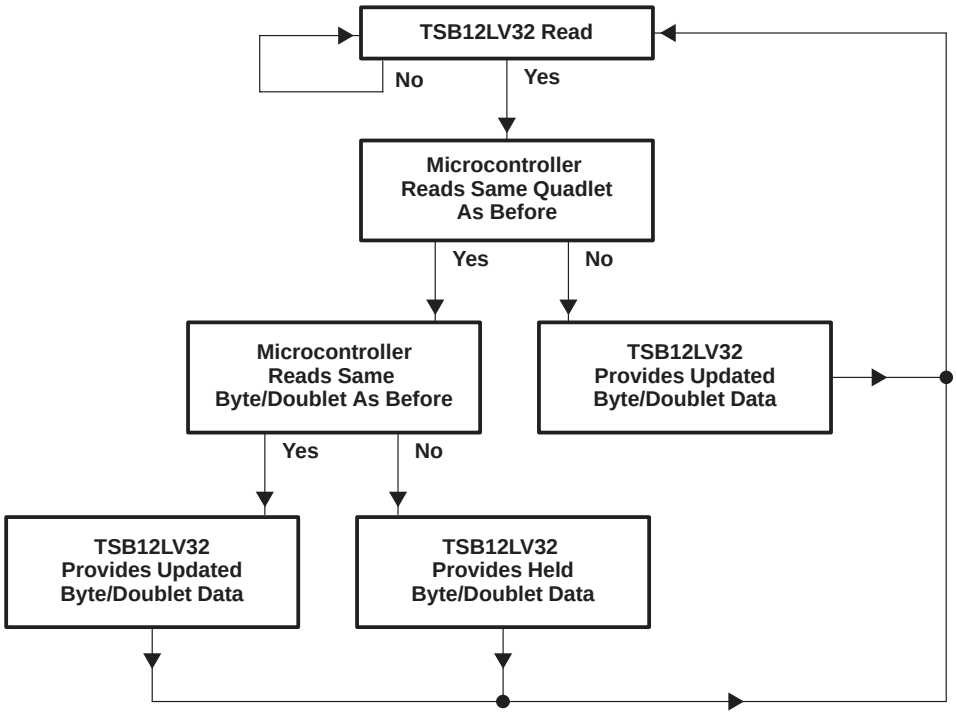


Figure 3–2. Microcontroller Byte Unstack Operation (Read)

3.3 Microcontroller Interface Read/Write Timing

The micro interface can be configured to operate in one of the following modes: handshake, fixed-timing, or ColdFire mode. Burst transfers are only supported in the latter two modes.

3.3.1 Microcontroller Handshake Mode

Byte handshake read and word handshake read are shown in Figure 3–3 and Figure 3–4, respectively.

The $\overline{\text{MCS}}$, $\overline{\text{MCA}}$ handshake timing sequence for a read transaction can be summarized as follows:

1. The host takes $\overline{\text{MCS}}$ low to signal the start of access. When the rising edge of BCLK samples $\overline{\text{MCS}}$ low and $\overline{\text{MWR}}$ high, the MD[0:15] lines are enabled and driven with the read value. For an 8-bit data bus, MD[0:7] lines are not used.
2. Following the next rising edge of BCLK, the TSB12LV32 takes $\overline{\text{MCA}}$ low to signal that the requested operation is complete. This is ensured to take place after two BCLK cycles. $\overline{\text{MCA}}$ remains low with the MD lines containing valid read data until the micro interface releases $\overline{\text{MCS}}$ (high state)
3. The host takes $\overline{\text{MCS}}$ high to signal the end of the process.
4. The TSB12LV32 takes $\overline{\text{MCA}}$ high to acknowledge the end of the access. This 3-states the MD lines.

Another read or write transaction can begin after the next rising edge of BCLK. Note that data size is determined by the MCMODE/SIZ1 and M8BIT/SIZ0 signals. The ColdFire signal is only asserted high when the micro interface is operating in ColdFire mode.

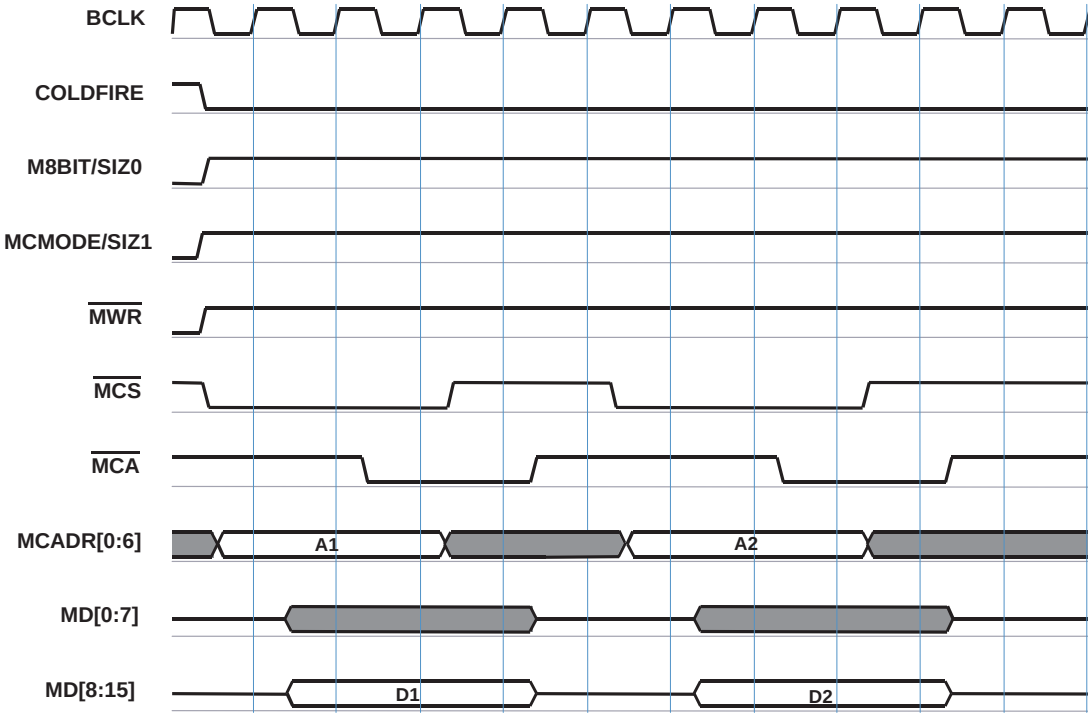


Figure 3–3. Byte Handshake Read

Figure 3–4 shows a word handshake read transaction. In this case, all 16 bits of the MD lines are used. Note that MD[0] contains the MSB and MD[15] contains the LSB. As in the byte read case, another read or write transaction can begin after the next rising edge of BCLK.

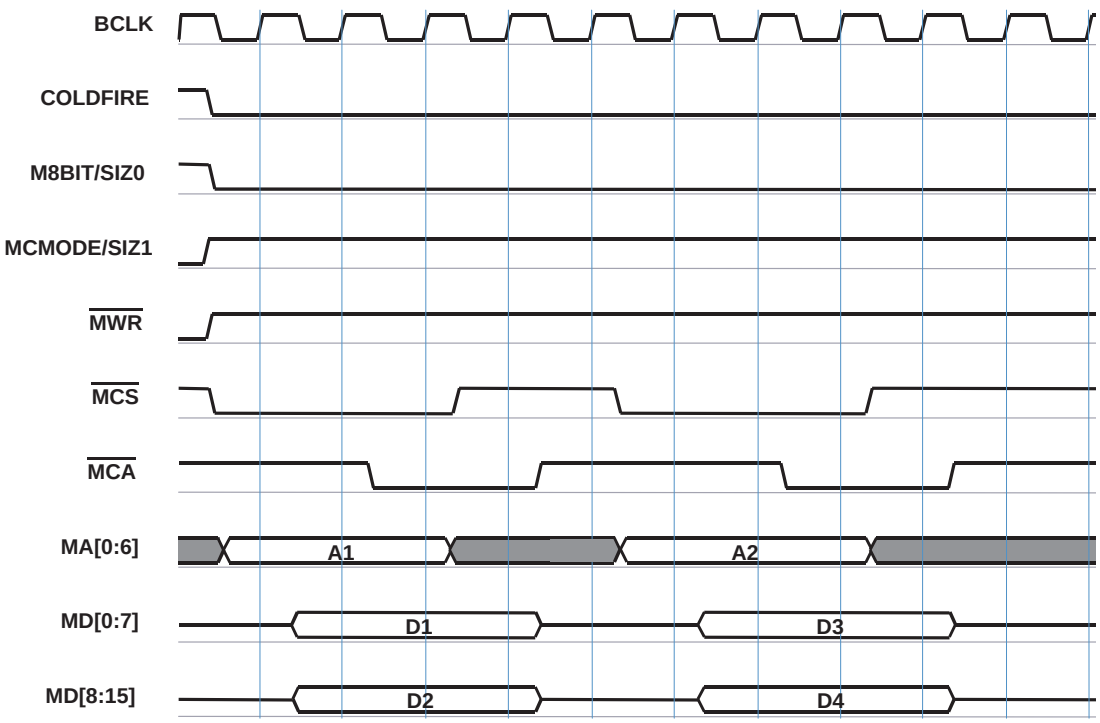


Figure 3–4. Word Handshake Read

Byte handshake write and word handshake write are shown in Figure 3–5 and Figure 3–6. In this case, the micro interface asserts MCA low immediately after MCS is sampled low. The micro interface keeps MCA low until it samples MCS high. For 8-bit accesses, the MD[0:7] lines are not used. If a transfer error condition occurs, TEA will be asserted low for one BCLK cycle. An error condition can occur if the MCMODE/SIZ1 or M8BIT/SIZ0 line changes state during the access cycle.

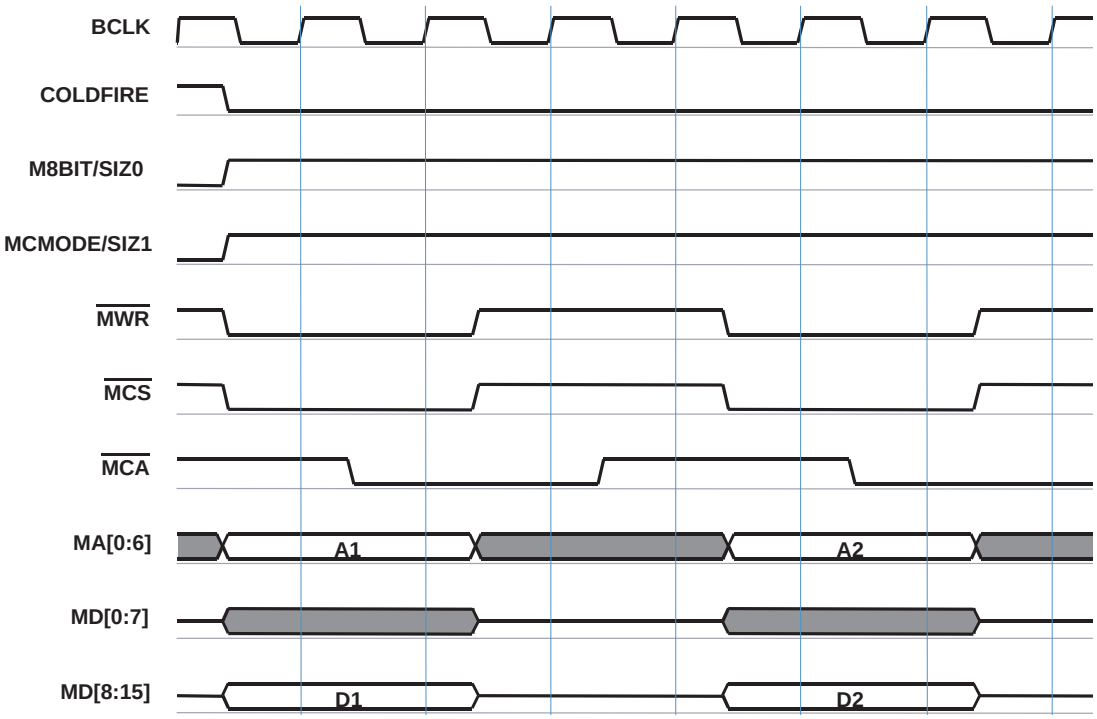


Figure 3–5. Byte Handshake Write

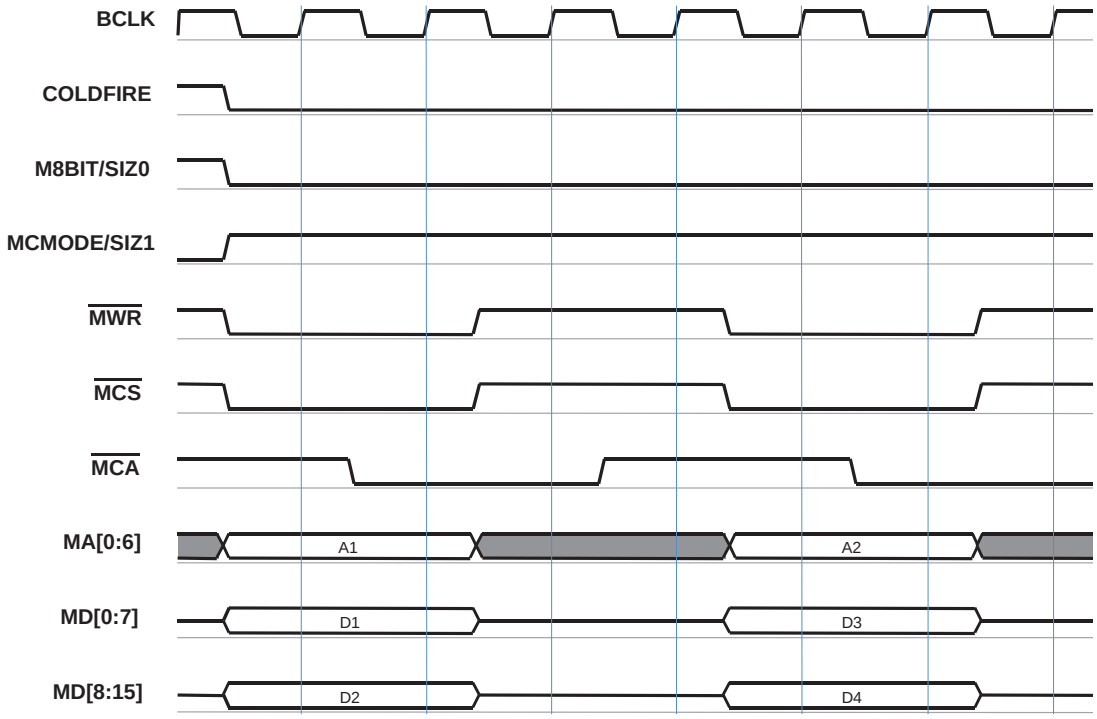


Figure 3–6. Word Handshake Write

3.3.2 Microcontroller Fixed-Timing Mode

Byte and word fixed-timing reads shown in Figure 3–7 and Figure 3–8. Fixed-timing mode supports burst transfers. If $\overline{MCS}$ is asserted low for more than one BCLK cycle, burst mode is enabled. The fixed-timing burst mode does not have a limit on the maximum burst size allowed.

The timing sequence in the fixed-timing a read transaction can be summarized as follows:

1. The host pulses $\overline{MCS}$ low to signal the start of access. Pulsing $\overline{MCS}$ low for more than once clock cycle will enable burst mode. The number of BCLK cycles during which $\overline{MCS}$ is asserted low determines the burst size.
2. When the rising edge of BCLK samples $\overline{MCS}$ low and $\overline{MWR}$ high, the register value or GRF data pointed to by MA is latched onto the MD lines. The MD lines will latch on every rising edge of BCLK if $\overline{MCS}$ is asserted low.
3. After 2 BCLK cycles, the TSB12LV32 pulses $\overline{MCA}$ low for one clock cycle to signal the completion of the requested operation. If $\overline{MCS}$ is pulsed low for *n* BCLK cycles, $\overline{MCA}$ will also be pulsed low for *n* cycles. Note that MA needs only contain valid data during the first cycle in which $\overline{MCS}$ is low. Except for the first one, every data transfer takes only one BCLK cycle. If a read transaction is accessing the CFR, it may not cross any register boundary.

Another read or write transaction can begin after the next rising edge of BCLK. Note that data size is determined by the MCMODE/SIZ1 and M8BIT/SIZ0 signals. The ColdFire signal is only asserted high when the micro interface is operating in ColdFire mode.

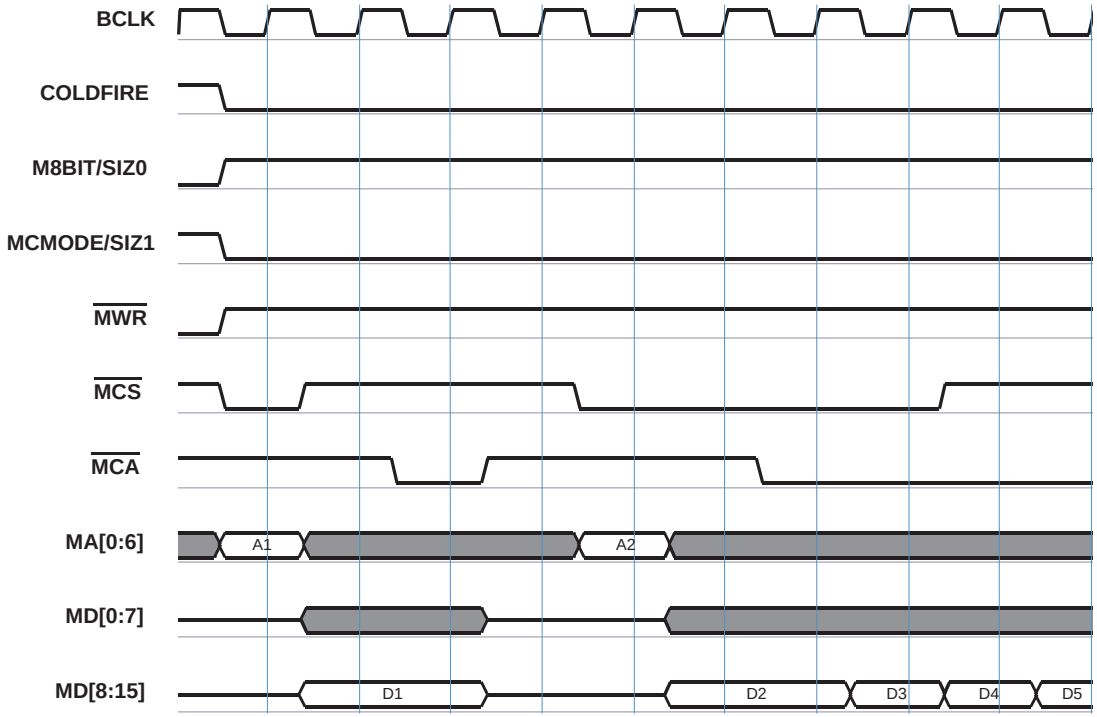


Figure 3–7. Byte Fixed-Timing Read

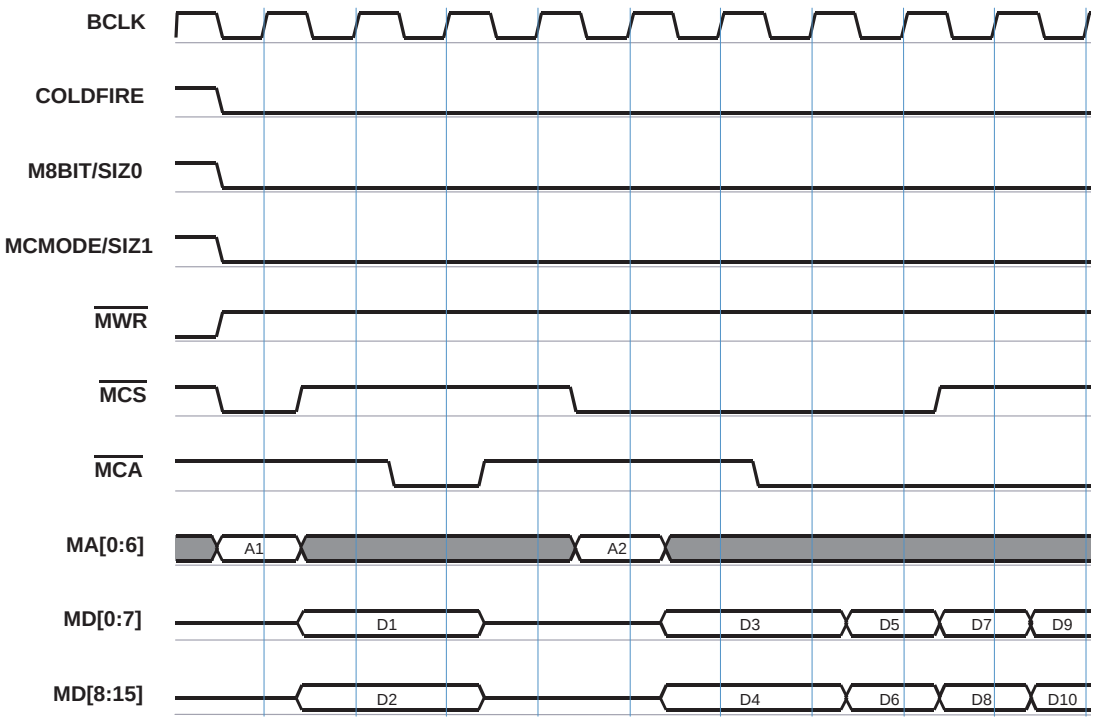


Figure 3–8. Word Fixed-Timing Read

Byte fixed timing write and word fixed timing write are shown in Figure 3–9 and Figure 3–10.

The first data transfer takes one extra wait cycle. All subsequent data transfers take only one BCLK cycle. For an 8-bit data bus, MD[0:7] is not used (don't care) and is driven with zeros. If the write transaction is accessing the CFR register, it may not cross any register boundary. First write data for each ATF quadlet must start at byte0. Write accesses to the ATF must be quadlet aligned. The micro interface will wait for all bytes of each quadlet to be available before creating a write request to the ATF. If a transfer error condition occurs, $\overline{\text{TEA}}$ will be asserted low for one BCLK cycle. An error condition can occur if the MCMODE/SIZ1 or M8BIT/SIZ0 lines transition during the access cycle.

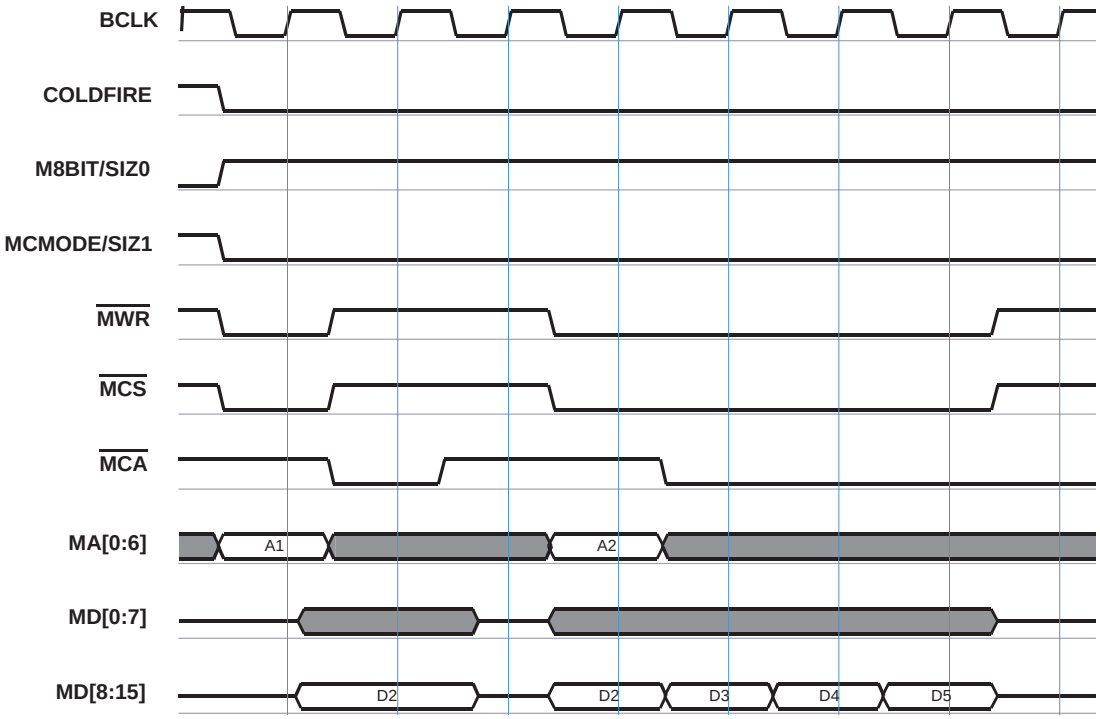


Figure 3–9. Byte Fixed-Timing Write

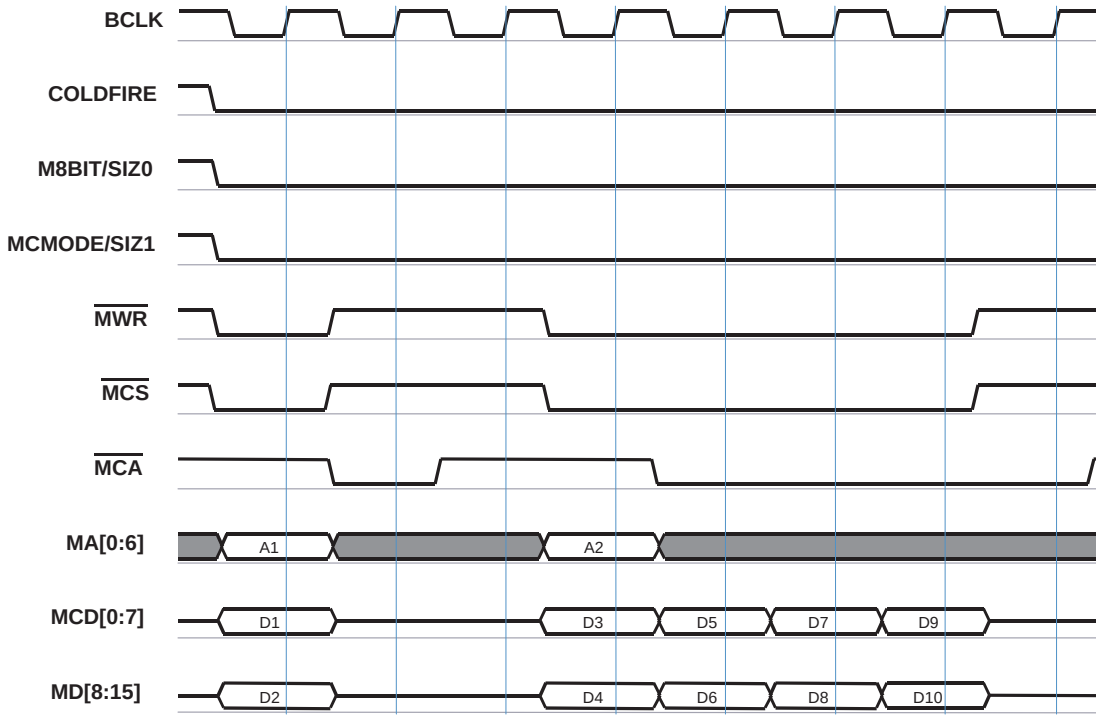


Figure 3–10. Word Fixed-Timing Write

3.3.2.1 GRF READ

The timing requirements when performing a GRF read access in fixed–timing mode are different from other access modes. In fixed–timing mode, the GRF must be accessed only on a quadlet boundary. In other words, only quadlet fetches are legal. If the microinterface is configured for a byte access, this means that MCSZ must be asserted low for 4 BCLK cycles, as shown in Figure 3–11. If configured for word access, then MCSZ must only be asserted for 2 BCLK cycles, as shown in Figure 3–12.

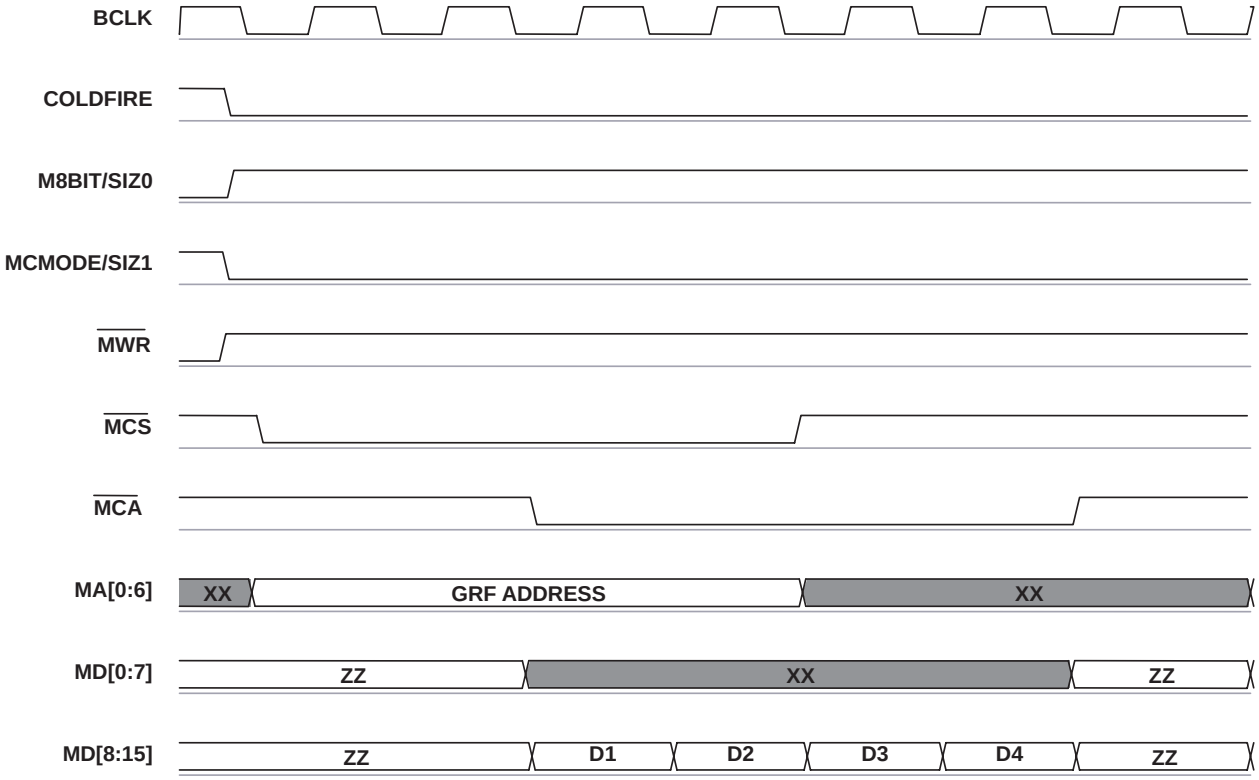


Figure 3–11. GRF Read Access (Byte Fixed-Timing Mode)

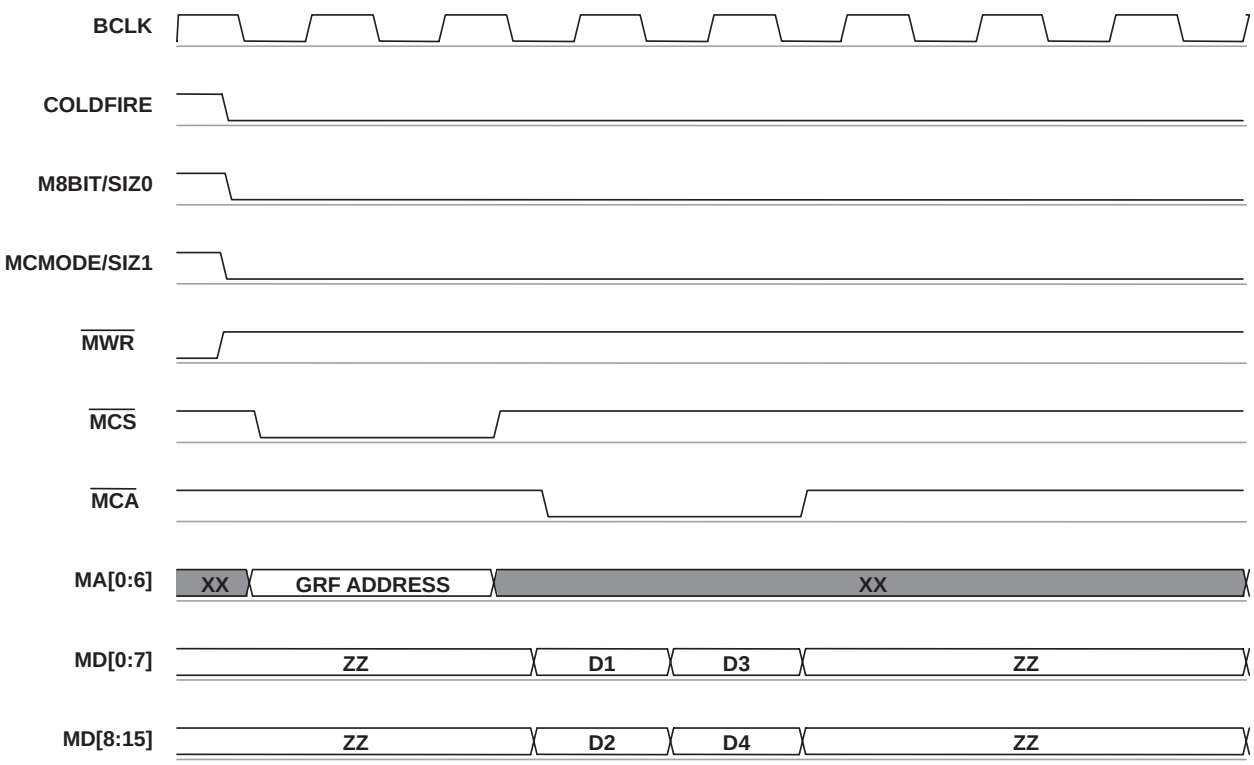


Figure 3–12. GRF Read Access (Word Fixed-Timing Mode)

3.3.3 Microcontroller ColdFire Mode

The TSB12LV32 supports a glueless interface to the ColdFire family of microcontrollers. To enable this mode, the ColdFire pin must be asserted and kept high for the entire access cycle. The timing diagram for a ColdFire read operation is shown in Figure 3–13.

The timing sequence for a ColdFire read access can be summarized as follows:

1. The ColdFire pulses $\overline{\text{MCS}}$ low for one BCLK cycle to signal the start of access.$\overline{\text{MCS}}$ must only be asserted for one clock cycle.
2. When the rising edge of BCLK samples $\overline{\text{MCS}}$ low and $\overline{\text{MWR}}$ high, MD lines are enabled, but do not yet contain valid data. The MA lines should contain the address information at this point. MA is only required to be available for one BCLK cycle. The data transfer size is determined by the state of the MCMODE/SIZ1 and M8BIT/SIZ0 lines.
3. The TSB12LV32 pulses $\overline{\text{MCA}}$ low for n clock cycles to signal the requested operation is complete. The number n depends on the data transfer size specified by the MCMODE/SIZ1 and M8BIT/SIZ0 lines. The CFR register value or GRF memory data pointed to by the MA lines is latched onto the MD lines. $\overline{\text{MCA}}$ will pulse for one clock cycle on every word (2-Bytes) transfer.

The microinterface does burst transfers if the MCMODE/SIZ1 and M8BIT/SIZ0 lines indicate more than 2-bytes (1 word) of data. The TSB12LV32 does not support 1-byte transfers in the ColdFire mode. If a transfer error condition occurs, $\overline{\text{TEA}}$ will be asserted low for one BCLK cycle. An error condition can occur if the MCMODE/SIZ1 and M8BIT/SIZ0 lines specify a transfer size of 1-byte or if their state changes during the access cycle. Note that all 16-bits of the MD lines are always used in this mode.

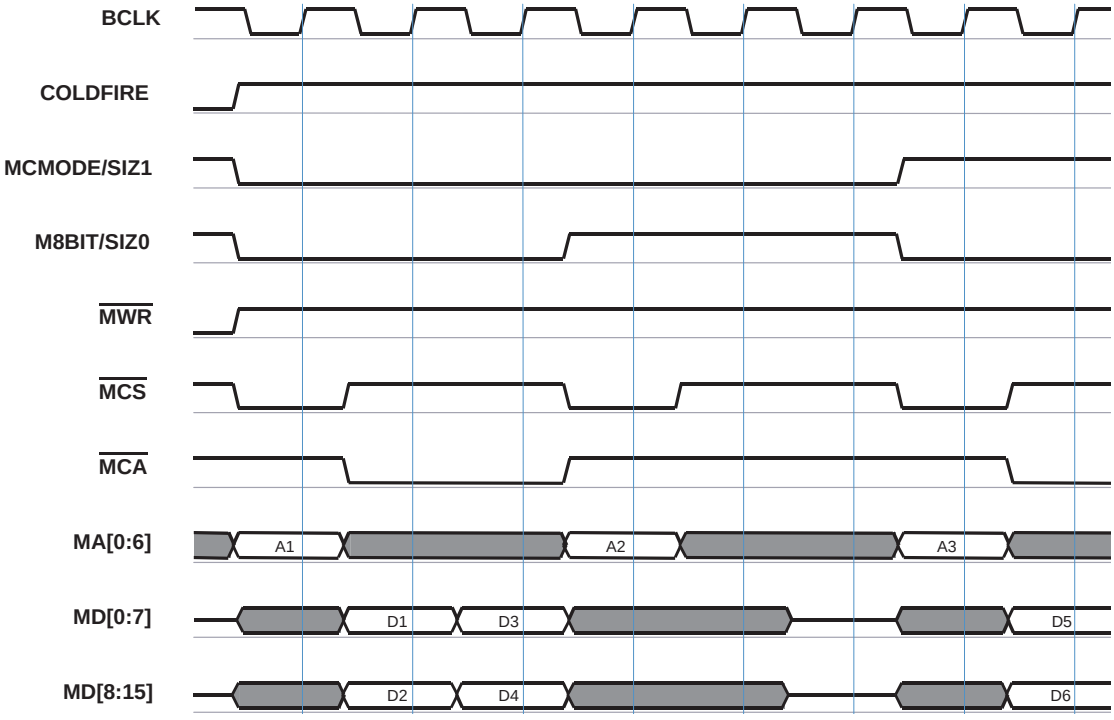


Figure 3–13. ColdFire Read

The ColdFire write transaction is shown in Figure 3–14. Unlike the handshake and fixed-timing write modes, the ColdFire write operation requires the data on the MD lines be available one BCLK cycle after the address on the MA lines is sampled. Violating this timing requirement may result in a transfer error, causing $\overline{\text{TEA}}$ to be asserted low for one BCLK cycle.

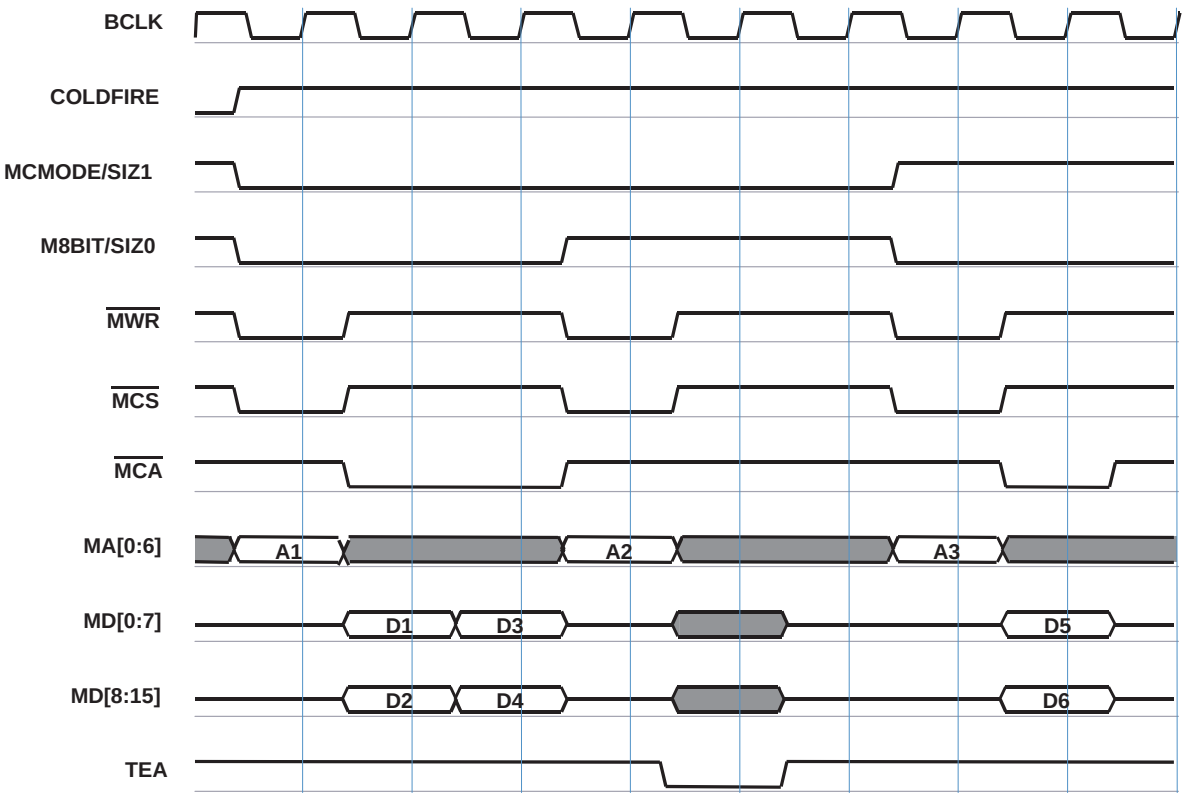
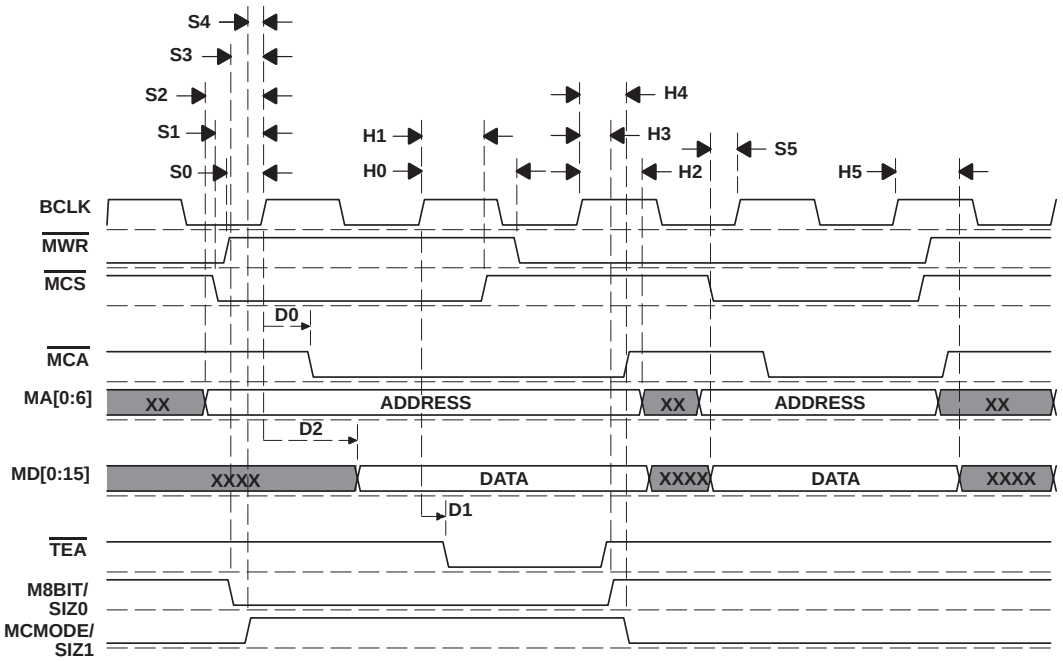


Figure 3–14. ColdFire Write

3.3.4 Microcontroller Critical TIming†

PARAMETER		TERMINAL NAME	ACCESS TYPE	MIN	MAX	UNIT
t_{d0}	Delay time (BCLK to Q)	$\overline{MCA}$	Read/Write	3.75	9.5	ns
t_{d1}		$\overline{TEA}$	Read/Write	3.75	9.5	
t_{d2}		MD[0:15]	Read	2.5	10.5	
t_{su0}	Setup time to BCLK	$\overline{MWR}$	Read/Write	4.5		ns
t_{su1}		$\overline{MCS}$	Read/Write	6.5		
t_{su2}		MA[0:6]	Read/Write	6.5		
t_{su3}		M8BIT/SIZ0	Read/Write	5		
t_{su4}		MCMODE/SIZ1	Read/Write	3.5		
t_{su5}		MD[0:15]	Write	3		
t_{h0}	Hold time from BCLK	$\overline{MWR}$	Read/Write	1.75		ns
t_{h1}		$\overline{MCS}$	Read/Write	1.5		
t_{h2}		MA[0:6]	Read/Write	2		
t_{h3}		M8BIT/SIZ0	Read/Write	1.5		
t_{h4}		MCMODE/SIZ1	Read/Write	1.75		
t_{h5}		MD[0:15]	Write	1.5		

† All parameters are referenced to the rising edge of BCLK.



3.3.5 Endian Swapping

The term *endianness* refers to the way data is referenced and stored in a processor's memory. For example, consider a 32-bit processor; any 32-word consists of four bytes which may be stored in memory in one of two ways. Of the four bytes, either byte 3 will be considered the most significant byte and byte 0 the least significant byte, or vice versa (see Figures 3–15 and 3–16). A little endian type memory considers byte 0 the least significant byte, whereas a big endian type memory considers byte 3 to be the least significant byte.

Byte #0 (Most Significant Byte)	Byte #1	Byte #2	Byte #3 (Least Significant Byte)

Figure 3–15. Big Endian Format

Byte #3 (Most Significant Byte)	Byte #2	Byte #1	Byte #0 (Least Significant Byte)

Figure 3–16. Little Endian Format

The TSB12LV32 configuration register space (CFR) and FIFO memory, both of which are 32-bits wide, use a big endian architecture. The TSB12LV32 uses the same endianness as the internal P1394 link core. This means that the most significant byte is the left-most byte (byte 0) and the least significant byte is the right most byte (byte 3).

3.3.5.1 Data and Address Invariance for Little Endian Processors

For little-endian processors, there are two modes of byte swapping, address invariant and data invariant. Address invariance preserves byte ordering between the internal system (GP2Lynx registers and FIFO) and external system (microcontroller/processor). Data invariance preserves the bit significance of the data, but changes the byte significance between the internal and external systems. The MDINV pin controls how the write/read data is swapped at the data bus (i.e., determines how the received bytes from the microcontroller are mapped into the TSB12LV32 internal registers and memory space). Note that when the COLDFIRE pin is high, the MDINV pin has no affect and data is always interpreted in as big endian. Refer to Literature

Number *SLLA021.pdf ENDIANNESS AND THE TSB12LV41 (MPEG2LYNX) MICROPROCESSOR INTERFACE* for a detailed description of endianness.

The pin settings for all the swapping operation are shown in Table 3–3. Note that in performing the byte swapping operation in the little-endian mode, only the two least significant bits of the 32-bit address inside are involved. This is because there is a total of four bytes associated with the swapping operation.

Table 3–3. Endian Swapping Operation

LENDIAN	M8BIT/SIZ0	MDINV	DESCRIPTION
0	X	X	Big endian mode, no manipulation on byte address and data bytes
1	1 (8-bits wide)	1	Little endian data invariance mode, swap the low order 2 bit address: **External low order 2-bit address** **Internal low order 2-bit address** Byte Address 00 ↔ Byte Address 11 Byte Address 01 ↔ Byte Address 10 Byte Address 00 ↔ Byte Address 11 Byte Address 11 ↔ Byte Address 00
1	01 (16-bits wide)	1	Little endian data invariance mode, swap the low order 2 bit address: **External low order 2-bit address** **Internal low order 2-bit address** Word Address 00 ↔ Word Address 10 Word Address 10 ↔ Word Address 00
1	1	0	16-bit little endian address invariance mode, swap data between MD[0:7] and MD[8:15].
1	1	0	8-bit little endian address invariance mode, no manipulation on byte address and data bytes.

Since the TSB12LV32's microprocessor interface is either 8 bits or 16 bits wide, but the internal configuration registers are 32 bits wide, a byte stacking (for writes) and a byte unstacking (for reads) operation must be performed on the data bus. For little endian processors, the TSB12LV32 can perform the swapping of bytes on the data bus required to allow both the processor and the TSB12LV42 to interpret the data the same. There are two methods of swapping the data bytes, address invariant and data invariant. Both of these methods are described below.

NOTE:

For the host processor to work correctly with the TSB12LV32, users *must* correctly connect the address and data busses of their microprocessor to the TSB12LV32's microprocessor port. Users must connect the MSB (most significant bit) of their address/data bus to the address/data MSB of the TSB12LV32. This must be done regardless of bit number labeling or which type of endianness their microprocessor uses.

3.3.5.2 Data Invariant System Design

Figure 3–17 shows a little endian data invariant system design example. In this system, the actual value of the data as it was stored in the processor's memory is preserved. Data invariant designs do not preserve the addresses when mapping between endian domains. If the data represents an integer, it is interpreted the same by both systems. If the data represents a string, an array, or some other type of byte indexed structure, it is interpreted differently by both systems.

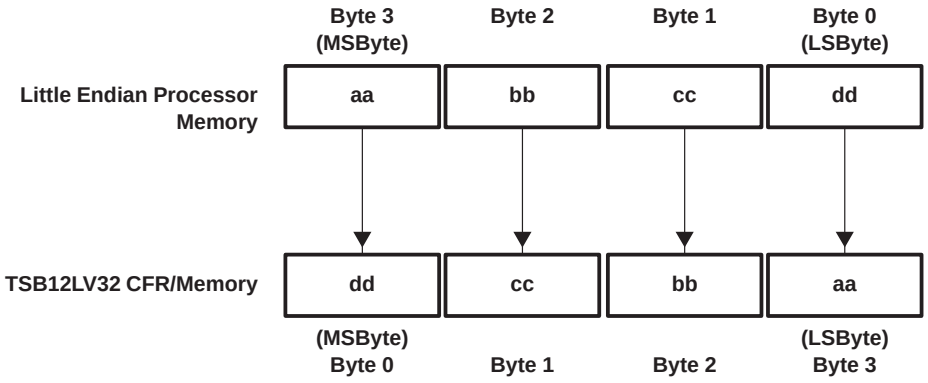


Figure 3–17. Little-Endian Data Invariance Illustration Chart

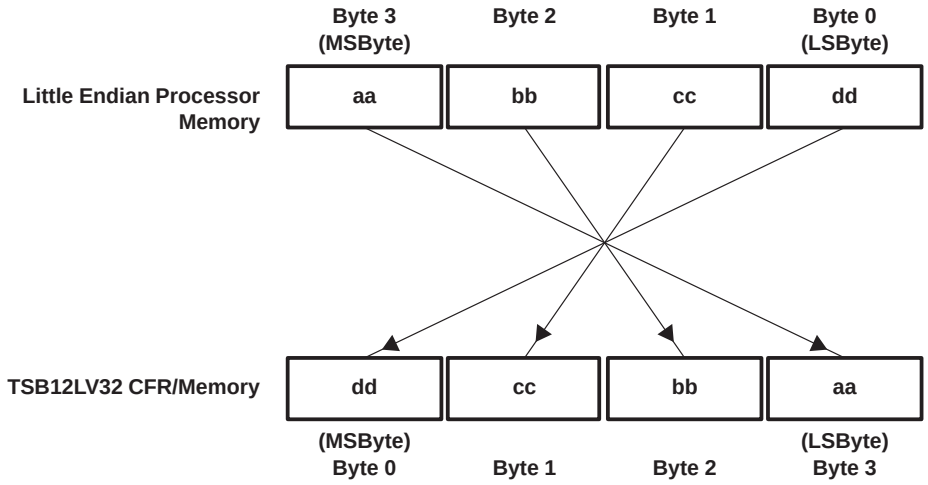


Figure 3–18. Little-Endian Address Invariance Illustration Chart

3.3.5.3 Address Invariant System Design

Figure 3–18 shows a little-endian address invariant system design example. In this case, the byte ordering between both systems is preserved (i.e., byte address is preserved). For example, byte 3 in the little endian processor memory is also byte 3 in the TSB12LV32 CFR space. As Figure 3–18 shows, the byte ordering is automatically maintained by the TSB12LV32 when in the address-invariance mode by swapping the order in which the incoming bytes on the microprocessor are written to the CFRs.

4 Link Core

This section describes the link core components and operations. Figure 4–1 shows the link core components.

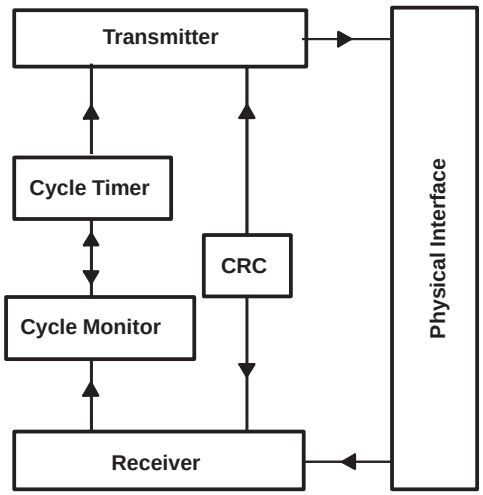


Figure 4–1. Link Core Components

4.1 Physical Interface

The physical (Phy) interface provides Phy-level services to the transmitter and receiver. This includes gaining access to the serial bus, sending packets, receiving packets, and sending and receiving acknowledge packets. The Phy interface module also interfaces to the Phy chip and implements Texas Instruments patent-pending bus-holder galvanic isolation.

4.2 Transmitter

The transmitter retrieves data from either the asynchronous transmit FIFO (ATF) or the data mover (DM) port and creates correctly formatted serial-bus packets to be transmitted through the Phy interface. When data is present at the ATF interface to the transmitter, the TSB12LV32 Phy interface arbitrates for the serial bus and sends a packet. When data is present at the DM Port, the TSB12LV32 arbitrates for the serial bus during the next isochronous cycle. The transmitter autonomously sends the cycle-start packets when the chip is a cycle master.

4.3 Receiver

The receiver takes incoming data from the Phy interface and determines if the incoming data is addressed to this node. When the incoming packet is addressed to this node, the CRC of the packet is checked. If the header CRC is good, the header is confirmed in the general-receive FIFO (GRF). For block and isochronous packets, the remainder of the packet is confirmed one quadlet at a time. The receiver places a status quadlet in the GRF after the last quadlet of the packet is confirmed in the GRF. The status quadlet contains the error code for the packet. In the case of asynchronous packets, the error code is the acknowledge code that is sent (returned) for that packet. For isochronous and broadcast packets that do not need acknowledge packets, the error code is the acknowledge code that would have been sent. This acknowledge code tells the transaction layer whether or not the data CRC is good or bad. If the header CRC is bad, the header is flushed and the rest of the packet is ignored. When a cycle-start packet is received, it is detected and the cycle-start packet data is sent to the cycle timer. Cycle-start packets are not placed in the GRF like other quadlet packets.

4.4 Cycle Timer

The cycle timer is only used by nodes that support isochronous data transfer. The cycle timer is a 32-bit cycle-timer register. Each node with isochronous data-transfer capability has a cycle-timer register as defined by the IEEE 1394–1995 specification. In the TSB12LV32, the cycle-timer register is implemented in the cycle timer located in the IEEE-1212 initial register space at location 200h and can also be accessed through the local bus at TSB12LV32 CFR address 14h. The low-order 12 bits of the timer are a modulo 3072 counter, which increments once every 24.576-MHz clock periods (or 40.69 ns). The next 13 higher-order bits are a count of 8,000-Hz (or 125-µs) cycles, and the highest 7 bits count seconds. The cycle timer contains the cycle-timer register. The cycle-timer register consists of three fields: cycle offset, cycle count, and seconds count. The cycle timer has two possible sources. First, when the cycle source (CYSRC) bit in the configuration register (bit 21 at 08h) is set, then the CYCLEIN input causes the cycle count field to increment for each positive transition of the CYCLEIN input (8 kHz) and the cycle offset resets to all zeros. CYCLEIN should only be the source when the node is the cycle master. The timer can also be disabled using the cycle-timer-enable bit (CYTEN) in the control register. The second cycle-source option is when the CYSRC bit is cleared. In this state, the cycle-offset field of the cycle-timer register is incremented by the internal 24.576-MHz clock. The cycle timer is updated by the reception of the cycle-start packet for the non-cycle master nodes. The cycle-offset field in the cycle-start packet is used by the cycle-master node to keep all nodes in phase and running with a nominal isochronous cycle of 125 µs. The cycle-start bit is set when the cycle-start packet is sent from the cycle master node or received by a noncycle master node.

4.5 Cycle Monitor

The cycle monitor is only used by nodes that support isochronous data transfer. The cycle monitor observes chip activity and handles scheduling of isochronous activity. When a cycle-start message is received or sent, the cycle monitor sets the cycle-started interrupt bit. It also detects missing cycle-start packets and sets the cycle-lost interrupt bit when this occurs. When the isochronous cycle is complete, the cycle monitor sets the cycle-done-interrupt bit. The cycle monitor instructs the transmitter to send a cycle-start message when the cyclemaster bit (CYMAS) is set in the control register.

4.6 Cyclic Redundancy Check (CRC)

The CRC module generates a 32-bit CRC for error detection. This is done for both the header and the data. The CRC module generates the header and data CRC for transmitting packets and checks the header and data CRC for received packets (see the IEEE-1394–1995 standard for details on the generation of the CRC).

4.7 Packet Routing Control Logic

Asynchronous and Isochronous receive packets will be routed to the DM port or the GRF via the receiver routing control logic. Any asynchronous packet addressed in the range of 0 to 0000_FFFF_FFFF and is of the *write request* type will be routed to the DM port according to Table 4–1. When bit IRCVALL (at 18h) is set, all isochronous data will be routed according to Table 4–1. Note that self-ID packets and Phy packets are always received by the GRF regardless of the routing control settings.

Table 4–1. Receiver Routing

AR0	AR1	DMASYNC	DMRX	DATA MOVER (DM)	GENERAL-RECEIVE FIFO (GRF)
0	0	X	0	Receives no data; Power-on default	Receives all data (asynchronous and isochronous); power-on default
		1	1	Receives Read Response Packets with a tlabel of 11XXXX	Receives all other asynchronous packets and isochronous packets
0	1	1	1	Receives asynchronous data only	Receives isochronous/asynchronous streaming data only
		X	0	Receives no data	Receives all data
		0	1	Receives isochronous/ asynchronous streaming data only	Receives asynchronous data
1	0	X	0	Receives no data	Receives all data
		0	1	Receive no data	Receives all data
		1	1	Receives addressed write request asynchronous packets in the address range of 0000_0000_0000 - 0000_FFFF_FFFFh	Receives addressed asynchronous packets in the address range of 0001_0000_0000-FFFF_FFFF_FFFFh. Also receives any asynchronous packets not going to the DM port regardless of the address. Receives isochronous packets too.
1	1	X	0	Receives no data	Receives all data
		X	1	Receive all data (asynchronous and isochronous)	Receives no data

5 Data Mover Port Interface

The data mover (DM) port in the TSB12LV32 is the physical medium by which autonomous streams of different types are piped to/from an application that uses the TSB12LV32. The DM port is meant to handle an external memory interface of large data packets.

The DM port can support three types of packets:

- Asynchronous
- Isochronous
- Asynchronous Streaming (1394a supported format)

The port can be configured to either transmit or receive data packets at one time (half duplex). A typical system diagram is shown in Figure 5–1:

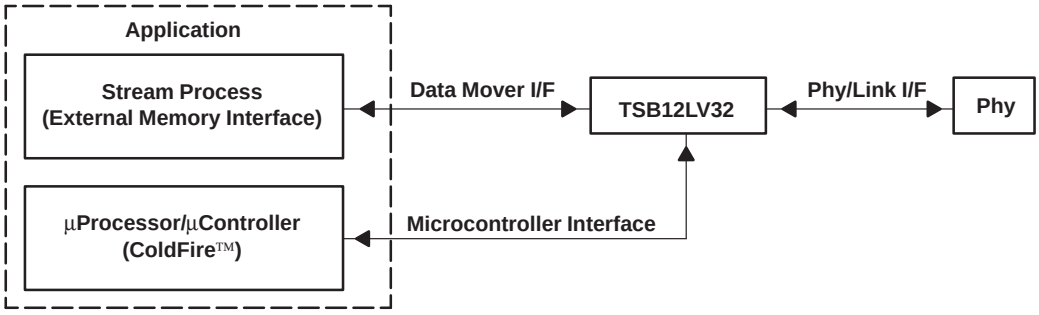
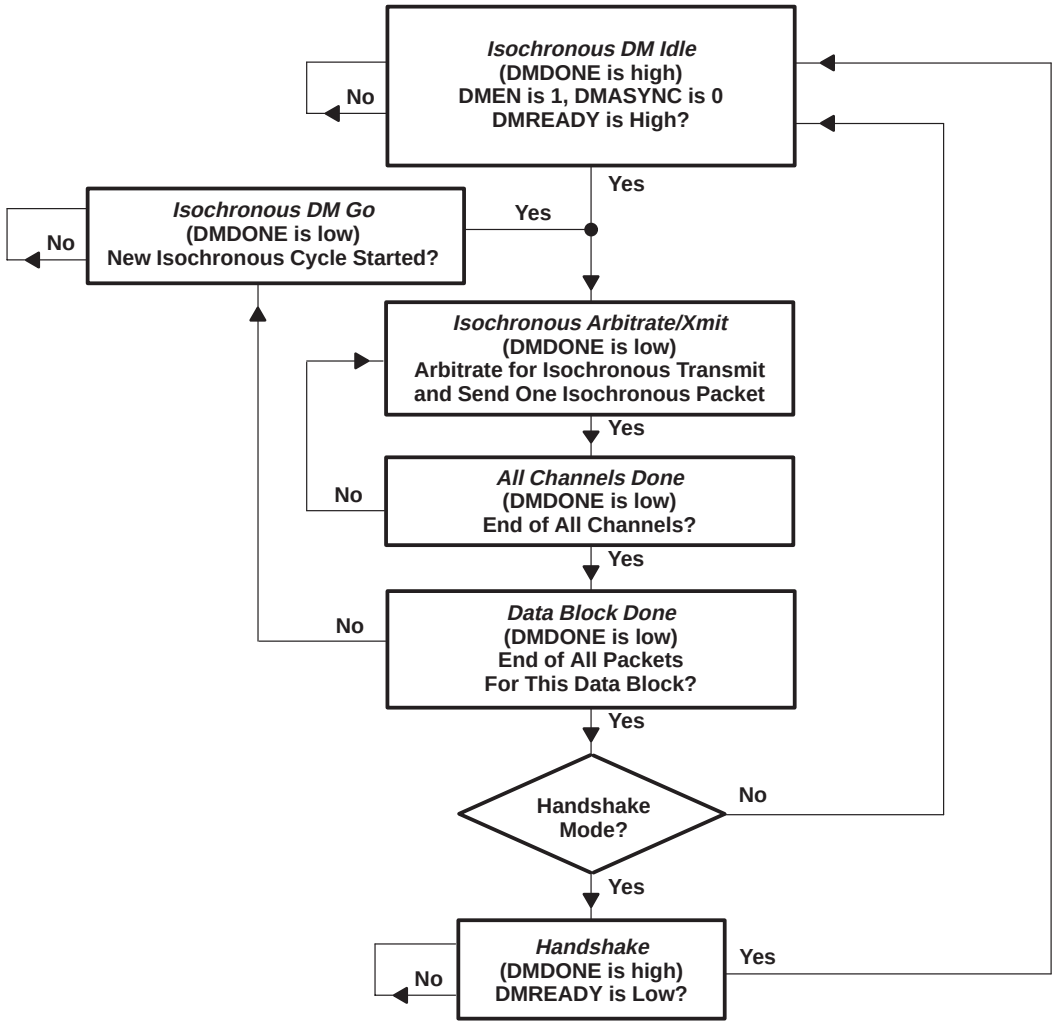


Figure 5–1. A Typical System Diagram

The DM port will perform all operations synchronously, utilizing a 24.576 MHz output clock called DMCLK. DMCLK is essentially SCLK/2. There is no asynchronous logic within the DM block. All data transfers are synchronized to the DMCLK output. The DM operates by setting the DM control register at 04h and the control register at 08h of the CFR. The DM interfaces internally with the configuration register (CFR) block and the link core (Link) block and interfaces externally with the data mover external interface.

The advantages of the DM port can be summarized as follows:

- Transmits or receives large blocks of data at speeds up to 400 Mbits/s.
- Allows for a large external FIFO specific to an individual application.
- Handles asynchronous, isochronous, or asynchronous streaming packets.



NOTE A: DMEN and DMASYNC are configuration register bits, DMREADY is an input terminal, and DMDONE is an output terminal.

Figure 5–2. Isochronous DM Flow Control (TSB12LV32 Transmit)

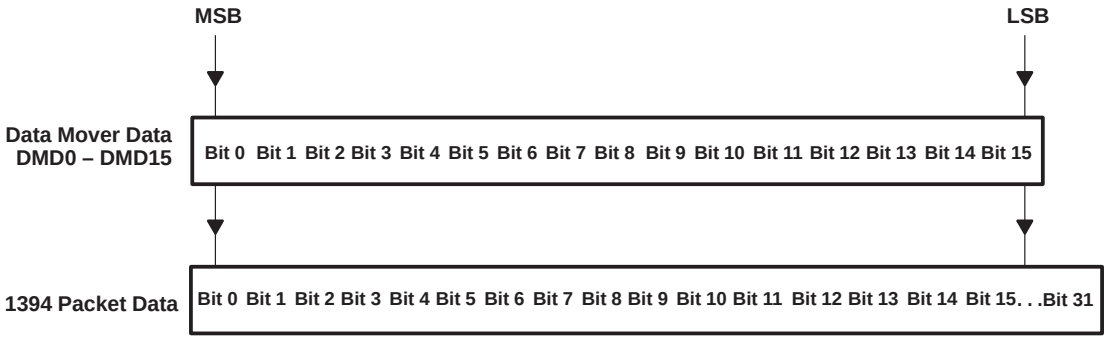
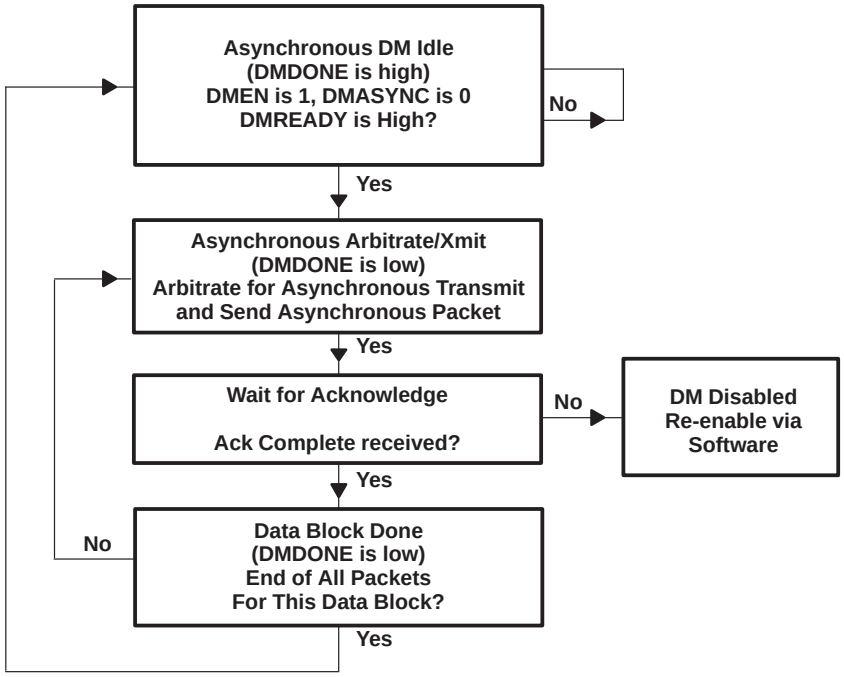


Figure 5–3. Transmit Data Path

The DM isochronous transmit reads data from the DM interface (DMD[0–15] lines) and passes it to the 1394 isochronous transmit interface in accordance with Figure 5–2. The data path is shown in Figure 5–3. The asynchronous header registers will contain the latest extracted header from the asynchronous stream when the header is supplied via the DM port. For automatic header insertion, the *datalength* field in the header register will be automatically updated by the payload size of the previous asynchronous transmit packet. This option can be turned off by setting the appropriate bits in the DM control registers. The DM asynchronous transmit reads data from the DM interface (DMD[0:15] lines) and passes it to the 1394 asynchronous interface in accordance with Figure 5–4



NOTE: DMEN and DMASYNC are configuration register bits, DMREADY is an input terminal, and DMDONE is an output terminal.

Figure 5–4. Asychronous DM Flow Control (TSB12LV32 Transmit)

The DM Interfaces to the configuration register (CFR), the link core (Link), and the external data mover interface (DMI).

5.1 Data Mover Data Flow Diagram

The data mover has eight modes of operation. There are four modes for transmit and four modes for receive.

Definitions

- Data mover port configured to operate in transmit mode means that the packet data is received through the data mover port and forwarded (unbuffered) to the link core transmit logic to be sent to the physical layer device (Phy), which will, in turn, *transmit* the data onto the 1394 bus.
- Data mover port configured to operate in receive mode means that the packet data is *received* by the link core receive logic from the 1394 bus through the Phy. The data is then routed by the link core to the data mover port without any internal buffering.

5.1.1 Isochronous Receive

In all the isochronous receive modes, the packet header information is always loaded into the header registers. The packet header quadlet is loaded into the Header0 register at 38h and the packet trailer quadlet is loaded into the trailer register at 48h.

5.1.1.1 Isochronous Packet Receive Without Header and Trailer

Step 1: Isochronous packet is received through the receiver logic of the link core

Step 2: The packet header is stripped off from the packet and loaded into the header0 register at 38h.

Step 3: Packet data (payload only) is routed directly to the DM port without any buffering.

Step 4: Trailer quadlet is loaded into the trailer register at 48h

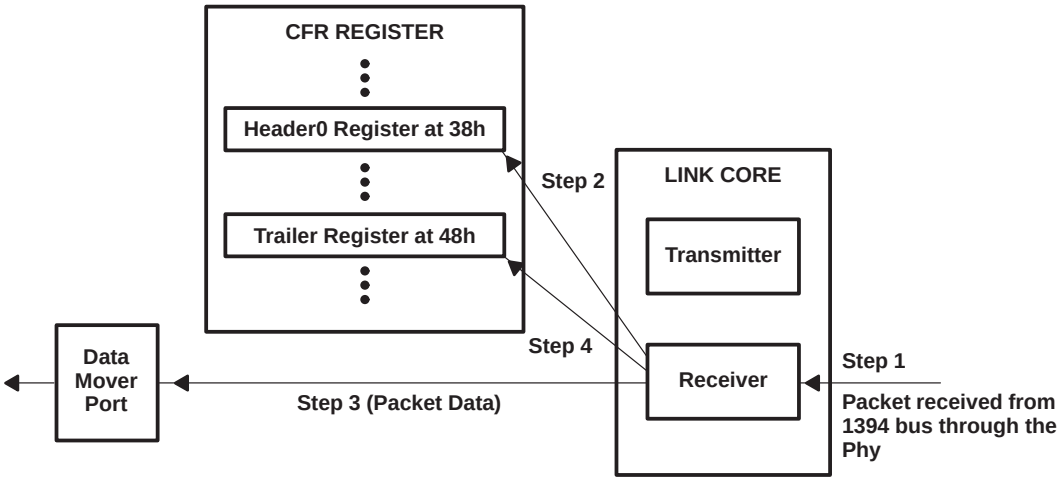


Figure 5–5. Isochronous Receive Without Header and Trailer

5.1.1.2 Isochronous Packet Receive With Header and Trailer

Step 1: Isochronous packet is received through the receiver block of the link core.

Step 2: The header quadlet is both loaded into the header0 register at 38h and routed to the DM port without any buffering.

Step 3: Packet data (payload only) is sent directly through the DM port only.

Step 4: Trailer quadlet is loaded into the trailer register at 48h. It is also forwarded to the DM port.

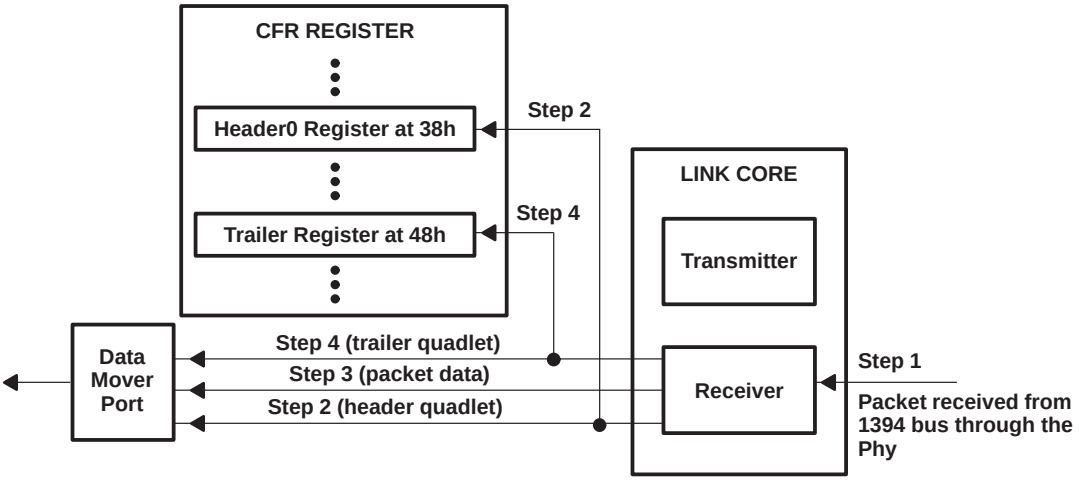


Figure 5–6. Isochronous Receive With Header and Trailer

5.1.2 Isochronous Transmit

There are two ways (modes of operation) to transmit isochronous data through the data mover:

- Isochronous packet transmit with automatic header insertion.
- Isochronous packet transmit without automatic header insertion.

The difference between the two modes lies in the mechanism in which the header information is inserted into the data stream. However, in both cases the header information is always loaded into the link core transmitter from the header register.

5.1.2.1 Isochronous Packet Transmit With Automatic Header Insertion

In this mode, the header information is first loaded into the header0 register through the microcontroller interface. The header will subsequently be automatically inserted into the data once the data mover starts streaming it through to the link core transmitter logic. The following steps further illustrate the process:

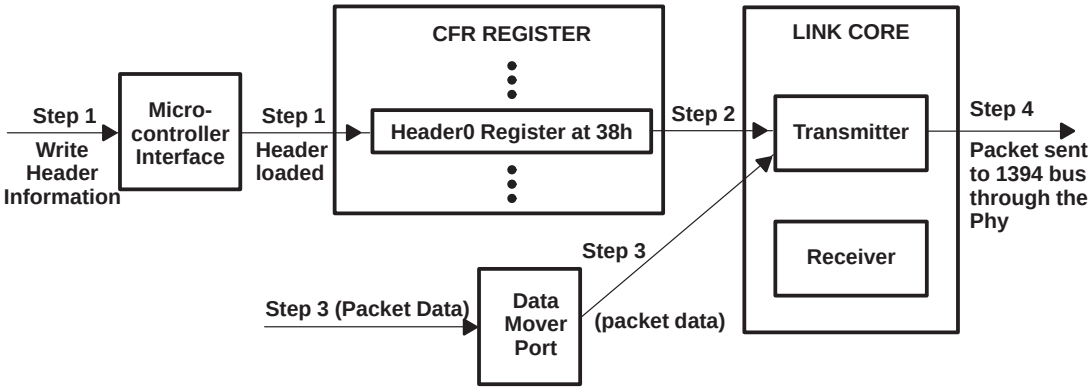


Figure 5–7. Isochronous Transmit With Auto Header Insertion

Step 1: Isochronous header quadlet is loaded into header0 register at 38h through a write operation from the microcontroller interface.

Step 2: Header quadlet is forwarded to the transmitter of the link core.

Step 3: Packet data (payload only) is transmitted through the data mover directly to the transmitter of the link core.

Step 4: Isochronous packet is sent to the 1394 bus through the Phy.

NOTE:

The data coming through the data mover port is typically supplied by an external fast memory block (i.e., FIFO, DRAM). This external memory logic may begin transmitting data through to the data mover port exactly one DMCLK cycle after the DMPRE output pin on the GP2Lynx is asserted high.

5.1.2.2 Isochronous Packet Transmit Without Automatic Header Insertion

In this mode, the packet header and data information is loaded through the data mover port. This mode is sometimes called isochronous packet transmit with manual header insertion. This is because the header quadlet is not preloaded into the header0 register via the microcontroller interface. Instead, it is inserted *manually* into the data stream at the same time as the rest of the packet. The following steps further illustrate the process:

Step 1: Isochronous header information (only one header quadlet in this case) is fetched into the header0 register at 38h through the data mover port.

Step 2: Header quadlet is forwarded to the transmitter of the link core.

Step 3: Packet data (payload only) is transmitted through the data mover directly to the transmitter of the link core.

Step 4: Isochronous packet is sent to the 1394 bus through the Phy.

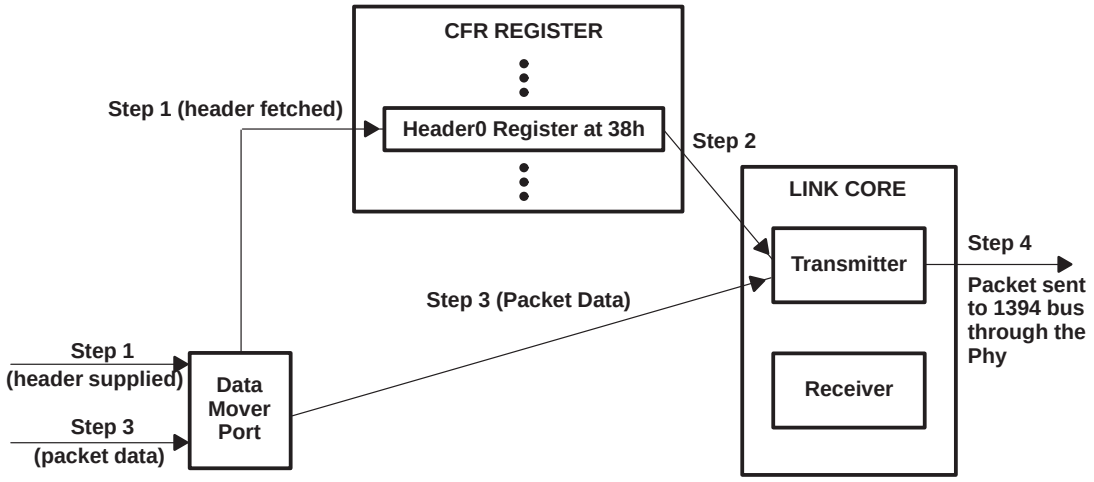


Figure 5–8. Isochronous Transmit Without Auto Header Insertion

5.1.3 Asynchronous Receive

In all the asynchronous receive modes, the packet header information is always loaded into the header registers. In quadlet receive mode, the first three header quadlets are loaded into the header0at 38h, header1at 3Ch, and header2at 40h registers, respectively. The trailer quadlet is loaded into the trailer register at 48h. In block receive mode, the only additional step performed is loading the fourth header quadlet received into the header3 register at 44h.

5.1.3.1 Asynchronous Packet Receive Without Headers and Trailer

Step 1: Asynchronous packet is received through the receiver logic of the link core

Step 2: The packet headers are stripped off from the packet and loaded into the header registers:

a) If in quadlet receive mode, the three header quadlets are loaded into the header0–header2 registers.

b) If in block receive mode, the four header quadlets are loaded into the header0–header3 registers.

Step 3: Packet data (payload only) is routed directly to the DM port without any buffering.

Step 4: Trailer quadlet is loaded into the trailer register at 48h

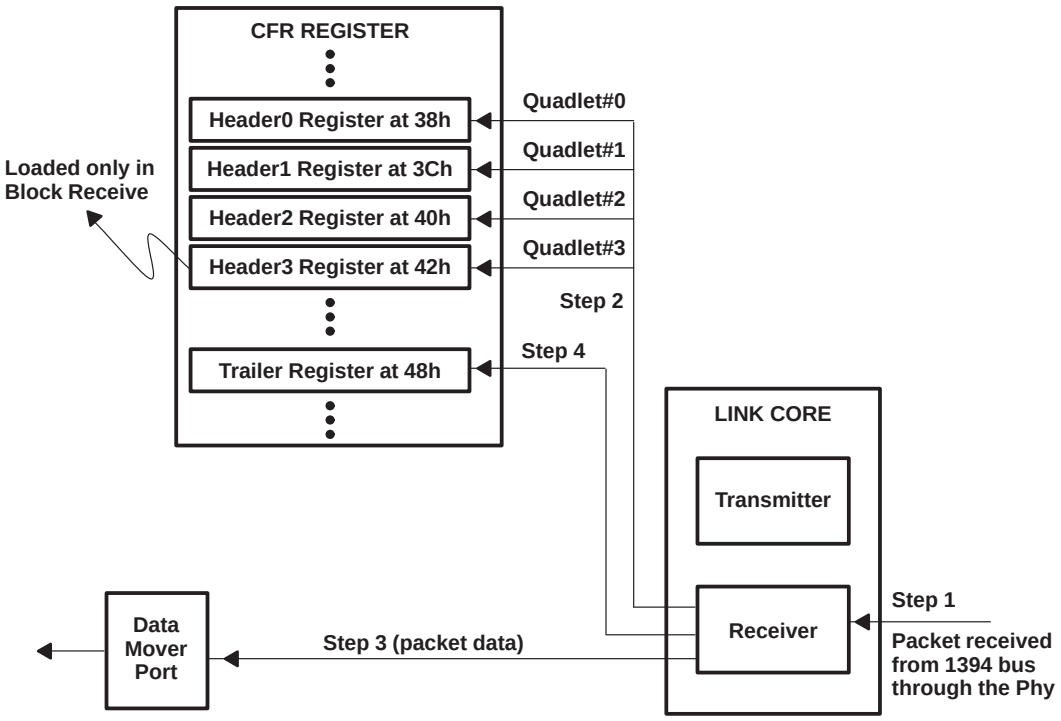


Figure 5–9. Asynchronous Receive Without Headers and Trailer

5.1.3.2 Asynchronous Packet Receive With Headers and Trailer

Step 1: Asynchronous packet is received through the receiver logic of the link core.

Step 2: The header quadlets are loaded into their respective header registers AND routed to the DM port without any buffering.

Step 3: Packet data is routed directly to the DM port (no buffering performed).

Step 4: Trailer quadlet is loaded into the trailer register at 48h.

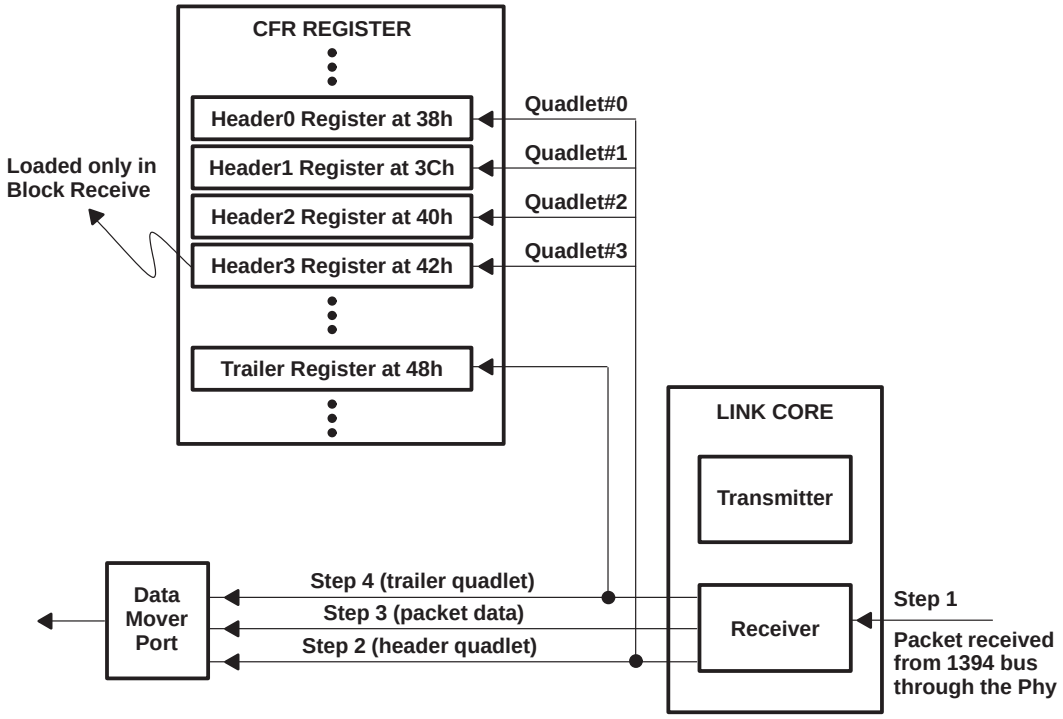


Figure 5–10. Asynchronous Receive With Headers and Trailer

5.1.4 Asynchronous Transmit

There are two ways (modes of operation) to transmit asynchronous data through the data mover:

- Asynchronous packet transmit with automatic header insertion.
- Asynchronous packet transmit without automatic header insertion.

The difference between the two modes lies in the mechanism in which the header information is inserted into the data stream. However, in both cases, the header information is always loaded into the link core transmitter from the header registers.

5.1.4.1 Asynchronous Packet Transmit With Automatic Header Insertion

In this mode, the header information is first loaded into the header0–header3 registers through the microcontroller interface. The headers will subsequently be automatically inserted into the data once the data mover starts streaming it through to the link core transmitter logic. The following steps further illustrate the process:

Step 1: Asynchronous header quadlets (3 quadlets in quadlet receive mode and 4 quadlets in block receive mode) are loaded into header0–header3 registers through a write operation from the microcontroller interface. Loading one header requires a single write operation.

Step 2: Header quadlets are forwarded to the transmitter of the link core.

Step 3: Packet data (payload only) is transmitted through the data mover directly to the transmitter of the link core.

Step 4: Asynchronous packet is sent to the 1394 bus through the Phy.

NOTE:

The data coming through the data mover port is typically supplied by an external fast memory block (i.e., FIFO, DRAM). This external memory logic may begin transmitting data through to the data mover port exactly one DMCLK cycle after the DMPRE output pin on the GP2Lynx is asserted high.

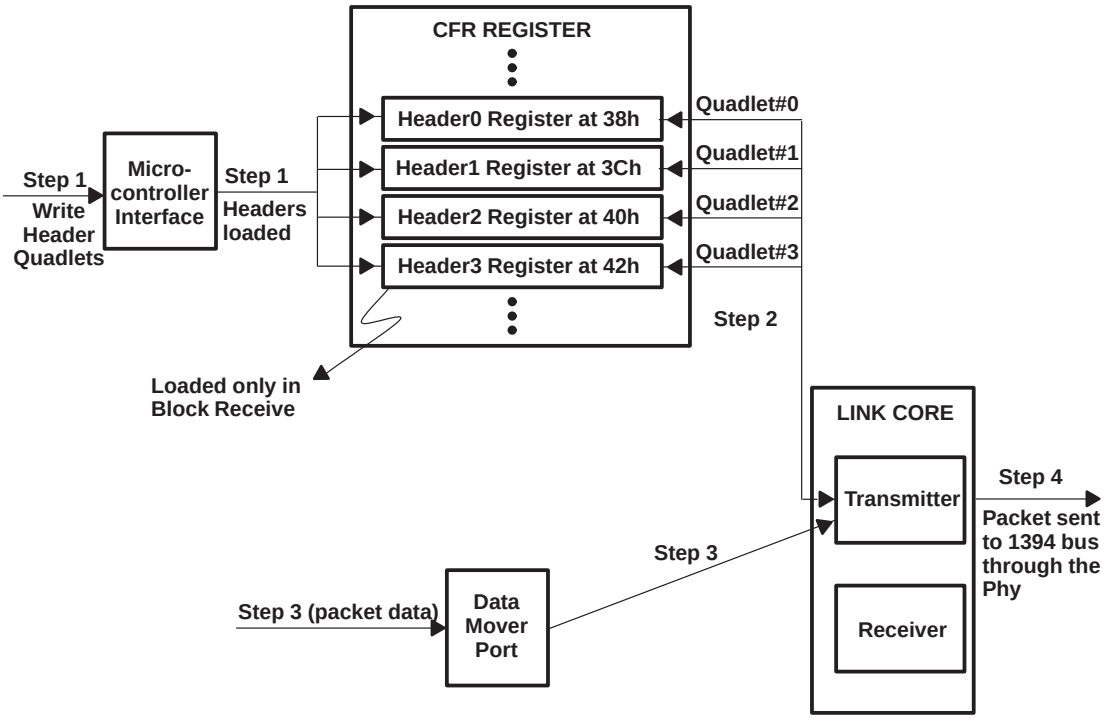


Figure 5–11. Asynchronous Transmit With Auto Header Insertion

5.1.4.2 Asynchronous Packet Transmit Without Automatic Header Insertion

In this mode, the packet headers and data information are loaded through the data mover port. This mode is sometimes called asynchronous packet transmit with manual header insertion. This is because the header quadlets are not preloaded into the header registers via the microcontroller interface. Instead, they are inserted *manually* into the data stream at the same time as the rest of the packet. The following steps further illustrate the process:

Step 1: Asynchronous header quadlets (3 quadlets in quadlet receive mode and 4 quadlets in block receive mode) are fetched into the header registers through the data mover port.

Step 2: The header quadlets are then forwarded to the transmitter of the link core.

Step 3: Packet data (payload only) is transmitted through the data mover directly to the transmitter of the link core.

Step 4: Asynchronous packet is sent to the 1394 bus through the Phy.

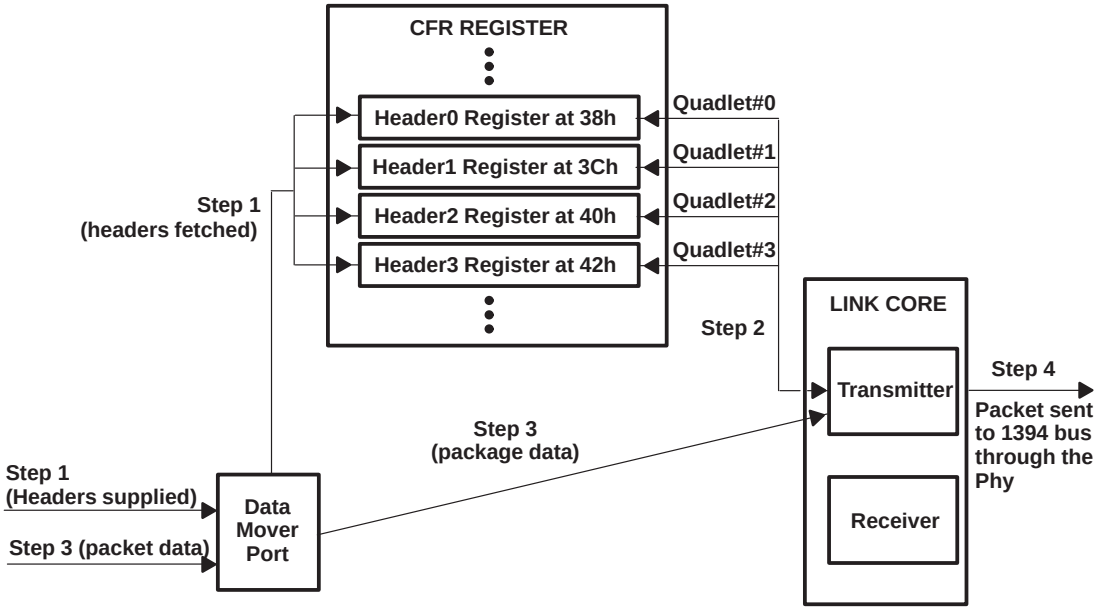


Figure 5–12. Asynchronous Transmit Without Auto Header Insertion

5.2 Data Mover Modes of Operation

The data mover (DM) port in the GP2Lynx is meant to handle an external memory interface of large data packets. The port can be configured to either transmit or receive data packets. The data can be either asynchronous or isochronous packets. All traffic through the data mover is synchronous to the rising edge of DMCLK. DMCLK is an output signal at 25 MHz.

The data mover operates by setting the DM control registers. If the DM is configured for transmit mode, it waits for DMREADY to be asserted before it can drive DMDONE low and fetch the entire block of data one packet at a time. Upon transmitting the last packet in the block, the DM will drive DMDONE high for a minimum of one DMCLK cycle (~ 40 ns). The next DMCLK cycle in which DMDONE finds DMREADY high, the process will be restarted.

The data mover has eight modes of operation which are specified by the DMASYNC, DMHDR, and DMRX bits in the DM control register at 04h. Table 5–1 shows all the DM modes of operation.

Table 5–1. Modes of Operation

DMASYNC	DMHDR	DMRX	MODE OF OPERATION
0	0	0	Isochronous packet transmit with auto header insertion
0	0	1	Isochronous packet receive without header and trailer
0	1	0	Isochronous packet transmit without header insertion
0	1	1	Isochronous packet receive with header and trailer
1	0	0	Asynchronous packet transmit with auto header insertion
1	0	1	Asynchronous packet receive without headers and trailer
1	1	0	Asynchronous packet transmit without header insertion
1	1	1	Asynchronous packet receive with headers and trailer

5.2.1 Isochronous Transmit With Automatic Header Insertion

Upon receiving a high on DMREADY, the following sequence of operations is performed:

Step 1: DMDONE will be asserted low (deactivated) at the next DMCLK cycle.

Step 2: The data mover will take the header that has been loaded into the header0 register at 38h and request the link core to transmit the data onto the 1394 bus.

Step 3: The link core will fetch the header from the header0 register.

Step 4: DMPRE will pulse for one DMCLK cycle before the first data quadlet is sent.

Step 5: The data mover will then begin to fetch the data payload by asserting DMRW high.

Step 6: When the link core has fetched the last data quadlet, the data mover checks if the number of channels specified by the control registers have been sent. If all channels have been sent the data mover waits for a subaction gap to occur before asserting DMDONE high to indicate the end of the cycle. Otherwise the data mover will provide the header in the next header register and then begin fetching the data payload until all channels are complete.

The timing diagrams in Figures 5–13 to 5–15 illustrate this mode of operation at different transmit speeds. For simplification, these diagrams show three quadlets of data payload.

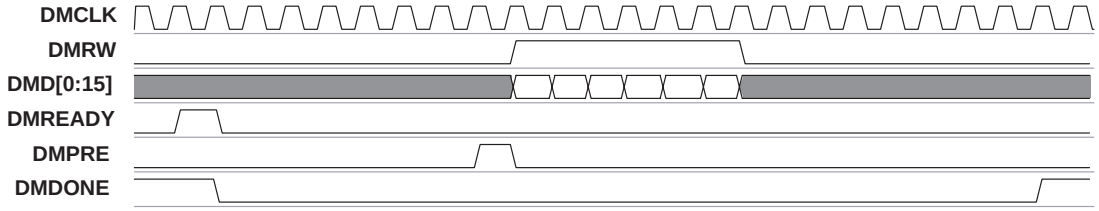


Figure 5–13. Isochronous Transmit With Auto Header Insertion at 400 Mbps

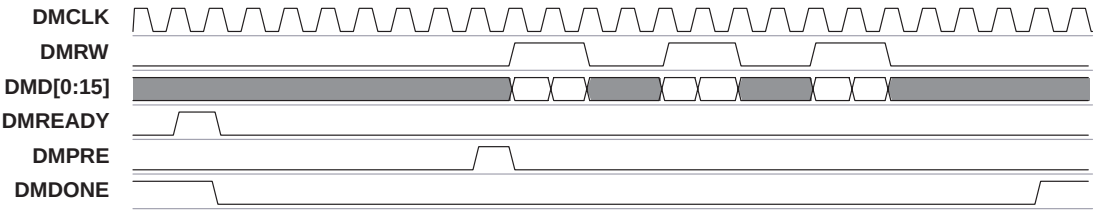


Figure 5–14. Isochronous Transmit With Auto Header Insertion at 200 Mbps

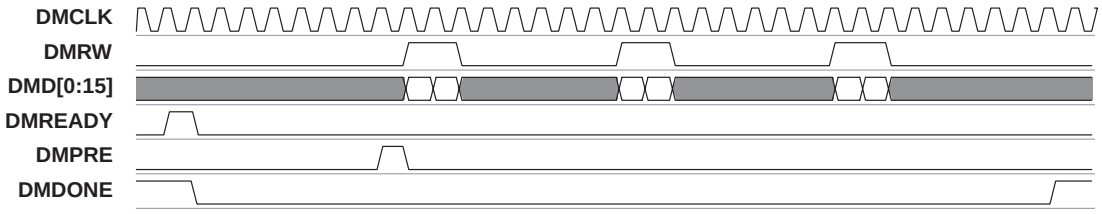


Figure 5–15. Isochronous Transmit With Auto Header Insertion at 100 Mbps

5.2.2 Isochronous Transmit Without Automatic Header Insertion

Upon receiving a high on DMREADY, the following sequence of operations is performed:

Step 1: DMDONE will be asserted low (deactivated) at the next DMCLK cycle.

Step 2: DMPRE will pulse for one DMCLK cycle before the first header quadlet is sent.

Step 3: The data mover will fetch the header by asserting DMRW high.

Step 4: The data mover will then load the header into the header0 register and request the data to be transmitted out on the 1394 bus by the link core.

Step 5: The link will fetch the header.

Step 6: DMPRE will pulse for one DMCLK cycle before the first data quadlet is sent.

Step 7: The data mover will then begin to fetch the data payload by asserting DMRW high.

Step 8: When the last data quadlet has been fetched by the link, the data mover will check if the number of channels specified by the control registers have been sent. If all channels have been sent the data mover will wait for a subaction gap to occur before asserting DMDONE high to indicate the end of the cycle. Otherwise the DM will fetch the next header and load it into the next header register and then begin fetching the data payload until all channels are complete.

Figure 5–16 shows the timing diagram for this mode at a data transmit rate of 400 Mbps. The dashed sections indicate repetitive behaviour (when the payload is more than two quadlets long).

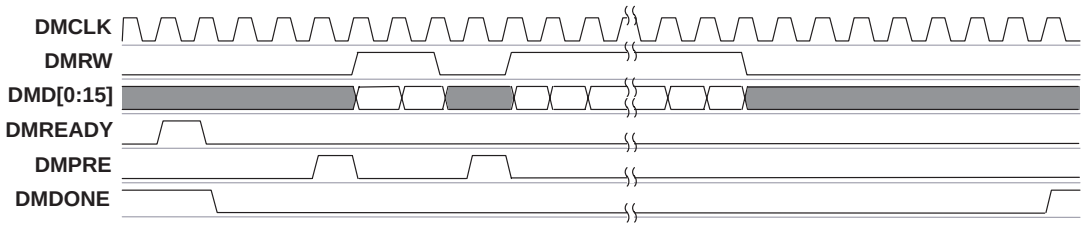


Figure 5–16. Isochronous Transmit Without Auto Header Insertion

5.2.3 Isochronous Packet Receive Without Header and Trailer

In this mode, when the link receives an isochronous packet that is addressed to it, the following sequence of operations are performed:

Step 1: The packet router control logic will route the packet to the data mover. If the sync bit field in the header quadlet matches a bit pattern in the ISYNCRCVN field of the isochronous port register at 18h, DMPRE will be asserted high for one DMCLK cycle.

Step 2: After the header is sent through, DMDONE will be asserted high for one DMCLK cycle. DMRW is then asserted high as the data payload comes through.

Step 3: After all data has been received on the DMD[0:15] lines, DMRW will be asserted low and the trailer quadlet will then come out on the DMD[0:15] lines.

PKTFLAG is never asserted high in this mode. Figure 5–17 shows the timing diagram for this mode at 400 Mbps. Figure 5–17 shows the case where DMPRE is asserted high for one DMCLK cycle to indicate that the sync bits of the received isochronous header matches the contents of the ISYNCRCVN field.

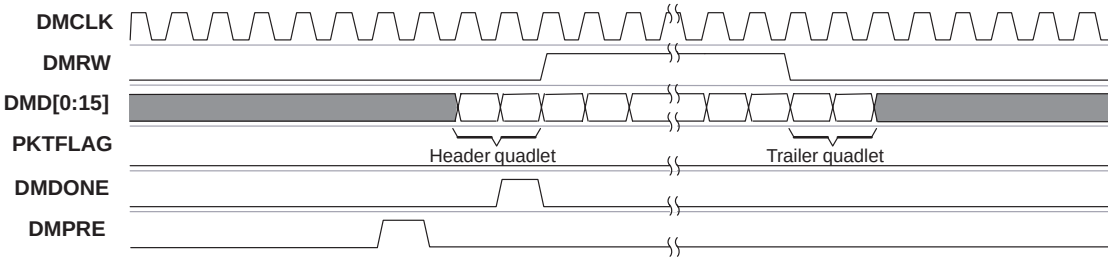


Figure 5–17. Isochronous Receive Without Header and Trailer

5.2.4 Isochronous Packet Receive With Header and Trailer

In this mode, when the link receives an isochronous packet that is addressed to it, the following sequence of operations are performed:

Step 1: The packet router control logic will route the packet to the data mover. If the sync bit field in the header quadlet matches a bit pattern in the ISYNCRCVN field of the isochronous port register at 18h, DMPRE will be asserted high for one DMCLK cycle. At the same time DMDONE will be asserted high for one DMCLK cycle.

Step 2: This is followed by DMRW asserted high as the packet comes through. PKTFLAG is only asserted high when the header quadlet is being received.

Step 3: After all the data payload has been received on the DMD[0:15] lines, PKTFLAG will be asserted high again as the trailer quadlet is being received. Once the entire packet is received, the DMRW line will be asserted low.

Figure 5–18 shows the timing diagram for this mode at 400 Mbps. Also, Figure 5–18 shows the case where DMPRE is asserted high for one DMCLK cycle to indicate that the sync bits of the received isochronous header matches the contents of the ISYNCRCVN field.

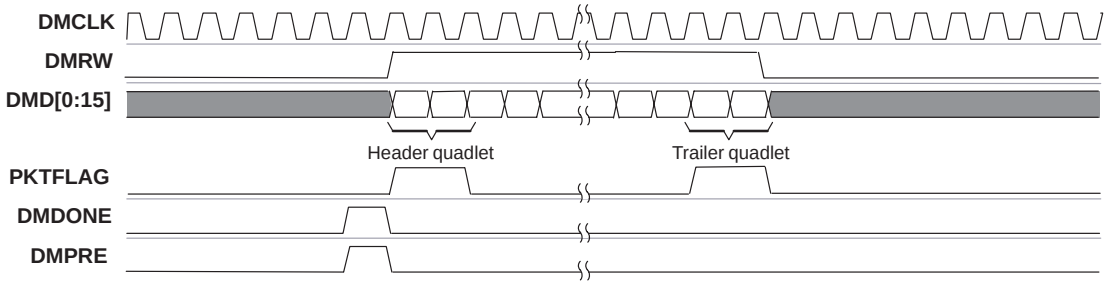


Figure 5–18. Isochronous Receive With Header and Trailer

Figure 5–19 shows the timing diagram at 200 Mbps when the received packet contains only one quadlet of payload.

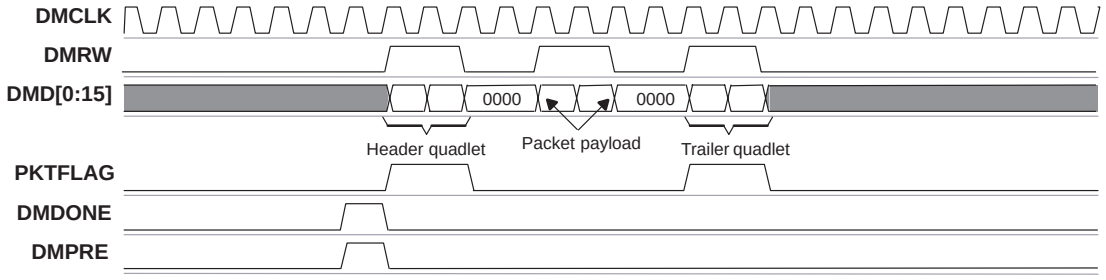


Figure 5–19. Isochronous Receive With Header and Trailer at 200 Mbps

5.2.5 Asynchronous Packet Transmit With Automatic Header Insertion

Upon receiving a high on DMREADY, the following sequence of operations are performed:

Step 1: DMDONE will be asserted low (deactivated) at the next DMCLK cycle.

Step 2: The data mover will take the headers that have been loaded into the header0–header3 registers and request the link core to transmit the data onto the 1394 bus.

Step 3: The link core will fetch the headers from the header0–header3 registers.

Step 4: DMPRE will pulse for one DMCLK cycle before the first data quadlet is sent.

Step 5: The data mover will then begin to fetch the data payload by asserting DMRW high.

Step 6: When the link core has fetched the last data quadlet, the data mover waits until the destination node returns an *ack_complete* immediate response. If an *ack_complete* is not received, the data mover will assert DMERROR high and become disabled.

Figure 5–20 and Figure 5–21 show the timing diagram for this mode for the quadlet transmit and the block transmit cases, respectively. For simplicity, a data block size of three quadlets was selected in Figure 5–20. Figure 5–22 shows the block transmit case at 400 Mbps.

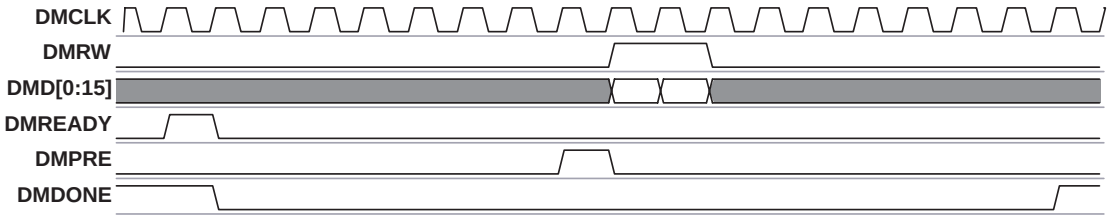


Figure 5–20. Asynchronous Quadlet Transmit With Automatic Header Insertion

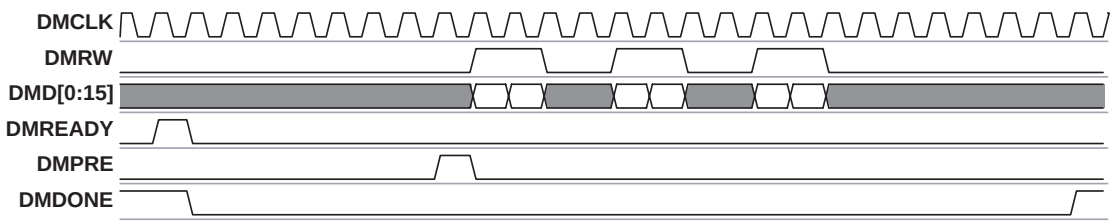


Figure 5–21. Asynchronous Block Transmit With Automatic Header Insertion at 200 Mbps

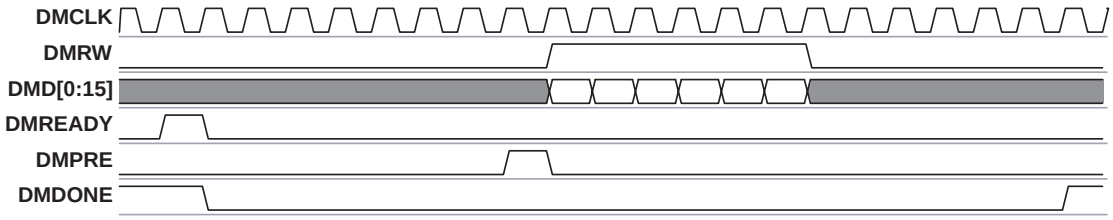


Figure 5–22. Asynchronous Block Transmit With Automatic Header Insertion at 400 Mbps

5.2.6 Asynchronous Packet Transmit Without Automatic Header Insertion

Upon receiving a high on DMREADY, the following sequence of operations are performed:

Step 1: DMDONE will be asserted low (deactivated) at the next DMCLK cycle.

Step 2: DMPRE will pulse for one DMCLK cycle before the header quadlets are sent.

Step 3: The data mover will fetch the headers by asserting DMRW high.

Step 4: The data mover will then load the headers into the header0–header3 registers and request the data to be transmitted out on the 1394 bus by the link core.

Step 5: The link will fetch the headers.

Step 6: DMPRE will pulse for one DMCLK cycle before the first data quadlet is sent.

Step 7: The data mover will then begin to fetch the data payload by asserting DMRW high.

Step 8: When the link core has fetched the last data quadlet, the data mover waits until the destination node returns an *ack_complete* immediate response. If an *ack_complete* is not received, the data mover will assert DMERROR high and become disabled.

Figure 5–23 and Figure 5–24 show the timing diagram for this mode for the quadlet transmit and the block transmit cases, respectively. For simplicity, a data block size of three quadlets was selected in Figure 5–24.

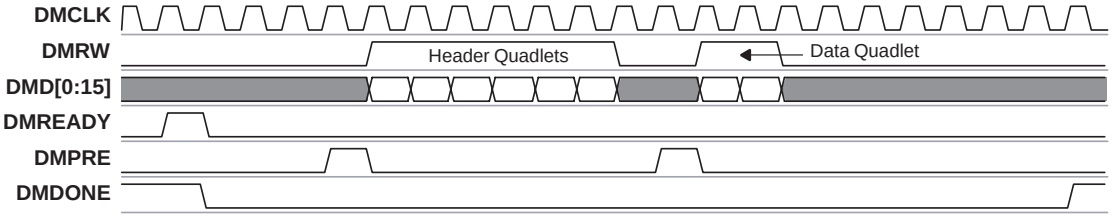


Figure 5–23. Asynchronous Quadlet Transmit Without Automatic Header Insertion at 400 Mbps

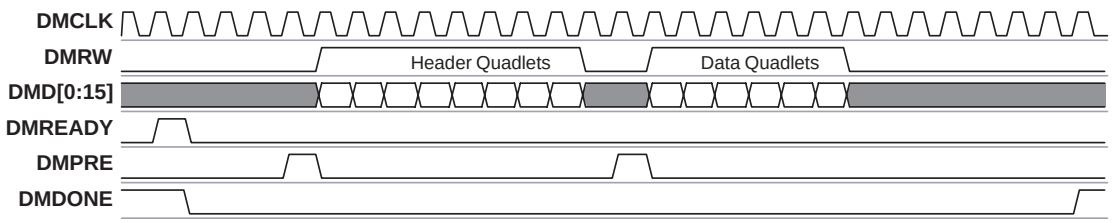


Figure 5–24. Asynchronous Block Transmit Without Automatic Header Insertion at 400 Mbps

5.2.7 Asynchronous Packet Receive With Headers and Trailer

In this mode, when the link receives an isochronous packet that is addressed to it, the following sequence of operations is performed:

Step 1: The packet router control logic will route the packet to the data mover. At the same time DMDONE will be asserted high for one DMCLK cycle.

Step 2: This is followed by DMRW asserted high as the packet comes through. PKTFLAG is only asserted high when the header quadlets are being received.

Step 3: After all the data payload has been received on the DMD[0:15] lines, PKTFLAG will be asserted high again as the trailer quadlet is being received. Once the entire packet is received, the DMRW line will be asserted low.

Figure 5–25 and Figure 5–26 show the timing diagram for this mode for the quadlet receive and the block receive cases, respectively. For simplicity, a data block size of three quadlets was selected in Figure 5–26

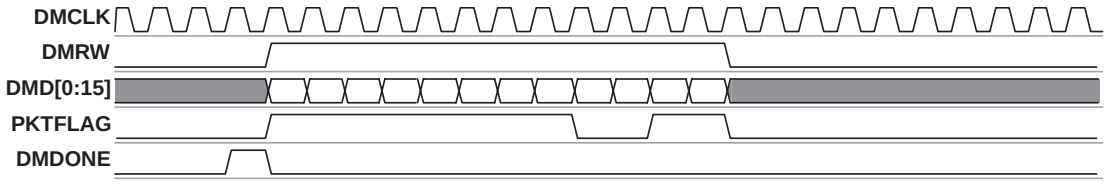


Figure 5–25. Asynchronous Quadlet Receive With Headers and Trailer at 400 Mbps

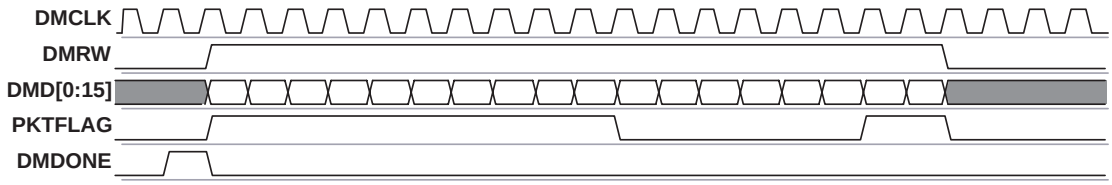


Figure 5–26. Asynchronous Block Receive With Headers and Trailer at 400 Mbps

5.2.8 Asynchronous Packet Receive Without Headers and Trailer

In this mode, when the link receives an isochronous packet that is addressed to it, the following sequence of operations are performed:

Step 1: The packet router control logic will route the packet to the data mover. After the headers are sent through, DMDONE will be asserted high for one DMCLK cycle.

Step 2: DMRW is then asserted high as the data payload comes through.

Step 3: After all data has been received on the DMD[0:15] lines, DMRW will be asserted low and the trailer quadlet will then come out on the DMD[0:15] lines.

Figure 5–27 and Figure 5–28 show the timing diagram for this mode for the quadlet receive and the block receive cases, respectively. For simplicity, a data block size of three quadlets was selected in Figure 5–28

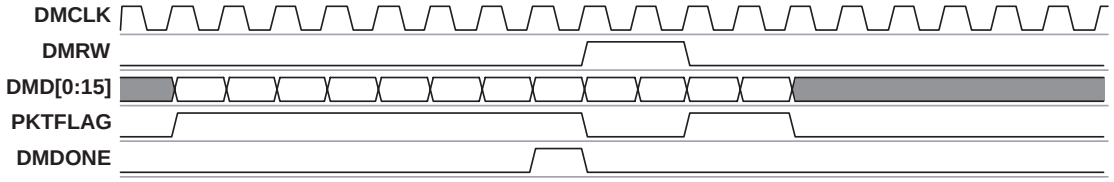


Figure 5–27. Asynchronous Quadlet Receive Without Headers and Trailer at 400 Mbps

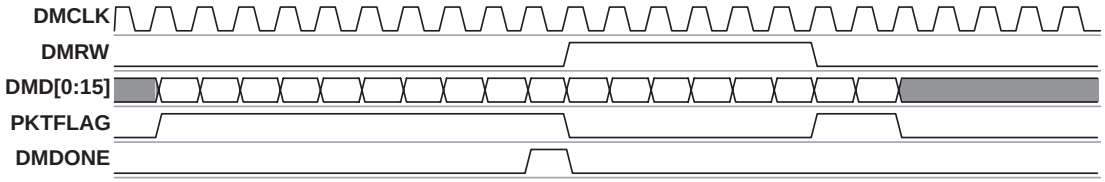


Figure 5–28. Asynchronous Block Receive Without Headers and Trailer at 400 Mbps

5.3 Data Mover Byte Mode

In this mode the DMD lines are only 1 byte wide => the maximum speed is only 200 Mbit/sec. Only bits 0–7 will be used for the data bus. DMERROR will be asserted if transmission of a 400 Mbit/sec packet is attempted.

5.4 Data Mover Endian Swapping

In this mode the DMD[0:15] bytes are swapped. If the data mover is in byte mode, the least significant byte is fetched first (Figure 5–29). If the data mover is not in byte mode, the least significant word is fetched first and the byte order is then swapped (Figure 5–30).

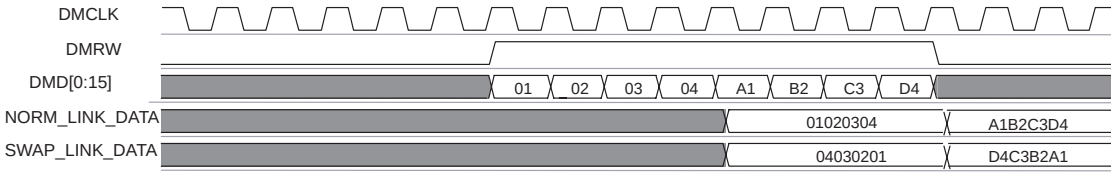


Figure 5–29. Endian Swapping in Byte Mode

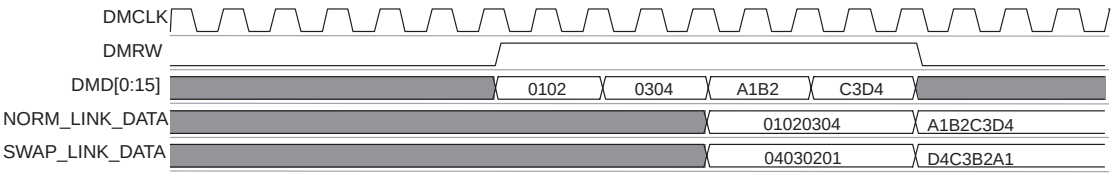


Figure 5–30. Endian Swapping in Word Mode

5.5 Data Mover Handshake Mode

In this mode, when DMDONE is asserted high it will check for DMREADY low as an acknowledge. This is equivalent to the mode used in the TSB12LV31 (GPLynx), as shown in Figure 5–31.

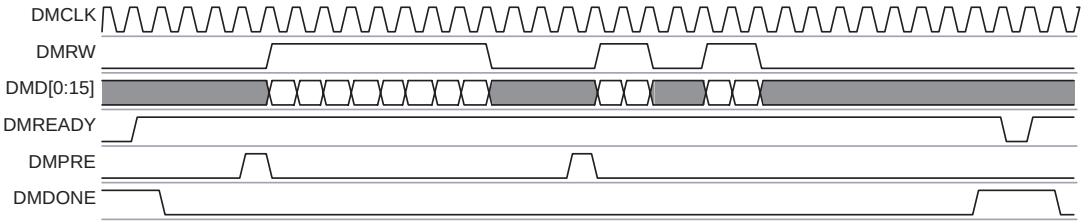


Figure 5–31. Data Mover Handshake Mode (GPLynx mode)

5.6 Data Mover Critical Timing

Table 5–2. CLK to Output Timing With Respect to DMCLK

PARAMETER		TERMINAL NAME	MIN	MAX	UNIT
t_{d0}	Delay time (DMCLK to Q)	DMDONE	1.75	8.5	ns
t_{d1}		DMRW	1.75	7.5	
t_{d2}		DMPRE	1.75	14.5	
t_{d3}		DMERROR	1.5	11.5	
t_{d4}		PKTFLAG	1.75	8.5	
t_{d5}		DMD[0:15]	0.5	9	
t_{su0}	Setup time to DMCLK	DMREADY	14		ns
t_{su1}		DMD[0:15]	14		
t_{h0}	Hold time from DMCLK	DMREADY	–1		ns
t_{h1}		DMD[0:15]	–0.75		

NOTE: All timing parameters are with respect to the rising edge of DMCLK

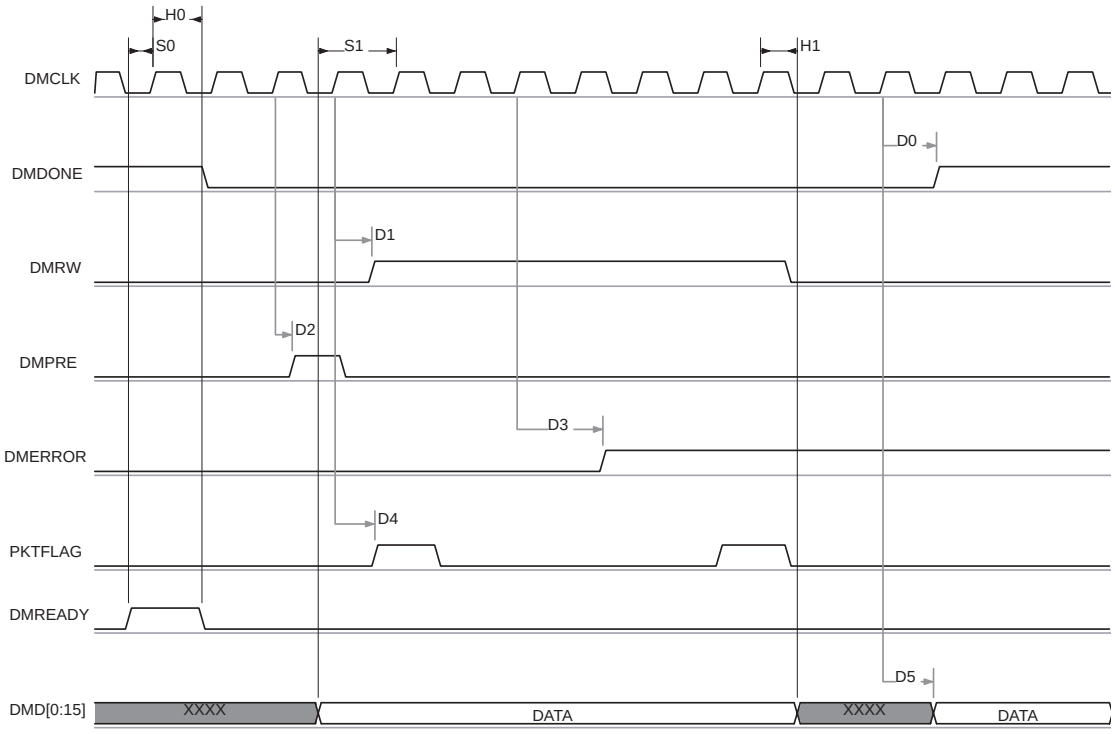


Figure 5–32. Clock to Output Timing With Respect to DMCLK

6 FIFO Memory Access

Access to all FIFO memories is fundamentally the same, only the addresses to where the write is made changes. Figure 6–1 shows the FIFO-address access map. The FIFO is separated into an asynchronous transmit FIFO (ATF) and a general receive FIFO (GRF), each of 517 quadlets (2 Kbytes). Since asynchronous packets may also be transmitted through the data mover port and the ATF always has priority and its data wil be transmitted first.

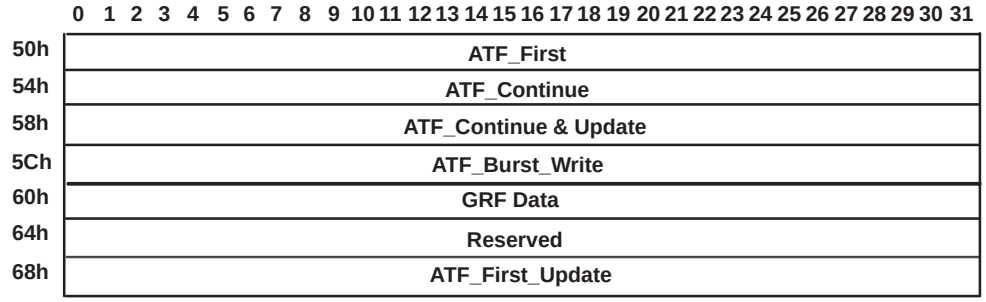


Figure 6–1. TSB12LV32 Controller-FIFO-Access Address Map

6.1 General

The suffix _First denotes a write to the FIFO location where the first quadlet of a packet should be written when the writer wants to transmit the packet. The first quadlet is held in the FIFO until a quadlet is written to an update location. The suffix _Continue denotes a write to the FIFO location where the second through n-1 quadlets of a packet should be written. The second through n-1 quadlets are held in the FIFO until a quadlet is written to an update location. The suffix_Continue & Update denotes a write to the FIFO location where the last quadlet of a multiple quadlet packet should be written.

6.2 ATF Access

The procedure for accessing the ATF for a quadlet write operation is accomplished in three successive steps. To ensure that an ATF underflow condition does not occur, loading of the ATF in the following manner is highly recommended:

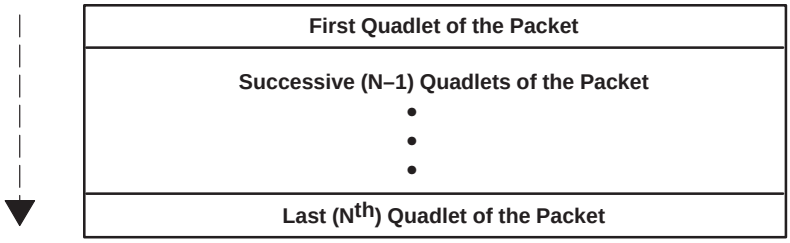


Figure 6–2. Asynchronous Packet With N Quadlets (ATV Loading Operation)

Each quadlet can be written into the ATF register on byte (8-bit) boundary or word (16-bit) boundary. To write to the ATF in a byte fashion, the following steps should be followed:

Step 1: Writing the first quadlet of the packet:
- a) Write the first 8-bits of the quadlet to ATF location 50h.
- b) Write the second 8-bits of the quadlet to ATF location 51h.
- c) Write the third 8-bits of the quadlet to ATF location 52h.
- d) Write the fourth 8-bits of the quadlet to ATF location 53h.

The data is not yet confirmed for transmission.

Step 2: Writing the next (n-1) quadlets of the packet:
a) Write the first 8-bits of each quadlet to ATF location 54h.
b) Write the second 8-bits of each quadlet to ATF location 55h.
c) Write the third 8-bits of each quadlet to ATF location 56h.
d) Write the fourth 8-bits of each quadlet to ATF location 57h.
The data is not yet confirmed for transmission.

Step 3: Last (N^{th}) quadlet of the packet:
a) Write the first 8-bits of the quadlet to ATF location 58h.
b) Write the second 8-bits of the quadlet to ATF location 59h.
c) Write the third 8-bits of the quadlet to ATF location 5Ah.
d) Write the fourth 8-bits of the quadlet to ATF location 5Bh.
The data is now confirmed for transmission.

To write to the ATF in a word fashion, the following steps should be followed:

Step 1: Writing the first quadlet of the packet:
a) Write the first 16-bits of the quadlet to ATF location 50h.
b) Write the second 16-bits of the quadlet to ATF location 52h.
The data is not yet confirmed for transmission.

Step 2: Writing the next (n-1) quadlets of the packet:
a) Write the first 16-bits of each quadlet to ATF location 54h.
b) Write the second 16-bits of each quadlet to ATF location 56h.
The data is not yet confirmed for transmission.

Step 3: Last (N^{th}) quadlet of the packet:
a) Write the first 16-bits of the quadlet to ATF location 58h.
b) Write the second 16-bits of the quadlet to ATF location 5Ah.
The data is now confirmed for transmission.

All writes to the ATF must be quadlet aligned (i.e., only an even number of write accesses is allowed). If the first quadlet of a packet is not written to the ATF_First location, the transmitter enters a state denoted by an ATStuck interrupt. An underflow of the ATF also causes an ATSTK interrupt. When this state is entered, no asynchronous packets can be sent until the ATF is cleared by way of the ATFCLR control bit (bit 0 at CFR 30h). However isochronous packets may still be sent while the ATF is in this state.

6.3 ATF Burst Access

It is allowable to perform a burst write into location 54h (ATF_Continue), which allows multiple quadlets to load into ATF, but the data is not confirmed for transmission. It is also allowable to perform burst write to location 58h (ATF_Continue & Update), which allows multiple quadlets to load into ATF, and the data is confirmed for transmission. Write accesses to address 5Ch (ATF_Burst Write) writes the whole packet into the ATF. The first quadlet written into ATF has the control bit set to 1 to indicate this is the first quadlet of the packet, and the rest of the quadlets have the CD bit set to 0. The last quadlet written into ATF confirms the packet for transmission.

To do a burst write operation the host bus master must continually drive $\overline{\text{MCS}}$ low. The TSB12LV32 loads MD0–MD15 to the ATF during each rising edge of BCLK while $\overline{\text{MCS}}$ is low. At the same time it asserts $\overline{\text{MCA}}$ ($\overline{\text{MCA}}$ is always one cycle behind $\overline{\text{MCS}}$) low. The CD bit is 0 for ATF_Continue and ATF_Continue & Update.

The ATF_First_Update is a unique address location optimised for transmitting zero length isochronous packets (including asynchronous streaming packets). A zero-length packet contains no data payload, and only the packet header and header CRC are transmitted.

6.4 General-Receive-FIFO (GRF)

Access to the GRF is done with a read from the GRF, which requires a read from address 60h. The GRF will accumulate self-ID packets upon bus-reset. All quadlets of a self-ID packet are saved in the GRF after

power up. Hardware will check to insure the second quadlet is indeed the complement (logical inverse) of the first quadlet. If there are any errors associated with the self-ID process, a self-ID interrupt will be generated and the self-ID check register at 38h will be updated to reflect the error(s). This option can be turned off by setting the FULLSID bit in the control register at 08h to 0.

6.5 GRF Stored Data Format

Each quadlet in the GRF is internally 33-bits wide. The most significant bit (extra bit) is used to indicate whether it is a packet token or a regular received quadlet (received header CRC and data CRC are checked and not stored in GRF). This bit is called the CD bit, which value is reflected in bit #16 of the FIFO status register. If CD bit is 1, the next quadlet read from the GRF is a packet token. If the CD bit is 0, the next quadlet read from the GRF is a regular received quadlet. A packet token is stored as the first quadlet for each received confirmed packet. The definition for packet token is shown in Table 6-1. Bit 0 is most significant bit and bit 32 is the least significant bit.

Table 6–1. Packet Token Definition

BITS	NAME	DESCRIPTION
0	CD	CD bit is 1 for packet token. This bit should only be read from the FIFO status Register at 30h
1–2	Reserved	Reserved
3–16	QUADLET_COUNT	Expected quadlet count after packet token for this received packet.
17–19	Reserved	Reserved
20–24	ackCode	If bit 20 is 0, bits[21:24] are used as the Ack code that was sent back to the transmitting node. If bit 20 is 1, it is an error condition and an error Ack code is sent to the transmitting node.
25–26	Reserved	Reserved
27–28	SPEED	The speed code for the received packet. 00 – 100 Mb/s 01 – 200 Mb/s 10 – 400 Mb/s
29–32	Reserved	Reserved

7 TSB12LV32 Data Formats

The data formats for transmission and reception of data are shown in the following sections. The transmit format describes the expected organization of data presented to the TSB12LV32 at the host-bus interface. The receive formats describe the data format that the TSB12LV32 presents to the host-bus interface.

7.1 Asynchronous Transmit (Host Bus to TSB12LV32)

Asynchronous transmit refers to the use of the asynchronous-transmit FIFO (ATF) interface. The general-receive FIFO (GRF) is shared by asynchronous data and isochronous data. There are two basic formats for data to be transmitted and received. The first is for quadlet packets, and the second is for block packets. For transmits, the FIFO address indicates the beginning, middle, and end of a packet. For receives, the data length, which is found in the header of the packet, determines the number of bytes in a block packet.

7.1.1 Quadlet Transmit

The quadlet-transmit format is shown in Figure 7–1 and 7–2, are described in Table 7–1. The first quadlet contains packet control information. The second and third quadlets contain the 64-bit, quadlet-aligned address. The fourth quadlet is data used only for write requests and read responses. For read requests and write responses, the quadlet data field is omitted.

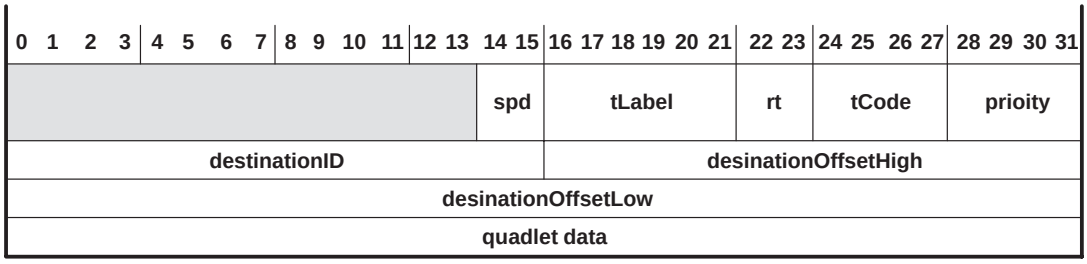


Figure 7–1. Quadlet-Transmit Format (Write Request)

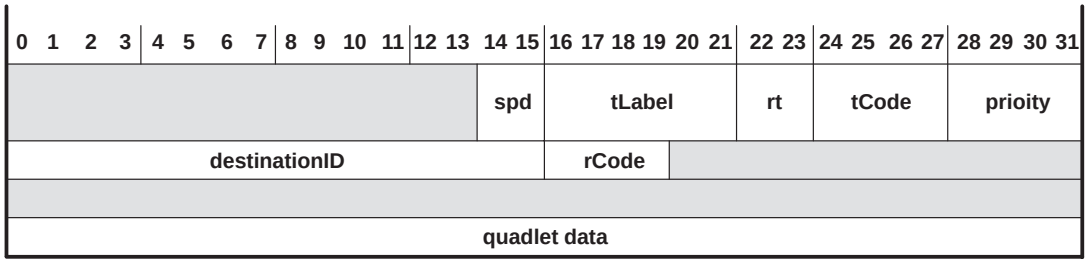


Figure 7–2. Quadlet-Transmit Format (Read Response)

Table 7–1. Quadlet-Transmit Format Functions

<table>
<tr><th>FIELD NAME</th><th colspan="3">DESCRIPTION</th></tr>
<tr><td>spd</td><td colspan="3">The spd field indicates the speed at which the current packet is to be sent. 00 = 100 Mb/s, 01 = 200 Mb/s, and 10 = 400 Mb/s, and 11 is undefined for this implementation.</td></tr>
<tr><td>tLabel</td><td colspan="3">The tLabel field is the transaction label, which is a unique tag for each outstanding transaction between two nodes. This field is used to pair up a response packet with its corresponding request packet.</td></tr>
<tr><td>rt</td><td colspan="3">The rt field is the retry code for the current packet is: 00 = new, 01 = retry_X, 10 = retryA, and 11 = retryB.</td></tr>
<tr><td>tCode</td><td colspan="3">The tCode field is the transaction code for the current packet (see Table 6–10 of IEEE–1394 standard).</td></tr>
<tr><td>priority</td><td colspan="3">The priority field contains the priority level for the current packet. For cable implementation, the value of the bits must be zero (for backplane implementation, see clause 5.4.1.3 and 5.4.2.1 of the IEEE–1394 standard).</td></tr>
<tr><td>destinationID</td><td colspan="3">The destinationID field is the concatenation of the 10-bit bus number and the 6-bit node number that forms the destination node address of the current packet.</td></tr>
<tr><td>destination OffsetHigh, destination OffsetLow</td><td colspan="3">The concatenation of these two fields addresses a quadlet in the destination node address space. This address must be quadlet aligned (modulo 4).</td></tr>
<tr><td>quadlet data</td><td colspan="3">For write requests and read responses, the quadlet data field holds the data to be transferred. For write responses and read requests, this field is not used and should not be written into the FIFO.</td></tr>
<tr><td>rcode</td><td colspan="3">Specifies the result of the read request transaction. The response codes that may be returned to the requesting agent are defined as follows:</td></tr>
<tr><td></td><td>Response Code</td><td>Name</td><td>Description</td></tr>
<tr><td></td><td>0</td><td>resp_complete</td><td>Node successfully completed requested operation</td></tr>
<tr><td></td><td>1–3</td><td>reserved</td><td></td></tr>
<tr><td></td><td>4</td><td>resp_conflict_error</td><td>Resource conflict detected by responding agent. Request may be retried.</td></tr>
<tr><td></td><td>5</td><td>resp_data_error</td><td>Hardware error. Data not available.</td></tr>
<tr><td></td><td>6</td><td>resp_type_error</td><td>Field within request packet header contains un supported or invalid vallue.</td></tr>
<tr><td></td><td>7</td><td>resp_address_error</td><td>Address location within specified node not accessible</td></tr>
<tr><td></td><td>8-Fh</td><td>reserved</td><td></td></tr>
</table>

7.1.2 Block Transmit

The block-transmit format is shown in Figure 7–3 and is described in Table 7–2. The first quadlet contains packet-control information. The second and third quadlets contain the 64-bit address. The first 16 bits of the fourth quadlet contains the dataLength field. This is the number of bytes of data in the packet. The remaining 16 bits represent the extended_tCode field (see Table 6–11 of the IEEE–1394 standard for more information on extended_tCodes). The block data, if any, follows the extended_tCode.

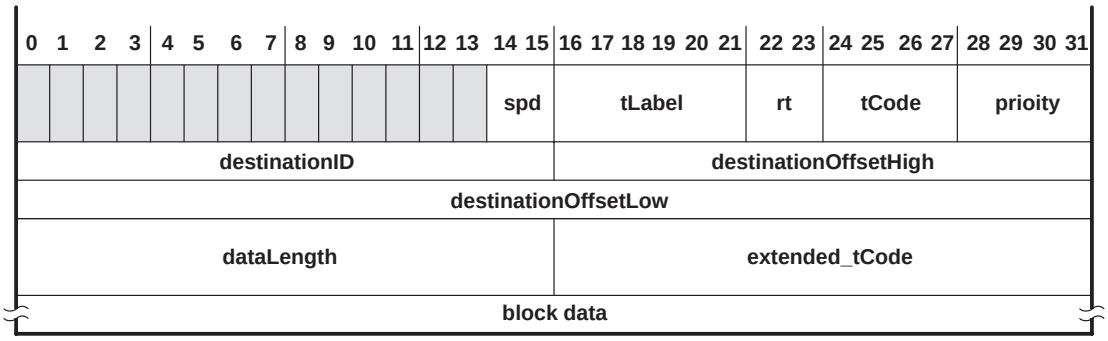


Figure 7–3. Block-Transmit Format

Table 7–2. Block-Transmit Format Functions

FIELD NAME	DESCRIPTION
Spd	The spd field indicates the speed at which the current packet is to be sent. 00 = 100 Mb/s, 01 = 200 Mb/s, and 10 = 400 Mb/s, and 11 is undefined for this implementation.
TLabel	The tLabel field is the transaction label, which is a unique tag for each outstanding transaction between two nodes. This field is used to pair up a response packet with its corresponding request packet.
Rt	The rt field is the retry code for the current packet is 00 = new, 01 = retry_X, 10 = retryA, and 11 = retryB.
TCode	tCode is the transaction code for the current packet (see Table 6–10 of IEEE–1394 standard).
priority	The priority level for the current packet. For cable implementation, the value of the bits must be zero. For backplane implementation, see clause 5.4.1.3 and 5.4.2.1 of the IEEE–1394 standard.
destinationID	The destinationID field is the concatenation of the 10-bit bus number and the 6-bit node number that forms the node address to which the current packet is being sent.
destination OffsetHigh, destination OffsetLow	The concatenation of the destination OffsetHigh and the destination OffsetLow fields addresses a quadlet in the destination node address space. This address must be quadlet aligned (modulo 4). The upper 4 bits of the destination OffsetHigh field are used as the response code for lock-response packets and the remaining bits are reserved.
dataLength	The dataLength filed contains the number of bytes of data to be transmitted in the packet.
extended_tCode	The block extended_tCode to be performed on the data in the current packet (see Table 6–11 of the IEEE–1394 standard).
block data	The block data field contains the data to be sent. If dataLength is 0, no data should be written into the FIFO for this field. Regardless of the destination or source alignment of the data, the first byte of the block must appear in byte 0 of the first quadlet.

7.1.3 Quadlet Receive

The quadlet-receive format through the FIFO is shown in Figure 7–4 and is described in Table 7–3. The first quadlet (trailer) contains the packet-reception status that is added by the TSB12LV32. The first 16 bits of the second quadlet contain the destination node and bus ID, and the remaining 16 bits contain packet-control information. The first 16 bits of the third quadlet contain the node and bus ID of the source, and the remaining 16 bits of the third quadlet and the fourth quadlet contain the 48-bit, quadlet-aligned destination offset address. The last quadlet contains data that is used by write requests and read responses. For read requests and write responses, the quadlet data field is omitted.

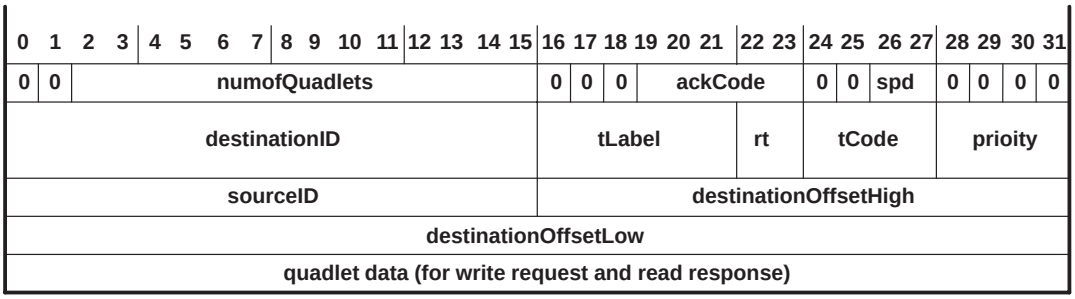


Figure 7–4. FIFO Quadlet-Receive Format

The quadlet-receive format through the DM is shown in Figure 7–5 and is described in Table 7–3. This format is similar to the quadlet receive format for the TSB12LV31(GPLynx). The first 16 bits of the first quadlet contain the destination node and bus ID, and the remaining 16 bits contain packet-control information. The first 16 bits of the second quadlet contain the node and bus ID of the source, and the remaining 16 bits of the second and third quadlets contain the 48-bit, quadlet-aligned destination offset address. The fourth quadlet contains data that is used by write requests and read responses. For read requests and write responses, the quadlet data field is omitted. The last quadlet (trailer) contains the packet-reception status that is added by the TSB12LV32.

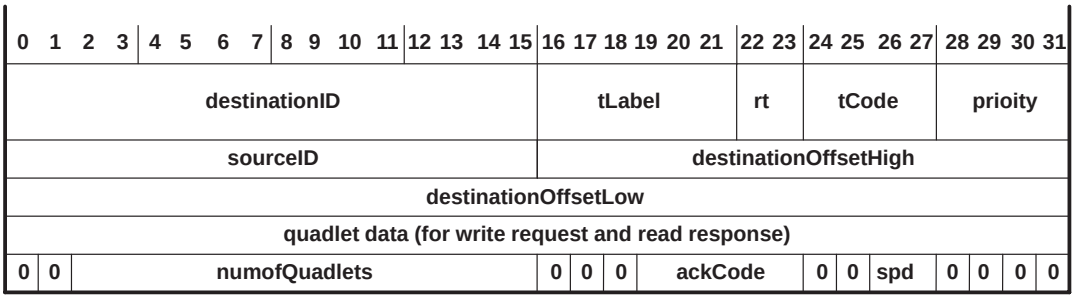


Figure 7–5. Data Mover Quadlet-Receive Format

Table 7–3. Quadlet–Receive Format Functions

FIELD NAME	DESCRIPTION
numofQuadlets	Total number of quadlets in the current packet (payload and header quadlets only).
ackCode	This 5-bit field holds the acknowledge code sent by the receiver for the current packet (see Table 6-13 in the draft standard).
Spd	The spd field indicates the speed at which the current packet was sent. 00 = 100 Mbits/s, 01 =200 Mbits/s, 10 = 400 Mbits/s, and 11 is undefined for this implementation.
destinationID	The destinationID field contains the concatenation of the 10-bit bus number and the 6-bit node number that forms the node address to which the current packet is being sent.
tLabel	The tLabel field is the transaction label, which is a unique tag for each outstanding transaction between two nodes. This field is used to pair up a response packet with its corresponding request packet.
rt	The rt field is the retry code for the current packet is 00 = new, 01 = retry_X, 10 = retryA, and 11 = retryB.
tCode	The tCode field is the transaction code for the current packet (see Table 6-10 of the IEEE–1394 standard).
priority	The priority field contains the priority level for the current packet. For cable implementation, the value of the bits must be zero (for backplane implementation, see clause 5.4.1.3 and 5.4.2.1 of the IEEE–1394 standard).
sourceID	The sourceID field contains the node ID of the sender of the current packet.
destination OffsetHigh, destination OffsetLow	The concatenation of the destination OffsetHigh and the destination OffsetLow fields addresses a quadlet in the destination nodes address space. This address must be quadlet aligned (modulo 4). (The upper four bits of the destination OffsetHigh field are used as the response code for lock-response packets, and the remaining bits are reserved.)
quadlet data	For write requests and read responses, the quadlet data field holds the transferred data. For write responses and read requests, this field is not present.

7.1.4 Block Receive

The block-receive format through the FIFO is shown in Figure 7-6 and is described in Table 7-5. The first 16 bits of the first quadlet contain the node and bus ID of the destination node, and the last 16 bits contain packet-control information. The first 16 bits of the second quadlet contain the node and bus ID of the source node, and the last 16 bits of the second quadlet and all of the third quadlet contain the 48-bit, quadlet-aligned destination offset address. All remaining quadlets, except for the last one, contain data that is used only for write requests and read responses. For block read requests and block write responses, the data field is omitted. The last quadlet contains packet-reception status.

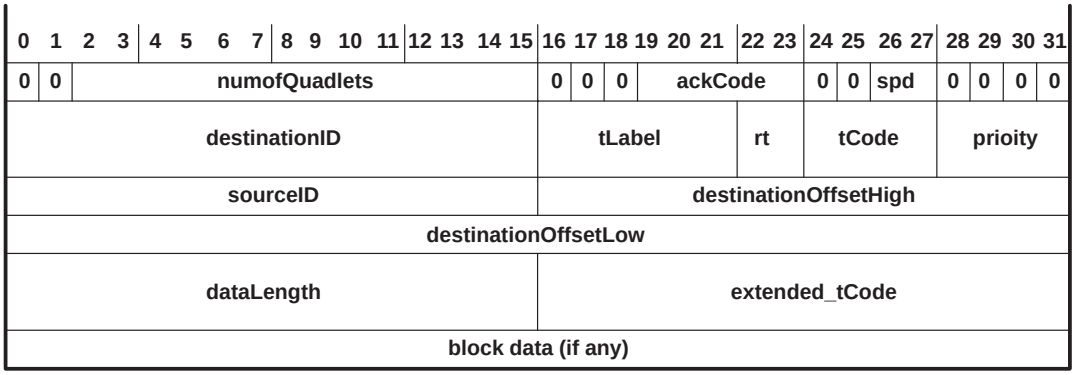


Figure 7–6. FIFO Block-Receive Format

The block-receive format through the FIFO is shown in Figure 7-7.

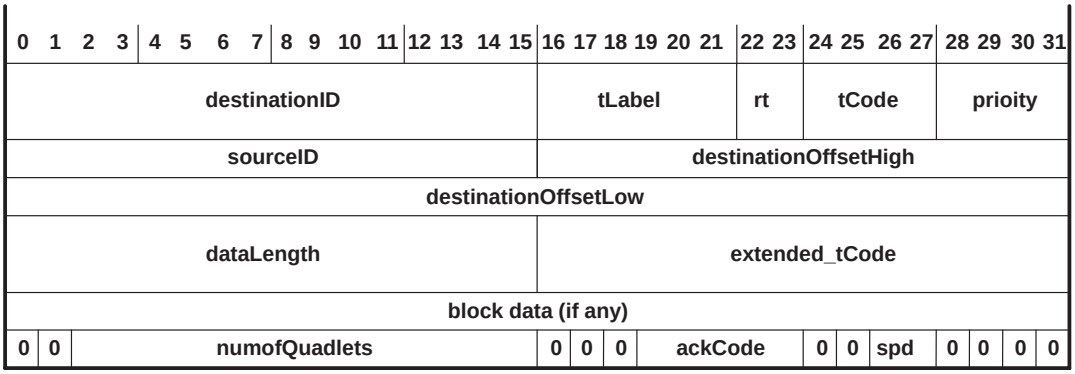


Figure 7–7. Data Mover Block-Receive Format

Table 7–4. Block-Receive Format Functions

FIELD NAME	DESCRIPTION
numofQuadlets	Total number of quadlets in the current packet (payload and header quadlets only)
ackCode	This 5-bit field holds the acknowledge code sent by the receiver for the current packet (see Table 6-13 in the draft standard).
destinationID	The destinationID field is the concatenation of the 10-bit bus number and the 6-bit node number that forms the node address to which the current packet is being sent.
tLabel	The tLabel field is the transaction label, which is a unique tag for each outstanding transaction between two nodes. This field is used to pair up a response packet with its corresponding request packet.
rt	The rt field contains the retry code for the current packet is 00 = new, 01 = retry_X, 10 = retryA, and 11 = retryB.
tCode	The tCode field is the transaction code for the current packet (see Table 6-10 of the IEEE-1394 standard).
priority	The priority field contains the priority level for the current packet. For cable implementation, the value of the bits must be zero (for backplane implementation, see clause 5.4.1.3 and 5.4.2.1 of the IEEE-1394 standard).
sourceID	The sourceID field contains the node ID of the sender of the current packet.
destination OffsetHigh, destination OffsetLow	The concatenation of the destination OffsetHigh and the destination OffsetLow fields addresses a quadlet in the destination nodes address space. This address must be quadlet aligned (modulo 4). The upper 4 bits of the destination OffsetHigh field are used as the response code for lock-response packets and the remaining bits are reserved.
dataLength	For write request, read responses, and locks, the dataLength field indicates the number of bytes being transferred. For read requests, the dataLength field indicates the number of bytes of data to be read. A write-response packet does not use this field. Note that the number of bytes does not include the head, only the bytes of block data.
extended_tCode	The extended_tCode field contains the block extended_tCode to be performed on the data in the current packet (see Table 6-11 of the IEEE-1394 standard).
block data	The block data field contains any data being transferred for the current packet. Regardless of the destination address or memory alignment, the first byte of the data appears in byte 0 of the first quadlet of this field. The last quadlet of the field is padded with zeros out to four bytes, if necessary.
spd	The spd field indicates the speed at which the current packet was sent. 00 = 100 Mb/s, 01 = 200 Mb/s, 10 = 400 Mb/s, and 11 is undefined for this implementation.

7.2 Isochronous Transmit (Host Bus to TSB12LV32)

The format of the isochronous-transmit packet is shown in Figure 7–8 and is described in Table 7–5. The data for each channel must be presented to the isochronous-transmit FIFO interface in this format in the order that packets are to be sent. The transmitter sends any packets available at the isochronous-transmit interface immediately following reception or transmission of the cycle-start message. The speed at which the current packet is sent is determined by the *speed* field in the DM control register (bits 22-23)

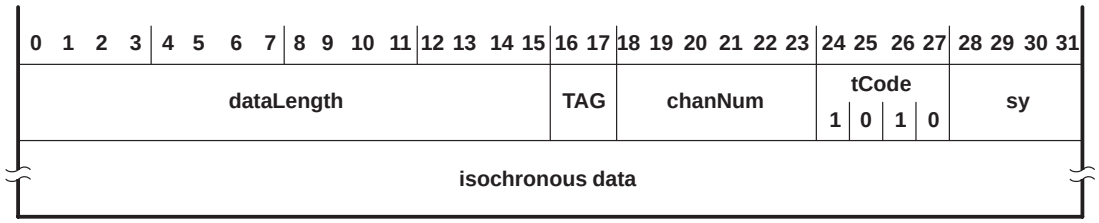


Figure 7–8. Isochronous-Transmit Format

Table 7–5. Isochronous-Transmit Functions

FIELD NAME	DESCRIPTION
dataLength	The dataLength field indicates the number of bytes in the current packet
TAG	The TAG field indicates the format of data carried by the isochronous packet (00 = formatted, 01 – 11 are reserved).
chanNum	The chanNum field carries the channel number with which the current data is associated.
tCode	The transaction code for the current packet (tCode=Ah).
sy	The sy field carries the transaction layer-specific synchronization bits.
isochronous data	The isochronous data field contains the data to be sent with the current packet. The first byte of data must appear in byte 0 of the first quadlet of this field. If the last quadlet does not contain four bytes of data, the unused bytes should be padded with zeros.

7.2.1 Isochronous Receive (TSB12LV32 to Host Bus)

The format of the iscohronous-receive data through the DM is shown in Figure 7–8 and is described in Table 7–6. The data length, which is found in the header of the packet, determines the number of bytes in an isochronous packet. For iscohronous-receive through the FIFO, the last quadlet will be inserted as the first quadlet in the receive data, as shown in Figure 7–9.

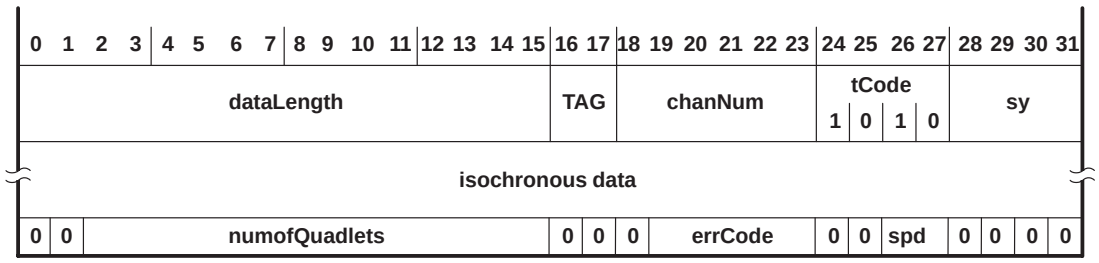


Figure 7–9. Data Mover Isochronous-Receive Format

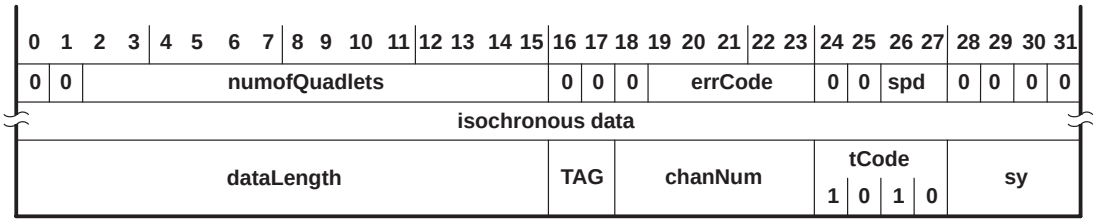


Figure 7–10. GRF Isochronous-Receive Format

Table 7–6. Isochronous-Receive Functions

FIELD NAME	DESCRIPTION
dataLength	The dataLength field indicates the number of bytes in the current packet.
TAG	The TAG field indicates the format of data carried by isochronous packet (00 = formatted, 01 –- 11 are reserved).
chanNum	The chanNum field contains the channel number with which this data is associated.
tCode	The tCode field carries the transaction code for the current packet (tCode = Ah).
sy	The sy field carries the transaction layer-specific synchronization bits.
isochronous data	The isochronous data field has the data to be sent with the current packet. The first byte of data must appear in byte 0 of the first quadlet of this field. The last quadlet should be padded with zeros.
spd	The spd field indicates the speed at which the current packet was sent.
numofQuadlets	Total number of quadlets in the current packet (payload and header quadlets only).
errCode	The errCode field indicates whether the current packet has been received correctly. The possibilities are Complete, DataErr, or CRCErr, and have the same encoding as the corresponding acknowledge codes.

7.3 Phy Configuration

The format of the Phy configuration packet is shown in Figure 7-12 and is described in Table 7-8. The Phy configuration packet transmit contains two quadlets, which are loaded into the ATF. The first quadlet is written to address 50h. The second quadlet is written to address 58h. The 00E0h in the first quadlet (bits 16–31) tells the TSB12LV32 that this quadlet is the Phy configuration packet. The Eh is then replaced with 0h before the packet is transmitted to the Phy interface.

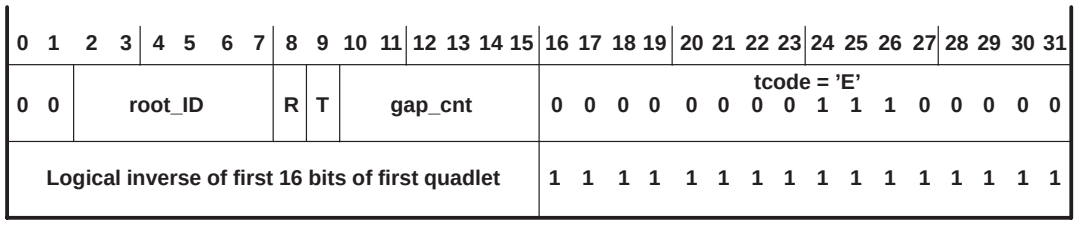


Figure 7–11. Phy Configuration Packet Format

The Phy configuration packet can perform the following functions:

- Set the gap count field of all nodes on the bus to a new value. The gap count, if set intelligently, can optimize bus performance.
- Force a particular node to be the bus root after the next bus reset.

It is not valid to transmit a Phy configuration packet with both the R bit and T bit set to zero. This would cause the packet to be interpreted as an extended Phy packet.

Table 7–7. Phy Configuration Packet Functions

FIELD NAME	DESCRIPTION
00	The 00 field is the Phy configuration packet identifier.
root_ID	The root_ID field is the physical_ID of the node to have its force_root bit set (only meaningful when R is set).
R†	When R is set, the force-root bit of the node identified in root_ID is set and the force_root bit of all other nodes are cleared. When R is cleared, root_ID is ignored.
T	When T is set, the PHY_CONFIGURATION.gap_count field of all the nodes is set to the value in the gap_cnt field.
gap_cnt	The gap_cnt field contains the new value for PHY_CONFIGURATION.gap_count for all nodes. This value goes into effect immediately upon receipt and remains valid after the next bus reset. After the second reset, gap_cnt is set to 63h unless a new Phy configuration packet is received.

The format of a received Phy-configuration packet is shown in Figure 7–12 and is described in Table 7–8. When PHY_PKT_ENA (bit 3 of the control register @08h) is set, all Phy packets will be received in the GRF. One HDRERR interrupt will be generated for every Phy packet received.

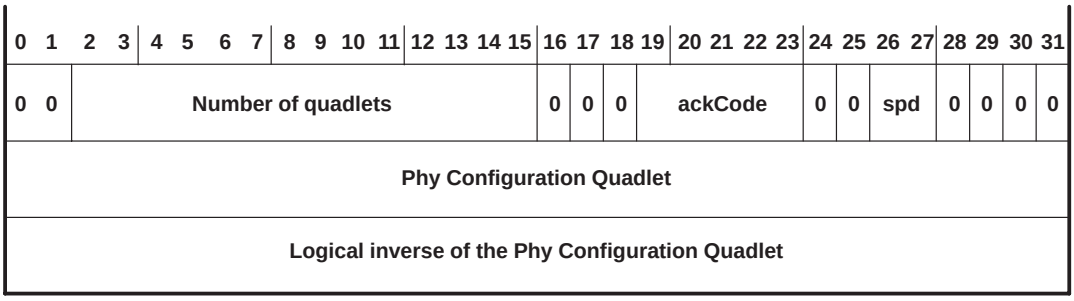


Figure 7–12. Received Phy Configuration Packet Format

Table 7–8. Receive Phy–Configuration Packet

FIELD NAME	DESCRIPTION
numofQuadlets	Total number of quadlets in the packet. This field is equal to 2 in this case.
ackCode	This 5–bit field holds the acknowledge code sent by the receiver for the current packet. In this case, the ackCode is equal to 1 (ack_complete)
spd	The spd field indicates the speed at which the current packet was sent. In this case, the spd field is equal to ‘00’ (S100)

7.3.1 Extended Phy Packets

7.3.1.1 Ping Packets

The reception of a Phy ping packet causes the node identified by Phy_ID to transmit Self-ID packet(s) that reflect the current configuration and status of the Phy. The ping packet provides a method of measuring the round-trip delay of packets between two nodes on the bus that are farthest from one another in terms of cable hops. The format of this packet is shown in Figure 7–13 and described in Table 7–9.

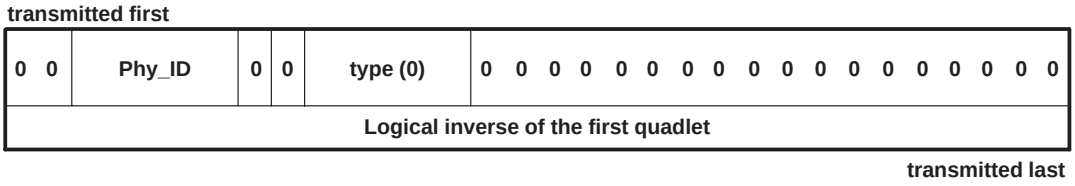


Figure 7–13. Ping Packet Format

Table 7–9. Ping Packet Fields

FIELD NAME	DESCRIPTION
Phy_ID	Physical node identifier of the destination of this packet
type	Extended Phy packet type (zero identifies ping packet)

7.3.1.2 Remote Access Packets

The remote access packet provides a method for a node to access the Phy registers of another node on the bus. The reception of a remote access packet causes the node identified by the Phy_ID field to read the selected Phy register and subsequntly return a remote reply packet that contains the current value of the Phy register. The format of this packet is shown in Figure 7–14 and described in Table 7–10.

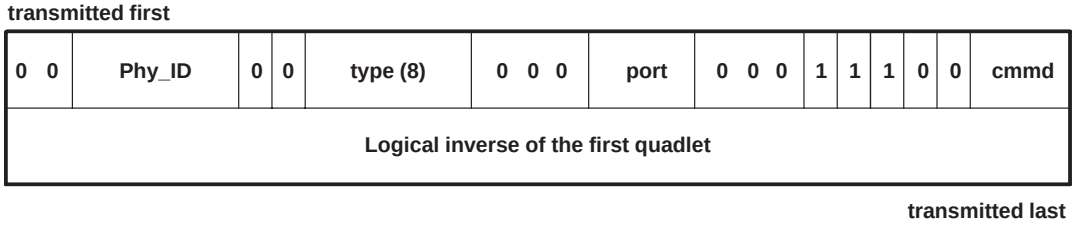


Figure 7–14. Remote Access Packet Format

Table 7–10. Remote Access Packet Fields

FIELD NAME	DESCRIPTION
Phy_ID	Physical node identifier of the destination of this packet
type	Extended Phy packet type (8 identifies command packet)
port	This field selects one of the Phy's ports
cmmd	Command: 0 = NOP (No operation) 1 = Transmit TX_DISABLE_NOTIFY then disable port 2 = Initiate suspend (i.e., become a suspend initiator) 4 = Clear the port's Fault bit to zero 5 = Enable port 6 = Resume port

7.3.1.3 Remote Command Packets

The remote command packet provides a method for one node to issue a number of Phy-specific commands to the selected port within the target Phy. The reception of a remote command packet shall request the node identified by the Phy_ID field to perform the operation specified in the *cmnd* field and subsequently return a remote confirmation packet. The format of this packet is shown in Figure 7–15 and described in Table 7–11.

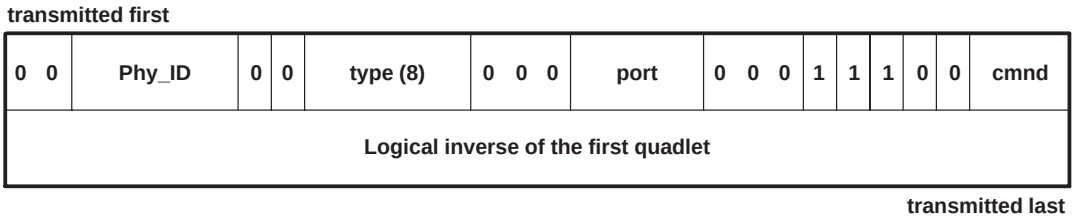


Figure 7–15. Remote Command Packet Format

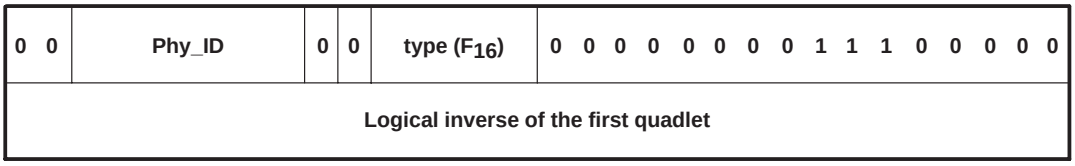
Table 7–11. Remote Command Packet Fields

FIELD NAME	DESCRIPTION
Phy_ID	Physical node identifier of the destination of this packet
type	Extended Phy packet type (8 identifies command packet)
port	This field selects one of the Phy's ports
cmnd	Command: 0 = NOP (No operation) 1 = Transmit TX_DISABLE_NOTIFY then disable port 2 = Initiate suspend (i.e., become a suspend initiator) 4 = Clear the port's Fault bit to zero 5 = Enable port 6 = Resume port

7.3.1.4 Resume Packets

The resume packet is a broadcast packet to all the Phys on the bus. It commands all suspended ports on the bus to resume normal operation. The reception of the resume packet causes any node to commence resume operations for all Phy ports that are both connected and suspended. A resume packet requires no reply. The format of this packet is shown in Figure 7–16 and described in Table 7–12.

transmitted first



transmitted last

Figure 7–16. Resume Packet Format

Table 7–12. Resume Packet Fields

FIELD NAME	DESCRIPTION
Phy_ID	Physical node identifier of the source of this packet
type	Extended Phy packet type (F identifies resume packet)

7.4 Receive Self-ID Packet

Based on the settings of the RXSID and FULLSID bits in the control register @08h, the self-ID packets can be either ignored or received into the GRF. Refer to Table 7–13.

Table 7–13. GRF Receive Self-ID Setup Using Control Register Bits (RXSID and FULLSID)

RXSID (bit 1)	FULLSID (bit 2)	OPERATION
0	X	Self-ID packets are not received by the link.
1	0	Only the data quadlet (first quadlet) of the self-ID packets are received into the GRF.
1	1	soth the data quadlet (first quadlet) and the logical inverse quadlet (second quadlet) of all Self-ID packets are received into the GRF.

Figures 7–17 and 7–18 show the format of a received self-ID packet. For completeness, the figures assume the cable Phy on the bus implements the maximum number of ports allowed by the P1394a specification. Both figures show one received self-ID packet. The contents are described in Table 7–14.

0 1 2 3	4 5 6 7	8 9 10 11	12 13 14 15	16 17 18 19	20 21 22 23	24 25 26 27	28 29 30 31
0 0	numofQuadlets			0 0 0	ackCode	0 0	spd	0 0	LPS_RESET	LPS_OFF
Self-ID Data Quadlet #0							
Logical Inverse of the Self-ID Quadlet #0							
Self-ID Data Quadlet #1							
Logical Inverse of the Self-ID Quadlet #1							
Self-ID Data Quadlet #2							
Logical Inverse of theSelf-ID Quadlet #2							

Figure 7–17. Receive Self-ID Packet Format (RXSID=1, FULLSID=1)

Figure 7–18 shows the format of the received self-ID packet when the FULLSID is cleared. In this case, only the first quadlet of each self-ID packet is received in the GRF.

0 1 2 3	4 5 6 7	8 9 10 11	12 13 14 15	16 17 18 19	20 21 22 23	24 25 26 27	28 29 30 31
0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	1 1 1 0	0 0 0 0
Self-ID Data Quadlet #0							
Self-ID Data Quadlet #1							
Self-ID Data Quadlet #2							

Figure 7–18. Receive Self-ID Packet Format (RXSID=1, FULLSID=0)

Table 7–14. Receive Self-ID Function

FIELD NAME	DESCRIPTION
Self-ID Data Quadlet	First 32-bits of the first self-ID packet
Logical Inverse of the Self-ID Quadlet	Second 32-bits of the first self-ID packet
ACK	When the ACK field is set (0001), the data in the Self-ID packet is correct. When ACK is ≠ 0001, the data in the self-ID packet is incorrect.

The cable Phy sends one to three self-ID packets at the base rate (100 Mbits/s) during the self-ID phase of arbitration or in response to a ping packet. The number of self-ID packets sent depends on the number of ports. Figures 7–19, 7–20, and 7–21 show the format of the cable Phy self-ID packets. Inside the GRF, the first received quadlet of a self-ID packet is always 0000_00E0h, and the final quadlet is always the quadlet containing the acknowledgement code.

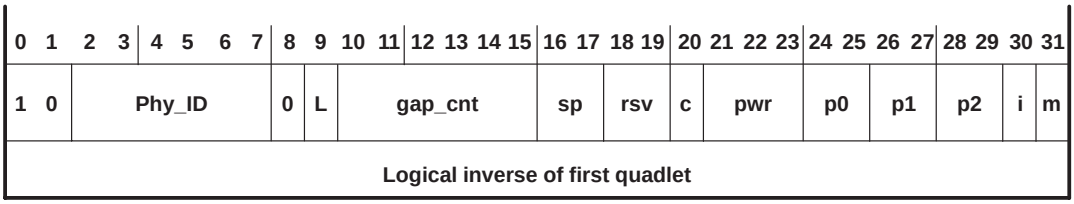


Figure 7–19. Phy Self-ID Packet #0 Format

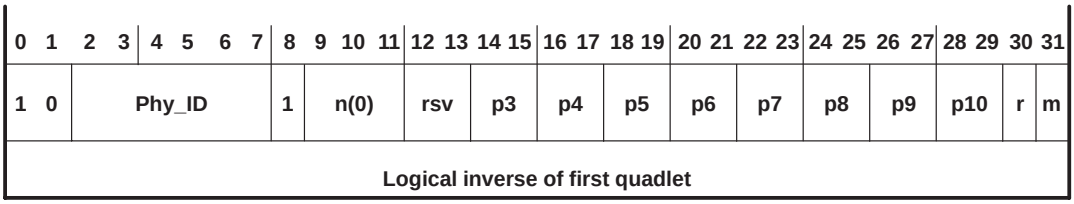


Figure 7–20. Phy Self-ID Packet #1 Format

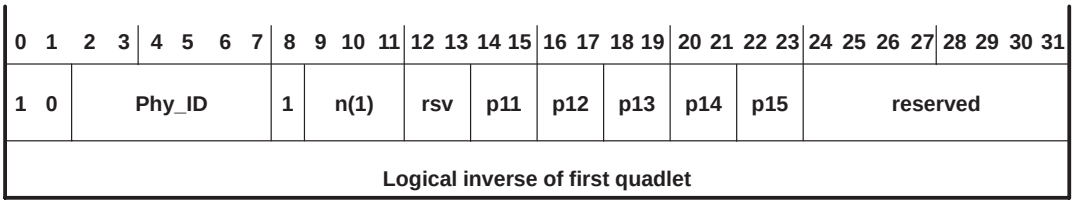


Figure 7–21. Phy Self-ID Packet #2 Format

When there is only one node (i.e., one Phy/link pair) on the bus, following a bus reset, the GRF contains 0000_00E0h and the acknowledge quadlet only.

Example: If there are three 1394.a compliant nodes on the bus, each with a Phy containing three or less ports, the GRF of any one of the links is shown below. The FULLSID bit is assumed to be set in this example.

GRF CONTENTS	DESCRIPTION
0000_00E0h	Header quadlet for Self-ID Phy packet
Self-ID1	Self_ID quadlet for Phy #1
Self-ID1 (inverse)	Logical inverse quadlet for Self_ID of Phy #1
Self-ID2	Self_ID quadlet for Phy #2
Self-ID2 (inverse)	Logical inverse quadlet for Self_ID of Phy #2
Self-ID3	Self_ID quadlet for Phy #3
Self-ID3 (inverse)	Logical inverse quadlet for Self_ID of Phy #3
0000_000_ACK	Trailing acknowledgement quadlet

GRF contents (following a bus reset) with three nodes on the bus

Table 7–15. Phy Self-ID Packet Fields

FIELD NAME	DESCRIPTION
10	The 10 field is the self-ID packet identifier.
L	If set, this node has an active link and transaction layers. In discrete Phy implementations, this shall be the logical AND of Link_active and LPS active.
gap_cnt	The gap_cnt field contains the current value for the current node PHY_CONFIGURATION.gap_count field.
sp	The sp field contains the Phy speed capability. The code is: 00 98.304 Mbits/s 01 98.304 Mbits/s and 196.608 Mbits/s 10 98.304 Mbits/s 196.608 Mbits/s, and 393.216 Mbits/s 11 Extended speed capabilities reported in Phy register 3
c	If set and the link_active flag is set, this node is contender for the bus or isochronous resource manager as described in clause 8.4.2 of IEEE Std 1394–1995.
pwr	Power consumption and source characteristics: 000 Node does not need power and does not repeat power. 001 Node is self-powered and provides a minimum of 15W to the bus. 010 Node is self-powered and provides a minimum of 30W to the bus. 011 Node is self-powered and provides a minimum of 45W to the bus. 100 Node may be powered from the bus and is using up to 3W. No additional power is needed to enable the link‡. 101 Reserved for future standaraization. 110 Node is powered from the bus and is using up to 3W. An additional 3W is needed to enable the link‡. 111 Node is powered from the bus and is using up to 3W. An additional 7W is needed to enable the link‡.
p0 – p15	The p0 – p15 field indicates the port connection status. The code is: 00 Not present on the current Phy 01 Not connected to any other Phy 10 Connected to the parent node 11 Connected to the child node
i	If set, this node initiated the current bus reset (i.e., it started sending a bus_reset signal before it received one)†.
m	If set, another self-ID packet for this node will immediately follow (i.e., if this bit is set and the next Self-ID packet received has a different Phy_ID, the a self-ID packet was lost)
n	Extended self-ID packet sequence number
rsv	Reserved and set to all zeros

† There is no way to ensure that exactly one node has this bit set. More than one node can be requesting a bus reset at the same time.

‡ The link is enabled by the link-on Phy packet described in clause 7.5.2 of the IEEE 1394.a spec.; this packet may also enable application layers.

8 TSB12LV32/Phy Interface

This section provides an overview of the digital interface between a TSB12LV32 and a physical layer device (Phy). The information that follows can be used as a guide through the process of connecting the TSB12LV32 to a 1394 Phy. The part numbers referenced, the TSB41LV03A and the TSB12LV32, represent the Texas Instruments implementation of the Phy (TSB41LV03A) and link (TSB12LV32) layers of the IEEE 1394-1995 and P1394a standards.

The specific details of how the TSB41LV03A device operates are not discussed in this document. Only those parts that relate to the TSB12LV32 Phy interface are mentioned.

8.1 Principles of Operation

The TSB12LV32 is designed to operate with a Texas Instruments physical-layer device. The following paragraphs describe the operation of the Phy-LLC interface assuming a TSB41LV03A Phy. The TSB41LV03A is an IEEE 1394a three port cable transceiver/arbiter Phy capable of 400 Mbits/s speeds.

The interface to the Phy consists of the SCLK, CTL0–CTL1, D0–D7, LREQ, LPS, LINKON, and $\overline{\text{DIRECT}}$ terminals on the TSB12LV32, as shown in Figure 8–1. Refer to Texas Instruments *Application Report SLLA044* for a detailed description of the electrical interface between the TSB12LV32 and TSB41LV03.

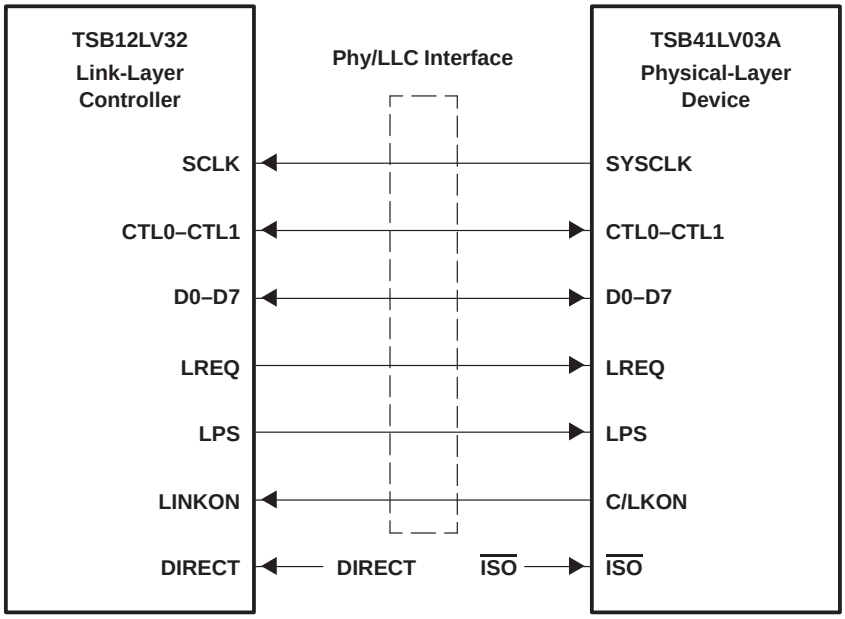


Figure 8–1. Phy-LLC Interface

The SYSCLK from the Phy terminal provides a 49.152 MHz interface clock. All control and data signals are synchronized to, and sampled on, the rising edge of SYSCLK.

The CTL0 and CTL1 terminals form a bidirectional control bus, which controls the flow of information and data between the TSB41LV03A and TSB12LV32.

The D0–D7 terminals form a bidirectional data bus, which is used to transfer status information, control information, or packet data between the devices. The TSB41LV03A supports S100, S200, and S400 data transfers over the D0–D7 data bus. In S100 operation only the D0 and D1 terminals are used; in S200 operation only the D0–D3 terminals are used; and in S400 operation all D0–D7 terminals are used for data

transfer. When the TSB41LV03A is in control of the D0–D7 bus, unused Dn terminals are driven low during S100 and S200 operations. When the TSB12LV32 is in control of the D0–D7 bus, unused Dn terminals are ignored by the TSB41LV03A.

The LREQ terminal is controlled by the TSB12LV32 to send serial service requests to the Phy in order to request access to the serial-bus for packet transmission, read or write Phy registers, or control arbitration acceleration.

The LPS and LINKON terminals are used for power management of the Phy and TSB12LV32. The LPS terminal indicates the power status of the TSB12LV32, and may be used to reset the Phy-LLC interface or to disable SYSCLK. The C/LKON terminal is used to send a wake-up notification to the TSB12LV32 and to indicate an interrupt to the TSB12LV32 when either LPS is inactive or the Phy register LCtrl bit is zero.

The $\overline{\text{DIRECT}}$ and $\overline{\text{ISO}}$ terminals are used to enable the output differentiation logic on the CTL0–CTL1 and D0–D7 terminals. Output differentiation is required when an Annex J type isolation barrier is implemented between the Phy and TSB12LV32.

The TSB41LV03A normally controls the CTL0–CTL1 and D0–D7 bidirectional buses. The TSB12LV32 is allowed to drive these buses only after the TSB12LV32 has been granted permission to do so by the Phy. There are four operations that may occur on the Phy-LLC interface: link service request, status transfer, data transmit, and data receive. The TSB12LV32 issues a service request to read or write a Phy register, to request the Phy to gain control of the serial-bus in order to transmit a packet, or to control arbitration acceleration.

The Phy may initiate a status transfer either autonomously or in response to a register read request from the TSB12LV32. The Phy initiates a receive operation whenever a packet is received from the serial-bus. The Phy initiates a transmit operation after winning control of the serial-bus following a bus-request by the TSB12LV32. The transmit operation is initiated when the Phy grants control of the interface to the TSB12LV32.

The encoding of the CTL0–CTL1 bus is shown in Table 8–1 and Table 8–2.

Table 8–1. CTL Encoding When the Phy Has Control of the Bus

CTL0	CTL1	NAME	DESCRIPTION
0	0	Idle	No activity (this is the default mode)
0	1	Status	Status information is being sent from the Phy to the TSB12LV32.
1	0	Receive	An incoming packet is being sent from the Phy to the TSB12LV32.
1	1	Grant	The TSB12LV32 has been given control of the bus to send an outgoing packet.

Table 8–2. CTL Encoding When the TSB12LV32 Has Control of the Bus

CTL0	CTL1	NAME	DESCRIPTION
0	0	Idle	The TSB12LV32 releases the bus (transmission has been completed)
0	1	Hold	The TSB12LV32 is holding the bus while data is being prepared for transmission, or indicating that another packet is to be transmitted (concatenated) without arbitrating
1	0	Transmit	An outgoing packet is being sent from the TSB12LV32 to the Phy.
1	1	Reserved	Reserved

8.2 TSB12LV32 Service Request

To request access to the bus, to read or write a Phy register, or to control arbitration acceleration, the TSB12LV32 sends a serial bit stream on the LREQ terminal as shown in Figure 8–2.

NOTE: Each cell represents one clock sample time, and n is the number of bits in the request stream.

Figure 8–2. LREQ Request Stream

The length of the stream will vary depending on the type of request as shown in Table 8–3.

Table 8–3. Request Stream Bit Length

REQUEST TYPE	NUMBER OF BITS
Bus Request	7 or 8
Read Register Request	9
Write Register Request	17
Acceleration Control Request	6

Regardless of the type of request, a start-bit of 1 is required at the beginning of the stream, and a stop-bit of 0 is required at the end of the stream. The second through fourth bits of the request stream indicate the type of the request. In the descriptions below, bit 0 is the most significant and is transmitted first in the request bit stream. The LREQ terminal is normally low.

Encoding for the request type is shown in Table 8–4.

Table 8–4. Request Type Encoding

LR1–LR3	NAME	DESCRIPTION
000	ImmReq	Immediate bus request. Upon detection of idle, the Phy takes control of the bus immediately without arbitration.
001	IsoReq	Isochronous bus request. Upon detection of idle, the Phy arbitrates for the bus without waiting for a subaction gap.
010	PriReq	Priority bus request. The Phy arbitrates for the bus after a subaction gap, ignores the fair protocol.
011	FairReq	Fair bus request. The Phy arbitrates for the bus after a subaction gap, follows the fair protocol.
100	RdReg	The Phy returns the specified register contents through a status transfer.
101	WrReg	Write to the specified register.
110	AccelCtl	Enable or disable asynchronous arbitration acceleration.
111	Reserved	Reserved.

For a bus request the length of the LREQ bit stream is 7 or 8 bits as shown in Table 8–5.

Table 8–5. Bus Request

BIT(S)	NAME	DESCRIPTION
0	Start Bit	Indicates the beginning of the transfer (always 1).
1–3	Request Type	Indicates the type of bus request (see Table 8–4).
4–6	Request Speed	Indicates the speed at which the Phy will send the data for this request (see Table 8–6) for the encoding of this field.
7	Stop Bit	Indicates the end of the transfer (always 0). If bit 6 is 0, this bit may be omitted.

The 3-bit request speed field used in bus requests is shown in Table 8–6.

Table 8–6. Bus Request Speed Encoding

LR4–LR6	DATA RATE
000	S100
010	S200
100	S400
All Others	Invalid

NOTE:

The TSB41LV03A will accept a bus request with an invalid speed code and process the bus request normally. However, during packet transmission for such a request, the TSB41LV03A will ignore any data presented by the TSB12LV32 and will transmit a null packet.

For a read register request the length of the LREQ bit stream is 9 bits as shown in Table 8–7.

Table 8–7. Read Register Request

BIT(S)	NAME	DESCRIPTION
0	Start Bit	Indicates the beginning of the transfer (always 1).
1–3	Request Type	A 100 indicating this is a read register request.
4–7	Address	Identifies the address of the Phy register to be read.
8	Stop Bit	Indicates the end of the transfer (always 0).

For a write register request the length of the LREQ bit stream is 17 bits as shown in Table 8–8.

Table 8–8. Write Register Request

BIT(S)	NAME	DESCRIPTION
0	Start Bit	Indicates the beginning of the transfer (always 1).
1–3	Request Type	A 101 indicating this is a write register request.
4–7	Address	Identifies the address of the Phy register to be written to.
8–15	Data	Gives the data that is to be written to the specified register address.
16	Stop Bit	Indicates the end of the transfer (always 0).

For an acceleration control request the length of the LREQ bit stream is 6 bits as shown in Table 8–9.

Table 8–9. Acceleration Control Request

BIT(S)	NAME	DESCRIPTION
0	Start Bit	Indicates the beginning of the transfer (always 1).
1–3	Request Type	A 110 indicating this is a acceleration control request.
4	Control	Asynchronous period arbitration acceleration is enabled if 1, and disabled if 0.
5	Stop Bit	Indicates the end of the transfer (always 0).

For fair or priority access, the TSB12LV32 sends the bus request (FairReq or PriReq) at least one clock after the Phy-LLC interface becomes idle. If the CTL terminals are asserted to the receive state (’b10) by the Phy, then any pending fair or priority request is lost (cleared). Additionally, the Phy ignores any fair or priority requests if the receive state is asserted while the TSB12LV32 is sending the request. The TSB12LV32 may then reissue the request one clock after the next interface idle.

The cycle master node uses a priority bus request (PriReq) to send a cycle start message. After receiving or transmitting a cycle start message, the TSB12LV32 can issue an isochronous bus request (IsoReq). The Phy will clear an isochronous request only when the serial bus has been won.

To send an acknowledge packet, the TSB12LV32 must issue an immediate bus request (ImmReq) during the reception of the packet addressed to it. This is required in order to minimize the idle gap between the end of the received packet and the start of the transmitted acknowledge packet. As soon as the receive packet ends, the Phy immediately grants control of the bus to the TSB12LV32. The TSB12LV32 sends an acknowledgment to the sender unless the header CRC of the received packet is corrupted. In this case, the TSB12LV32 does not transmit an acknowledge, but instead cancels the transmit operation and releases the interface immediately; the TSB12LV32 must not use this grant to send another type of packet. After the interface is released the TSB12LV32 may proceed with another request.

The TSB12LV32 may make only one bus request at a time. Once the TSB12LV32 issues any request for bus access (ImmReq, IsoReq, FairReq, or PriReq), it cannot issue another bus request until the Phy indicates that the bus request was lost (bus arbitration lost and another packet received), or won (bus arbitration won and the TSB12LV32 granted control). The Phy ignores new bus requests while a previous bus request is pending. All bus requests are cleared upon a bus reset.

For write register requests, the Phy loads the specified data into the addressed register as soon as the request transfer is complete. For read register requests, the Phy returns the contents of the addressed register to the TSB12LV32 at the next opportunity through a status transfer. If a received packet interrupts the status transfer, then the Phy continues to attempt the transfer of the requested register until it is successful. A write or read register request may be made at any time, including while a bus request is pending. Once a read register request is made, the Phy ignores further read register requests until the register contents are successfully transferred to the TSB12LV32. A bus reset does not clear a pending read register request.

The TSB41LV03A includes several arbitration acceleration enhancements, which allow the Phy to improve bus performance and throughput by reducing the number and length of interpacket gaps. These enhancements include autonomous (fly-by) isochronous packet concatenation, autonomous fair and priority packet concatenation onto acknowledge packets, and accelerated fair and priority request arbitration following acknowledge packets. The enhancements are enabled when the EAA bit in Phy register 5 is set.

The arbitration acceleration enhancements may interfere with the ability of the cycle master node to transmit the cycle start message under certain circumstances. The acceleration control request is therefore provided to allow the TSB12LV32 to temporarily enable or disable the arbitration acceleration enhancements of the TSB41LV03A during the asynchronous period. The TSB12LV32 typically disables the enhancements when its internal cycle counter rolls over indicating that a cycle start message is imminent, and then re-enables the enhancements when it receives a cycle start message. The acceleration control request may be made at any time, however, and is immediately serviced by the Phy. Additionally, a bus reset or isochronous bus request will cause the enhancements to be re-enabled, if the EAA bit is set.

8.3 Status Transfer

A status transfer is initiated by the Phy when there is status information to be transferred to the TSB12LV32. The Phy waits until the interface is idle before starting the transfer. The transfer is initiated by the Phy asserting status (‘b01) on the CTL terminals, along with the first two bits of status information on the D[0:1] terminals. The Phy maintains CTL = status for the duration of the status transfer. The Phy may prematurely end a status transfer by asserting something other than status on the CTL terminals. This occurs if a packet is received before the status transfer completes. The Phy continues to attempt to complete the transfer until all status information has been successfully transmitted. There is at least one idle cycle between consecutive status transfers.

The Phy normally sends just the first four bits of status to the TSB12LV32. These bits are status flags that are needed by the TSB12LV32 state machines. The Phy sends an entire 16-bit status packet to the TSB12LV32 after a read register request, or when the Phy has pertinent information to send to the TSB12LV32 or transaction layers. The only defined condition where the Phy automatically sends a register to the TSB12LV32 is after self-ID, where the Phy sends the physical-ID register that contains the new node

address. All status transfers are either 4 or 16 bits unless interrupted by a received packet. The status flags are considered to have been successfully transmitted to the TSB12LV32 immediately upon being sent, even if a received packet subsequently interrupts the status transfer. Register contents are considered to have been successfully transmitted only when all 8 bits of the register have been sent. A status transfer is retried after being interrupted only if any status flags remain to be sent, or if a register transfer has not yet completed.

The definition of the bits in the status transfer are shown in Table 8–9 and the timing is shown in Figure 8–3.

Table 8–10. Status Bits

BIT(S)	NAME	DESCRIPTION
0	Arbitration Reset Gap	Indicates that the Phy has detected that the bus has been idle for an arbitration reset gap time (as defined in IEEE Std 1394–1995). This bit is used by the TSB12LV32 in the busy/retry state machine.
1	Subaction Gap	Indicates that the Phy has detected that the bus has been idle for a subaction gap time (as defined in IEEE Std 1394–1995). This bit is used by the TSB12LV32 to detect the completion of an isochronous cycle.
2	Bus Reset	Indicates that the Phy has entered the bus reset start state.
3	Interrupt	Indicates that a Phy interrupt event has occurred. An interrupt event may be a configuration time-out, cable-power voltage falling too low, a state time-out, or a port status change.
4–7	Address	This field holds the address of the Phy register whose contents are being transferred to the TSB12LV32.
8–15	Data	This field holds the register contents.

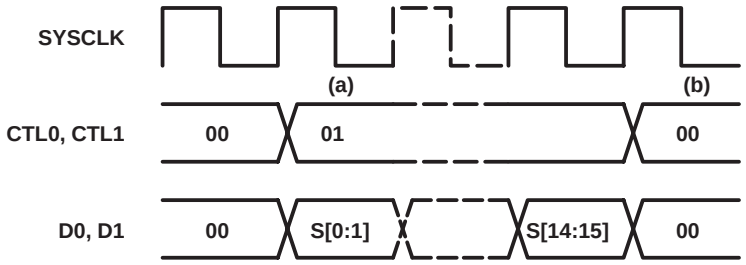


Figure 8–3. Status Transfer Timing

The sequence of events for a status transfer is as follows:

- Status transfer initiated. The Phy indicates a status transfer by asserting status on the CTL lines along with the status data on the D0 and D1 lines (only 2 bits of status are transferred per cycle). Normally (unless interrupted by a receive operation), a status transfer will be either 2 or 8 cycles long. A 2-cycle (4-bit) transfer occurs when only status information is to be sent. An 8-cycle (16-bit) transfer occurs when register data is to be sent in addition to any status information.
- Status transfer terminated. The Phy normally terminates a status transfer by asserting idle on the CTL lines. The Phy may also interrupt a status transfer at any cycle by asserting receive on the CTL lines to begin a receive operation. The Phy shall assert at least one cycle of idle between consecutive status transfers.

8.4 Receive Operation

Whenever the Phy detects the data-prefix state on the serial bus, it initiates a receive operation by asserting Receive on the CTL terminals and a logic 1 on each of the D terminals (data-on indication). The Phy indicates the start of a packet by placing the speed code (encoded as shown in Table 8–11 on the D terminals, followed by packet data. The Phy holds the CTL terminals in the receive state until the last symbol of the packet has

been transferred. The Phy indicates the end of packet data by asserting idle on the CTL terminals. All received packets are transferred to the TSB12LV32. Note that the speed code is part of the Phy-LLC protocol and is not included in the calculation of CRC or any other data protection mechanisms.

It is possible for the Phy to receive a null packet, which consists of the data-prefix state on the serial bus followed by the data-end state, without any packet data. A null packet is transmitted whenever the packet speed exceeds the capability of the receiving Phy, or whenever the TSB12LV32 immediately releases the bus without transmitting any data. In this case, the Phy will assert receive on the CTL terminals with the data-on indication (all 1s) on the D terminals, followed by idle on the CTL terminals, without any speed code or data being transferred. In all cases, the TSB41LV03A sends at least one data-on indication before sending the speed code or terminating the receive operation.

The TSB41LV03A also transfers its own self-ID packet, transmitted during the self-ID phase of bus initialization, to the TSB12LV32. This packet it transferred to the TSB12LV32 just as any other received self-ID packet.

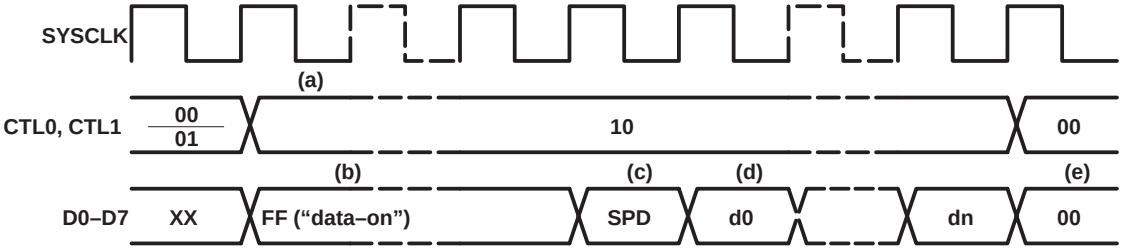


Figure 8–4. Normal Packet Reception Timing

The sequence of events for a normal packet reception is as follows:

- Receive operation initiated. The Phy indicates a receive operation by asserting receive on the CTL lines. Normally, the interface is idle when receive is asserted. However, the receive operation may interrupt a status transfer operation that is in progress so that the CTL lines may change from status to receive without an intervening idle.
- Data-on indication. The Phy asserts the data-on indication code on the D lines for one or more cycles preceding the speed-code.
- Speed-code. The Phy indicates the speed of the received packet by asserting a speed-code on the D lines for one cycle immediately preceding packet data. The link decodes the speed-code on the first receive cycle for which the D lines are not the data-on code. If the speed-code is invalid, or indicates a speed higher than that which the link is capable of handling, the link should ignore the subsequent data.
- Receive data. Following the data-on indication (if any) and the speed-code, the Phy asserts packet data on the D lines with receive on the CTL lines for the remainder of the receive operation.
- Receive operation terminated. The Phy terminates the receive operation by asserting idle on the CTL lines. The Phy asserts at least one cycle of idle following a receive operation.

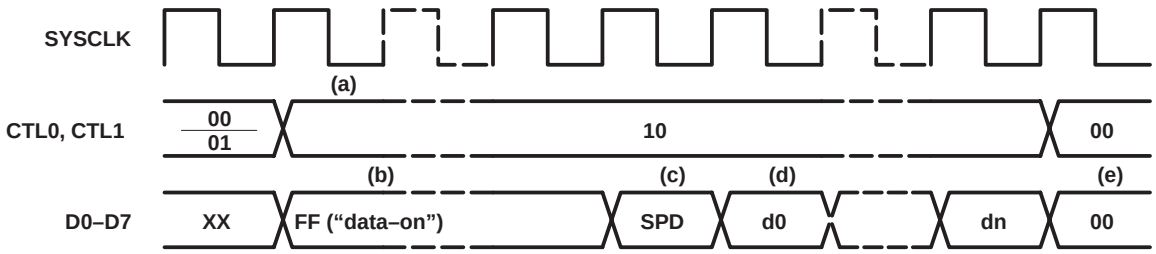


Figure 8–5. Null Packet Reception Timing

The sequence of events for a null packet reception is as follows:

- Receive operation initiated. The Phy indicates a receive operation by asserting receive on the CTL lines. Normally, the interface is idle when receive is asserted. However, the receive operation may interrupt a status transfer operation that is in progress so that the CTL lines may change from status to receive without an intervening idle.
- Data-on indication. The Phy asserts the data-on indication code on the D lines for one or more cycles.
- Receive operation terminated. The Phy terminates the receive operation by asserting idle on the CTL lines. The Phy asserts at least one cycle of idle following a receive operation.

Table 8–11. Receive Speed Codes

D0–D7	DATA RATE
00XX XXXX	S100
0100 XXXX	S200
0101 0000	S400
1YYY YYYY	Data-on indication

NOTE: X = Output as 0 by Phy, ignored by TSB12LV32.
Y = Output as 1 by Phy, ignored by TSB12LV3

8.5 Transmit Operation

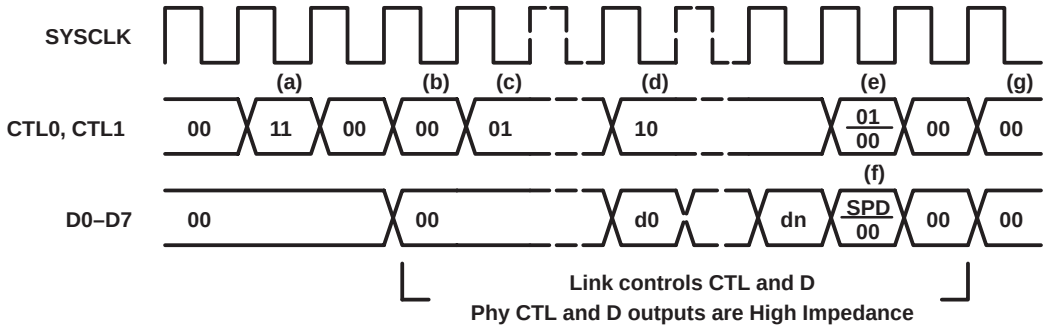
When the TSB12LV32 issues a bus request through the LREQ terminal, the Phy arbitrates to gain control of the bus. If the Phy wins arbitration for the serial bus, the Phy-LLC interface bus is granted to the TSB12LV32 by asserting the grant state (’b11) on the CTL terminals for one SYSCLK cycle, followed by idle for one clock cycle. The TSB12LV32 then takes control of the bus by asserting either idle (’b00), hold (’b01) or transmit (’b10) on the CTL terminals. Unless the TSB12LV32 is immediately releasing the interface, the TSB12LV32 may assert the idle state for at most one clock before it must assert either hold or transmit on the CTL terminals. The hold state is used by the TSB12LV32 to retain control of the bus while it prepares data for transmission. The TSB12LV32 may assert hold for zero or more clock cycles (i.e., the TSB12LV32 need not assert hold before transmit). The Phy asserts data-prefix on the serial bus during this time.

When the TSB12LV32 is ready to send data, the TSB12LV32 asserts transmit on the CTL terminals as well as sending the first bits of packet data on the D lines. The transmit state is held on the CTL terminals until the last bits of data have been sent. The TSB12LV32 then asserts either hold or idle on the CTL terminals for one clock cycle, and then asserts idle for one additional cycle before releasing the interface bus and placing its CTL and D terminals in high impedance. The Phy then regains control of the interface bus.

The hold state asserted at the end of packet transmission indicates to the Phy that the TSB12LV32 requests to send another packet (concatenated packet) without releasing the serial bus. The Phy responds to this concatenation request by waiting the required minimum packet separation time and then asserting grant as before. This function may be used to send a unified response after sending an acknowledge, or to send

consecutive isochronous packets during a single isochronous period. Unless multispeed concatenation is enabled, all packets transmitted during a single bus ownership must be of the same speed (since the speed of the packet is set before the first packet). If multispeed concatenation is enabled (when the EMSC bit of Phy register 5 is set), the TSB12LV32 must specify the speed code of the next concatenated packet on the D terminals when it asserts hold on the CTL terminals at the end of a packet. The encoding for this speed code is the same as the speed code that precedes received packet data as given in Table 8–11.

After sending the last packet for the current bus ownership, the TSB12LV32 releases the bus by asserting Idle on the CTL terminals for two clock cycles. The Phy begins asserting Idle on the CTL terminals one clock after sampling Idle from the link. Note that whenever the D and CTL terminals change direction between the Phy and the TSB12LV32, there is an extra clock period allowed so that both sides of the interface can operate on registered versions of the interface signals.



NOTE: SPD = Speed code, see Table 8–11, d0–dn = Packet data

Figure 8–6. Normal Packet Transmission Timing

The sequence of events for a normal packet transmission is as follows:

- Transmit operation initiated. The Phy asserts grant on the CTL lines followed by Idle to hand over control of the interface to the link so that the link may transmit a packet. The Phy releases control of the interface (i.e., it places its CTL and D outputs in a high-impedance state) following the idle cycle.
- Optional idle cycle. The link may assert at most one Idle cycle preceding assertion of either hold or transmit. This idle cycle is optional; the link is not required to assert Idle preceding either hold or transmit.
- Optional hold cycles. The link may assert hold for up to 47 cycles preceding assertion of transmit. These hold cycle(s) are optional; the link is not required to assert hold preceding transmit.

8.6 TSB12LV32/Phy Interface Critical Timing

PARAMETER†	PIN NAME(S)	MIN	MAX	UNIT
td0, delay time (SCLK to Q)	LREQ	3	21	ns
td1, delay time (SCLK to Q)	CTL[0:1]	3	21	ns
td2, delay time (SCLK to Q)	D[0:7]	3.5	21	ns
tsu0, setup time to SCLK	CYCLEIN	2		ns
tsu1, setup time to SCLK	CONTNDR	3		ns
tsu2, setup time to SCLK	CTL[0:1]	3		ns
tsu3, setup time to SCLK	D[0:7]	3		ns
th0, hold time from SCLK	CYCLEIN	2		ns
th1, hold time from SCLK	CONTNDR	2		ns
th2, hold time from SCLK	CTL[0:1]	0		ns
th3, hold time from SCLK	D[0:7]	0		ns

† All timing parameters are referenced to the rising edge of SCLK on the TSB12LV32 side.

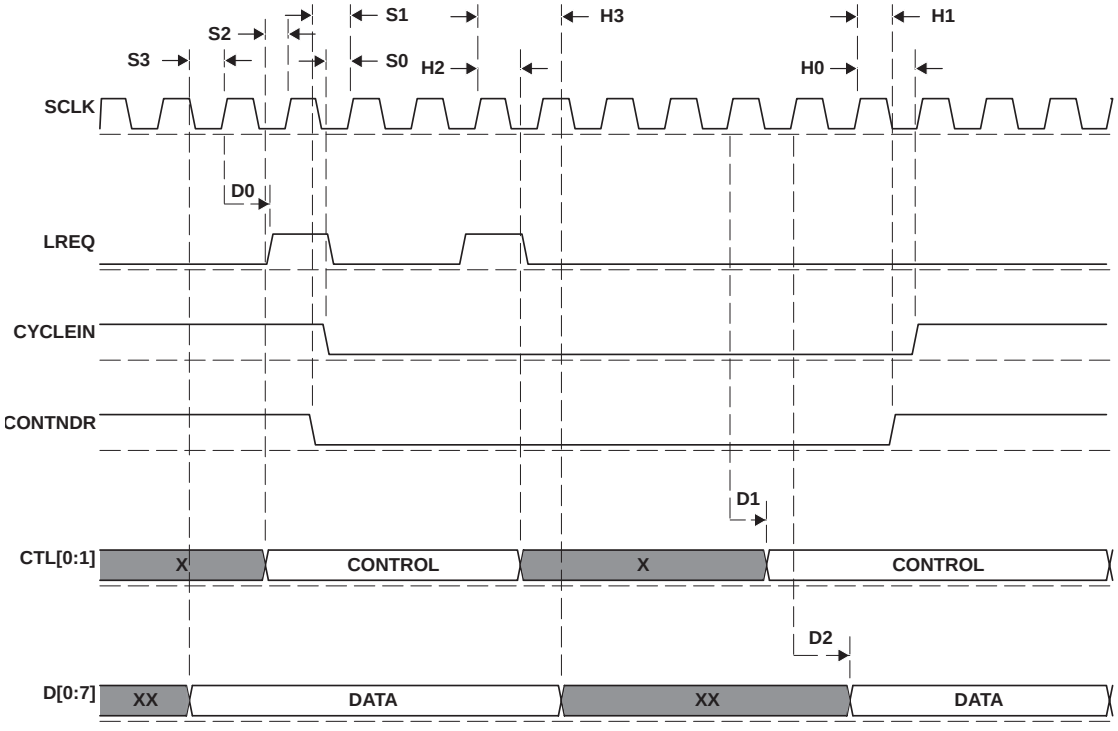


Figure 8–7. Critical Timing for the TSB12LV32/Phy Interface

9 Electrical Characteristics

9.1 Absolute Maximum Ratings Over Operating Free-Air Temperature Range (Unless Otherwise Noted)†

Supply voltage range, V_{CC} . . . . . –0.5 V to 3.6 V
Supply voltage range, $V_{CC}5V$ . . . . . –0.5 V to 5.5 V
Input voltage range, V_I . . . . . –0.5 V to $V_{CC}5V$ + 0.5 V
Output voltage range, V_O . . . . . –0.5 V to $V_{CC}5V$ + 0.5 V
Input clamp current, I_{IK} (V_I < 0 or V_I > V_{CC} . . . . . ±20 mA
Output clamp current, I_{OK} (V_O < 0 or V_O > V_{CC} . . . . . ±20 mA
Operating free-air temperature range: TSB12LV32 . . . . . 0°C to 70°C
TSB12LV32I . . . . . –40°C to 85°C
Storage temperature range . . . . . –65°C to 150°C

† Stresses beyond those listed under “absolute maximum ratings” may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under “recommended operating conditions” is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTES: 1. This applies to external input and bidirectional buffers. For 5-V tolerant terminals, use $V_I > V_{CC}5V$.
2. This applies to external output and bidirectional buffers. For 5-V tolerant terminals, use $V_I > V_{CC}5V$.

DISSIPATION RATING TABLE

PACKAGE	$T_A \le 25°C$ POWER RATING	DERATING FACTOR ABOVE $T_A = 25°C$	$T_A = 70°C$ POWER RATING	$T_A = 85°C$ POWER RATING
PZ	1500 mW	16.9 mW/°C	737 mW	482 mW

9.2 Recommended Operating Conditions

		MIN	NOM	MAX	UNIT
Supply voltage, V_{CC}		3	3.3	3.6	V
Supply voltage, $V_{CC}5V$		3	4.5	5.5	V
Input voltage, V_I		0		$V_{CC}5V$	V
Output voltage, V_O†		0		V_{CC}	V
High–level input voltage, V_{IH}	TSB12LV32, $\overline{\text{RESET}}$	2.6		$V_{CC}5V$	V
	TSB12LV32, other inputs	2.0		$V_{CC}5V$	
	TSB12LV32I	2.2		$V_{CC}5V$	
Low–level input voltage, V_{IL}	TSB12LV32	0		0.8	V
	TSB12LV32I	0		0.7	
Input transition time, t_f and t_r (10% to 90%)		0		25	ns
Operating free-air temperature, T_A	TSB12LV32	0	25	70	°C
	TSB12LV32I	–40	25	85	
Virtual junction temperature, T_{JC}‡		0	25	115	°C

† This applies to external output buffers.
‡ The junction temperatures listed reflect simulation conditions. The absolute maximum junction temperature is 150°C. The customer is responsible for verifying the junction temperature.

9.3 Electrical Characteristics Over Recommended Ranges of Supply Voltage and Operating Free-Air Temperature (Unless Otherwise Noted)

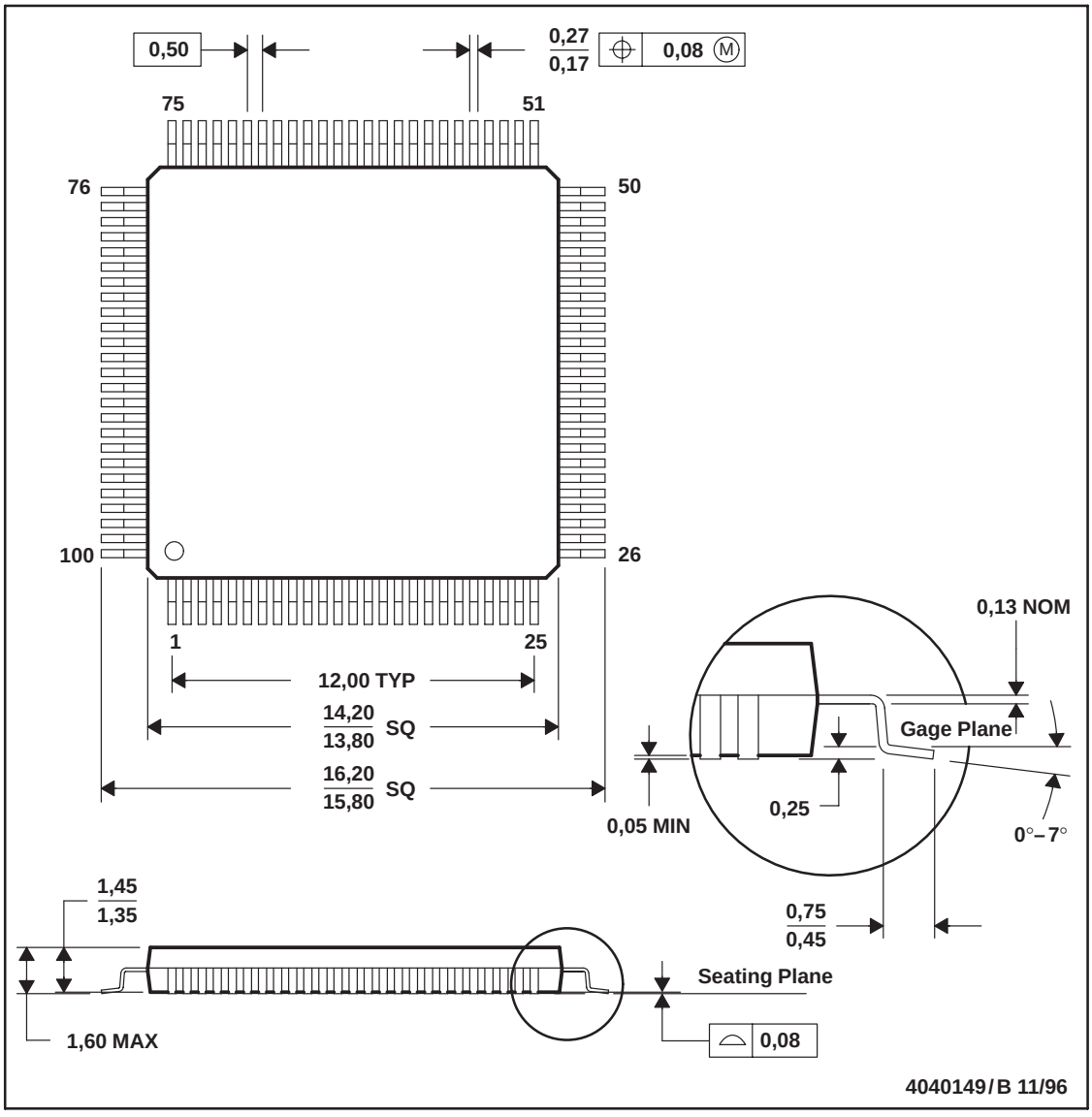
PARAMETER		TEST CONDITIONS	MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage	I_{OH} = –8 mA	V_{CC}–0.6			V
V_{OL}	Low-level output voltage	I_{OL} = 8 mA			0.5	V
I_{IL}	Low-level input current	$V_I = V_{IL}$			–20	µA
I_{IH}	High-level input current	$V_I = V_{IH}$			20	µA
I_{OZ}	High-impedance output current	$V_O = V_{CC}$ or GND			±20	µA
$I_{CC(Q)}$	Static supply current	I_O = 0		10		µA

† All typical characteristics are measured at V_{CC} = 3.3 V and T_A = 25°C.

10 Mechanical Information

The TSB12LV32 is packaged in a high-performance 100-pin PZ package. The following shows the mechanical dimensions of the PZ package.

PZ (S-PQFP-G100) **PLASTIC QUAD FLATPACK**



NOTES: A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Falls within JEDEC MS-026

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its semiconductor products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Customers are responsible for their applications using TI components.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty or endorsement thereof.