CX28500
Multichannel Synchronous Communications Controller

Data Sheet

500052D
November 2002

MINDSPEED™

Ordering Information

Model Number	Package	Operating Temperature
CX28500EBG	35 mm TBGA	–40–85 °C

Revision History

Revision	Level	Date	Description
A	—	July 2001	Created.
B	—	February 2002	Updated to Revision B.
C	—	July 2002	Updated to Revision C.
D	—	November 2002	Updated to Revision D.

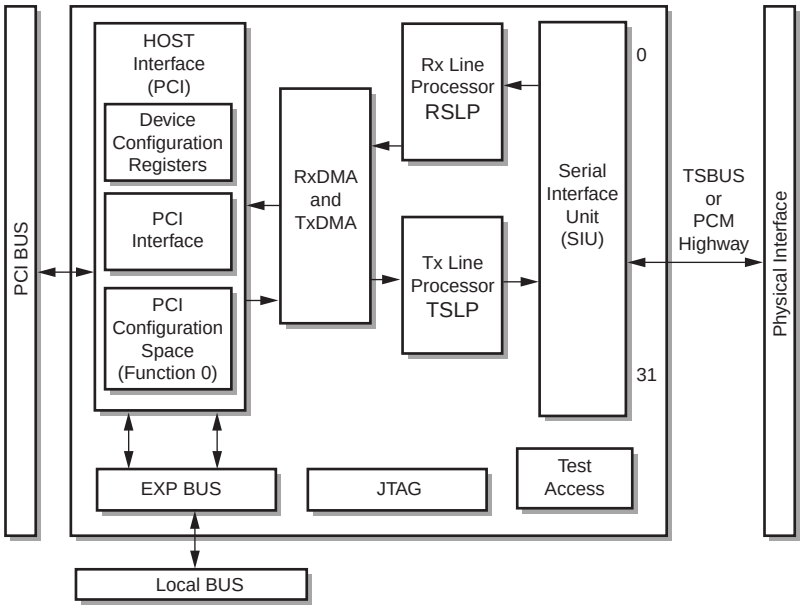
CX28500

Multichannel Synchronous Communications Controller

The CX28500 is an advanced Multichannel Synchronous Communications Controller. It formats and deformats up to 1024 High-level Data Link Control (HDLC) channels in a CMOS integrated circuit. CX28500 operates at Layer 2 of the Open Systems Interconnection (OSI) protocol reference model. It provides a comprehensive, high-density solution for processing of HDLC channels for internetworking applications such as Frame Relay, Integrated Services Digital Network (ISDN), D-channel signaling, X.25, Signaling System 7 (SS7), Data Exchange Interface (DXI), Inter System Link Protocol (ISLP), and LAN/WAN data transport. Under minimal Host supervision, CX28500 manages table-like data structures of channel data buffers in Host memory by performing Direct Memory Access (DMA) of up to 1024 channels.

CX28500 interfaces to 32 independent serial data streams, such as T1/E1 signals. It then transfers data across the popular 32-bit or 64-bit Peripheral Component Interface (PCI) bus to system memory at a rate up to 66 MHz. The CX28500 has an aggregate data throughput of 390 Mbps. Each serial interface can be operated up to 13.0 MHz. Six Serial Interfaces can be operated at rates up to 52 MHz. Logical channels can be mapped as any combination of Digital Signal Level 0 (DS0) time slots to support ISDN hyperchannels (N x 64 Kbps). Additionally, logical channels can operate in subchanneling mode (N x 8 Kbps) by mapping a combination of DS0 time slots and/or the individual bits of a DS0 time slot (8 bits). For example, a 56 Kbps channel can be achieved by mapping 7 bits out of 8 possible bits in a time slot (7 x 8 Kbps = 56 Kbps). CX28500 also includes a 32-bit expansion port for bridging the PCI bus to local microprocessors or peripherals. A Joint Test Action Group (JTAG) port enables boundary-scan testing to replace bed-of-nails board testing.

Functional Block Diagram



Distinguishing Features

- 1024-channel HDLC controller
- OSI Layer 2 protocol support
- General purpose HDLC (ISO 3309)
  - X.25 (LAPB)
  - Frame relay (LAPF/ANSI T1.618)
  - ISDN D-channel (LAPD/Q.921)
  - ISLP support
- 32 Independent serial interfaces, which support:
  - Mixed Data Rates (combination of T1/E1/T3/E3, etc.) as long as they do not exceed each port's respective bandwidth limitation and the overall device bandwidth of 390 Mbps per direction
  - 32 T1/E1 data streams
  - 6 HSSI interfaces (52 Mbps)
  - DC to 13.0 Mbps serial interfaces
  - 32 x 8.192 MHz TDM busses
- Configurable logical channels
  - Standard DS0 (56, 64 Kbps)
  - Subchanneling (N x 8 Kbps)
  - Hyperchannel (N x 64 Kbps)
  - Unchannelized mode
- Per-channel protocol mode selection
  - Non-FCS mode
  - 16-bit FCS mode
  - 32-bit FCS mode
  - Transparent mode (unformatted data)
- Hardware Flow Control (CTS)
- Selectable Endian configuration on data
- Per-channel DMA buffer management
  - Table-like data structures
  - Variable size transmit/receive FIFO
- Per-channel message length check
  - Select no length checking
  - Select from three 14-bit registers to compare message length
- Direct PCI bus interface
  - 32/64-bit, 33/66 MHz operation
  - Bus master and slave operation
  - PCI Version 2.1
- Host back-to-back transaction over the PCI
- HSSI interfaces (52 Mbps)
- Local expansion bus interface (EBUS)
  - 32-bit multiplexed address/data bus
- TSBUS
- Support of 64-bit ECC host memory
- Low power, 3.3 V CMOS operation
- JTAG boundary scan access port
- 35 mm x 35 mm 580-pin BGA

Contents

Figures

Tables

1 Introduction

The CX28500 HDLC controller (CX28500) is a 1024-channel communications controller that operates at Layer 2 of the Open System Interconnection (OSI) protocol reference. It provides a comprehensive, high density solution for HDLC channels for internetworking applications.

- HDLC/SDLC
- LAPB, LAPD
- Digital Access Cross-Connect (DAC)
- Frame Relay Switches and Access Devices (FRAD)
- ISDN-D channel signalling
- X.25
- SMDS/ATM DXI
- LAN/WAN access data
- SONET/SDH add/drop multiplexers (ADMs)
- Terminal multiplexers (TMs)
- High Range/Gigabit Routers

CX28500 interfaces to 32 independent serial data streams such as T1/E1, T3/E3, STS-1/STM-1, and DS3. It signals and transfers data across the high performance 32/64-bit Peripheral Component Interface (PCI) bus to system memory at a rate up to 66 MHz. Each serial port can be configured to support different types of interfaces. All of the CX28500's serial ports are individually programmable to operate as conventional or TSBUS serial ports.

1.1 CX28500's Operational Modes

Table 1-1 explains the CX28500's supported serial port mode.

Table 1-1. Supported Serial Port Modes

CX28500 Serial Port Modes	Description
Conventional Unchannelized(1)	In this mode, Transmit Synchronization (TSYNC) and Receive Synchronization (RSYNC) are ignored. The serial input/output data stream is a bit-stream without framing or alignment. The bit-stream belongs to a single logical channel. The CX28500 conventional, unchannelized mode can operate up to 13 Mbps for all 32 serial ports. The first six ports can operate unchannelized T3/E3, HSSI, or STS-1/STM-1 bit-stream up to 52 Mbps per serial interface (for reference, see Section 1.2).
Conventional Channelized(1)(2)	The serial bit-stream is treated as a frame of *N* time slots (where *N* is less than or equal to 4096, given that other restrictions are met). The maximum bandwidth embedded into the PCM highway for the first six ports is STS-1 rate (51.84 Mbps). The byte and frame synchronization performed is based on receive and transmit sync pulse (RSYNC and TSYNC). (For a detailed description of these signals, see *Chapter 5.)*
Conventional T1 mode(1)	The serial bit-stream is treated as a frame of *N* time slots (where *N* has the same meaning as in channelized mode). If the serial port is configured in T1 mode then the port operates according to the T1 framing definition (i.e., 24 time slots, where the first bit is the F-bit). The difference between channelized mode and T1 mode is that in T1 mode, the F-bit of the T1 frame is masked off and is dropped on input, and is generated for the output (for proper timing only—a T1 framer is needed to generate the correct F-bit).
TSBUS(2)(3)	The TSBUS serial interface bit-stream is treated as a frame of *N* time slots (where *N* is defined as greater than or equal to 6, and the aggregate number of time slots across all ports in any direction does not exceed the 4096 available time slots in each direction, receive or transmit) or variable bandwidth time slots called Virtual Serial Ports (VSPs). Byte synchronization and frame synchronization is performed based on the TSBUS sync pulse STB (i.e., bus strobe). Mixed T1/E1 paths in one T3, mixed VT1.5/VT2 paths mapped to VTGs in one STS1, and mixed VC11/VC12 paths mapped to TUG2 in STM-1 are allowed using this serial port configuration.

FOOTNOTE:

(1) A conventional serial port is defined as seven input/output signals as follows: Transmit Clock (TCLK), Transmit Synchronization (TSYNC), Transmit Data (TDAT), Receive Clock (RCLK), Receive Synchronization (RSYNC), Receive Data (RDAT), Receive Out-Of-Frame, or Clear To Send (ROOF/ CTS). (For a detailed description of these signals, see Section 2.1.)

(2) Channelized mode refers to a data bit stream segmented into frames. Each frame consists of a series of 8-bit time slots. The frame synchronization is maintained in both the transmit and receive direction by using the Transmit Synchronization (TSYNC) and Receive Synchronization (RSYNC) input signals.

(3) A Time Slot Bus (TSBUS) is defined as seven input/output signals as follows: Transmit Clock (TCLK), Transmit Stuff (TSTUFF), Transmit Data (TDAT), Strobe (STB), Receive Clock (RCLK), Receive Stuff (RSTUFF), Receive Data (RDAT). (For a detailed description of these signals, see *Chapter 5*).

1.2 CX28500 Serial Port Throughput Limits

Each of the CX28500 serial ports can be configured to operate in any of the preceding operational modes. The following restrictions apply:

1. The overall number of time slots cannot exceed 4096.
2. The overall number of channels cannot exceed 1024.
3. Only the first 6 ports can be configured to operate as high speed ports—T3 (44.7 Mbps), E3 (34.4 Mbps), and HSSI (52 Mbps).
4. The total accumulated data rate of all serial port clocks in either receive or transmit direction must be less than 250 Mbps for a PCI bus rate of 33 MHz and 390 Mbps for a PCI bus rate of 66 MHz.

Allowed CX28500 port configurations are represented in Table 1-2.

Table 1-2. Allowed CX28500 Port Configurations

Speed of Port	4 Mbps	8 Mbps	13 Mbps	34.4 Mbps	44.7 Mbps	51.8 Mbps
Number of Ports	32	32	32	6(1)	6(1)	6(1)

FOOTNOTE:
(1) For high speed ports such as T3 (44.736 Mbps), E3 (34.368 Mbps), HSSI (51.840 Mbps) the data path of the remaining ports may operate at less than or equal to 4 Mbps or 8 Mbps (see Table 1-3).

Table 1-3. Data Path Combinations

Path Combinations	Low Speed					High Speed			Aggregate Port Speed
	2 Mbps (2.048)	4 Mbps (4.096)	8 Mbps (8.192)	10 Mbps (10.240)	13 Mbps (12.960)	34.4 Mbps(1) (E3-34.368)	44.7 Mbps(1) (T3-44.736)	51.8 Mbps(1) (HSSI-51.840)	
Allowed Number of Ports	—	17	—	—	—	—	—	6	380.7 Mbps
	—	26	—	—	—	6	—	—	312.7 Mbps
	—	—	14	—	—	—	6	—	383.1 Mbps
	—	—	22	—	—	6	—	—	386.4 Mbps
	—	—	9	—	—	—	—	6	384.8 Mbps
	—	—	—	11	—	—	6	–	381.1 Mbps
	—	—	—	7	—	—	—	6	382.7 Mbps
	—	—	—	17	—	6	–	—	380.3 Mbps
	—	—	—	—	9	—	6	—	385.1 Mbps
	—	—	—	—	6	—	—	6	388.8 Mbps
	—	—	—	—	14	6	—	—	387.6 Mbps
	26	—	—	—	—	—	6	—	321.7 Mbps
	26	—	—	—	—	—	—	6	364.3 Mbps
	26	—	—	—	—	6	—	—	259.5 Mbps

GENERAL NOTE:

1. These data path combinations are meant to illustrate the flexibility in data path configuration, and are not suggesting that these are the only supported configurations. Again, they are here for illustration purposes only. Other data rates and path combinations are achievable, as long as device-aggregate bandwidth restrictions and per-port bandwidth restrictions are met.

FOOTNOTE:

(1) A high speed port can be E3 (34.4 Mbps), T3 (44.7 Mbps), or HSSI (52 Mbps) without exceeding the maximum of six serial ports.

CX28500's configuration options are extremely flexible. Each logical channel can be assigned to a physical stream ranging from 8 Kbps (sub-channeling mode) to 52 Mbps.

CX28500's serial ports can interface to a standard PCM highway or TSBUS bus, which can be configured to operate at any of the serial port rates given in Tables 1-4 and 1-5.

Table 1-4. Examples of Serial Port Configurations With 33 MHz PCI Clock

Configuration	Total Bandwidth	PCI Clock
32 x T1	49.408 Mbps	33 MHz
32 x E1	65.536 Mbps	33 MHz
2 x 52 + 30 x E1	165.44 Mbps	33 MHz
32 x 2E1	131.072 Mbps	33 MHz
Maximum	250 Mbps	33 MHz

Table 1-5. Examples of Serial Port Configurations With 66 MHz PCI Clock

Configuration	Total Bandwidth	PCI Clock
4 x 52	208 Mbps	66 MHz
32 x 10	320 Mbps	66 MHz
6 x 52 + 6 x 13	390 Mbps(1)	66 MHz
32 x 4E1	262.144 Mbps	66 MHz
4 x 52 + 28 x T1	251.232 Mbps	66 MHz
6 x T3 + 26 x T1	310 Mbps	66 MHz
26 x 8 + 4 x T3	387 Mbps	66 MHz
Maximum	390 Mbps	66 MHz

GENERAL NOTE: STS-1 data rates (51.84Mbps) are achieved when used in combination with the M29503 over the TSBUS.

FOOTNOTE:
(1) Half of the OC-12 data rate, including overhead.

The send and receive data can be formatted in the HDLC messages or left unformatted (transparent mode) over any combination of bits within a selected time slot. CX28500's protocol message type is specified on a per-channel basis.

1.3 CX28500's Bus Interfaces

1.3.1 PCI—Peripheral Components Interface

An on-device PCI 2.1 compliant controller, known as the Host interface, is provided. Access to CX28500 is available through PCI read, write, and configuration cycles. The PCI bus interface supports DMA bursts for extremely high message throughput applications, up to 780 Mbps of the aggregate serial port bandwidth (i.e., 390 Mbps per direction).

1.3.2 EBUS—Local Expansion Bus

The CX28500 provides an on-device 32-bit local expansion bus (EBUS) controller that allows a Host processor to access peripheral memory space on the EBUS. Physical devices interface directly to CX28500 over the PCI using the configurable memory mapping features.

Although EBUS utilization is optional, the most notable application for the EBUS is the connection to peripheral devices (e.g., CX28365, a 12-port T3/E3 framer) local to CX28500's serial ports.

1.3.3 TSBUS—Time Slot Bus

CX28500 provides a TSBUS interface for variable bandwidth time slots, Virtual Serial Port (VSP). A VSP is defined as an entity—quantified by clock bus rate divided by number of time slots—which provides multiple asynchronous paths over a single serial port. A programmable number of VSPs per TSBUS are allowed by using the existing start and end address time slot pointer mechanism. The pointer mechanism allows CX28500 to allocate any number of VSPs on a given serial port, providing that the total number of VSPs allocated across all ports does not exceed 4096, the total number of logical channels does not exceed 1024, and the serial port clock speed does not exceed 52 MHz. While operating in TSBUS mode, the minimum number of time slots required is 8 per serial port. The programmable number of time slots, implemented by the pointer mechanism (i.e., configurable start and end addresses) allows any number of VSPs to be concatenated into a single logical channel. This concatenation allows mixed VTG path options without changing the number of time slots assigned to the TSBUS port. In the TSBUS transmit direction, CX28500 requires the stuff status for each time slot to be presented at its TSTUFF input exactly 8 time slots in advance of the actual time slot for which the stuff status is applied. The amount of the TSTUFF advance is fixed at 8 time slots even though the number of time slots within a frame might vary. For the receive direction CX28500 requires the stuff status for each time slot to be presented at its RSTUFF input on the current time slot for which the stuff is applied (For a detailed description, see the timing diagrams in Appendix E.)

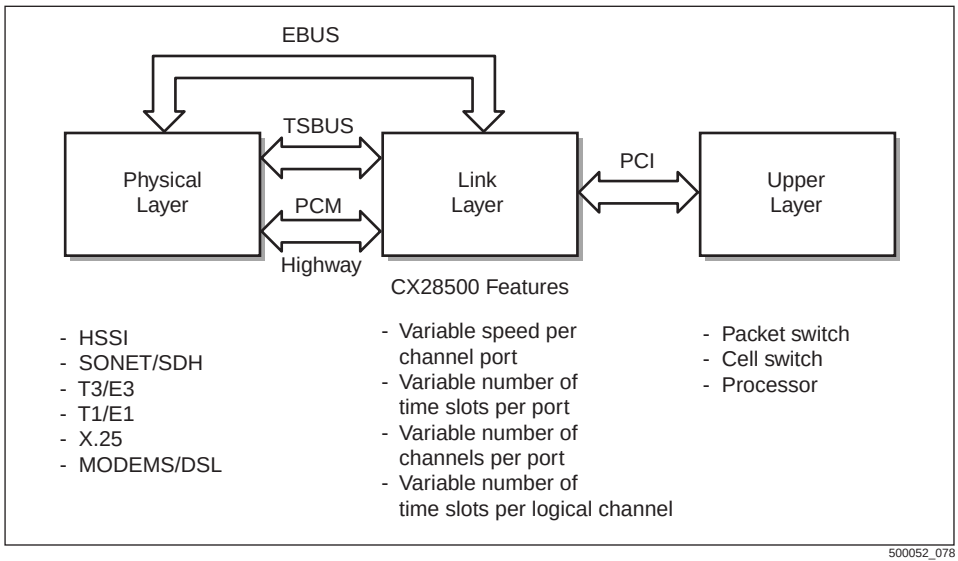
1.4 CX28500 Layering Model

Independent of the particular layering scheme used, or the functions of the layers, the operation of layered protocols is based on the fundamental idea of a layering principle (refer to Figure 1-1).

Level 2 of the protocol stack specifies how data travels between a Host and the packet switch to which it connects. Because raw hardware delivers only a stream of bits, the level 2 protocol must define the format of frames and specify how the machines recognize frame boundaries. This function is one of CX28500's features, including error detection (e.g., a frame checksum). Usually, protocols use the term "frame" to refer to a unit of data as it passes between a Host and a packet switch.

CX28500, as a high-throughout communications controller for synchronous or asynchronous path applications, multiplexes and demultiplexes up to 1024 data channels and allows upper protocols to recognize frame boundaries and check if a frame has been transferred successfully. Examples of CX28500's potential applications as a layer 2 protocol device and examples of device communication with other upper layers are described in Section 1.5.

Figure 1-1. CX28500 as a Data Link Layer Device



1.5 CX28500's Applications

The potential applications for CX28500 are in data communications and telecommunications markets, such as the following:

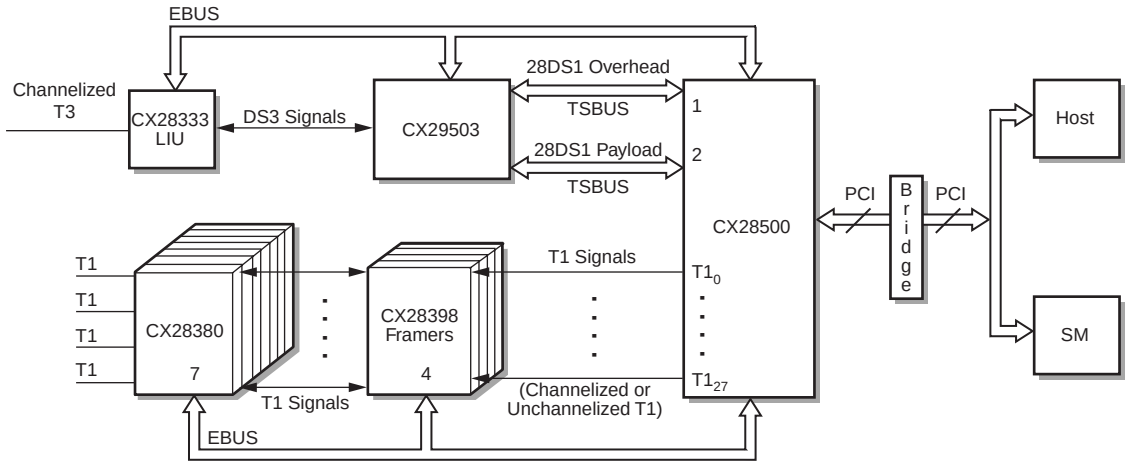
- WAN access equipment
  - Router
  - Remote Access Server
  - Frame Relay Switch
  - ISDN basic-rate (BRI) or primary-rate interfaces (PRI)
  - Edge Switch (ATM or Frame Relay)
  - Protocol Converter
  - Enterprise Switch (Frame Relay or ATM)
  - SONET/SDH Packet Switches
  - High-Range Router/Gigabit Router
- Transmission Equipment
  - Digital Access Cross-Connects (DACs)
  - Digital Loop Carries (DLCs)
  - Wireless Communication
  - Cellular Base Station Controllers

1.6 CX28500 Applications Examples

1.6.1 T1/T3 WAN Access

An example of a Mid-Range Router Application for channelized DS3 and unchannelized/channelized DS1 line interface is illustrated in Figure 1-2.

Figure 1-2. Mid-Range Router Application



LEGEND:

CX28333-DS3/E3 line Interface Unit.
CX29503–M13 Framers; 28 DS1 signals can be multiplexed into and demultiplexed from 1 DS3 signal or various combinations of E1 and DS1 signals that can be multiplexed into and demultiplexed from 1 DS3 signal.
7xCX28380–T1/E1 Line Interface Unit.
4xCX28398 or 2xCX28395–T1/E1/Framers.
CX28500–HDLC Controller.

GENERAL NOTE:

The application handles T1/E1 and T3/E3 signals both in North American digital telephony hierarchy and in the European digital telephony hierarchy.

500052_079

The block diagram illustrates how the CX28500 interconnects with 3 different buses:

- PCI bus used for communication between CX28500, central Host processor and shared memory.
- EBUS used for communication with peripheral devices.
- TSBUS used to provide multiple asynchronous paths between the DS3 framers and the central Host processor. The TSBUS also provides paths for inter-processor control (IPC) channels between the DS3 framers and the central Host processor. This example illustrates how 30 of CX28500's 32 serial ports are used in a Mid-Range Router Application. The T3/E3 data path (payload) occupies only one TSBUS port and the overhead path occupies another TSBUS port.

Feature List:

- 30 ports
- PCI bus that operates to redirect packets from and to local RAM via a PCI bridge
- EBUS that provides a path of control and performance monitoring for local devices
- Two TSBUS ports that provide the path for 28 DS1 payload data and 28 DS1 overhead data as a mixed combination of asynchronous paths
- Simultaneous connection of CX28500's serial port modes, including channelized TSBUS mode and T1/E1, either channelized or unchannelized
- Variable speed per port and channel
- Configurable allocation of the time slots
- Sufficient bandwidth to perform 28 DS1 paths and 1 DS3 path simultaneously
- Low cost and powerful line card interface
- High level chip integration
- High performance HDLC controller

This application uses 30 of the available 32 serial ports. For PCI bus operation, the central Host processor examines the packet's sequence and number, and manipulates the table header in local RAM to perform HDLC packet formatting and de-formatting.

1.6.2 T3/E3 Frame Relay Switch

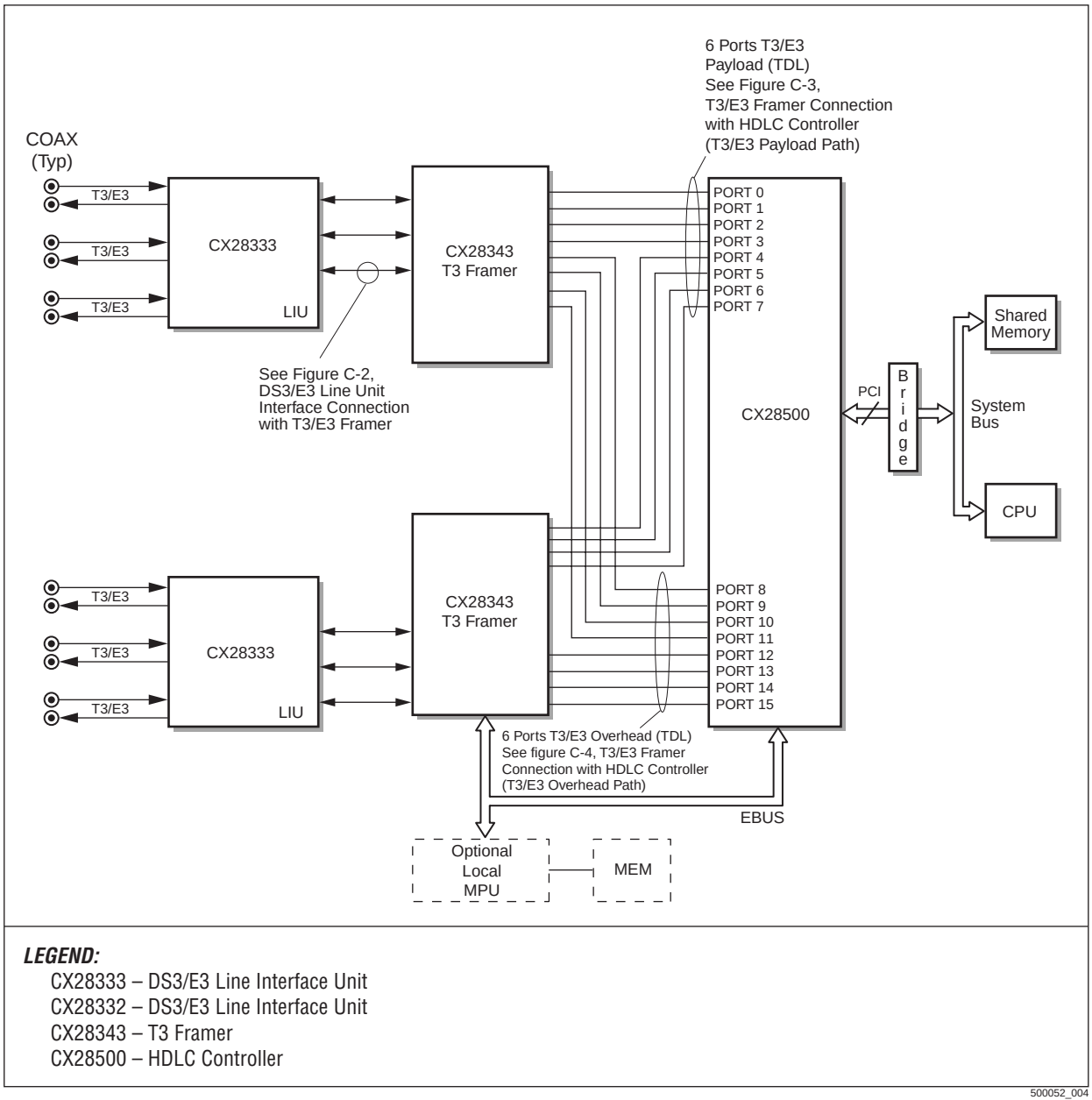
This application illustrates a connection between CX28500 and CX28344 T3/E3 Framer specifying an 6 DS3 serial ports application for an unchannelized DS3 High-End Router.

Feature List:

- Supports full-rate DS3 byte streams with T3/E3 overhead and payload
- Supports a byte stream of 32 dwords burst size that travels over the PCI bus/bridge and is directly accessible as shared memory for the Host to process
- Uses a high-speed port priority scheme. Figure 1-3 illustrates how the first payload data path ports are connected at the lower port numbers and the Terminal Data Link (TDL) overhead data paths are connected to the low priority ports.
- Uses framers that can be optimally configured through an optional microprocessor. This path can be used as a inter-processor communication (IPC) path for control and performance monitoring.
- Supports concurrent paths of payload and overhead. (The first six ports run simultaneously at 44.2 Mbps, the others run simultaneously at the TDL rate (192 Kbps).

NOTE: For the detailed description of the application's device connections, see Appendix C.

Figure 1-3. High-Range Router Application



LEGEND:

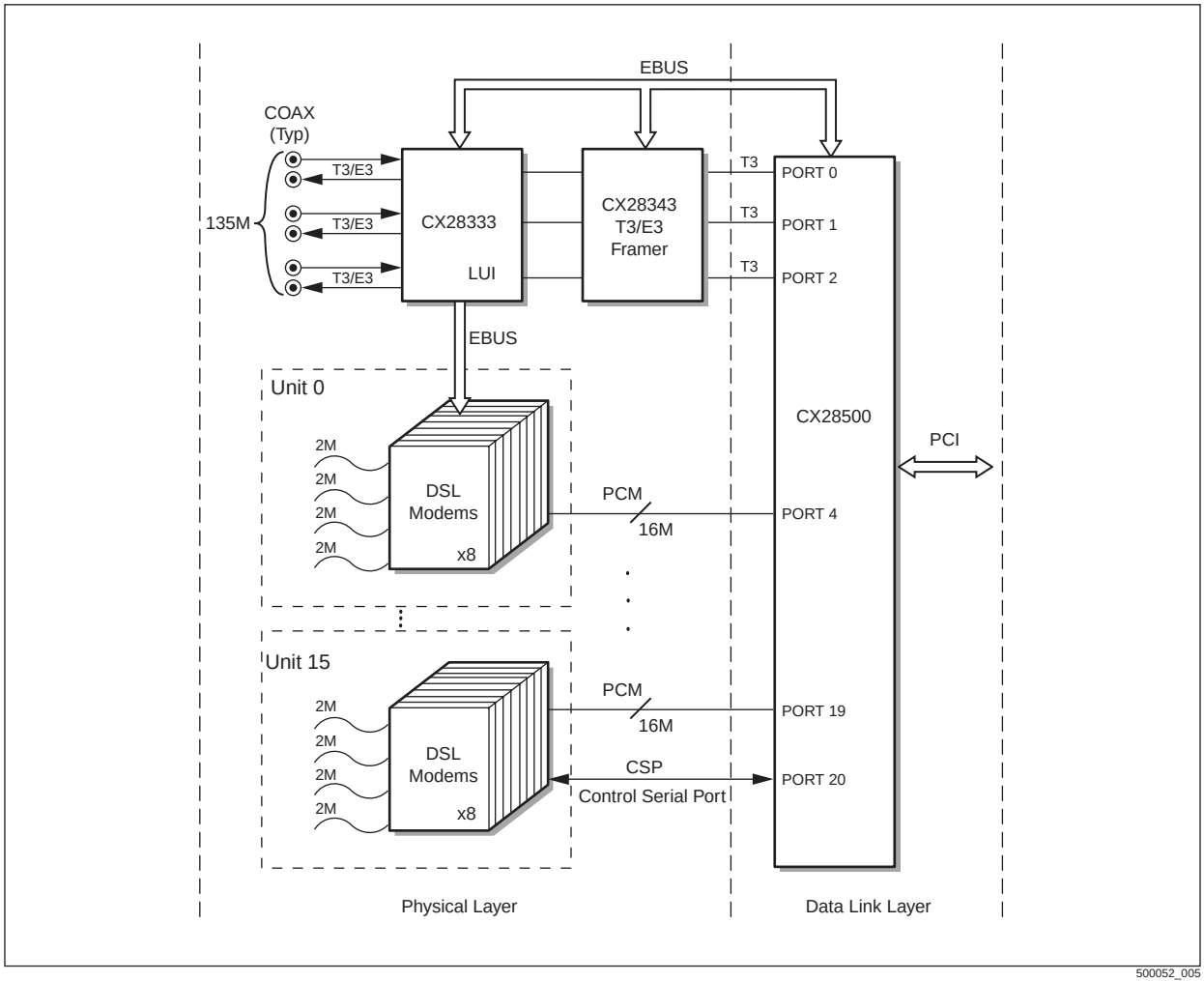
CX28333 – DS3/E3 Line Interface Unit
CX28332 – DS3/E3 Line Interface Unit
CX28343 – T3 Framer
CX28500 – HDLC Controller

500052_004

1.6.3 128 Port DSL Access Concentrator

Figure 1-4 illustrates a scalable application of an Internet Access Device (IAD).

Figure 1-4. DSLAM or IAD



CX28500, the HDLC controller, which operates at Layer 2 of OSI, communicates with the physical layer devices—CX28333 (T3/E3 Line Interface Framer) and eight units of eight DSL modems. The modems are connected to CX28500's serial port at the 16 Mbps rate into a PCM highway. CX28500's first three ports run channelized T3 and the other 16 ports run at 16 Mbps so that the total port aggregated rate is 260.6 Mbps. The example shows how a bank of eight modems can be configured by the serial line Control Serial Port (CSP). The CSP port provides the advantage of a Message Buffer Inter-Processor Control (MBIPC) channel.

1.6.4 SONET/SDH Mapper

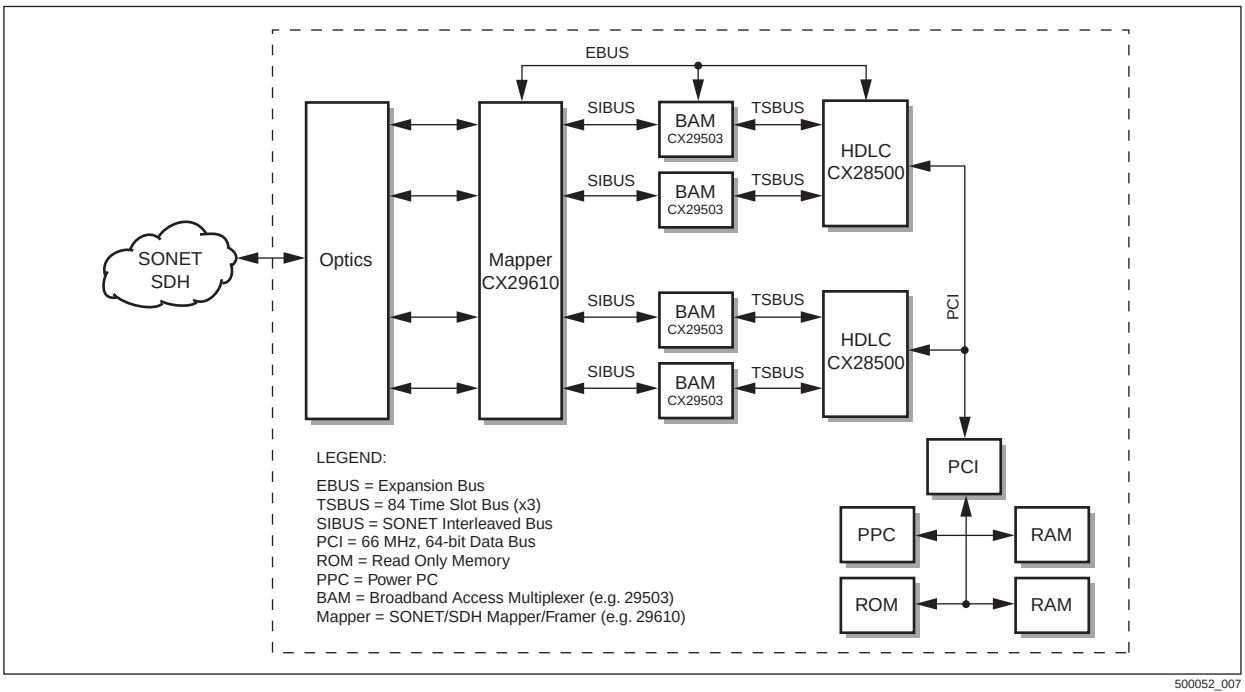
Figure 1-5 illustrates a Gigabit Router Line Card which can perform packet processing for one OC-12/STM-4 line.

Feature List:

- EBUS bus configuration to local devices
- PCI bus which acts as CX28500's configuration/status interface
- Power solutions for HDLC processing messages (two HDLC controllers running at 66 MHz)
- IPC channel processor
- Performance monitoring
- Scalable configuration up to OC-48
- T3/E3 data link over the TSBUS channel

NOTE: This configuration allows the Power PC (PPC) to receive and transmit data link messages over these channels via its local RAM (using DMA and buffer table) instead of having to service individual channel FIFOs with direct read/write access on each device. This effectively removes all PPC real-time constrains, the only remaining real-time activity is to handle the single line interrupt from the Mapper to the PPC.

Figure 1-5. Gigabit Router Line Card for OC-12



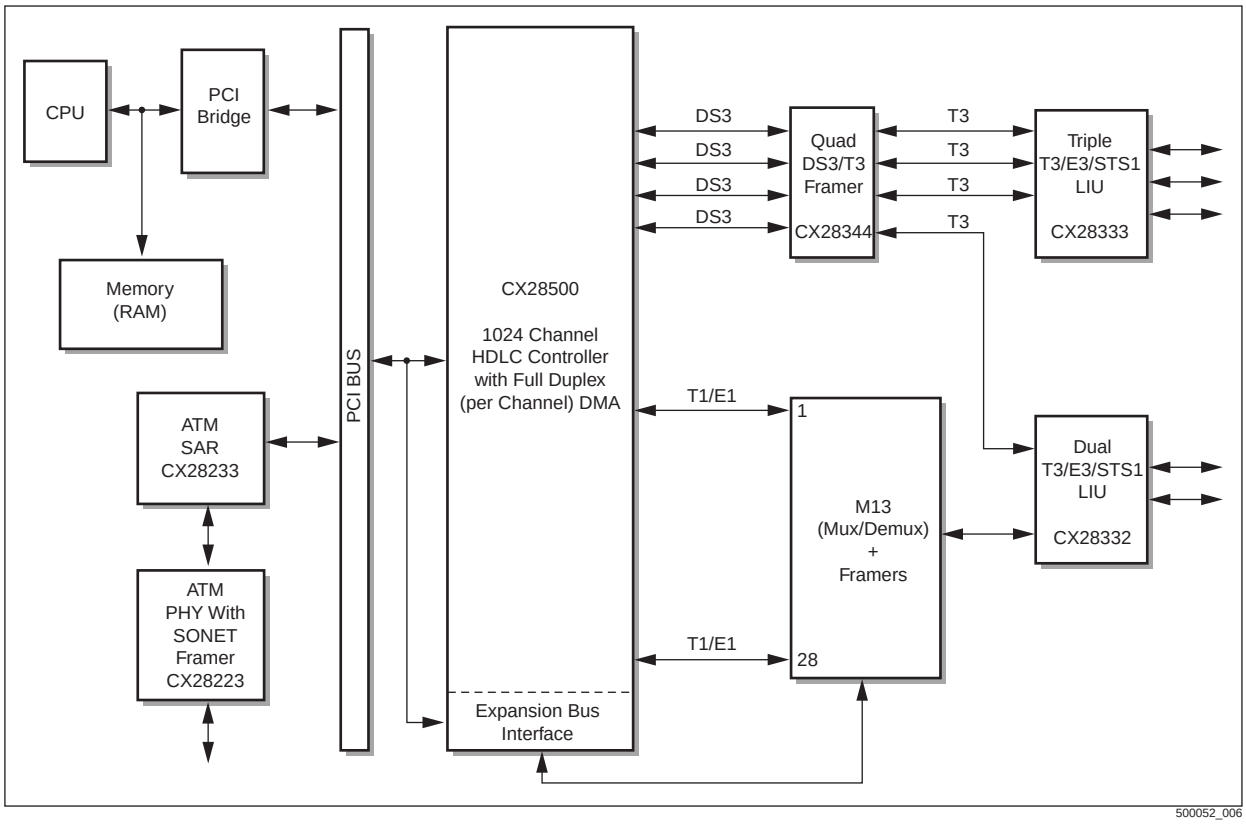
1.6.5 Line Card SONET/ATM SAR

Figure 1-6 illustrates the Line Card SONET/ATM SAR.

Feature List:

- Higher data aggregation
- Higher number of interfaces
- Chip cost reduced by increasing the number of interfaces serving a large number of logical channels
- Efficient bus occupancy when data is transferred between Link Layer Controller and memory
- Fully independent serial port configuration
- Different types of interfaces handled simultaneously (channelized DS3 and channelized T1/E1)
- 32 independent serial ports
- Conventional mode supported by seven signals (RDAT, RCLK, RSYNC, ROOF, TDAT, TCLK, and TSYNC)
- Local bus architecture
- Per-channel DMA buffer management, linked list data structure handled in local memory
- Increased PCI utilization accomplished by supporting different classes of traffic with fully per-channel programmable threshold values of internal buffers. These configurable buffer thresholds allow the user to take advantage of PCI bursts in transferring data between CX28500's internal memory and shared memory.

Figure 1-6. Line Card SONET/ATM SAR



1.7 Feature Summary

The CX28500's feature list includes the following:

- A 1024-channel, full-duplex link layer controller for synchronous applications is provided.
- 32 full-duplex physical interfaces (i.e., ports) with independent clock rates are provided. CX28500 implements 32 serial ports that are individually programmable to operate either as conventional serial ports or TSBUS serial ports. Conventional serial ports' input/out data streams can be configured as channelized or unchannelized bit streams. As for TSBUS serial ports, they are channelized by definition. Clock rates may be as high as 52 MHz (for the first six ports) or 13 MHz (for the remaining 26 ports, see Table 1-2).
- General purpose HDLC (ISO 3309) is supported.
  - HDLC/SDLC
  - HSSI
  - ISDN D-channel (LAPD/Q.921)
  - X.25 (LAPB)
  - SS7
  - Frame Relay (LAPF/ANSI T1.618)
  - Inter System Link Protocol (ISLP) support
  - LAPDm support
  - ATM/SMDS DXI
  - Transparent unformatted mode
  - Point-to-Point-Protocol (PPP)
- Hyperchannels and subchannels are supported.
  - Hyperchannel mode
    - ISDN Primary Rate Interface (PRI)
    - ISDN Primary Rate Adapter (PRA)
    - Fractional T1 (FT1)
    - Fractional E1 (FE1)
    - Fractional Nx64K
    - SONET/SDH/PDH paths connected via TSBUS, which include: Mixed VT1.5/VT2 paths, Mixed TU-11/TU-12 paths, and Mixed T1/E1 paths
    - Multiple lines muxed to 1 port, which include: Digital Subscriber Line Access Multiplexer (DSLAM) and T1/E1 Frame Relay
  - Subchanneling mode
    Each channel can be programmed to either use a complete DS0 time slot or mask any subset of a time slot. A signal mask is defined per-channel basis that has an enabled bit per each time slot. The mask bit indicates whether the whole DS0 or part of it is enabled
    - ISDN Basic Rate Interface (BRI)
    - ISDN Basic Rate Adapter (BRA)
    - Frame Relay 56K and Nx56K
    - Compressed Voice Transparent Channels (e.g., ADPCM)
    - Centralized Signaling Channel Controllers, which include: Link Access Procedure D-Channel (LAPD), Common Channel Signaling (CCS), and Signaling System #7 (SS7)

- Unchannelized mode
  - Digital Comm/Termination Equipment (DCE/DTE) interfaces
  - High Speed Serial Interface (HSSI)
  - Inter-Process Communication (IPC)
  - V-Series DTE/DCE Interfaces (V.35)
  - SDSL Modems and Access Concentrators
  - T3/E3 Frame Relay

- Variable path primitives are supported.
  - Path Payload
    - Sub-channeling (N × 8 Kbps) where N is between 1 and 7, the fractional bits in an 8-bit time slot.
    - DS0 (64 Kbps)
    - N × 64 Kbps, allows all types of hyperchanneling, channelized, unchannelized, or path payload as long as the bandwidth does not exceed the respective port's bandwidth limitation
  - Higher speed ports
    - Unchannelized T3 (44.736 Mbps)
    - Unchannelized E3 (34.368 Mbps)
    - HSSI (52 Mbps)
  - Path overhead (Performance Monitoring and Provisioning) T1
    - Facilities Data Link (FDL)
    - Common Channel Signalling (CCS)
    - T3/E3 Terminal Data Link (TDL)
    - V.51 and V.52 signalling channels
- Per-channel protocol selection is supported
  - Non-FCS mode
  - 16-bit FCS mode
  - 32-bit FCS mode
  - Transparent mode
- Configurable logical channels are supported
  - Standard DS0
  - Hyperchannel
  - Subchannel
- Programmable time slot allocation is supported
  - Pointer mechanism
- Per-channel DMA buffer management is supported
  - 32 KB per direction receive and transmit (64 Kb per chip) internal FIFO
  - Configurable DMA threshold per-channel basis
  - Programmable FIFO size per-channel basis
  - Configurable number of buffer descriptors per channel
  - Flexible buffer descriptor handling
- Self Service mechanism
- Clear to Send (CTS) per-channel control of data transmission
- Direct PCI bus interface is supported
  - PCI Bus Interface (rev. 2.1)
  - 32/64-bit multiplexed Address/Data bus minimizes pincount
  - 33/66 MHz operation
  - Burst DMA capability (i.e., up to 32 dwords) minimizes the bus occupancy

- EBUS—Expansion Bus Interface is provided
  - 32-bit multiplexed Address/Data
  - Allows Host to control other local devices
  - Facilitates Host access to any local memory
- TSBUS interface
  - Variable bandwidth time slot
  - Multiple asynchronous paths over single port
  - Allows SONET/SDH/PDH paths connection
    - Mixed VT1.5/VT2 paths
    - Mixed TU-11/TU-12 paths
    - Mixed T1/E1 paths
- 580-pin BGA package is used
- 3.3 V/2.5 V supply; 5 V-tolerant inputs
- JTAG access is provided
- Low power CMOS technology is used

1.8 System Overview

CX28500 supports 32 fully independent serial ports that can be configured to run in channelized, unchannelized, T1 or TSBUS mode.

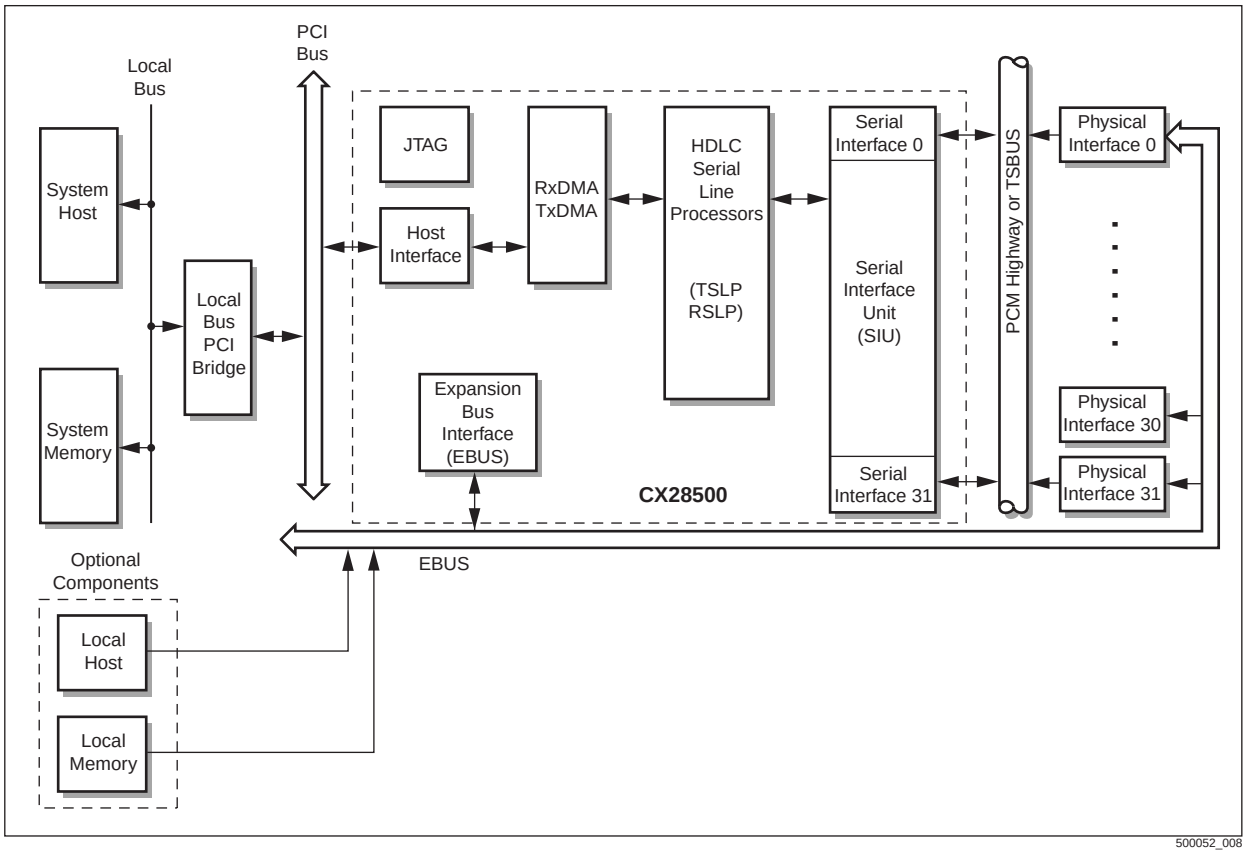
For example, in the channelized mode, the first six ports can operate at 51.84 Mbps (STS-1 rate) while another 22 ports can operate at 8.192 Mbps (4xE1 rate). Four more ports will be unused. Each STS-1 frame transports 28xT1, 1xT3, 21xE1 or mixed T1/E1 VTG paths. The configuration is valid as long as the overall number of time slots per the whole device is 4096 time slots or less. For other restrictions see Section 1.2.

Alternatively, any of CX28500's ports can interface unchannelized data streams (HDLC or unformatted). In this mode, each of the first six ports can be configured to operate up to 52 Mbps and any of the remaining 26 ports up to 13 Mbps. The restriction is that the overall bandwidth must not exceed 390 Mbps (per direction) while operating at 66 MHz PCI clock cycles or 250 Mbps (per direction) while operating at 33 MHz PCI clock cycles.

CX28500 manages buffer memory for each of the active data channels with common table processing structures. The on-device features allow data transmission between buffer memory and the serial interfaces with minimum Host processor intervention. This allows the Host processor to concentrate on managing the higher layers of the protocol stack.

Figure 1-7 illustrates CX28500's system overview.

Figure 1-7. System Overview



The TSBUS interface provides multiple asynchronous paths where all TSBUS frames run at 51.84 Mbps. The supported TSBUS framer configurations that are mapped into and from VSPs are as follows:

- There are 28 PDH framers and 28 SONET/SDH framers. Either of these two categories of framers can be mapped directly to the VSPs for a given configuration.
- When using the PDH framers, each framer can be independently configured as a DS1 framer or as an E1 framer.
- The SONET/SDH category of framers has two subcategories as the name implies. When using the SONET/SDH framers, they all have to be configured as either SONET framers (one subcategory) or as SDH framers (the other subcategory). The configurations of the SONET/SDH framers cannot have mixed subcategories. When configuring for the SONET subcategory of framers, each framer can be independently configured as a VT1.5 framer or as a VT2.0 framer. The SDH subcategory of framers is similarly configured. Each SDH framer can be independently configured as a VC-11 framer or as an E1 framer.

The supported TSBUS frame structures are DS1s, E1s mapped via DS2, VT1.5, VT2.0, VC-11, VC-12.

The following mixed mappings are also supported by selectively configuring each framer:

- DS1s and E1s extracted from mixed DS2s via the PDH framers
- DS1s and E1s extracted from mixed VTGs via the SONET framers
- DS1s and E1s extracted from mixed VTGs via the SONET framers
- VC-11s and VC-12s extracted from mixed TUG-2s via the SDH framers

The frame structure is designed to transport the unchannelized STS-1 Synchronous Payload Envelope (SPE).

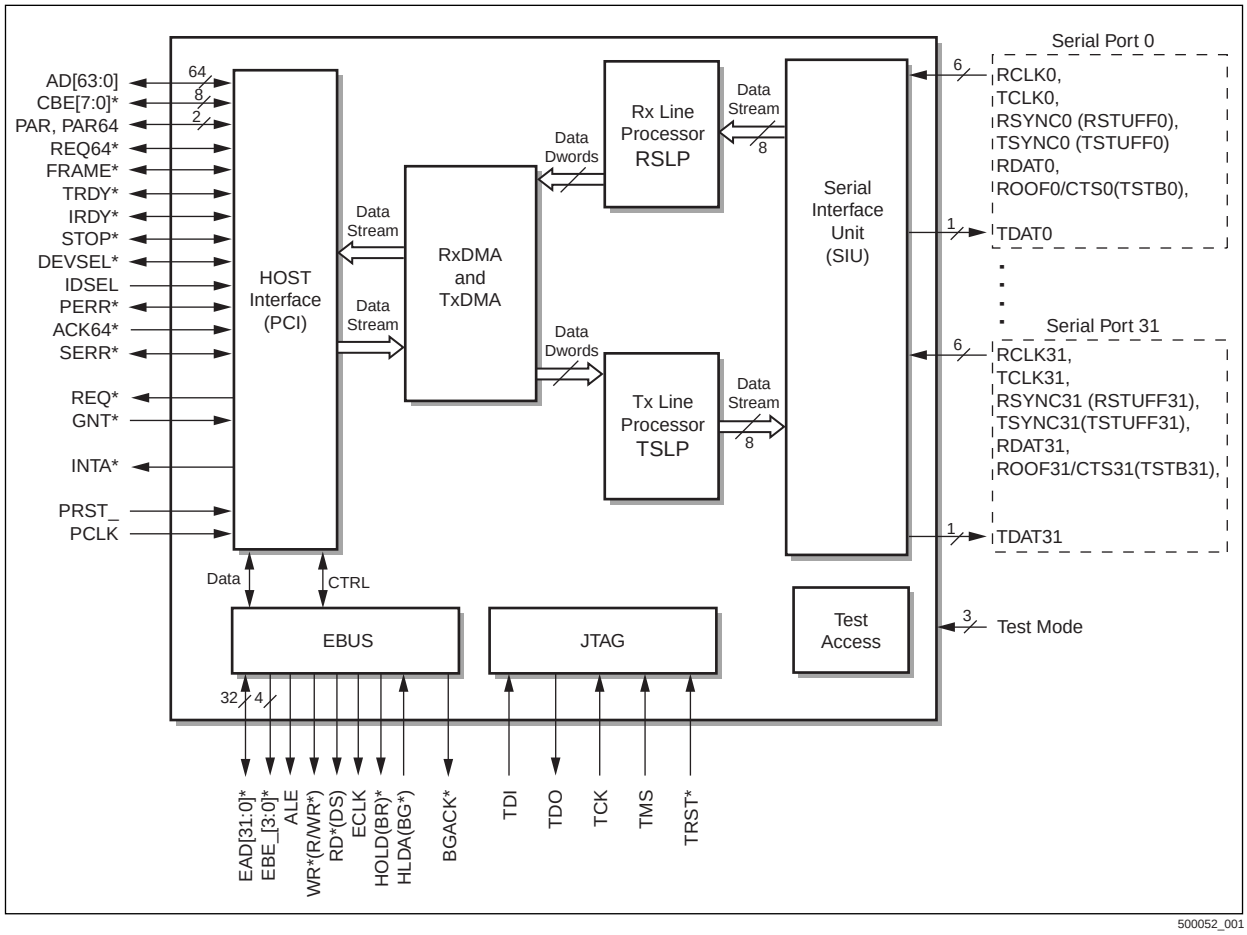
The frame structure is designed to transport the unchannelized DS3 payload.

The frame structure is designed to transport 16 E1 signals that are mapped to and from E3.

1.9 Block Diagram

Figure 1-8 illustrates the CX28500 conceptual block diagram.

Figure 1-8. CX28500 Top Level Block Diagram



The following is a description of the block diagram.

- Host Interface (PCI): This block provides the communication path between the Host and CX28500.
- Expansion Bus (EBUS): The EBUS is an extension of the Host Interface, which provides the Host with access to control other devices on the local PC board.
- Serial Interface Unit (SIU): This block provides the interface between 32 serial ports and the Receive and Transmit Serial Line Processors block. A temporal buffering space is provided by SIU that is 48 bits per port, divided as 32 bits (4 bytes) for the transmit direction and 16 bits (3 bytes) for the receive direction. SIU controls the data access to the Rx and Tx Serial Line Processors. Given that CX28500 supports two types of serial ports, one is the conventional interface, the other TSBUS interface—SIU needs to operate depending on serial port type (for detailed descriptor information, see Chapter 5).
- Transmit Serial Line Processor (TSLP): This block provides the interface between the DMA and SIU. The data provided by DMA is processed by TSLP according to the channel type (transparent/HDLC, etc.) before it is transmitted to the line.

- Receive Serial Line Processor (RSLP): This block provides the interface between the SIU and DMA. The data provided by SIU is processed by RSLP according to the channel type (transparent/HDLC, etc.) before it is transmitted to the line.
- RxDMA and TxDMA: This block provides the interface between the Host Memory (PCI) and the Transmit and Receive Serial Line Processors (TSLP and RSLP). The DMA contains the main storage of data—a dual port RAM of 32 KB per each direction, receive and transmit. This space acts as a holding buffer for incoming (Rx) and outgoing (Tx) data.
- JTAG: This is a special test port used for serial boundary scan on a PCB, as well as access to internal scan paths and embedded memory for test purposes.

1.10 Receive Data Path

At the SIU level, all line signals are synchronized with the system PCI clock, which runs either at 33 MHz or 66 MHz. The received data serial stream is stored in SIU local buffers.

Each cycle, the SIU transfers a byte received from one of the 32 serial ports to the Receive Serial Line Processor (RSLP). The order in which ports get serviced by the SIU depends on their priorities. A lower numbered port has a higher priority than a higher numbered port, hence lower number ports are serviced before higher number ports. For each serial port that was served, SIU translates the time slot into a channel number by using the internal map, and provides specific parameters for RSLP to further process the incoming data.

The RSLP is a three–stage pipeline processor that performs the functions of an HDLC processor (formatting data), as well as data concatenation (i.e., to merge bytes in double words) required by RxDMA. The RSLP processed data is transferred to RxDMA together with a related status, which indicates the status of data.

The RSLP received data is stored in the internal memory before it is transferred to the Host. The internal memory can be dynamically configured on a per-channel basis in programmable units called the channel's internal buffer space or channel's FIFO. The user must configure the channel FIFO in quad dwords granularity by programming the start and end address of each FIFO (see Table 6-24, *RDMA Buffer Allocation Register, bit fields RDMA_ENDAD and RDMA_STARTAD*). When the threshold is crossed, a request to the RxDMA to serve that specific channel is generated. If either an HDLC complete message (defined by an opening and closing flag) or an EOM message resides in the channel FIFO, then a request will be generated towards RxDMA to serve the channel regardless of the threshold value. The request is queued in the RxDMA Service Request Queue, which contains all the channels pending service requests. The request is serviced when it reaches the top of the RxDMA Service Requests Queue. If the channel request gets to the top of the RxDMA Service Request Queue, then the RxDMA will transfer the content of the channel FIFO to the Host memory. After each transaction is completed, the channel is queued at the end of RxDMA Service Request Queue until all channel data content is transferred to the Host memory.

1.11 Transmit Data Path

For transmit direction, the user allocates the buffer space in quad dwords granularity for each channel by programming the start and end addresses of each channel FIFO (see *Table 6-32, TDMA Buffer Allocation Register*, *bit fields TDMA_ENDAD* and *TDMA_STARTAD*). Transmission begins on a request generated by the TxSIU when it is ready for transmission. The request, in terms of a channel number, is passed to the TSLP. If the TSLP has data for the requested channel, then it will give that data to the TxSIU. Otherwise, it will generate a request, also a channel number, to the TxDMA. The TxDMA serves the channel based on its FIFO fill-state and its data availability. A channel is queued in TxDMA Service Request Queue when the channel FIFO contains less than the threshold value, or at least 32 empty dwords for the channel.

When a channel request gets to the top of the TxDMA Service Requests Queue, the TxDMA will serve this channel until the channel FIFO is full. Since this may involve more than one PCI transaction, the channel is inserted at the end of the TxDMA Service Request Queue after each transaction. The TxDMA gets data from shared memory on a per-channel basis when the internal FIFO is not full. If TxDMA has either enough data or a complete HDLC message, then it will start transferring data to the TSLP. The TSLP, in turn, does the necessary formatting, if any, and sends the data to the TxSIU for transmission. The SIU uses a fixed priority scheme where port A has priority over port B (when A is smaller than B). For the selected port, the SIU uses a internal map conversion to identify which channel number data is transferred.

1.12 Pin Configuration

Figure 1-9 provides a diagram of the CX28500 Pin Configuration, and Table 1-6 provides a pin description.

Figure 1-9. Pin Configuration Diagram

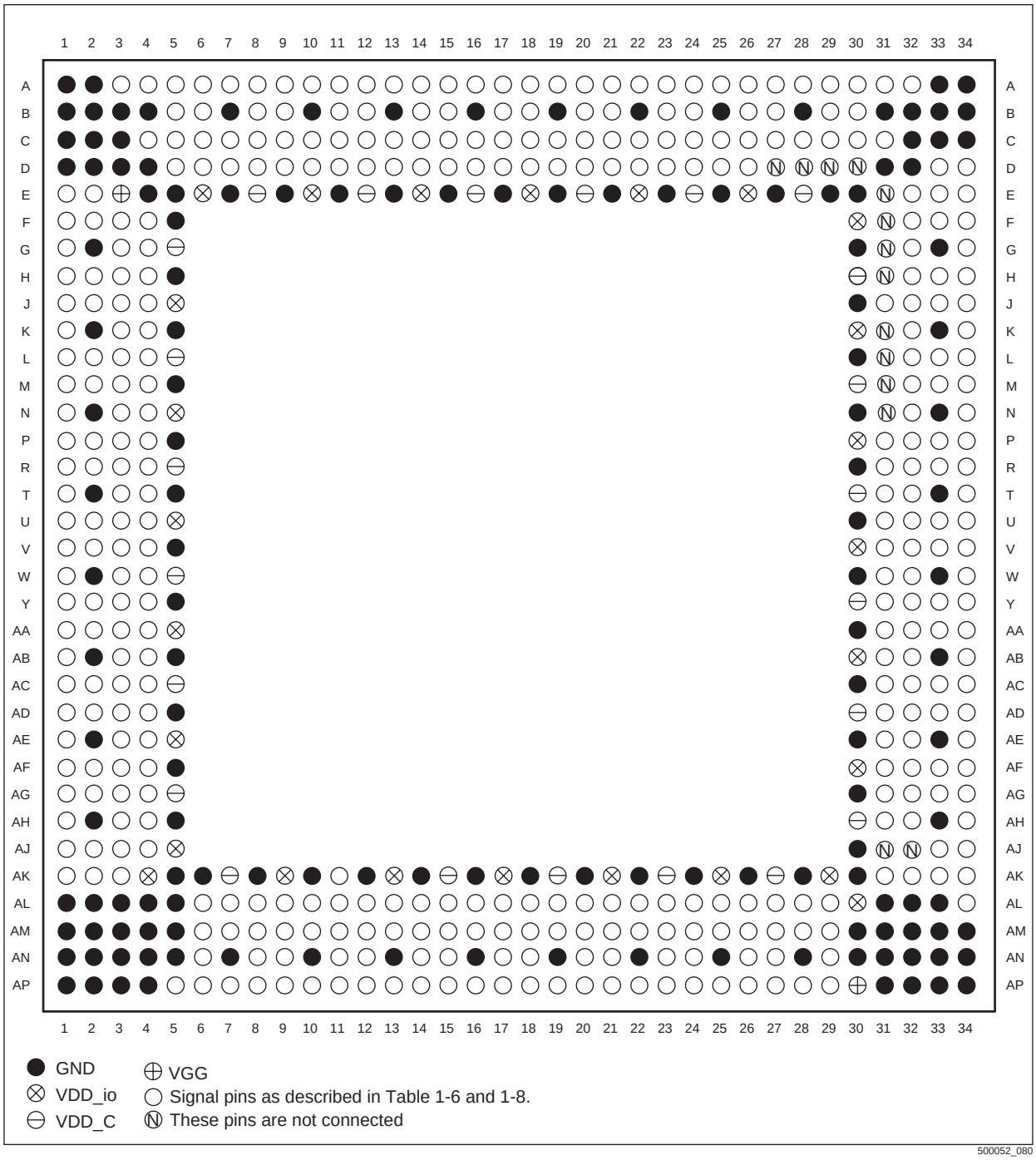


Table 1-6. Pin Description

Pin Number	Pin Label
A1	GND
A2	GND
A3	TDAT[17]
A4	TCLK[17]
A5	RDAT[17]
A6	TCLK[16]
A7	RDAT[16]
A8	TCLK[15]
A9	ROOF/CTS/STB/SPORT [14]
A10	TCLK[14]
A11	ROOF/CTS/STB/SPORT [13]
A12	RCLK[13]
A13	ROOF/CTS/STB/SPORT [12]
A14	RCLK[12]
A15	TDAT[11]
A16	RCLK[11]
A17	TDAT[10]
A18	TSYNC/TSTUFF[10]
A19	RDAT[10]
A20	TSYNC/TSTUFF[9]
A21	RDAT[9]
A22	TCLK[8]
A23	RDAT[8]
A24	TCLK[7]
A25	ROOF/CTS/STB/SPORT [6]
A26	TCLK[6]
A27	ROOF/CTS/STB/SPORT [5]
A28	TCLK[5]
A29	RSYNC/RSTUFF[5]
A30	TDAT[4]
A31	RCLK[4]
A32	RDAT[4]
A33	GND
A34	GND
B1	GND
B2	GND
B3	GND
B4	GND
B5	RSYNC/RSTUFF[17]
B6	TSYNC/TSTUFF[16]
B7	GND
B8	TSYNC/TSTUFF[15]
B9	RDAT[15]
B10	GND
B11	RDAT[14]
B12	TCLK[13]
B13	GND
B14	TCLK[12]
B15	ROOF/CTS/STB/SPORT [11]
B16	GND
B17	ROOF/CTS/STB/SPORT [10]
B18	TCLK[10]
B19	GND
B20	TCLK[9]
B21	ROOF/CTS/STB/SPORT [8]
B22	GND
B23	ROOF/CTS/STB/SPORT [7]
B24	RCLK[7]
B25	GND
B26	RCLK[6]
B27	TDAT[5]
B28	GND
B29	RDAT[5]
B30	TSYNC/TSTUFF[4]
B31	GND
B32	GND
B33	GND
B34	GND
C1	GND
C2	GND
C3	GND
C4	TSYNC/TSTUFF[17]
C5	ROOF/CTS/STB/SPORT [17]
C6	TDAT[16]
C7	RSYNC/RSTUFF[16]
C8	TDAT[15]
C9	RSYNC/RSTUFF[15]
C10	TSYNC/TSTUFF[14]
C11	RSYNC/RSTUFF[14]
C12	TSYNC/TSTUFF[13]
C13	RDAT[13]
C14	TSYNC/TSTUFF[12]
C15	RDAT[12]
C16	TCLK[11]
C17	RDAT[11]
C18	RCLK[10]
C19	ROOF/CTS/STB/SPORT [9]
C20	RCLK[9]
C21	TDAT[8]
C22	RCLK[8]
C23	TDAT[7]
C24	RSYNC/RSTUFF[7]
C25	TDAT[6]

Pin Number	Pin Label
C26	RSYNC/RSTUFF[6]
C27	TSYNC/TSTUFF[5]
C28	RCLK[5]
C29	ROOF/CTS/STB/SPORT [4]
C30	TCLK[4]
C31	RSYNC/RSTUFF[4]
C32	GND
C33	GND
C34	GND
D1	GND
D2	GND
D3	GND
D4	GND
D5	RCLK[17]
D6	ROOF/CTS/STB/SPORT [16]
D7	RCLK[16]
D8	ROOF/CTS/STB/SPORT [15]
D9	RCLK[15]
D10	TDAT[14]
D11	RCLK[14]
D12	TDAT[13]
D13	RSYNC/RSTUFF[13]
D14	TDAT[12]
D15	RSYNC/RSTUFF[12]
D16	TSYNC/TSTUFF[11]
D17	RSYNC/RSTUFF[11]
D18	RSYNC/RSTUFF[10]
D19	TDAT[9]
D20	RSYNC/RSTUFF[9]
D21	TSYNC/TSTUFF[8]
D22	RSYNC/RSTUFF[8]
D23	TSYNC/TSTUFF[7]
D24	RDAT[7]

Pin Number	Pin Label
D25	TSYNC/TSTUFF[6]
D26	RDAT[6]
D27	No Connect
D28	No Connect
D29	No Connect
D30	No Connect
D31	GND
D32	ROOF/CTS/STB/SPORT [3]
D33	GND
D34	TDAT[3]
E1	RSYNC/RSTUFF[18]
E2	RDAT[18]
E3	VGG
E4	GND
E5	GND
E6	VDD_io
E7	GND
E8	VDD_c
E9	GND
E10	VDD_io
E11	GND
E12	VDD_c
E13	GND
E14	VDD_io
E15	GND
E16	VDD_c
E17	GND
E18	VDD_io
E19	GND
E20	VDD_c
E21	GND
E22	VDD_io
E23	GND
E24	VDD_c
E25	GND

Pin Number	Pin Label
E26	VDD_io
E27	GND
E28	VDD_c
E29	GND
E30	GND
E31	No Connect
E32	TSYNC/TSTUFF[3]
E33	TCLK[3]
E34	RCLK[3]
F1	TDAT[18]
F2	TSYNC/TSTUFF[18]
F3	TCLK[18]
F4	RCLK[18]
F5	GND
F30	VDD_io
F31	No Connect
F32	RSYNC/RSTUFF[3]
F33	RDAT[3]
F34	ROOF/CTS/STB/SPORT [2]
G1	RSYNC/RSTUFF[19]
G2	GND
G3	RDAT[19]
G4	ROOF/CTS/STB/SPORT [18]
G5	VDD_c
G30	GND
G31	No Connect
G32	TDAT[2]
G33	GND
G34	TSYNC/TSTUFF[2]
H1	TDAT[19]
H2	TSYNC/TSTUFF[19]
H3	TCLK[19]
H4	RCLK[19]
H5	GND

Pin Number	Pin Label
H30	VDD_c
H31	No Connect
H32	TCLK[2]
H33	RCLK[2]
H34	RSYNC/RSTUFF[2]
J1	RCLK[20]
J2	RSYNC/RSTUFF[20]
J3	RDAT[20]
J4	ROOF/CTS/STB/SPORT [19]
J5	VDD_io
J30	GND
J31	RDAT[2]
J32	ROOF/CTS/STB/SPORT [1]
J33	TDAT[1]
J34	TSYNC/TSTUFF[1]
K1	TDAT[20]
K2	GND
K3	TSYNC/TSTUFF[20]
K4	TCLK[20]
K5	GND
K30	VDD_io
K31	No Connect
K32	TCLK[1]
K33	GND
K34	RCLK[1]
L1	RCLK[21]
L2	RSYNC/RSTUFF[21]
L3	RDAT[21]
L4	ROOF/CTS/STB/SPORT [20]
L5	VDD_c
L30	GND
L31	No Connect
L32	RSYNC/RSTUFF[1]

Pin Number	Pin Label
L33	RDAT[1]
L34	ROOF/CTS/STB/SPORT [0]
M1	ROOF/CTS/STB/SPORT [21]
M2	TDAT[21]
M3	TSYNC/TSTUFF[21]
M4	TCLK[21]
M5	GND
M30	VDD_c
M31	No Connect
M32	TDAT[0]
M33	TSYNC/TSTUFF[0]
M34	TCLK[0]
N1	RCLK[22]
N2	GND
N3	RSYNC/RSTUFF[22]
N4	RDAT[22]
N5	VDD_io
N30	GND
N31	No Connect
N32	RCLK[0]
N33	GND
N34	RSYNC/RSTUFF[0]
P1	ROOF/CTS/STB/SPORT [22]
P2	TDAT[22]
P3	TSYNC/TSTUFF[22]
P4	TCLK[22]
P5	GND
P30	VDD_io
P31	RDAT[0]
P32	ROOF/CTS/STB/SPORT [24]
P33	TDAT[24]
P34	TSYNC/TSTUFF[24]
R1	TCLK[23]

Pin Number	Pin Label
R2	RCLK[23]
R3	RSYNC/RSTUFF[23]
R4	RDAT[23]
R5	VDD_c
R30	GND
R31	TCLK[24]
R32	RCLK[24]
R33	RSYNC/RSTUFF[24]
R34	RDAT[24]
T1	ROOF/CTS/STB/SPORT [23]
T2	GND
T3	TDAT[23]
T4	TSYNC/TSTUFF[23]
T5	GND
T30	VDD_c
T31	ROOF/CTS/STB/SPORT [25]
T32	TDAT[25]
T33	GND
T34	TSYNC/TSTUFF[25]
U1	TDO
U2	TMS
U3	TRST
U4	TCK
U5	VDD_io
U30	GND
U31	TCLK[25]
U32	RCLK[25]
U33	RSYNC/RSTUFF[25]
U34	RDAT[25]
V1	TDI
V2	TM[0]
V3	TM[1]
V4	TM[2]
V5	GND

Pin Number	Pin Label
V30	VDD_io
V31	TCLK[26]
V32	TSYNC/TSTUFF[26]
V33	TDAT[26]
V34	ROOF/CTS/STB/SPORT [26]
W1	EBE[0]
W2	GND
W3	EBE[1]
W4	EBE[2]
W5	VDD_c
W30	GND
W31	RDAT[26]
W32	RSYNC/RSTUFF[26]
W33	GND
W34	RCLK[26]
Y1	EBE[3]
Y2	BGACK
Y3	HLDA
Y4	HOLD
Y5	GND
Y30	VDD_c
Y31	TCLK[27]
Y32	TSYNC/TSTUFF[27]
Y33	TDAT[27]
Y34	ROOF/CTS/STB/SPORT [27]
AA1	ALE
AA2	ECLK
AA3	RD
AA4	WR
AA5	VDD_io
AA30	GND
AA31	ROOF/CTS/STB/SPORT [28]
AA32	RDAT[27]

Pin Number	Pin Label
AA33	RSYNC/RSTUFF[27]
AA34	RCLK[27]
AB1	EAD[31]
AB2	GND
AB3	EAD[30]
AB4	EAD[29]
AB5	GND
AB30	VDD_io
AB31	TCLK[28]
AB32	TSYNC/TSTUFF[28]
AB33	GND
AB34	TDAT[28]
AC1	EAD[28]
AC2	EAD[27]
AC3	EAD[26]
AC4	EAD[25]
AC5	VDD_c
AC30	GND
AC31	ROOF/CTS/STB/SPORT [29]
AC32	RDAT[28]
AC33	RSYNC/RSTUFF[28]
AC34	RCLK[28]
AD1	EAD[24]
AD2	EAD[23]
AD3	EAD[22]
AD4	EAD[21]
AD5	GND
AD30	VDD_c
AD31	RCLK[29]
AD32	TCLK[29]
AD33	TSYNC/TSTUFF[29]
AD34	TDAT[29]
AE1	EAD[20]
AE2	GND
AE3	EAD[19]

Pin Number	Pin Label
AE4	EAD[18]
AE5	VDD_io
AE30	GND
AE31	ROOF/CTS/STB/SPORT [30]
AE32	RDAT[29]
AE33	GND
AE34	RSYNC/RSTUFF[29]
AF1	EAD[17]
AF2	EAD[16]
AF3	EAD[15]
AF4	EAD[14]
AF5	GND
AF30	VDD_io
AF31	RCLK[30]
AF32	TCLK[30]
AF33	TSYNC/TSTUFF[30]
AF34	TDAT[30]
AG1	EAD[13]
AG2	EAD[12]
AG3	EAD[11]
AG4	EAD[10]
AG5	VDD_c
AG30	GND
AG31	TDAT[31]
AG32	ROOF/CTS/STB[31]
AG33	RDAT[30]
AG34	RSYNC/RSTUFF[30]
AH1	EAD[9]
AH2	GND
AH3	EAD[8]
AH4	EAD[7]
AH5	GND
AH30	VDD_c
AH31	RCLK[31]
AH32	TCLK[31]

Pin Number	Pin Label
AH33	GND
AH34	TSYNC/TSTUFF[31]
AJ1	EAD[6]
AJ2	EAD[5]
AJ3	EAD[4]
AJ4	EAD[3]
AJ5	VDD_io
AJ30	GND
AJ31	No Connect
AJ32	No Connect
AJ33	RDAT[31]
AJ34	RSYNC/RSTUFF[31]
AK1	EAD[2]
AK2	EAD[1]
AK3	EAD[0]
AK4	VDD_io
AK5	GND
AK6	GND
AK7	VDD_c
AK8	GND
AK9	VDD_io
AK10	GND
AK11	VDD_c
AK12	GND
AK13	VDD_io
AK14	GND
AK15	VDD_c
AK16	GND
AK17	VDD_io
AK18	GND
AK19	VDD_c
AK20	GND
AK21	VDD_io
AK22	GND
AK23	VDD_c

Pin Number	Pin Label
AK24	GND
AK25	VDD_io
AK26	GND
AK27	VDD_c
AK28	GND
AK29	VDD_io
AK30	GND
AK31	BS[1]
AK32	BS[2]
AK33	BS[3]
AK34	BS[4]
AL1	GND
AL2	GND
AL3	GND
AL4	GND
AL5	GND
AL6	PRST*
AL7	AD[31]
AL8	AD[28]
AL9	AD[24]
AL10	AD[22]
AL11	AD[19]
AL12	C/BE[2]*
AL13	DEVSEL
AL14	SERR*
AL15	AD[14]
AL16	AD[10]
AL17	C/BE[0]*
AL18	AD[01]
AL19	REQ64*
AL20	C/BE[4]*
AL21	AD[61]
AL22	AD[58]
AL23	AD[54]
AL24	AD[50]

Pin Number	Pin Label
AL25	AD[47]
AL26	AD[43]
AL27	AD[39]
AL28	AD[36]
AL29	AD[32]
AL30	VDD_io
AL31	GND
AL32	GND
AL33	GND
AL34	BS[0]
AM1	GND
AM2	GND
AM3	GND
AM4	GND
AM5	GND
AM6	PCLK
AM7	AD[30]
AM8	AD[27]
AM9	C/BE[3]*
AM10	AD[21]
AM11	AD[18]
AM12	FRAME*
AM13	STOP*
AM14	PAR
AM15	AD[13]
AM16	AD[09]
AM17	AD[07]
AM18	AD[02]
AM19	ACK64*
AM20	C/BE[5]*
AM21	AD[62]
AM22	AD[59]
AM23	AD[55]
AM24	AD[51]
AM25	AD[48]

Pin Number	Pin Label
AM26	AD[44]
AM27	AD[40]
AM28	AD[37]
AM29	AD[33]
AM30	GND
AM31	GND
AM32	GND
AM33	GND
AM34	GND
AN1	GND
AN2	GND
AN3	GND
AN4	GND
AN5	GND
AN6	GNT*
AN7	GND
AN8	AD[26]
AN9	IDSEL
AN10	GND
AN11	AD[17]
AN12	IRDY*
AN13	GND
AN14	C/BE[1]*
AN15	AD[12]
AN16	GND
AN17	AD[06]
AN18	AD[03]
AN19	GND
AN20	C/BE[6]*
AN21	AD[63]
AN22	GND
AN23	AD[56]
AN24	AD[52]
AN25	GND
AN26	AD[45]

Pin Number	Pin Label
AN27	AD[41]
AN28	GND
AN29	AD[34]
AN30	GND
AN31	GND
AN32	GND
AN33	GND
AN34	GND
AP1	GND
AP2	GND
AP3	GND
AP4	GND
AP5	INTA*
AP6	REQ*
AP7	AD[29]
AP8	AD[25]
AP9	AD[23]
AP10	AD[20]
AP11	AD[16]
AP12	TRDY*
AP13	PERR*
AP14	AD[15]
AP15	AD[11]
AP16	AD[08]
AP17	AD[05]
AP18	AD[04]
AP19	AD[00]
AP20	C/BE[7]*
AP21	PAR64
AP22	AD[60]
AP23	AD[57]
AP24	AD[53]
AP25	AD[49]
AP26	AD[46]
AP27	AD[42]

Pin Number	Pin Label
AP28	AD[38]
AP29	AD[35]
AP30	VGG
AP31	GND
AP32	GND
AP33	GND
AP34	GND

1.13 CX28500 Hardware Signals Description

CX28500 is packaged in a 35 mm × 35 mm, 580-pin BGA.

The pin input/output functions are defined in Table 1-7.

Pin labels, signal names, I/O functions, and signal definitions are provided in Table 1-8. An active low signal is always denoted with a trailing asterisk (*).

Table 1-7. I/O Pin Types

I/O	Definition
I	Input. High impedance, TTL.
O	Output. CMOS.
I/O	Input/Output. TTL input/CMOS output.
t/s	Three-state. Bidirectional three-state input/output pin.
s/t/s	Sustained three-state. This is an active-low, three-state signal owned by only one driver at a time. The driver that drives an s/t/s signal low must drive it high for at least one clock cycle before allowing it to float. A pullup is required to sustain the deasserted value.
o/d	Open drain. This output is driven low only, and a pullup is required to sustain the deasserted value.
FOOTNOTE: All outputs are CMOS drive levels and can be used with CMOS or TTL-logic.	

Table 1-8. RS28500 Hardware Signal Definitions *(1 of 7)*

	Pin Label	Signal Name	I/O	Definition
Expansion Bus Interface	ECLK	Expansion Bus Clock	t/s O	The EBUS clock can be configured to operate at the PCI clock rate (i.e., 33 MHz or 66 MHz), or at half of the PCI clock rate (i.e., 33/2 MHz or 66/2 MHz), in which case the PCI clock is divided by two (see field EC_KDIV in Table 6-21).
	EAD[31:0]*(6)*	Expansion Bus Address and Data	t/s I/O	EAD[31:0] is a multiplexed address/data bus.
	EBE[3:0]*	Expansion Bus Byte Enables	s/t/s O	EBE* contains byte-enabled information for the EBUS transaction. For details on how these signals are used and controlled by the Host, see Chapter 4.
	WR* (R/WR*)	Write Strobe	s/t/s O	High-to-low transition enables write data from CX28500 into peripheral device. Rising edge defines write. (In Motorola® mode, R/WR* is held high throughout read and held low throughout write. Determines meaning of DS* strobe.)
	RD* (DS*)	Read Strobe	s/t/s O	High-to-low transition enables read data from peripheral into CX28500. Held high throughout write operation. (In Motorola mode, DS* transitions low for both read and write operations and is held low throughout the operation.)
	ALE (AS*)	Address Latch Enable	s/t/s O	High-to-low transition indicates that EAD[30:0] bus contains valid address. Remains asserted low through the data phase of the EBUS access. (In Motorola mode, high-to-low transition indicates EBUS contains a valid address. Remains asserted for the entire access cycle.)
	HOLD (BR*)	Hold Request (Bus Request)	t/s O	When asserted, CX28500 requests control of the EBUS.
	HLDA (BG*)	Hold Acknowledge (Bus Grant)	I	When asserted, CX28500 has access to the EBUS. It is held asserted when there are no other masters connected to the bus, or asserted as a handshake mechanism to control EBUS arbitration. If in Intel mode, then it should be pulled-down. If in Motorola mode, then it should be pulled-up.
	BGACK*	Bus Grant Acknowledge	t/s O	When asserted, CX28500 acknowledges to the bus arbiter that the bus grant signal was detected and a bus cycle is sustained by CX28500 until this signal is deasserted. This is used in Motorola mode only.

Table 1-8. RS28500 Hardware Signal Definitions *(2 of 7)*

	Pin Label	Signal Name	I/O	Definition
Serial Interface	TCLK[31:0]	Transmit Clock	I	If the serial port is configured in conventional mode, then TCLK controls the rate at which data is transmitted and synchronizes transitions for TDATx and sampling of TSYNCx and CTSx, if CTSxis enabled. If the port is configured as a TSBUS port, then TCLK controls the rate at which data is transmitted and synchronizes transitions for TDATx and sampling of TSTUFFx and STBx (STBx only for transmit circuitry).
	TSYNC[31:0]/ TSTUFF[31:0]	Transmit Synchronization/ TSBUS Transmit Stuff	I	If the serial port is configured in conventional mode, then this signal is defined as TSYNC. TSYNC is sampled on the specified active edge of the corresponding TCLKx clock. (See TSYNC_EDGE bit field in Table 6-36, *TSIU Port Configuration Register*.) When TSYNCx signal goes from low to high then the start of transmit frame is indicated. TSYNCx is ignored if the serial port is configured to operate in unchannelized mode. If the serial port is configured in T1 mode, then the corresponding data bit that latched out during the same bit time period (but not necessarily sampled at the same clock edge) is the F-bit of the T1 frame. If the serial port is configured in channelized mode, then the corresponding data bit that latched out during the same bit time period (but not necessarily sampled at the same clock edge) is Bit-0 of the first time slot of the N×64 frame. Since CX28500's flywheel mechanism is always used in channelized mode, no other synchronization signal is required to track the start of each subsequent frame. However, if a SYNC pulse, if it exists, occurs anywhere but at the frame boundaries, which are tracked by the flywheel mechanism, a COFA is generated for that port. If the port is configured to operate as TSBUS port, then this signal is defined as TSTUFF. The TSTUFF values are to either stuff (no TDAT output) or not stuff (TDAT valid). TSTUFF is sampled on the specified active edge of the corresponding TCLKx. (See TSTUFF_EDGE bit field in Table 6-36, *TSIU Port Configuration Register*.) If the serial port operates in channelized TSBUS mode, then TSTUFF assertion indicates that no data needs to be transmitted in the 8th time slot after the assertion of the TSTUFF. Note that while operating in channelized TSBUS mode, CX28500 requires the following: **1.** The stuff status for each time slot to be presented at its TSTUFF input exactly eight time slots in advance of the actual time slot for which the stuff status is to be applied. The amount of the TSTUFF advance is fixed at eight time slots, even though the number of time slots within a frame may vary. **2.** Assertion of this signal anytime during the time slot is required. But, for good practice, it is recommended that this signal is asserted within the first two bits of the time slot.

Table 1-8. RS28500 Hardware Signal Definitions *(3 of 7)*

	Pin Label	Signal Name	I/O	Definition
Serial Interface (Continued)	TDAT[31:0]	Transmit Data	t/s O	Serial data latched out on active edge of transmit clock, TCLKx. If channel is unmapped to time slot, data bit is considered invalid and CX28500 outputs either three-state signal or logic 1 depending on TRITx bit field value in Table 6-36, *TSIU Port Configuration Register*. To avoid collision with other drivers, this signal is three-stated until the detection of the first TSYNC pulse, and during COFA.
	RCLK[31:0]	Receive Clock	I	If the serial port is configured in conventional mode, then RCLK controls the rate at which data is transmitted and synchronized transitions for RDATx and sampling of RSYNCx and SPORT, if SPORT is enabled. If the port is configured as a TSBUS port, then RCLK controls the rate at which data is received and synchronizes transitions for RDATx and sampling of RSTUFFx.
	RSYNC[31:0]/ RSTUFF[31:0]	Receive Synchronization/ Receive Stuff	I	If the port operates in a conventional mode, then this signal is defined as RSYNC. RSYNC is sampled on the specified active edge of the corresponding receive clock, RCLKx. (See RSYNC_EDGE bit field in Table 6-28, *RSIU Port Configuration Register.)* RSYNCx is ignored if the serial port is configured to operate in unchannelized mode. If RSYNCx signal goes from low to high, then the start of a receive frame is indicated. For T1 mode, the corresponding sampled and stored data bit during the same bit-time period (not necessarily sampled on the same clock edge) is the F-bit. For the channelized mode, the corresponding data bit sampled and stored during the same bit-time period (not necessarily sampled on the same clock edge) is Bit-0 of the first time slot of the N×64 frame. Since CX28500's flywheel mechanism is always used in channelized mode, no other synchronization signal is required to track the start of each subsequent frame. However, if a SYNC pulse, if it exists, occurs anywhere but at the frame boundaries, which are tracked by the flywheel mechanism, a COFA is generated for that port. If the port operates as a TSBUS port then this signal is RSTUFF. The RSTUFF is sampled on the specified active edge of the corresponding TCLKx. (See RSTUFF_EDGE bit field in Table 6-28, *RSIU Port Configuration Register.)* RSTUFF in this case assertion indicates that this time slot contains no data.
	RDAT[31:0]	Receive Data	I	Serial data sampled on active edge of receive clock, RCLKx. If the channel is mapped to a time slot, input bit is sampled and transferred to memory. If the channel is unmapped to time slot, data bit is considered invalid and CX28500 ignores the received sample.

Table 1-8. RS28500 Hardware Signal Definitions *(4 of 7)*

	Pin Label	Signal Name	I/O	Definition
Serial Interface (Continued)	ROOF[31:0][(2)(3)(5)] /ROOF/ CTS[31.0][(2)(3)(5)] /STB/ SPORT[31:0][(2)(4)(5)]	Receiver Out-Of-Frame/ Channelized Clear To Send/TSBUS Strobe	I	If ROOFx, the signal is sampled on the specified active edge of the corresponding receive clock RCLKx (*see ROOF_EDGE bit field in* Table 6-28, *RSIU Port Configuration Register*). When it is asserted high, an Out-Of-Frame (OOF) condition interrupt is indicated, if OOFIEN bit field is set to 1 in *RSIU Port Configuration Register*. While ROOFx is asserted, the received serial data stream is considered Out-Of-Frame. If OOFABT bit field is set to 1 in Table 6-28, *RSIU Port Configuration Register*, the receive process is disabled for the entire port and it remains disabled until ROOFx is deasserted, otherwise, the receive process is enabled. Upon ROOFx deassertion, if OOFIEN bit field is set to 1 in *RSIU Port Configuration Register*, an interrupt Frame Recovery (FREC) is generated. The data processing resumes for all affected channels. This signal can also operate as a general Serial Port Interrupt (SPORT) by clearing the OOFABT bit field and setting the OOFIEN bit field in *RSIU Port Configuration Register* (i.e., OOFABT = 0 and OOFIEN = 1). When the ROOFx signal transitions from high-to-low (deassertion), a SPORT interrupt is generated and data stream is not affected. If this signal is used as a general purpose interrupt, no interrupt is generated until this signal goes from high to low. If CTSx, the signal is sampled on the specified active edge of the corresponding transmit clock, TCLKx. (*See CTS_EDGE bit field in* Table 6-28, *RSIU Port Configuration Register*.) If CTS transitions from high-to-low (is deasserted), then the channel assigned to the time slot will send continuous idle characters after the current message has been completely transmitted. The message transmission data continues when this CTS transitions from low to high again (is asserted). The response time to CTS is a 32 bit-time, meaning that a new message might be transmitted if the message starts within the next 32 bits after CTS was deasserted. If TSTBx, the signal is sampled twice: 1. Once by the receive circuitry on the specified edge of the corresponding receive clock, RCLKx. (*See RSTB_EDGE bit field in* Table 6-28, *RSIU Port Configuration Register*.) 2. Once by the transmit circuitry on the specified edge of the corresponding transmit clock, TCLKx. (*See TTSTB_EDGE bit field in* Table 6-36, *TSIU Port Configuration Register*.) If TSTB transitions from low to high assertion, then it marks the first bit of time slot 0 within the TSBUS frame. Since there is a single TSTB for both directions, receive and transmit, the number of configured time slots (*RSIU Time Slot Pointers Assignment Register and TSIU Time Slot Pointers Register)* and the RPORT_TYPE or TPORT_TYPE *value (RSIU Port Configuration Register and TSIU Port Configuration Register*) specifying whether the serial port operates in channelized or unchannelized mode must be identically configured for both directions per serial port. Unexpected CX28500 behavior may be generated if this restriction is violated.

Table 1-8. RS28500 Hardware Signal Definitions *(5 of 7)*

	Pin Label	Signal Name	I/O	Definition
PCI Interface	AD[63:0]	PCI Address and Data	t/s I/O	AD[63:0] is a multiplexed address/data bus. A PCI transaction consists of an address phase during the first clock period followed by one or more data phases. AD[7:0] is the LSB. As a master, CX28500 supports both 32- and 64-bit operations. As a target, it supports only 32-bit operations.
	PCLK	PCI Clock	I	PCLK provides timing for all PCI transitions. All PCI signals except PRST*, INTA*, and INTB* are synchronous to PCLK and are sampled on the rising edge of PCLK. CX28500 supports a PCI clock up to 66 MHz.
	PRST*	PCI Reset	I	This input resets all functions on CX28500.
	CBE[7:0]*	PCI Command and Byte Enables	t/s I/O	During the address phase, CBE[3:0]* contain command information while CBE[7:4]* are unused; during the data phases, CBE[7:0]* contain information denoting which byte lanes are valid. PCI commands are defined as follows: CBE[3:0] — Command Type Oh 0000b Interrupt Acknowledge 1h 0001b Special Cycle 6h 0110b Memory Read 7h 0111b Memory Write Ah 1010b Configuration Read Bh 1011b Configuration Write Ch 1100b Memory Read Multiple Dh 1101b Dual Address Cycle Eh 1110b Memory Read Line Fh 1111b Memory Write and Invalidate Note that the CX28500 does not accept target (slave) transactions if the CBE bits are not all 0 (zeroes). Such transactions successfully complete on the PCI bus, but are silently ignored by the CX28500 device. The CBE bits function normally during all other times.
	PAR	PCI Parity	t/s I/O	The number of 1s on PAR, AD[31:0], and CBE[3:0]* is an even number. PAR always lags AD[31:0] and CBE* by one clock. During address phases, PAR is stable and valid one clock after the address; during the data phases it is stable and valid one clock after TRDY* on reads and one clock after IRDY* on writes. It remains valid until one clock after the completion of the data phase.
	PAR64	PCI MSB parity	t/s I/O	Same as PAR for CBE[7:4]* and AD[63:32].
	FRAME*	PCI Frame	s/t/s I/O	FRAME* is driven by the current master to indicate the beginning and duration of a bus cycle. Data cycles continue as FRAME* stays asserted. The final data cycle is indicated by the deassertion of FRAME*. For a non-burst, one-data-cycle bus cycle, this pin is only asserted for the address phase.
	REQ64*	Request 64-bit Transfer	t/s I/O	CX28500 asserts this signal when it needs to perform a 64-bit transfer. This signal is used during PCI reset to inform the system that the PCI is 64-bits wide.

Table 1-8. RS28500 Hardware Signal Definitions *(6 of 7)*

	Pin Label	Signal Name	I/O	Definition
PCI Interface (Continued)	ACK64*	Acknowledge 64-bit Transfer	s/t/s I/O	ACK64* asserted indicates the selected target is ready to perform a 64-bit transaction.
	STOP*	PCI Stop	s/t/s I/O	STOP* asserted indicates the selected target is requesting the master to stop the current transaction.
	IRDY*	PCI Initiator Ready	s/t/s I/O	IRDY* asserted indicates the current master's readiness to complete the current data phase.
	TRDY*	PCI Target Ready	s/t/s I/O	TRDY* asserted indicates the target's readiness to complete the current data phase.
	DEVSEL*	PCI Device Select	s/t/s I/O	When asserted, DEVSEL* indicates that the driving device has decoded its address as the target of the current cycle.
	IDSEL	PCI Initialization Device Select	I	This input is used to select CX28500 as the target for configuration read or write cycles.
	SERR*	System Error	o/d O	Any PCI device can assert SERR* to indicate a parity error on the address cycle or parity error on the data cycle of a special cycle command or any other system error where the result will be catastrophic. CX28500 will assert SERR* if it detects a parity error on the address cycle or encounters an abort condition while operating as a PCI master. Since SERR* is not an s/t/s signal, restoring it to the deasserted state is done with a weak pullup (same value as used for s/t/s). Note that CX28500 does not input SERR*. It is assumed that the Host will reset CX28500 in the case of a catastrophic system error.
	PERR*	Parity Error	s/t/s I/O	PERR* is asserted by the agent receiving data when it detects a parity error on a data phase. It is asserted one clock after PAR is driven, which is two clocks after the AD and CBE* parity was checked. If CX28500 masters a PCI write cycle and—after supplying the data during the data phase of the cycle—detects this signal being asserted by the agent receiving the data, then CX28500 generates a PERR interrupt. If CX28500 masters a PCI read cycle and—after receiving the data during the data phase of the cycle—calculates that a parity error has occurred, CX28500 asserts this signal and also generates the PERR Interrupt Descriptor towards the Host.
	INTA*	PCI CX28500 Interrupt	o/d O	INTA* is driven by CX28500 to indicate a CX28500 Layer 2 interrupt condition to the Host processor.
	REQ*	PCI Bus Request	t/s O	CX28500 drives REQ* to notify the PCI arbiter that it desires to master the bus. Every master in the system has its own REQ*.
	GNT*	PCI Bus Grant	I	The PCI bus arbiter asserts GNT* when CX28500 is free to take control of the bus, assert FRAME*, and execute a bus cycle. Every master in the system has its own GNT*.
	RBS [4:0]	PCI Read Burst Size	o/d 0	Reports the number of dwords CX28500 attempts to read during its subsequent master burst read transaction. These lines are driven only during address phase of the master read transaction, where a value of 00000 means one-dword and a value of 11111 means 32 dwords.

Table 1-8. RS28500 Hardware Signal Definitions *(7 of 7)*

	Pin Label	Signal Name	I/O	Definition
Boundary Scan and Test Access	TCK	JTAG Clock	I	Used to clock in the TDI and TMS signals and as clock out TDO signal.
	TRST*	JTAG Enable	I	An active-low input used to put the chip into a special test mode. This pin should be pulled up in normal operation.
	TMS	JTAG Mode Select	I	The test signal input decoded by the TAP controller to control test operations.
	TDO	JTAG Data Output	t/s O	The test signal used to transmit serial test instructions and test data.
	TDI	JTAG Data Input	I	The test signal used to receive serial test instructions and test data.
	TM[0] TM[1] TM[2]	Test Mode	I	Encodes test modes: all 0's in normal mode. TM[0] TM[1] TM[2] Description 0 0 0 Normal Operation. Tie to ground. 1 1 1 All outputs three-stated.
Power and Ground	VDD_c VDD_io	Power	—	VDD_c is power supply for internal logics. VDD_io is power supply for input and output.
	VGG	Input Tolerance	—	ESD diode clamp supply for 5 volts tolerant input where VGG = 5 volts, otherwise VGG = VDDi = VDDo = 3.3 volts.
	GND	Ground	—	Ground pins for internal logics, input, and output are connected to a common ground.
No Connection	No Connect	No Connect	—	These pins have no connection. They are reserved for future revisions.

FOOTNOTE:

(1) While operating in TSBUS mode, there is no damage expected when sampling STBx twice, since the RCLKx and TCLKx are the same signals for a specific port. However, this may require some additional restrictions for the board designers when these clocks are routed.

(2) This signal is used either as Receiver Out-Of-Frame or a Transmit Clear to Send or a TSBUS strobe. (OOF/FREC behavior selected by OOFABT = 1, CTS behavior selected by CTSENB = 1, STB behavior selected by TPORT_TYPE or RPORT_TYPE.) See related bit fields configuration (i.e., *RSIU Port Configuration Register* and *TSIU Port Configuration Register*). ROOF/CTS/STB/SPORT signals are from the same pin.

(3) If the serial port operates in conventional mode, then this signal is used either as a ROOFx or CTSx signal.

(4) If the port operates channelized TSBUS mode, then the signal is used as the TSBUS strobe signal, which indicates the beginning of the TSBUS frame.

(5) Only one pin in the device defines all these functions.

(6) The address line A31 must be asserted in all transactions.

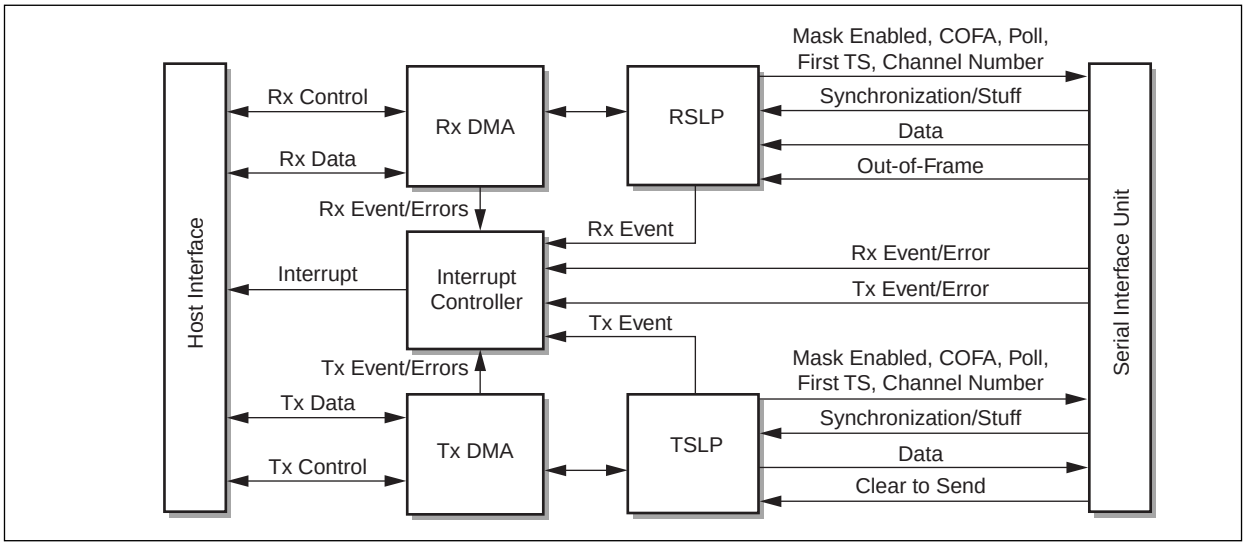
2 Internal Architecture

The CX28500 consists of the following functional blocks:

- Serial Interface Unit (SIU)—TSIU and RSIU for the transmit and receive directions
- Serial Line Processing (SLP)—TSLP and RSLP for the transmit and receive directions
- Direct Memory Access Controller (DMA)
- Interrupt Controller (INTC)

Figure 2-1 illustrates the different signal connection between the SIU and the host interface while the CX28500 is configured to operate in conventional or channelized TSBUS mode.

Figure 2-1. Serial Interface Functional Block Diagram



2.1 Serial Interface Unit (SIU)

The SIU is the module that logically connects the 32 serial ports (line interface unit) with serial line processing by performing the serial-to-parallel conversion for the receive side, and parallel to serial for the transmit side. The SIU contains two main blocks, RSIU for the receive path and TSIU for the transmit path. The RSIU main function is multiplexing 32 serial ports into one logical port for the receive serial line processing block. The TSIU main function is demultiplexing one logical port from the transmit serial line processing block to 32 serial ports.

The SIU main functions:

- Multiplexing/demultiplexing 32 serial ports to one port for the receive path, and one port to 32 serial ports for the transmit path, respectively.
- Performs frame integrity check while operating in channelized mode. SIU verifies the length of incoming/outgoing frames according to the configured number of time slots. In case of error a Change Of Frame Alignment (COFA) is reported (see *COFA description both modes convention and TSBUS mode*).
- Translates the time slot to logical channel number using the configured receive and transmit time slot map.
- Provides a poll indication per port, see RPOLLTH or TPOLLTH bit fields in RSIU Port Configuration register and TSIU Port Configuration register, which counts the frames of the poll interval of 1, 2, 4, 8, 16, 64, 128, or 256 frame interval.
- Generates the following interrupts:
  - RxOOF, where ROOF signal is asserted
  - RxFREC, where ROOF signal is deasserted
  - RxCOFA, where RSYNC signal is asserted in an unexpected place
  - RxCREC, where COFA condition is off for the receive port
  - TxCOFA, where TSYNC signal is asserted in an unexpected place
  - TxCREC, where COFA condition is off for the transmit port

TSIU (Transmit Serial Interface Unit) is responsible for informing TSLP (Transmit Serial Line Processor) when it is time to perform a transmit channel poll by counting a programmable (TPOLLTH) number of frames. For the TSBUS mode, one frame is defined as the STB strobe interval. TSIU is also responsible for requesting transmit data from TSLP for each transmitted time slot (i.e., for each VSP).

As for the RSIU, when data comes through the serial ports, it is stored in local buffers. At this stage, all line signals are synchronized to the system clock, or the PCI bus rate (either 33 MHz or 66 MHz). At each cycle, the RSIU transfers a byte received from one of the 32 ports to the Receive Serial Line Processor (RSLP). For the port being served, the RSIU translates the time slot to a channel number (using an internal map) and provides certain parameters that are needed by the RSLP to process the incoming data. Similar to the TSIU, for the TSBUS mode, one frame is defined as the STB strobe interval.

2.2 Serial Line Processor (SLP)

The serial line processors (RSLP and TSLP) service the bytes in the receive and transmit path. The SLP coordinates all byte-level transactions between SIU and DMA. The SLP also interacts with the INTC to notify the Host of events and errors during the serial line processing.

The RSLP main functions are the following:

- HDLC mode handling
  - Search opening and closing flag (7Eh)
  - Abort detection (7Fh)
  - Check max/min message length
  - Verify byte alignment
  - Check FCS
  - Detect change of pad-fill
  - Zero deletion
- Transparent mode
  - Start to receive data from the first time slot assigned to the logical channel
- Bit polarity inversion
- Handle channel activation/deactivation
- Handle OOF/COFA and overflow
- Invert incoming data
- Handle subchanneling
- Interrupts
  - BUFF, SHT, CHIC, CHABT

The TSLP main functions are the following:

- HDLC mode handling
  - Generate opening/closing /shared flag (7Eh)
  - Zero insertion after five consecutive 1 s bit-stuffing
  - Generate FCS depending upon the protocol
  - Handle CTS
  - Generate pad fill between frames
- Transparent mode
  - Start to transmit data from the first time slot assigned to the logical channel
  - Generate pad fill between messages
- Handle channel activation/deactivation
- Handle COFA and under-run
- Invert outgoing data
- Handle subchanneling
- Interrupts
  - BUFF, and EOM

2.3 Direct Memory Access Controller

The direct memory access controller (RxDMA and TxDMA) manages all of the memory operations between a correspondent's SLP and the Host interface. DMA takes requests from SLP to either fill or flush internal FIFO buffers, sets up an access to the data buffers in shared memory, and requests access to the PCI bus through the Host interface.

2.3.1 General Feature List

- Handles 1024 logical channels.
- Supports 32- or 64-bit DMA transactions for 32- or 64-bit PCI bus transactions, respectively.
- Configurable Internal Buffer Allocation
- Full control of internal buffer size and internal buffer threshold per channel
  - FIFO flushing capability (after soft chip reset, channel activation, and channel deactivation service request)
- Buffer Descriptor Handling
  - Separate Buffer Descriptors per channel
  - LAST bit field set in buffer descriptor indicates the Buffer Descriptor table length (maximum length is 4096 Buffer Descriptors entries)
  - Automatic fetch of Transmit and Receive Head Pointer Table (THPT and RHPT)
  - Automatic fetch of Transmit Head pointer Table (THPT) and Receive Head Pointer Table (RHPT) enables the Buffer Descriptor tables switching without interfering normal operation (channel Jump)
- Autonomous management of Buffer Descriptors
- Automatic fetch for next Buffer Descriptor
- Automatic Buffer Status Descriptor update and interrupts for End Of Buffer (EOB), Ownership (ONR) and End Of Message (EOM) conditions
  - Automatic polling of Buffer Descriptor
- Complete Buffer Control
  - Buffer Pointer
  - Buffer Length
  - Ownership (Host/CX28500)
  - End Of Buffer Interrupt Mask (EOBIEN)
  - Poll/No poll Control
- Self-service mechanism
  - Zero host intervention required
  - Support self-service for receive to transmit loopback
  - Automatic Tx Abort Command generation

Per Channel Configuration

- INHRBD—Status descriptor update/ignore
- EOMIEN—Error-free End Of Message (EOM) interrupt enable/disable
- ERRIEN Errored EOM enable/disable
- ONRIEN—Ownership error

Interrupts

- EOM—End Of Service Request without an error
- EOM—End Of Message with error (Overflow, OOF, COFA, FCS, ALIGN, ABT LNG)
- ONR—Ownership error
- EOB—End Of Buffer

2.4 Interrupt Controller

The Interrupt Controller takes receive and transmit events/errors from RSIU, RSLP, RxDMA and TSIU, TSLP, and TxDMA respectively. The Interrupt Controller coordinates the transfer of internally queued descriptors to an interrupt queue in shared memory and coordinates notification of pending interrupts to the Host.

3 Host Interface

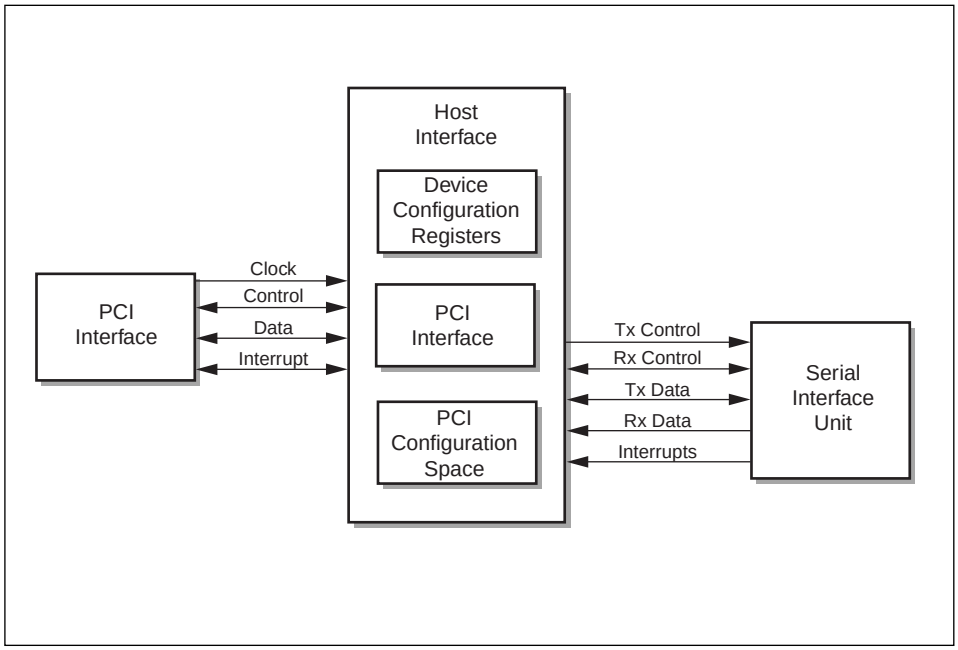
CX28500's Host interface performs the following major functions:

- Transfers data between the serial interface and shared memory over the PCI bus.
- Stores configuration state information.

CX28500 does not support Function 1. The access to the EBUS is performed via a service request command, which provides a better utilization of the PCI bus for local devices located on the EBUS.

Figure 3-1 illustrates the Host interface block diagram.

Figure 3-1. Host Interface Functional Block Diagram



3.1 PCI Interface

The Host interface in CX28500 is compliant with the *PCI Local Bus Specification* 2.1. CX28500 provides a PCI interface specific to 3.3 V and 33/66 MHz operation and supports as master a 32-bit or 64-bit bus with multiplexed address and data lines, and as a slave, a 32-bit PCI bus.

NOTE: The *PCI Local Bus Specification* (Revision 2.1, June 1, 1995) is an architectural, timing, electrical, and physical interface standard that provides a mechanism for a device to interconnect with processor and memory systems over a standard bus.

The Host interface can act as a PCI master and a PCI slave, and contains CX28500's PCI configuration space and internal registers. When CX28500 needs to access shared memory, it masters the PCI bus and completes the memory cycles without external intervention.

3.1.1 PCI Initialization

Generally, when a system initializes a module containing a PCI device, the configuration manager reads the configuration space of each PCI device on a PCI bus. Hardware signals select a specific PCI device based on a bus number, a slot number, and a function number. If a device that is addressed (via signal lines) responds to the configuration cycle by claiming the bus, then that function's configuration space is read out from the device during the cycle. Since any PCI device can be a multifunction device, every supported function's configuration space needs to be read from the device. Based on the information read, the configuration manager will assign system resources to each supported function within the device. Sometimes new information needs to be written into the function's configuration space. This is accomplished with a configuration write cycle.

CX28500 is a single function device that has device-resident memory to store the required configuration information. CX28500 supports Function 0 only.

3.1.2 PCI Bus Operations

CX28500 behaves either as a PCI master or a PCI slave device at any time and switches between these modes as required during device operation. CX28500 supports only dword write transactions.

As a PCI slave, CX28500 responds to the following PCI bus operations:

- Memory Read
- Memory Write
- Configuration Read
- Configuration Write
- Memory Read Multiple (treated like Memory Read in slave mode)
- Memory Read Line (treated like Memory Read in slave mode)
- Memory Write and Invalidate (treated like Memory Write)

NOTE: As a PCI slave, CX28500 does not support bursted read or write PCI transactions.

As a PCI master, CX28500 generates the following PCI bus operations:

- Memory Read
- Memory Read Line
- Memory Read Multiple (generated only in master mode)
- Memory Write

3.1.3 Fast Back-to-Back Transactions

Fast back-to-back transactions allow agents to utilize bus bandwidth more effectively. CX28500 supports PCI fast back-to-back transactions both as a bus target and bus master. CX28500 can also execute fast back-to-back transactions regardless of the PCI configuration settings (for details see *bit 9 TARGET_FBTB bit field, in* Table 6-20, *Global Configuration Descriptor*).

Fast back-to-back transactions are allowed on PCI when contention on TRDY*, DEVSEL*, STOP*, or PERR* is avoided.

CX28500, as a master supporting fast back-to-back transactions, places the burden of avoiding contention on itself. While acting as a slave, CX28500 places the burden on all the potential targets. As a master, CX28500 may remove the Idle state between transactions when it can guarantee that no contention occurs. This can be accomplished when the master's current transaction is to the same target as the previous transaction. While supporting this type of fast back-to-back transaction, CX28500 understands the address boundaries of the potential target, so that no contention occurs. The target must be able to detect a new assertion of FRAME* without the bus going to Idle state.

3.1.3.1 Operation Mode

During a fast back-to-back transaction, the master starts the next transaction if GNT* is still asserted. If GNT* is deasserted, the master has lost access to the bus and must relinquish the bus to the next master. The last data phase completes when FRAME* is deasserted, and IRDY* and TRDY* (or STOP*) are asserted. The current master starts another transaction on the clock following the completion of the last data phase of the previous transaction. During fast back-to-back transaction, only the master and target involved need to distinguish intermediate transaction boundaries using only FRAME* and IRDY* (there is no bus Idle state). When the transaction is over, all the agents see an Idle state.

3.1.3.2 Example of an Arbitration for Fast Back-to-Back and Non-Fast Back-to-Back Transactions

Appendix B shows an example of an arbitration for fast back-to-back and non-fast back-to-back transactions. The transactions shown are bursts of 2, 3, 4, 5, or 6 dwords read-write transferred while the address-data is either 32-bit or 64-bit wide.

3.1.4 PCI Configuration Space

This section describes how CX28500 implements the required PCI configuration register space. The intent of PCI configuration space definition is to provide an appropriate set of configuration registers that satisfy the needs of current and anticipated system configuration mechanisms, without specifying those mechanisms or otherwise placing constraints on their use. These registers allow for the following:

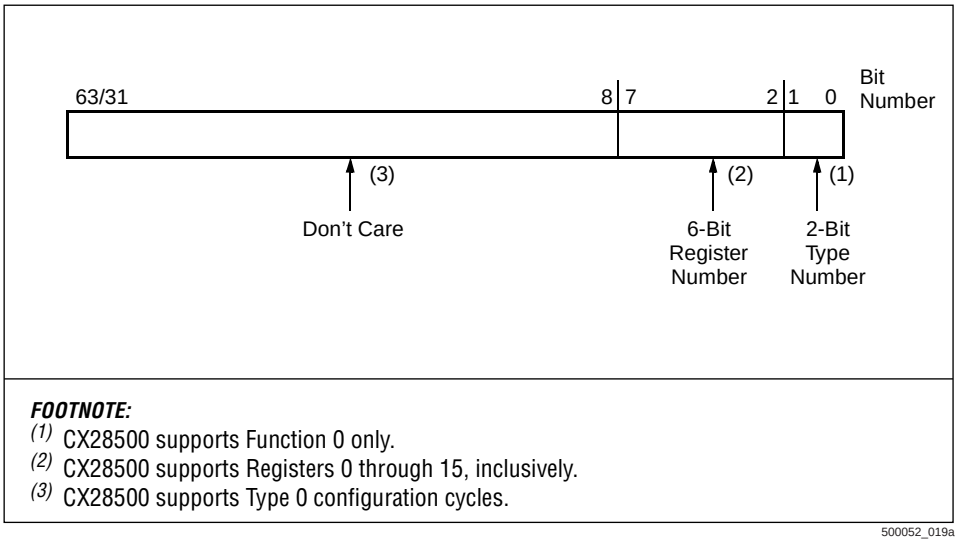
- Full device relocation, including interrupt binding
- Installation, configuration, and booting without user intervention
- System address map construction by device-independent software

CX28500 only responds to Type 0 configuration cycles. Type 1 cycles, which pass a configuration request on to another PCI bus are ignored.

CX28500 is a single function PCI agent; therefore, it implements configuration space for Function 0 only.

The address phase during a CX28500 configuration cycle indicates the function number and register number being addressed, which can be decoded by observing the status of the address lines AD[63:0]. Figure 3-2 illustrates the address lines during configuration cycle.

Figure 3-2. Address Lines During Configuration Cycle



The value of the signal lines AD[10:8] selects the function being addressed. Since CX28500 supports Function 0 only, it ignores these bits.

The value of the signal lines AD[7:2] during the address phase of configuration cycles selects the register of the configuration space to access. Valid values are 0 through 15. Accessing registers outside this range results in an all 0s value being returned on reads, and no action being taken on writes.

The value of the signal lines AD[1:0] must be 00b for CX28500 to respond. If these bits are 0 and the IDSEL* signal line is asserted, then CX28500 responds to the configuration cycle.

The Base Code register contains the Class Code, Sub Class Code, and Register Level Programming Interface registers. Table 3-1 illustrates the PCI Configuration Space.

Table 3-1. PCI Configuration Space

Register Number	Byte Offset (hex)	31 24	16	8	0
0	00h	Device ID		Vendor ID	
1	04h	Status		Command	
2	08h	Base Code			Revision Id
3	0Ch	Reserved	Header Type	Latency Timer	Reserved
4	10h	CX28500 Base Address Register (BAR)			
5	14h	—			
—	—	Reserved			
14	38h	—			
15	3Ch	Max Latency	Min Grant	Interrupt Pin	Interrupt Line

All writable bits in the configuration space are reset to 0 by the hardware reset, PRST* asserted. After reset, CX28500 is disabled and only responds to PCI configuration write and PCI configuration read cycles. Write cycles to reserved bits and registers have no effect. Read cycles to reserved bits always result in 0 being read.

3.2 PCI Configuration Registers

3.2.1 PCI Master and Slave

CX28500 is a single function PCI device that provides the necessary configuration space for a PCI bus controller to query and configure CX28500's PCI interface. PCI configuration space consists of a device-independent header region (64 bytes) and a device-dependent header region (192 bytes). CX28500 provides the device-independent header section only. Access to the device-dependent header region results in 0s being read, and no effect on writes.

Three types of registers are available in CX28500:

1. Read-Only (RO)—Return a fixed bit pattern if the register is used or a 0 if the register is unused or reserved.
2. Read-Resettable (RR)—Can be reset to 0 by writing a 1 to the register.
3. Read/Write (RW)—Retain the value last written to it.

Sixteen dword registers make up CX28500's PCI Configuration Space. Tables 3-2 through 3-8 specify the contents of these registers.

3.2.1.1 Register 0, Address 00h

Table 3-2. Register 0, Address 00h

Bit Field	Name	Reset Value	Type
31:16	Device Id	8500h—64 channels 8501h—384 channels 8502h—676 channels 8503h—1024 channels	RO
15:0	Vendor Id	14F1h	RO

3.2.1.2 Register 1, Address 04h

The Status register records status information for PCI bus related events. The Command register provides coarse control to generate and respond to PCI commands.

At reset, CX28500 sets the bits in this register to 0, meaning CX28500 is logically disconnected from the PCI bus for all cycle types except configuration read and configuration write cycles.

Table 3-3. Register 1, Address 04h *(1 of 2)*

Bit Field	Name	Reset Value	Type	Description
31	Status	0	RR	Detected Parity Error. This bit is set by CX28500 whenever it detects a parity error, even if parity error response is disabled.
30		0	RR	Detected System Error. This bit is set by CX28500 whenever it asserts SERR*.
29		0	RR	Received Master Abort. This bit is set by CX28500 whenever a CX28500-initiated cycle is terminated with master-abort.
28		0	RR	Received Target Abort. CX28500 sets this bit when a CX28500-initiated cycle is terminated by a target-abort.
27		0	RO	Unused.
26:25		01b	RO	DEVSEL Timing. Indicates CX28500 is a medium-speed PCI device. This means the longest time it will take CX28500 to return DEVSEL* when it is a target is 3 clocks.
24		0	RR	Data Parity Detected. CX28500 sets this bit when three conditions are met: 1. CX28500 asserted PERR* or observed PERR*. 2. CX28500 was the master for that transaction. 3. Parity Error Response bit is set.
23		1b	RO	Fast Back-to-Back Capable. Read Only. Indicates that when CX28500 is a target, it is capable of accepting fast back-to-back transactions when the transactions are not to the same agent.
22		0	RO	Unused.
21		1b	RO	66 MHz Capable. Read Only. Indicates the PCI interface is capable of operating at 66 MHz rate.
20:16		0	RO	Unused.

Table 3-3. Register 1, Address 04h *(2 of 2)*

Bit Field	Name	Reset Value	Type	Description
15:10	Command	0	RO	Unused.
9		0	RW	Fast back-to-back enable. This bit controls whether or not CX28500, while acting as master, can perform fast back-to-back transactions to different devices. The configuration software routine sets this bit if all bus agents in the system are fast back-to-back capable. • If 1, CX28500 can generate fast back-to-back transactions to different agents. • If 0, CX28500 can generate fast back-to-back transactions to the same agent. Note that this bit would be presumably set by the system configuration routine after ensuring that all targets on the same bus had the Fast Back-to-Back Capable Bit set. If the target is unable to provide the fast back-to-back capability, the target does not implement this bit and it is automatically retuned as zero when Status register is read.
8		0	RW	SERR enable. • If 1, disables CX28500's SERR* driver. • If 0, enables CX28500's SERR* driver and allows reporting of address parity errors.
7		0	RO	Wait cycle control. CX28500 does not support address stepping.
6		0	RW	Parity error response. This bit controls CX28500's response to parity errors. • If 1, CX28500 takes normal action when a parity error is detected on a cycle as the target. • If 0, CX28500 ignores parity errors.
5		0	RO	VGA palette snoop. Unused.
4		0	RO	Memory write and invalidate. The only write cycle type CX28500 generates is memory write.
3		0	RO	Special cycles. Unused. CX28500 ignores all special cycles.
2		0	RW	Bus master. • If 1, CX28500 is permitted to act as bus master. • If 0, CX28500 is disabled from generating PCI accesses.
1		0	RW	Memory space. Access control. • If 1, enables CX28500 to respond to memory space access cycles. • If 0, disables CX28500's response.
0		0	RO	I/O space accesses. CX28500 does not contain any I/O space registers.

GENERAL NOTE:
1. An active low signal is detected by a trailing asterisk (*).

3.2.1.3 Register 2, Address 08h

This location contains the Class Code and Revision ID registers. The Class Code register contains the Base Code, Sub Class, and Register Level Programming Interface fields. These are used to specify the generic function of CX28500. The Revision ID register denotes the version of the device.

Table 3-4. Register 2, Address 08h

Bit Field	Name	Reset Value	Type	Description
31:24	Class Code	02h	RO	Function: Network Controller.
23:16	Sub Class Code	80h	RO	Type: Other.
15:8	Register Level Programming Interface	0	RO	Indicates there is nothing special about programming CX28500.
7:0	Revision Id	01h	RO	Denotes the revision number of CX28500. This revision ID is divided into two 4-bit fields. Upper nibble indicates Die ID which starts with 0x1 for this device. The lower nibble is used for rev number, Rev A = 1, Rev B = 2, etc.

3.2.1.4 Register 3, Address 0Ch

Table 3-5. Register 3, Address 0Ch

Bit Field	Name	Reset Value	Type	Description
31:24	Reserved	0	RO	Unused.
23:16	Header Type	0h	RO	CX28500 is a single function device with the standard layout of configuration register space.
15:11	Latency Timer	0	RW	The latency timer is an 8-bit value that specifies the maximum number of PCI clocks that CX28500 can keep the bus after starting the access cycle by asserting its FRAME*. The latency timer ensures that CX28500 has a minimum time slot for it to own the bus, but places an upper limit on how long it owns the bus.
10:8		0	RO	
7:0	Reserved	0	RO	Unused.

GENERAL NOTE:
1. An active low signal is detected by a trailing asterisk (*).

3.2.1.5 Register 4, Address 10h

Table 3-6. Register 4, Address 10h

Bit Field	Name	Reset Value	Type	Description
31:20	CX28500 Base Address Register	0	RW	Allows for 1 MB-bounded PCI bus address space to be blocked off as CX28500 space. CX28500 will respond as a PCI slave with DEVSEL* to all memory cycles whose address bits 31:20 match the value of bits 31:20 of this register, and those upper address bits are non-0, and memory space is enabled in the Register 1, COMMAND bit field. Reads to addresses within this space that are not implemented read back 0; writes have no effect.
19:4		0	RO	When appended to bits 31:20, these bits specify a 1 MB bound memory range. 1 MB is the only amount of address space that a CX28500 can be assigned.
3		0	RO	CX28500 memory space is not prefetchable.
2:1		0	RO	CX28500 can be located anywhere in 32-bit address space.
0		0	RO	This base register is a memory space base register, as opposed to I/O mapped.

GENERAL NOTE:
1. An active low signal is detected by a trailing asterisk (*).

3.2.1.6 Register 5–14, Address 14h–38h

Table 3-7. Register 5-14, Address 14h–38h

Bit Field	Name	Reset Value	Type	Description
31:0	Reserved	0	RO	Unused.

3.2.1.7 Register 15, Address 3Ch

Table 3-8. Register 15, Address 3Ch

Bit Field	Name	Reset Value	Type	Description
31:24	Maximum Latency	0Fh	RO	Specifies how quickly CX28500 needs to gain access to the PCI bus. The value is specified in 0.25 μs increments and assumes a 33 MHz clock. A value of 0Fh means CX28500 needs to gain access to the PCI bus every 130 PCI clocks, expressed as 3.75 μs in this register.
23:16	Minimum Grant	01h	RO	This value specifies, in 0.25 μs increments, the minimum burst period CX28500 needs. CX28500 does not have any special MIN_GNT requirements. In general, the more channels CX28500 has active, the worse the bus latency and the shorter the burst cycle.
15:8	Interrupt Pin	01h	RO	Defines which PCI interrupt pin CX28500 uses. 01h means CX28500 uses pin INTA*.
7:0	Interrupt Line	0	RW	Communicates interrupt line routing. System initialization software writes a value to this register indicating which Host interrupt controller input is connected to CX28500's INTA* pin.

GENERAL NOTE:
1. An active low signal is detected by a trailing asterisk (*).

3.2.2 PCI Reset

CX28500 resets all internal functions when it detects the assertion of the PRST* signal line. Upon reset, the following occurs:

- All PCI output signals are three-stated immediately and asynchronously with respect to the PCI clock input, PCLK.
- All EBUS output signals are three-stated immediately and asynchronously with respect to the EBUS clock output, ECLK.
- All writable/resettable internal register bits are reset to their default values.
- All PCI data transfers are terminated immediately.
- All serial data transfers are terminated immediately.
- CX28500 is disabled and responds only to PCI configuration cycles.

3.2.3 PCI Throughput and Latency Considerations

For reference to PCI throughput and latency considerations see Appendix A.

3.2.4 Host Interface

After a hardware reset, the PCI configuration space within CX28500 needs to be configured by the Host as follows:

- Base address register
- Fast back-to-back enable/disable
- SERR* signal driver enable/disable
- Parity error response enable/disable
- Latency timer register
- Interrupt line register
- Bus mastering enable/disable
- Memory space access enable/disable

4 Expansion Bus (EBUS)

CX28500 provides access to a local bus interface on the CX28500 called the Expansion Bus (EBUS), which provides a Host processor to access any address in the peripheral memory space on the EBUS.

Although EBUS utilization is optional, the most notable applications for the EBUS are the connections to peripheral devices (e.g., Bt8370/Bt8398 T1/E1 framers, CX28398 (Octal DS1/E1 framers), CX28314/CX28313 (multiplexer-demultiplexer DS1 to DS3 plus framer) that are local to CX28500's serial port.

Unlike other generations of Conexant's HDLC controllers (CX28478/CX28474A/CX28472A/CX28471A), CX28500 provides access to the EBUS through an interface similar to a mailbox interface. This interface provides all the EBUS read and write accesses to be carried over the PCI bus, allowing PCI bursts. This mechanism improves the PCI utilization when multiple EBUS accesses are necessary for accessing the configuration of the peripheral devices. The PCI Function 1 is disabled. Therefore, CX28500's EBUS service requests are capable of accepting or generating burst on the PCI bus. However, the EBUS interface signals are not capable of performing burst.

Figures 4-1 and 4-2 illustrate block diagrams of the EBUS interface with and without local microprocessor (MPU).

Figure 4-1. EBUS Functional Block Diagram with Local MPU

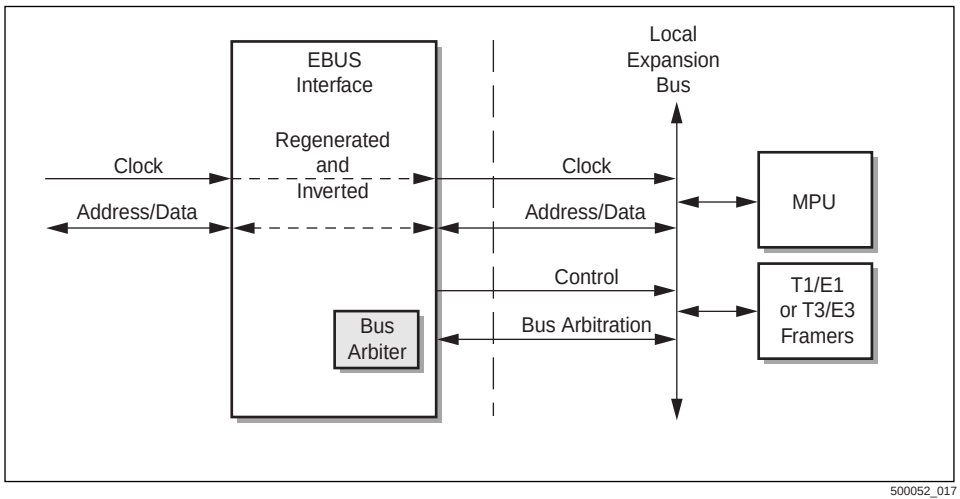
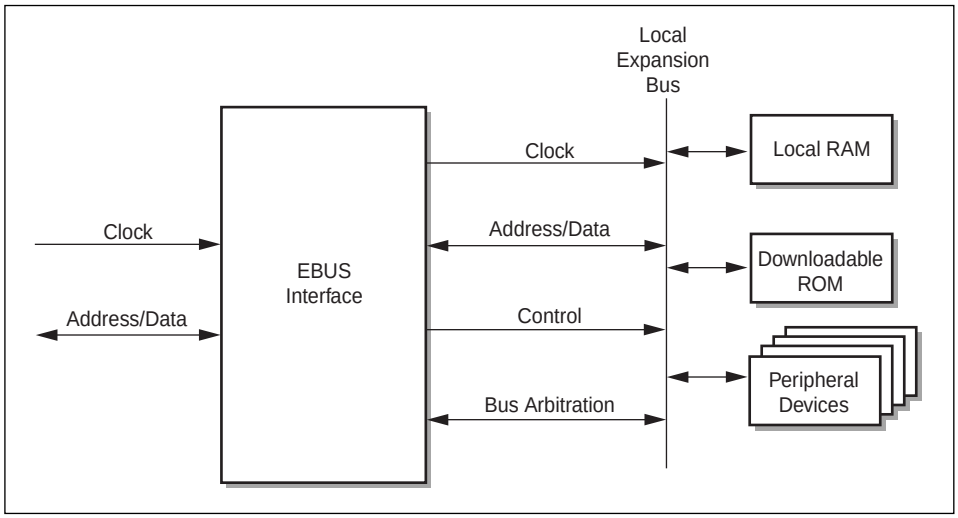


Figure 4-2. EBUS Functional Block Diagram without Local MPU



4.1 EBUS—Operational Mode

4.1.1 Initialization

After reset and after the PCI configuration is completed, CX28500 provides the Host the ability to read and write peripheral devices located on the EBUS (refer to Table 4-1). The Host Service Request mechanism allows the Host to instruct CX28500 to perform specific EBUS operations. CX28500 can perform bulk service request commands. The Service Request Acknowledge (SACK) can be generated either after each service request command or at the end of each bulk service request, depending on the value of SACKIEN bit field set in the service request configuration descriptor. (See Table 4-2.) CX28500 processes an SRQ by reading the HSRP register which contains the address of the first entry in the HSDT (or refer to Section 7.1.2.2). Once configured and enabled, the Host can configure local devices connected to the EBUS by issuing the EBUS Access Service Request (EBUS_WR or EBUS_RD). The command is a three dword memory location that contains the following dword-fields:

- Access Control Field
- Shared Memory Pointer (Buffer Address) representing the starting address of the buffer location where the device structure resides
- EBUS Base Address Offset (the address of the first EBUS transaction)

Table 4-1. EBUS Service Request Descriptor

Dword Number	Bit 31					Bit 0
dword 0	OPCODE[31:27]	SACKIEN[26]	Reserved [25:19]	FIFO_BURST[18]	EBUS Byte Enable [17:14]	Length[13:0]
dword 1	Shared Memory Pointer[31:2](2)					
dword 2	EBUS Base Address Offset(3)					
dword 3	Reserved(1)					

FOOTNOTE:
(1) All reserved bits must be written with 0's for forward compatibility.
(2) The two LSB's must be equal to zero for dword alignment.
(3) The EBUS Base Address Offset is only 31 bits wide. The MSB (bit 31) must be set to 1 for all transactions.

Table 4-2. EBUS Service Request Field Descriptions

Dword Number	Descriptor Field	Size (Bits)	Value	Description
dword 0	OPCODE	5	6	EBUS Write command (EBUS_WR)
			7	EBUS Read command (EBUS_RD)
	SACKIEN	1	—	Enable (1) or disable (0) acknowledge via interrupt in the end of the command execution
	Reserved	7	0	Reserved bits should be written with 0s.
	FIFO_BURST	1	0	Do increment EBUS address (address on the target device) by one after each EBUS access. This is used to access a continuous segment or block of memory on the target device that is connected to the EBUS.
			1	Do not increment EBUS address for this access. On some devices, memory accesses are carried out the writing/reading of one memory location. By setting FIFO_BURST to one, CX28500 does not increment the EBUS address after an access. Hence, the address stays the same for the next EBUS access.
	EBUS Byte Enable (EBE)	4	—	The value driven over EBE[3:0]*. Each bit controls a corresponding byte access on the EBUS. For example, an EBE[3:0] value of 0001 means that Host data passes to the device attached to the EBUS on byte 0, the least significant byte, of the EBUS while the other three bytes are inaccessible.
	Length	14	—	Number of EBUS transactions.
dword 1	Shared Memory Pointer	32	—	The Shared Memory Pointer (Buffer Address) is a dword–aligned address of the first buffer to or from which data needs to be transferred from or to the EBUS. The two LSB's must be equal to zero for dword alignment.
dword 2	EBUS Base Address Offset	31	—	The EBUS Base Address Offset is the address for the first EBUS transaction. Bit 31 of this dword must be set to 1 in every transaction.

When an EBUS_RD is issued, CX28500 executes a PCI bursted write of EBUS transactions and will store the data (EAD[31:0]) in an internal buffer. When the EBUS transaction ends, CX28500 bursts the data over the PCI to the location specified by Shared Memory Pointer (Buffer Address). The EBE[3:0]* drives the programmed EBUS Byte Enabled (EBE) value set in the Access Control Field dword. If EBE[3:0]* is different from 0000, the Host must determine which bytes are valid for access.

If an EBUS Write command is enabled, CX28500 transfers—via a PCI burst read—the data from the Host memory into an internal buffer. The data is transferred over the EBUS in a series of write transactions. The EBE[3:0]* drives the programmed value EBUS Byte Enabled (EBE) value set in the Access Control Field dword. If EBE[3:0]* is different from 0000, the Host must insert the valid bytes into the appropriate location.

4.1.2 Clock

The ECLK, Expansion Bus Clock, can be configured to operate at the PCI bus rate of either 33 MHz or 66 MHz, or at half of the PCI bus rate of 33/2 MHz and 66/2 MHz, respectively. This option is selectable by setting the value of ECLKDIV bit field in EBUS Configuration register. The signal is output on the ECLK signal line. Whether or not a device on the EBUS requires a synchronous interface, the ECLK signal is available all the time the PCI clock is available (PCLK). The EBUS clock output can be disabled by appropriately setting the ECKEN bit field in EBUS Configuration register. If ECLK is disabled, the ECLK output is three-stated.

After PCI reset, the ECLK output pin is three-stated and the ECKEN field in EBUS Configuration register is cleared.

4.1.3 Interrupt

Unlike Conexant's other HDLC controllers (CX28478/CX28474/CX28472), CX28500 is not connected to the EINT* pin of the EBUS. The EBUS interrupt line should be connected to PCI interrupt INTB* directly, if it is needed.

4.1.4 Address Duration

CX28500 is able to extend the duration that the address bits are valid for any given EBUS address phase. This is accomplished by specifying a value between 0 and 7 in the ALAPSE bit field in EBUS Configuration register. The value specifies the additional ECLK periods the address bits remain asserted. That is, a value of 0 specifies the address remains asserted for one ECLK period, and a value of 7 specifies the address remains asserted for 8 ECLK periods. Disabling the ECLK signal output does not affect the delay mechanism.

Both pre- and post-address cycles are always present during the address phase of an EBUS cycle. The pre-address cycle is one ECLK period long and provides CX28500 time to transition between the address phase and the following data phase. The pre- and post-cycles are not included in the Address Duration.

4.1.5 Data Duration

CX28500 is able to extend the duration that the data bits are valid for any given EBUS data phase. This is accomplished by specifying a value between 0 and 15 in the ELAPSE bit field in EBUS Configuration register. The value specifies the additional ECLK periods the data bits remain asserted. That is, a value of 0 specifies the data remains asserted for one ECLK period, and a value of 15 specifies the data remains asserted for 16 ECLK periods. Disabling the ECLK signal output does not affect the delay mechanism.

A pre-data and post-data cycle is always present during the data phase of an EBUS cycle. The pre-data cycle is one ECLK period long and provides CX28500 sufficient setup and hold time for the data signals. The post-data cycle is one ECLK period long and provides CX28500 sufficient time to transition between the data phase and the following bus cycle termination. The pre- and post-cycles are not included in the Data Duration.

4.1.6 Bus Access Interval

CX28500 can be configured to wait a specified amount of time after it releases the EBUS and before it requests the EBUS a subsequent time. This is accomplished by specifying a value between 0 and 15 in the BLAPSE bit field in EBUS configuration register. The value specifies the additional ECLK periods CX28500 waits immediately after releasing the bus. That is, a value of 0 specifies CX28500 will wait for one ECLK period, and a value of 15 specifies 16 ECLK periods. Disabling the ECLK signal output does not affect this wait mechanism.

The bus grant signal (HLDA/BG*) is deasserted by the bus arbiter only after the bus request signal (HOLD/BR*) is deasserted by CX28500. As the amount of time between bus request deassertion and bus grant deassertion can vary from system to system, it is possible for a misinterpretation of the old bus grant signal as an approval to access the EBUS. CX28500 provides the flexibility— through the bus access interval feature—to wait a specific number of ECLK periods between subsequent bus requests. If the signal (HLDA/BG*) is permanently asserted, then there is a minimum of 3 ECLK periods between transactions.

Refer to EBUS timing diagrams—Figure 10-7, *EBUS Write/Read Cycle, Intel-Style* (Intel) and Figure 10-8, *EBUS Write/Read Cycle, Motorola-Style* (Motorola).

4.1.7 PCI to EBUS Interaction

CX28500 provides a significant improvement in the EBUS interface compared to previous Conexant HDLC devices. PCI utilization is dramatically improved by enabling the EBUS accesses, reads and writes, to be burst over the PCI bus—when EBUS is extensively used to access EBUS peripheral during normal operation.

4.1.8 Microprocessor Interface

A microprocessor can be added to handle peripheral devices that require additional processing power. The MPUSEL bit field in EBUS Configuration register specifies the type of microprocessor interface to use for the EBUS.

If Intel-style protocol is selected, the following signals are effective:

- ALE–Address Latch Enable, asserted high by CX28500 to indicate that the address lines contain a valid address. This signal remains asserted for the duration of the access cycle.
- RD*–Read, strobed low by CX28500 to enable data reads out of the device and is held high during writes.
- WR*–Write, strobed low by CX28500 to enable data writes into the device and is held high during reads.
- HOLD–Hold Request, asserted high by CX28500 when it requests the EBUS from a bus arbiter.
- HLDA–Hold Acknowledge, asserted high by bus arbiter in response to HOLD signal assertion. Remains asserted until after the HOLD signal is deasserted. If the EBUS is connected and there are no bus arbiters on the EBUS, then this signal must be asserted high at all times.
- HLDA is treated as an asynchronous signal.

If Motorola-style protocol, the following signals are effective:

- AS*–Address Strobe, driven low by CX28500 to indicate that the address lines contain a valid address. This signal remains asserted for the duration of the access cycle.
- DS*–Data Strobe, strobed low by CX28500 to enable data reads or data writes for the addressed device.
- R/WR*–Read/Write, held high throughout read operation and held low throughout write operation by CX28500. This signal determines the meaning (read or write) of DS*.
- BR*–Bus Request, asserted low by CX28500 when it requests the EBUS from a bus arbiter.
- BG*– Hold Acknowledge, asserted low by bus arbiter in response to BR* signal assertion. Remains asserted until after the BR* signal is deasserted. If the EBUS is connected and there are no bus arbiters on the EBUS, then this signal must be asserted low at all times.
- BGACK*–Bus Grant Acknowledge, asserted low by CX28500 when it detects BGACK* currently deasserted. As this signal is asserted, CX28500 begins the EBUS access cycle. After the cycle is finished, this signal is deasserted indicating to the bus arbiter that CX28500 has released the EBUS.

4.1.9 Arbitration

The HOLD and HLDA (Intel) or BR* and BG* (Motorola) signal lines are used by CX28500 to arbitrate for the EBUS.

For Intel-style interfaces, the arbitration protocol is as follows (refer to Figure 10-7, *EBUS Write/Read Cycle, Intel-Style*).

1. CX28500 three-states EAD[31:0], EBE*[3:0]. WR*, RD*, and ALE*.
2. CX28500 requires EBUS access and asserts HOLD.
3. CX28500 checks for HLDA assertion by bus arbiter.
4. If HLDA is found to be deasserted, CX28500 waits for the HLDA signal to become asserted before continuing the EBUS operation.
5. If HLDA is found to be asserted, CX28500 continues with the EBUS access as it has control of the EBUS.
6. CX28500 drives the address lines (EAD[30:0]), EBE*[3:0], WR*, RD*, and ALE*. The data lines (EAD[31:0]) are driven one cycle later than the other aforementioned signals.
7. CX28500 completes EBUS access and deasserts HOLD.
8. Bus arbiter deasserts HLDA shortly thereafter.
9. CX28500 three-states EAD[31:0], EBE*[3:0]. WR*, RD*, and ALE*.

For Motorola-style interfaces, the arbitration protocol is as follows (refer to Figure 10-8, *EBUS Write/Read Cycle, Motorola-Style*).

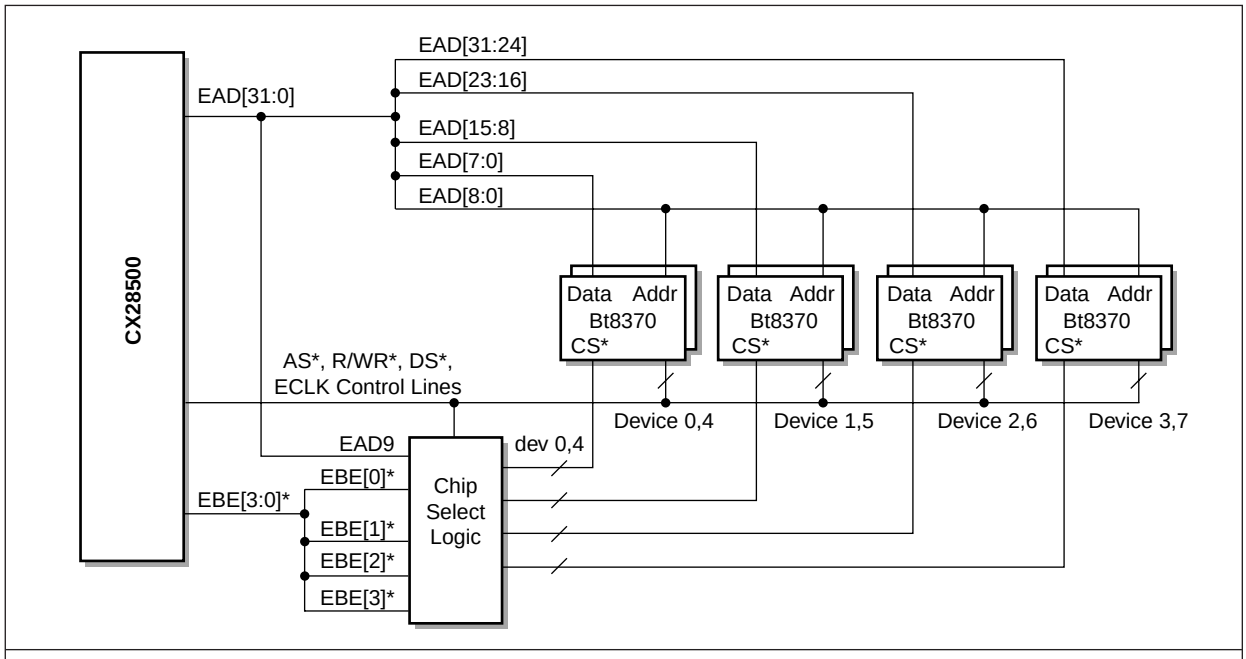
1. CX28500 three-states EAD[31:0], EBE*[3:0]. R/WR*, DS*, and AS*.
2. CX28500 requires EBUS access and asserts BR*.
3. CX28500 checks for BG* assertion by bus arbiter.
4. If BG* is found to be deasserted, CX28500 waits for the BG* signal to become asserted before continuing the EBUS operation.
5. If BG* is found to be asserted, CX28500 continues with the EBUS access as it has control of the EBUS.
6. If BGACK* is not asserted CX28500 assumes control of the EBUS by asserting BGACK*.
7. CX28500 drives the address lines (EAD[30:0]), EBE*[3:0], R/WR*, DS*, AS*. The data lines (EAD[31:0]) are driven one cycle later than the other aforementioned signals.
8. Shortly after the EBUS cycle is started, CX28500 deasserts BR*.
9. Bus arbiter deasserts BG* shortly thereafter.
10. CX28500 completes EBUS cycle.
11. CX28500 deasserts BGACK*.
12. CX28500 three-states EAD[31:0], EBE*[3:0]. R/WR*, DS*, and AS*.

4.1.10 Connection

Utilizing the EBUS address lines, EAD[17:0], and the byte enable lines, EBE[3:0]*, the EBUS can be connected in either a multiplexed or non-multiplexed address and data mode.

Figures 4-3 and 4-4 illustrate two examples of non-multiplexed address and data modes. Figure 4-3 illustrates four separate byte-wide framer devices connected to the EBUS with each byte enable line used as the chip select for separate devices, which allows a full dword data transfer over the EBUS.

Figure 4-3. EBUS Connection, Non-Multiplexed Address/Data, 8 Framers, No Local MPU



GENERAL NOTE:

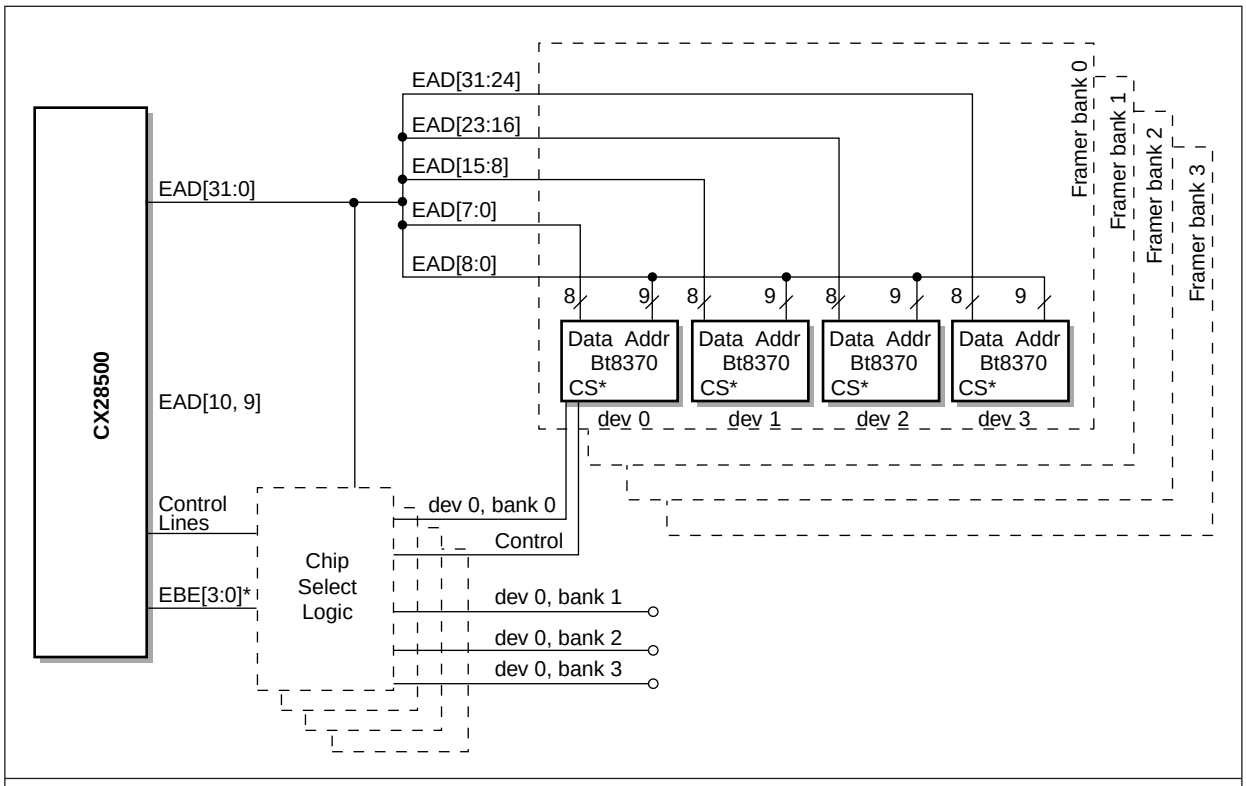
1. EBEx[3:0]* selects device x in each framer block.
2. EAD[31:0], AS* are supplied to each framer block.
3. EBEx*, AS* are supplied to each chip select block.

500052_050

Figure 4-4 illustrates how additional address lines can be combined with each byte enable line during the address phase to support multiple framer banks with each bank containing four byte-wide framer devices.

The framers configuration in shared memory is that only the Least Significant Byte (LSB) contains the information of one frame configuration, the others are unused.

Figure 4-4. EBUS Connection, Non-Multiplexed Address/Data, 16 Framers, No Local MPU



GENERAL NOTE:

1. EBEx[3:0]* selects device x in each framer block.
2. EAD[31:0], AS* are supplied to each framer block.
3. EBEx*, AS* are supplied to each chip select block.

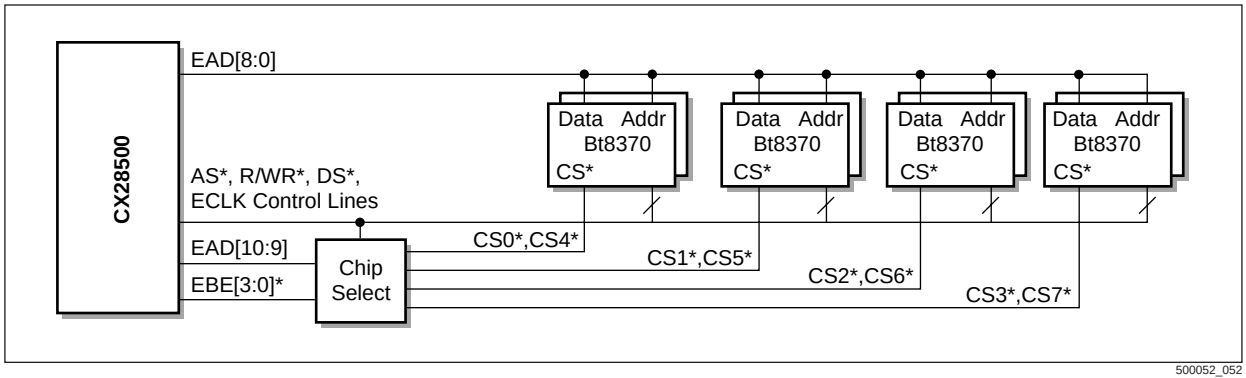
500052_051

In the multiplexed address and data mode, four byte-wide peripheral devices are connected to the EBUS. In this mode, 8 bits of the 32-bit EBUS transfer data to and from each device individually.

NOTE: The multiplexed address and data mode example does not allow for 4-byte data transfers.

Figure 4-5 illustrates the EBUS connection, multiplexed address/data, 8 Framers, no local MPU.

Figure 4-5. EBUS Connection, Multiplexed Address/Data, 8 Framers, No Local MPU

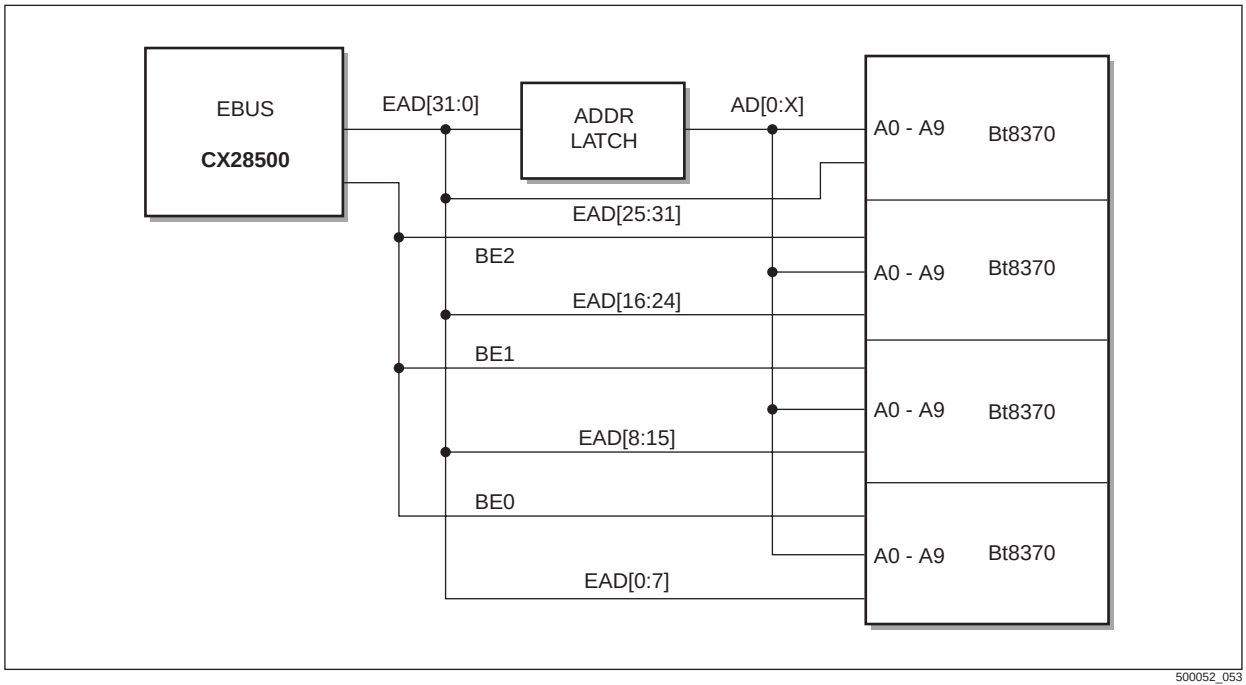


4.1.10.1 Multiplexing Address

Figure 4-6 illustrates the EBUS connections of four 8-bit peripheral devices.

The four devices are multiplexing the address in shared memory. The framer's configuration software has to read the whole block of framers configuration before it starts demultiplexing data per device.

Figure 4-6. EBUS Connection, Multiplexed Address/Data, 4 Framers, No Local MPU



5 Serial Interface

The Serial Interface Unit (SIU) is the module that logically connects the 32 serial ports (line interface unit) with serial line processing by performing the serial-to-parallel conversion for the receive side, and parallel to serial for the transmit side. The SIU contains two main blocks, RSIU for the receive path and TSIU for the transmit path. The RSIU main function is multiplexing 32 serial ports into one logical port for the receive serial line processing block. The TSIU main function is demultiplexing one logical port from the transmit serial line processing block to 32 serial ports.

The SIU main functions:

- Multiplexing/demultiplexing 32 serial ports to one port for the receive path, and one port to 32 serial ports for the transmit path, respectively.
- Performs frame integrity check while operating in channelized mode. SIU verifies the length of incoming/outgoing frames according to the configured number of time slots. In case of error a Change Of Frame Alignment (COFA) is reported (see *COFA description both modes convention and TSBUS mode*).
- Translates the time slot to logical channel number using the configured receive and transmit time slot map.
- Provides a poll indication per port, see RPOLLTH or TPOLLTH bit fields in RSIU Port Configuration register and TSIU Port Configuration register, which counts the frames of the poll interval of 1, 2, 4, 8, 16, 64, 128, or 256 frame interval.
- Generates the following interrupts:
  - RxOOF, where ROOF signal is asserted
  - RxFREC, where ROOF signal is deasserted
  - RxCOFA, where RSYNC signal is asserted in an unexpected place
  - RxCREC, where COFA condition is off for the receive port
  - TxCOFA, where TSYNC signal is asserted in an unexpected place
  - TxCREC, where COFA condition is off for the transmit port

TSIU (Transmit Serial Interface Unit) is responsible for informing TSLP (Transmit Serial Line Processor) when it is time to perform a transmit channel poll by counting a programmable (TPOLLTH) number of frames. For the TSBUS mode, one frame is defined as the STB strobe interval. TSIU is also responsible for requesting transmit data from TSLP for each transmitted time slot (i.e., for each VSP).

As for the RSIU, when data comes through the serial ports, it is stored in local buffers. At this stage, all line signals are synchronized to the system clock, or the PCI bus rate (either 33 MHz or 66 MHz). At each cycle, the RSIU transfers a byte received from one of the 32 ports to the Receive Serial Line Processor (RSLP). For the port being served, the RSIU translates the time slot to a channel number (using an internal map) and provides certain parameters that are needed by the RSLP to process the incoming data. Similar to the TSIU, for the TSBUS mode, one frame is defined as the STB strobe interval.

5.1 Serial Port Interface Definition in Conventional Mode

A receive serial port unit (RSIU) connects to four input signals: RCLK, RDAT, RSYNC, and ROOF. A transmit serial port unit (TSIU) connects to three input signals and one output signal: TCLK, TSYNC, CTS, and TDAT, respectively. The SIU is responsible for receiving and transmitting data bytes to the Transmit Serial Line Processor (TSLP) and Receive Serial Line Processor (RSLP). The receive and transmit data and synchronization signals are synchronous to the receive and transmit line clocks, respectively. CX28500 can be configured to sample in and latch out data signals and sample in status and synchronization signals on either the rising or falling edges of the respective line clock, namely RCLK and TCLK. This configuration is accomplished by setting the ROOF_EDGE, RSYNC_EDGE, RDAT_EDGE, TSYNC_EDGE, and TDAT_EDGE bit fields as detailed in Table 6-28, *RSIU Port Configuration Register* and Table 6-36, *TSIU Port Configuration Register*. The default, after reset, is to sample in and latch out data synchronization and status on the falling edges of the respective line clock. The port mode is configured by programming the RPORT_TYPE and TPORT_TYPE bit fields in *RSIU Port Configuration Register* and *TSIU Port Configuration Register*, respectively. When configured to operate in conventional mode, the receive and transmit directions are not related to each other so that each direction can be programmed independently of the other.

5.1.1 Frame Synchronization Flywheel

CX28500 utilizes the TSYNC and RSYNC signals to maintain a time-base, which keeps track of the active bit in the current time slot. The mechanism is referred to as the frame synchronization flywheel. The flywheel counts the number of bits per frame and automatically rolls over the bit count according to the programmed mode. The TSYNC or RSYNC input marks the first bit in the frame. The mode specified in the RPORT_TYPE bit field in Table 6-28, *RSIU Port Configuration Register* and TPORT_TYPE bit field in Table 6-36, *TSIU Port Configuration Register* and the start and end address of time slot pointer determines the number of bits in the frame. A flywheel exists for both the transmit and the receive functions for every port.

The flywheel is synchronized when CX28500 detects TSYNC = 1 or RSYNC = 1, for transmit or receive functions, respectively. Once synchronized, the flywheel maintains synchronization without further assertion of the synchronization signal. Additional sync pulses do not interfere with the flywheel mechanism as long as they are synchronized correctly.

A time slot counter within each port is reset once in each frame and tracks the current time slot being serviced. In the receive side, synchronization starts at the beginning of frame. In the transmit side, however, synchronization starts four time slots later since there is an associated latency between when the TSIU requests the TSLP for data and when the TSLP supplies the TSIU with the requested data.

NOTE: In unchannelized mode, CX28500 ignores the TSYNC and RSYNC signals and the frame synchronization flywheel mechanism is ignored.

5.1.2 Change Of Frame Alignment (COFA)

A Change Of Frame Alignment (COFA) condition is defined as a frame synchronization event detected when it was not expected, and also includes the detection of the first occurrence of frame synchronization when none was present. In unchannelized mode, there are no COFA conditions because the TSYNC and RSYNC signals are ignored in this mode.

In the receive direction, when a COFA condition is detected by the serial interface, an internal COFA signal is asserted until the COFA condition is declared off. A COFA condition is declared “off” when there was a complete frame without an unexpected SYNC pulse. Thus, an internal COFA signal is asserted for at least two frame periods. During the frame period that the internal COFA is asserted, CX28500’s serial line processor (SLP) terminates all messages that are found to be active during the COFA condition. For each receiver and transmitter channel found to be active and processing a message, the corresponding message descriptor’s owner bit is returned to the Host, and a Buffer Status Descriptor is written with the COFA error encoding. The Buffer Status Descriptor is written if the INHRBSD bit field in the RDMA Channel Configuration register is disabled (set to 1). CX28500 then proceeds to the next message descriptor (MD) from the *Receive Message Descriptor Table* (RMDT). In the transmit direction, however, CX28500 cannot recover from COFA without the Host’s intervention. For example, a new channel activation is required.

Assertion of COFA condition generates a COFA interrupt encoded in the Interrupt Status Descriptor (ISD) toward the Host if this interrupt is unmasked (see Table 6-28, *RSIU Port Configuration Register* and Table 6-36, *TSIU Port Configuration Register*, respectively *RCOFA_EN* or/and *TCOFA_EN bit fields*).

If a synchronization signal (SYNC) is received (low to high transition on TSYNC or RSYNC) while the internal COFA is asserted, an Interrupt Descriptor with the COFA interrupt encoding is generated immediately if this interrupt is not masked.

When the internal COFA is deasserted, CX28500 generates an Interrupt Descriptor with CREC event encoding if the interrupt is unmasked.

The receive serial bit stream processing resumes when the COFA condition is declared off. If channels are configured in HDLC mode, then channels resume immediately if the COFA condition is declared off. While in Transparent mode, channels start operating in the first time slot assigned to the logical channel. Thus, after a RxCOFA, no channel recovery action is required because the channel recovers automatically.

For each transmitter path, the active channel (regardless of message processing) is immediately deactivated. As a recovery channel action, the Host needs to reactivate the channel upon termination of the COFA condition. COFA detection is not applicable in unchannelized mode. When COFA condition occurs, the transmit output is three-stated. If operating in T1 mode, the F-bit cannot be three-stated after a COFA condition.

5.1.3 Out Of Frame (OOF)/Frame Recovery (FREC)

The Receiver Out-Of-Frame (ROOF) signal is asserted by the serial interface sourcing the channelized data to CX28500. This signal indicates that the interface device has lost frame synchronization.

In the case of multiplexed E1 lines (2xE1, 4xE1), any given port ROOF signal can be asserted and deasserted as the time slots are received from an Out-Of-Frame E1 followed by an in-frame E1.

ROOF assertion is detected by the Receiver Serial Interface (RSIU). If ROOF is asserted high and OOFIEN bit field in the RSIU Port Configuration Descriptor is set, an OOF interrupt is generated toward the Host. For each receive HDLC message that encountered an OOF condition, the corresponding Message Descriptor's owner bit is returned to the Host and a Buffer Status Descriptor is written with the OOF error encoding, along with EOM. The EOM indicates that the current message is effectively ended as a result of a receiving error. The Buffer Status Descriptor is written to Host memory only if configured to do so on a per-channel basis in the RDMA Channel Configuration register. CX28500 then proceeds to the next Message Descriptor in the list of messages. As for the RSIU, it starts to search for the opening flag of the next frame. For Transparent mode channels, the OOF causes the data that is being transferred to the Host to be replaced by an all 1s sequence. No special actions are taken in this case, and the Host must rely on the OOF interrupt to learn about the OOF.

While ROOF is asserted, if OOFABT bit field in the RSIU Port Configuration Descriptor is set, the receive process is disabled. Thus CX28500 terminates any active messages for all active channels operating over the port; otherwise, the receive process is enabled.

Notice that the OOF signal is examined on a per-time slot basis. Therefore, OOF assertion affects only those logical channels that are mapped to time slots where OOF is asserted. The remaining time slots on the same serial port are not affected by the OOF assertion on a specific time slot.

For ROOF to be deasserted, CX28500 must detect at least one frame without any OOF's. As ROOF is deasserted, CX28500 immediately restarts normal processing on all active channels. One to three time slots after deassertion of ROOF is detected, CX28500 generates an interrupt descriptor with the FREC (Frame Recovery) interrupt encoding if the interrupt is not masked (OOFIEN = 1, RSIU Port Configuration Descriptor).

5.1.4 General Serial Port Interrupt

The ROOF signal can be used as a general serial port interrupt (SPORT).

If OOFABT is zero, OOFIEN is set, and ROOF signal deasserts or transitions from high to low, SPORT interrupt is generated and the data stream processing is not affected. When ROOF transitions from low to high, the SPORT interrupt is cleared.

5.1.5 Channel Clear To Send (CTS)

CX28500 transmit path can be configured to obey a Channelized Clear To Send (CTS) external signal on a per-port basis by enabling the CTS_ENB bit in the TSIU Port Configuration register. CTS is sampled on the specified active edge of TCLK depending on CTS_EDGE in Table 6-36, *TSIU Port Configuration Register*.

If CTS is deasserted (low), the channel assigned to the time slot sends continuous idle characters after the current message has been completely transmitted. If CTS is asserted (high), message transmission continues.

When configured to operate in CTS mode, the channels of this specific port will not start a new message transmission if the CTS is a logical 0. The channel response time to react to changes in the channelized CTS signal is up to 32 line clocks.

5.1.6 Frame Alignment

CX28500 utilizes the TSYNC and RSYNC signals to maintain a timebase that keeps track of the active bit in the current time slot. The mechanism is referred to as the frame synchronization flywheel. The flywheel counts the number of bits per frame and automatically rolls over the bit count according to the programmed mode. The TSYNC or RSYNC inputs mark the first bit in the frame. The flywheel is synchronized when CX28500 detects TSYNC = 1 or RSYNC = 1, for transmit or receive functions, respectively. Once synchronized, the flywheel maintains synchronization without further assertion of the synchronization signal.

The serial data stream that CX28500 can manage consists of either packetized data or unpacketized data. CX28500 supports two types of data-stream modes: HDLC and Transparent.

In Transparent mode, message processing for every channel begins in the first time slot marked as the first time slot in the channel's frame structure. A user needs to configure the first time slot in the RSIU Time Slot Configuration Descriptor. In the transmission side, the user needs to mark the last time slot where the channel appears in the frame. This is done by setting the LAST_TS bit in the TSIU Time Slot Configuration Descriptor.

For a channel configured in HDLC mode—either transmit or receive direction, the channel waits for a synchronization signal from the internal frame synchronization flywheel before starting processing of a new message after channel activation. A Frame Synchronization signal must be provided once; after that, CX28500 keeps track of subsequent frame bit location with its flywheel mechanism.

The Frame Alignment is not relevant when the port is configured in Unchannelized mode, although in Unchannelized mode each time slot is treated as the first time slot. By configuring more than one time slot in unchannelized mode, (i.e., using TTS_ENDAD /RTS_ENDAD and TTS_STARTAD/RTS_STARTAD mechanism to define one frame) the number of time slots between frames are considered as a virtual frame synchronization for controlling the polling interval. (See Table 6-36, *TSIU Port Configuration Register* and Table 6-28, *RSIU Port Configuration Register*).

5.1.7 Polling

After channel activation, CX28500 must fetch Message Descriptors (MD) from shared memory to start the message flow in and out of shared memory.

As an MD is fetched, CX28500 checks the owner bit to verify whether the buffer is available for CX28500. If the owner bit indicates that the Host still owns the buffer, the Host has not prepared the data for processing in the data buffers. If poll bit is enabled, CX28500 polls the Buffer Descriptor until the owner bit is switched to CX28500-owned.

If the owner bit is still Host-owned and no poll (NP = 1) and if TONRIEN/RONRIEN bit field is set to 1 in TDMA Channel Configuration register or RDMA Channel Configuration register, an (ONR) interrupt is generated toward the Host.

If the Host owns the buffer and polling is enabled, the channel direction is suspended from processing messages, and CX28500 periodically polls the owner bit in the Buffer Descriptor until the owner bit is CX28500-owned. The channel is capable of leaving this suspended state autonomously. Figures 5-1 and 5-2 illustrate the details.

Figure 5-1. RDMA State Machine

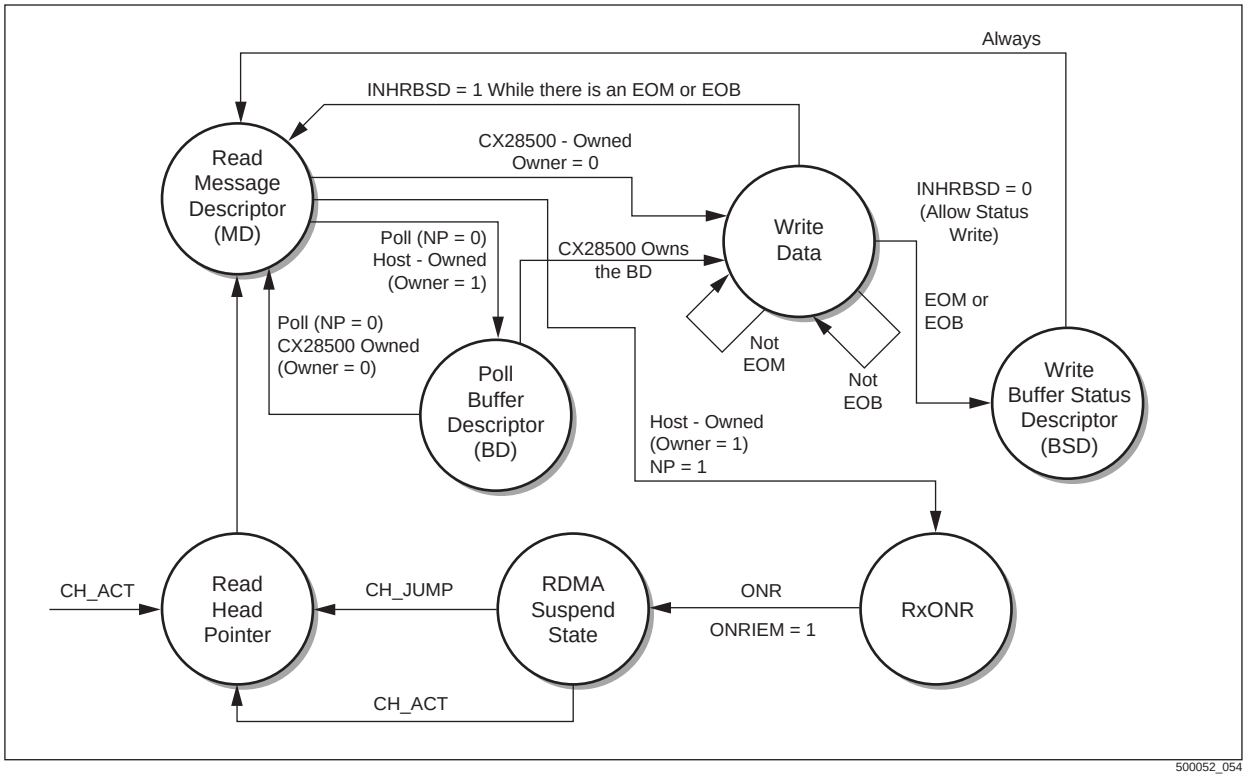
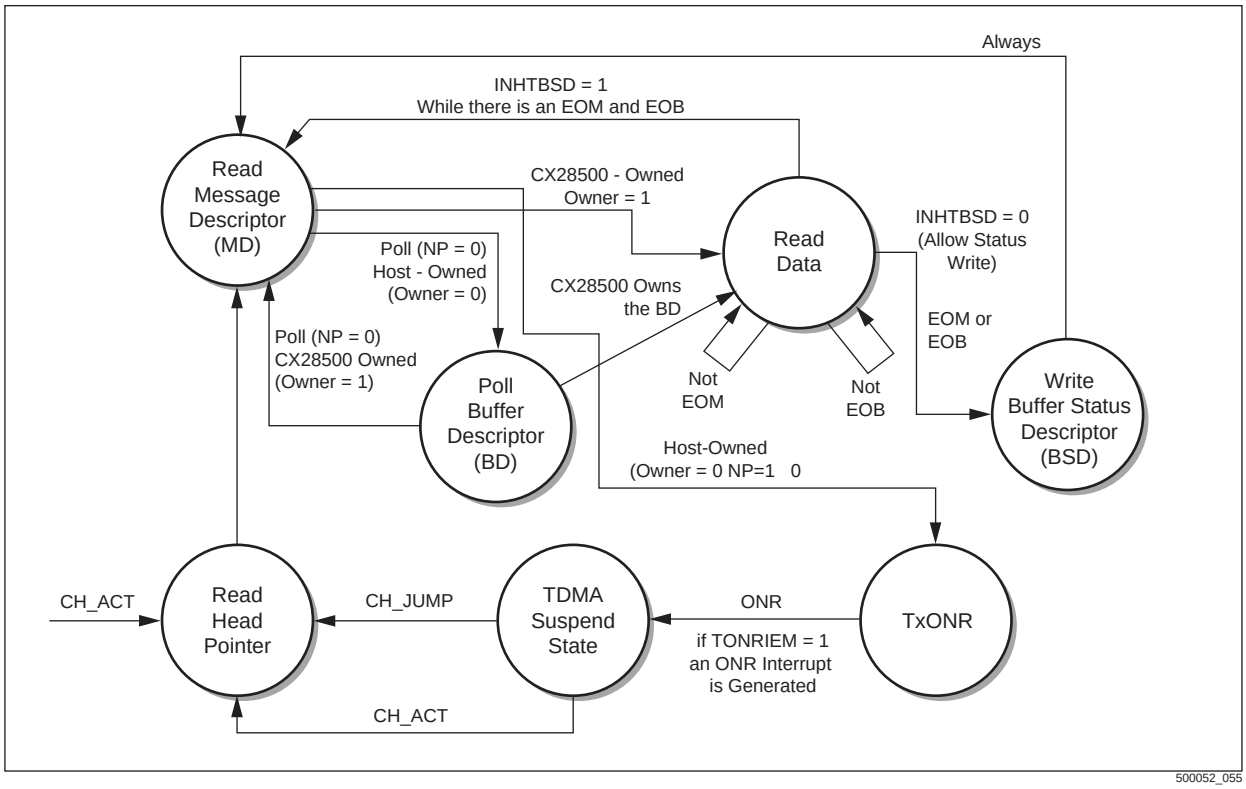


Figure 5-2. TDMA State Machine



The frequency of polling is controlled independently for each port by the RPOLLTH or TPOLLTH bit fields in RSIU Port Configuration register and TSIU Port Configuration register (poll throttle) bit fields in Table 6-28, *RSIU Port Configuration Register* for the Rx direction and Table 6-36, *TSIU Port Configuration Register* for the Tx direction.

The RPOLLTH or TPOLLTH bit fields in RSIU Port Configuration register and TSIU Port Configuration register, bit field specifies how often CX28500 checks the owner bit in a Host-owned Buffer Descriptor. The values correspond to 1, 2, 4, 8, 16, 64, 128, and 256 frame periods.

NOTE: For polling to work properly and efficiently, it is mandatory that the FIRST (receive time slot)/LAST (transmit time slot) indications be configured for each channel regardless of the operational mode (i.e., HDLC, transparent, etc.). The polling mechanism in the CX28500 uses these markings as triggering points. When configured correctly, CX28500 polls on a channel once at the selected poll throttle rate instead of polling at every time slot allocated to that channel, as in the case of hyperchannels. When a channel consists of only one time slot, as in a DSO channel, its corresponding receive time slot should have the FIRST_TS bit set. Likewise, its corresponding transmit time slot should have the LAST_TS bit set.

If the serial port is configured to unchannelized mode, this mechanism is still implemented. The user programs the Section 6.7.6 (for the transmit) or Section 6.6.6 (for the receive) as follows:

1. Program the STARTAD to point to the single entry that is used for this port in the Time Slot Configuration Descriptor.
2. Program the ENDAD to point at the entry STARTAD+N where N is the size of the frame to be used for the poll mechanism; N may be zero, too.

5.2 Serial Port Interface Definition in TSBUS Mode

A port operation mode is configured by programming the TPORT_TYPE and RPORT_TYPE bit fields in RSIU and TSIU Port Configuration registers. TPORT_TYPE and RPORT_TYPE must have the same number of time slots configured for each TSBUS port, and STB sampling edges must be programmed the same for both receive and transmit directions. When configured to operate in TSBUS mode, the frame synchronizing signals for both receive and transmit directions are driven by the STB signal. This leaves the RSYNC and TSYNC unused, which become RSTUFF and TSTUFF signals, respectively. Additionally, both RxClk and TxClk signals must be synchronized so that the sampling of STB in both directions is consistent.

5.2.1 TSBUS Frame Synchronization Flywheel

The CX28500's TSTB signal maintains a time-base that keeps track of the active bit in the current time slot. The mechanism is referred to as the frame synchronization flywheel. The flywheel counts the number of bits per frame and automatically rolls over the bit count according to the programmed mode. The TSTB input marks the first bit in the frame. A flywheel exists for both the transmit and receive directions for each port. The TSB assertion works the first bit of time slot in the TSBUS frame. The flywheel is synchronized when CX28500 detects TSTB = 1. Once synchronized, the flywheel maintains synchronization without further assertion of the synchronization signal.

A time slot counter within each port is reset at the beginning of each frame and tracks the current time slot being serviced.

5.2.2 TSBUS Change Of Frame Alignment (COFA)

A COFA condition is defined as a frame synchronization event detected when it is not expected. A COFA condition also detects if the first occurrence of frame synchronization was not present. A COFA condition can occur only if TSTB is asserted in any time slot position that is not the first time slot of the frame. The flywheel always counts the number of time slots allocated to a specific TSBUS port (when the port is enabled). If TSTB is asserted at any time other than a time coincident with CX28500's interval flywheel rollover (the first bit of TS0), COFA is reported on both receive and transmit port directions.

When a COFA condition is detected by the serial interface, an internal COFA signal is asserted until the COFA condition is declared off. A COFA condition is declared off when there is a complete frame without an unexpected TSTB pulse. Thus, an internal COFA signal is asserted for at least two frame periods. During the frame period when the internal COFA is asserted, CX28500's serial line processor (SIU) terminates all messages that are found to be active during the COFA condition. For each receiver and transmitter channels found to be active and processing a message, the corresponding message descriptor's owner bit is returned to the Host, and a Buffer Status Descriptor is written with the COFA error encoding. The Buffer Status Descriptor is written if the INHRBSD bit field in the RDMA Channel Configuration register is disabled (set to 1). CX28500 then proceeds to the next message descriptor (MD) from Table 6-39, *Transmit or Receive Message Descriptor Table (TMDT) or (RMDT) Content*.

Assertion of COFA condition generates a COFA interrupt encoded in the Interrupt Status Descriptor (ISD) toward Host if this interrupt is unmasked. (See Table 6-28, *RSIU Port Configuration Register* and Table 6-36, *TSIU Port Configuration Register*, respectively *RCOFA_EN* or/and *TCOFA_EN bit fields*.)

If a synchronization signal (TSTB) is received (low to high transition on TSTB) while the internal COFA is asserted, an Interrupt Descriptor with the COFA interrupt encoding is generated immediately if this interrupt is not masked.

When the internal COFA is deasserted, CX28500 generates an Interrupt Descriptor with CREC event encoding if the interrupt is unmasked.

The receive serial bit stream processing resumes when the COFA condition is declared off. If channels are configured in HDLC mode, channels resume immediately when the COFA condition is declared off. While in Transparent mode, channels start operating in the first time slot assigned to the logical channel. Thus, after a RxCOFA, no channel recovery action is required since the channel recovers automatically.

For each transmitter path, the active channel (regardless of message processing) is immediately deactivated. As a recovery channel action, Host needs to reactivate the channel upon termination of the COFA condition. COFA detection is not applicable in unchannelized mode. When a COFA condition occurs, the transmit output is three-stated.

5.2.3 TSBUS Out Of Frame (OOF)/Frame Recovery (FREC)

There is no Out Of Frame (OOF) condition while operating in TSBUS mode. The ROOF signal is used as a TSTB input pin (for reference see Figure E-1, *CX28500 Time Slot Interface Pins*).

5.2.4 TSBUS Frame Alignment

The serial data stream that CX28500 can manage consists of either packetized data or unpacketized data. CX28500 supports two types of data-stream modes: HDLC and Transparent.

In Transparent mode, message processing for every channel begins in the time slot marked as the first time slot in the channel's structure. Regardless of the channel protocol, the user needs to configure the first time slot for both receive and transmit directions (see *RFIRST_TS and TFIRST-TS bit fields* in Table 6-26, *RSIU Time Slot Configuration Descriptor* and Table 6-34, *TSIU Time Slot Configuration Descriptor*).

For a channel configured for HDLC mode, either transmit or receive direction, the channel will wait for a synchronization signal from the internal frame synchronization flywheel before starting processing new messages after channel activation.

A Frame Synchronization Signal (TSTB) must be provided one time, after that, CX28500 keeps track of subsequent frame bit location within the flywheel mechanism.

5.2.5 TSBUS Polling

The polling mechanism is the same as in the conventional mode.

5.2.6 TSBUS Channel Clear To Send

While operating in TSBUS mode, there is no CTS signal since the related input pin is defined to be TSTB (for reference see Figure E-1, *CX28500 Time Slot Interface Pins*).

5.2.7 TSBUS Interface

The TSBUS is a time slot interface. The digital communication data paths and overhead channels consist of payload data and overhead data derived from either SONET or SDH data streams, and payload and overhead data derived from either electrical DS3 or E3 data streams. One of the overhead channels may consist of HDSL messages generated and received by the Command Status Processor (CSP). The messages are provided by the local processor that is connected to access and configure local device registers. The TSBUS interface is capable of full-duplex (bidirectional) transmission of data between one device and the CX28500 device. The interface consists of two, 1-bit wide serial interfaces: a bidirectional payload TSBUS and bidirectional overhead TSBUS.

A TSBUS frame structure is defined as an integer multiplication of bytes. There are seven signals that define the TSBUS interface. In the TSBUS mode, the Rx and Tx are synchronous (i.e., the first bit of Tx and Rx frame is sampled on the falling edge or rising edge of RCLK/TCLK on TDAT/RDAT). TSTB defines the frame synchronization, which marks the first bit of Rx/Tx frame. TSTUFF acts as a flow control signal that indicates “stuff” (update) to be sent on the following time slot mapped to the logical channel. TSTUFF is sampled in the first two bits of the channel’s time slot.

In the TSBUS transmit direction, CX28500 requires the stuff status for each time slot to be presented at its TSTUFF input exactly eight time slots in advance of the actual time slot for which the stuff status is applied. The amount of the TSTUFF advance is fixed at eight time slots even though the number of time slots within a frame might vary.

For the receive direction, CX28500 requires the stuff status for each time slot to be presented at its RSTUFF input on the current time slot for which the stuff is applied.

5.2.8 Payload TSBUS

Examples of the payload TSBUS operates at a data rate of 51.84 Mbps. It carries the data path signals derived from either SONET, SDH, electrical DS3, or the electrical E3 signals.

The data on the Payload TSBUS is framed and consists of 84 time slots.

Time Slot Bus Features

Payload Data Paths are as follows:

- SONET/SDH Payload
- Electrical DS3 to DS1—x28 DS1 672 time slots

NOTE: F-bits not mapped with DS1 signals.

- Electrical E3 to E1—x16 E1 Framers 651 time slots

NOTE: Time Slot 0 not mapped.

- STS-1 to DS3/E3 to X28 DS1 (672 time slots) / X21 E1 (651 time slots)
- x28 DS1—672 time slots
- x21 E1—651 time slots
- x16 E1—496 time slots
- VT1.5—672 time slots
- VT2.0—651 time slots
- VT1.5 to DS1—672 time slots
- VT2.0 to E1—21 time slots
- TUG-2 to DS1—672 time slots
- TUG-2 to E1—651 time slots
- DS1 F-bits—1 time slot
- E1 Si bits—1 time slot

5.2.9 Overhead TSBUS

Examples of the overhead TSBUS operates at a data rate of 12.96 Mbps. It carries PDH, or SDH overhead communication channels and it carries the data monitoring and data configuration for the device that communicates through TSBUS interface with CX28500.

The data on the overhead TSBUS is framed and consists of 84 time slots.

5.2.9.1 Overhead Data Channels

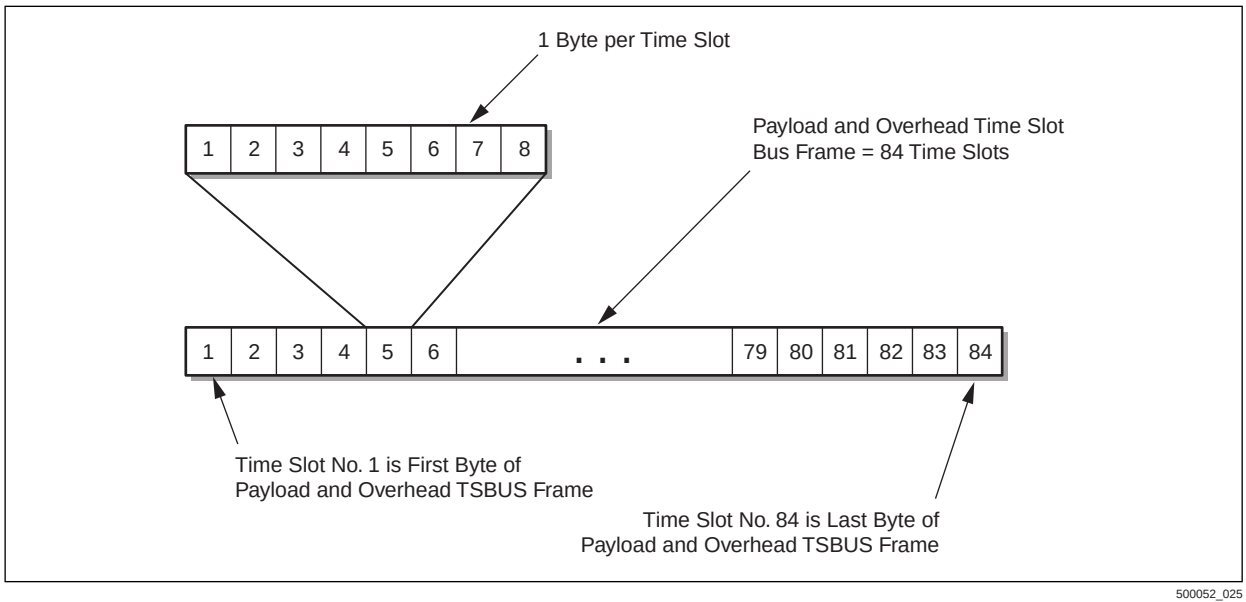
The sources and destinations of overhead data transferred to and from the Overhead TSBUS are as follows:

- SONET
  - SONET/SDH Section DCCR—3 time slots
  - Line CDDM Overhead—9 time slots
  - SPE Path F2 User Data—1 time slot
  - SPE Path F3 User Data—1 time slot
  - SPE N1 Tandem Connection—1 time slot
- DX3 – DS3 TDL Overhead—1 time slot
- Command Status Processor (CSP)—100 time slots

5.2.9.2 TSBUS Time Slots

The number of time slots configured for data payload or overhead are illustrated in Figure 5-3.

Figure 5-3. TSBUS Number of Time Slots into a Payload or Overhead Frame



TSBUS References

For a detailed description of the TSBUS interface see related documents:

1. Conexant: Time Slot Bus Interface Block Description
2. Conexant: Time Slot Bus (TSBUS) Interface Specification.

6 Memory Organization

CX28500 interfaces with a system Host using a set of data structures located in shared memory. CX28500 also contains a set of internal registers, which control CX28500, that the Host can configure. This section describes the various shared memory data structures and the layout of individual registers that are required for the operation of CX28500.

6.1 Memory Architecture

CX28500 supports a descriptor-based memory architecture wherein data is continually moved into and out of a table of data buffers in shared memory for each active channel. This assumes a system topology in which a Host and CX28500 both have access to shared memory for data control and data flow. The data structures are defined in a way that the control structures and the data structures may or may not reside in the same physical memory and may or may not be contiguous. In other words, this data structure is a table of descriptors with pointers to data buffers. The Host allocates and de-allocates the required memory space as well as the size and number of data buffers within that space.

6.1.1 Register Map and Shared Memory Access

During CX28500's PCI initialization, the system controller allocates a dedicated 1 MB-memory range to CX28500. The memory range allocated to CX28500 must not map to any other physical or shared memory. Instead, the system configuration manager allocates a logical memory address range and notifies the system or bus controllers that any access to these ranges must result in a PCI access cycle. CX28500 is assigned these address ranges through the PCI configuration cycle. Once configured, CX28500 becomes a functional PCI device on the bus.

As the Host accesses CX28500's allocated address ranges, the Host initiates access cycles on the PCI bus. It is up to individual CX28500 devices on the bus to claim these access cycles. As CX28500's address ranges are accessed, CX28500 behaves as a PCI slave device while data is being read or written by the Host. CX28500 responds to all access cycles where the upper 12 bits of a PCI address match the upper 12 bits of CX28500's Base Address register (see Table 3-6, *Register 4, Address 10h*).

For CX28500, a 1 MB-memory space is assigned to the CX28500 Base Address register, which is written into the PCI configuration space Address 10h, register 4 in PCI Configuration registers. Once a base address is assigned, a register map is used to access individual device resident registers. The 1 MB-memory range assigned to CX28500 does not restrict CX28500's PCI interface from attempting to access this

memory space. However, CX28500 cannot respond to an access cycle that CX28500 itself initiates as the bus master.

The register map provides the byte offset from the Base Address register where registers reside. The register map layout is given in Table 6-1, *PCI Register Map*. It should be noted that there are two address spaces. The first one includes registers that are directly accessed by the Host through the PCI, and the second includes the registers existing in shared memory that are accessed by CX28500 only through the Service Request Mechanism. Therefore, the same address offsets, or registers, may appear in both register maps memory spaces.

The only registers that can be directly accessed by the Host as slave reads or writes are the Rx Port Alive, the Tx Port Alive, the Interrupt Status Descriptor, the Interrupt Queue Pointer, the Interrupt Queue Length, the Service Request Length, the Service Request Pointer, and the Soft Reset registers specified in CX28500's Register Map. When the Host writes directly into a corresponding register, CX28500 behaves as a PCI slave while this write is performed.

All other registers need to be accessed through the Service Request Mechanism. After the PCI reset, when CX28500 is ready for configuration, these registers are updated with the appropriate shared memory values through a Configuration Write Service Request. After the Host has configured the shared memory image of CX28500's registers, and CX28500 has finished its local configuration (i.e., SRQ_LEN bit field in Service Request Length is reset to zero), the Host issues a service request by writing directly into the Service Request Length register. Writing to this location the actual value of the Service Request Descriptor Table Length from shared memory causes CX28500 to start performing the Service Request Descriptor Table.

PCI Configuration Register 4

CX28500 **Base Address Register**

Table 6-1. PCI Register Map

Register	Byte Offset	Number of instances	Reset Value	Access Type
Receive Port Alive	00000h	Per Port	0	Read Only
Transmit Port Alive	00004h	Per Port	0	Read Only
Interrupt Status Descriptor(3)	00008h	Per Chip	0	Read/Write
Interrupt Queue Pointer	0000Ch	Per Chip	0	Read/Write
Interrupt Queue Length	00010h	Per Chip	0	Read/Write
Service Request Length	00014h	Per Chip	3FF(2)	Read/Write
Service Request Pointer	00018h	Per Chip	0	Read/Write
Soft Chip Reset	00020h	Per Chip	0	Write Only

FOOTNOTE:

(1) There are two address spaces. The first address space includes registers that are directly accessed by Host through the PCI. The second address space (shown in Table 6-2) represents the CX28500's register map accessible by CX28500 via the Service Request Mechanism. Therefore, all the registers shown in this table can be directly accessed by the Host.

(2) During internal initialization, the Service Request Length becomes non-0 to prevent the Host from writing to this register since the Host can only modify this register when its value is 0. After initialization, this register is returned to 0, allowing the Host to modify its content.

(3) The Interrupt Status Descriptor is partially readable and partially writable. The write pointer cannot be modified, but the read pointer is modifiable. Any write accesses to the write pointer are ignored by CX28500.

Table 6-2. Indirect Register Map Address Accessible via Service Request Mechanism

Register	Address	Number of dword Registers	Number of instances	Reset Value	Access Type
RSLP Channel Status	00000h	1024	Per Channel	0	Read Only
RSLP Channel Configuration	01000h	1024	Per Channel	—	Read/Write
RDMA Buffer Allocation	02000h	1024	Per Channel	—	Write Only
RDMA Configuration	03000h	1024	Per Channel	—	Write Only
RSIU Time Slot Configuration	04000h	4096	Per Time Slot	—	Read/Write
RSIU Time Slot Pointer	08000h	32	Per Port	—	Read/Write
RSIU Port Configuration	08080h	32	Per Port	0	Read/Write
RSLP Max Message Length1	08100h	1	Per Chip	—	Read/Write
RSLP Max Message Length2	08104h	1	Per Chip	—	Read/Write
RSLP Max Message Length3	08108h	1	Per Chip	—	Read/Write
Receive Base Address Head Pointer (RBAHP)	0810Ch	1	Per Chip	0	Read/Write
Transmit Base Address Head Pointer (TBAHP)	08110h	1	Per Chip	0	Read/Write
EBUS Configuration	08114h	1	Per Chip	0	Read/Write
Global Configuration	08118h	1	Per Chip	0	Read/Write
TSLP Channel Status	10000h	1024	Per Channel	0	Read Only
TSLP Channel Configuration	11000h	1024	Per Channel	—	Read/Write
TDMA Buffer Allocation	12000h	1024	Per Channel	—	Read/Write
TDMA Configuration	13000h	1024	Per Channel	—	Read/Write
TSIU Time Slot Configuration	14000h	4096	Per Time Slot	—	Read/Write
TSIU Time Slot Pointer	18000h	32	Per Port	—	Read/Write
TSIU Port Configuration	18080h	32	Per Port	0	Read/Write

GENERAL NOTE:

1. These registers must be accessed through the Service Request Mechanism. As shown in Table 6-7, these registers' corresponding addresses are written into bits [18:0] at the Indirect Register Map Address field, which is located in the Device Configuration Descriptor.
2. For better performance and ease of implementation, it is recommended that this Service Request memory block be separated into two blocks for access. Service Request Block 1 starts at address 00000h and ends at address 0811Ch. Service Request Block 2 starts at address 10000h and ends at address 18100h. This separation is made possible by the unused memory gap occurring between the Global Configuration register and the TSLP Channel Status registers.

It is critically important that upon channel activation, shared memory and internal registers must be initialized, valid, and available to CX28500. CX28500 uses the information within the shared memory descriptors to transfer data between the serial interface and shared memory. CX28500 assumes the information is valid once a channel is activated.

6.2 Global Registers

6.2.1 Service Request Mechanism

The registers that need to be configured and checked to enable the activity of the service request mechanism are as follows:

- Service Request Length register (see Table 6-3)
- Service Request Pointer register (see Table 6-4)

The availability of the device to the Host for access is an example of information provided by querying the Service Request Length register. After the PCI reset, CX28500 sets the SRQ_LEN field in Service Request register to all 1s or 0x3FF, until it performs all the internal initialization. Since the Host cannot modify the Service Request Length register while it is non-0, the setting of this register to all 1s by CX28500 effectively prevents the Host from making additional service requests that could interfere with its internal initialization. When CX28500 is finished with the internal initialization, it clears this field to 0. The cleared SRQ_LEN indicates to the Host that it is ready. From this point, the Host is able to directly write this field with the actual number of service requests that CX28500 needs to perform to configure its registers. The number of SRQs written by the Host is stored in the SRQ_LEN field. While processing the service request commands, the SRQ_LEN field indicates how many commands are yet to be processed by CX28500 before a new command can be issued. Host slave writes to this register triggers the execution of the service request list specified by the Service Request Pointer register.

NOTE: Host slave writes to SRQ_LEN field, while the previous list of service requests has not been processed (i.e., SRQ_LEN is reset) implies unpredictable behavior.

Table 6-3. Service Request Length Register

Bit	Field Name	Value	Description
31:10	RSVD	0	Reserved.
9:0	SRQ_LEN [9:0]	—	Service Request Length. After a PCI reset, Host reads the SRQ_LEN bit field through PCI slave access. While the SRQ_LEN value equals 0x3FF, CX28500 is not ready to start the configuration of the device. If CX28500 resets this value, the device is ready to be configured. Host directly writes at this location the number of Service Request Descriptors (SRD) which were allocated in Service Request Descriptor Table (SRDT) i.e., shared memory. The SRDs was previously initialized and configured in SRDT. This value represents the number of service request commands queued by Host (i.e., the SRDT), that are waiting to be performed. Real-time reads from SRQ_LEN provides the number of service request commands that are waiting to be served. Do not write SRQ_LEN until after the SRQ_PTR, or the address of the Service Request Descriptor Table, is initialized. Because CX28500 starts to process service requests whenever SRQ_LEN is non-0 by modifying SRQ_LEN to the non-0 before a valid SRQ_PTR is written, CX28500 processes the invalid SRQ_PTR, resulting in unpredictable behaviors.

The Service Request Pointer register provides the address of the Service Request Descriptor Table (see Table 6-4). Host needs to allocate and initialize this table in shared memory.

Table 6-4. Service Request Pointer Register

Bit	Field Name	Value	Description
31:3	SRQ_PTR [31:3]	—	Service Request Pointer. These 29 bits are appended with 000b to form a 64-bit address Quadword aligned. This address points to the first entry on the Service Request Descriptor Table allocated in shared memory.
2:0	SRQ_PTR [2:0]	0	To ensure Quadword alignment.

6.2.1.1 Service Request Descriptor

A Service Request Descriptor (SRD) is a 4-dword location in shared memory. Actually, one represents an entry in the SRD table. The SRD is defined as a union type in C, which allows different commands to be configured in a 4-dword space. The SRD can handle three different configurations: Device Configuration Descriptor (DCD), EBUS Configuration Descriptor (ECD), and Channel Configuration Descriptor (CCD).

A list of service request commands is defined as a sequence of SRDs. The following instructions, referred to in the document as OPCODE, are supported:

- Configure a port/channel
- Read specific descriptors
- Activate or reactivate a channel
- Deactivate a channel
- Jump to new Buffer Descriptor table
- No-operation command
- Perform EBUS access (read or write)

Table 6-5 defines the Service Request Descriptor OPCODE.

A service request is issued to a specific channel, or per whole device. Each service request command is acknowledged by sending a service request acknowledge interrupt descriptor back to the Host if the SACKIEN bit is enabled in SRD. It is possible for the Host to issue multiple service requests successively without expecting or receiving acknowledgments from each request if the SACKIEN bit was not set accordingly in the SRD.

Host can only set the SACKIEN bit in the last SRD so if a SACK Service Request Acknowledge interrupt is received, it validates the whole list of service request commands.

Activate, Deactivate, or Jump commands could take a long time before they are actually executed by CX28500. CX28500 returns the SACK (if SACKIEN bit is set) immediately after it started the command execution. Therefore, the Host may not assume the command was actually executed just by detecting the SACK was returned. Another interrupt, End Of Command Execution (EOCE), is defined for each of these three commands. The Host may assume the command was actually executed only after receiving the appropriate EOCE.

In general, SACK is only issued after each service table entry is completed. However, for Activate, Deactivate, and Jump commands, SACKs are issued after SLP is completed, but before DMA completes (see End of Command interrupt).

Table 6-5. Service Request Descriptor—OPCODE Description *(1 of 2)*

OPCODE	Value	Description
NOP	0h	No Operation. This service request performs no action other than to facilitate a Host Service Acknowledge Interrupt (SACK). This would be used as a UNIX ping-like operation to detect the presence of CX28500.
CONFIG_WR	1h	Configuration Write. This is a request to copy from shared memory data into CX28500's internal registers. This service request can be issued for either one register or for the whole CX28500 register map (up to 16 K-1 registers), depending on the value of LENGTH bit field set in Service Request Descriptor. Note that the Service Request Descriptor used for this command is Device Configuration Descriptor. The LENGTH bit value in this descriptor is up to 16 K. Assuming that the Host configures an 16 K register structure in shared memory and the LENGTH bit field is set accordingly, CX28500 configures the entire configuration in one service request command in bursts of 32 dwords (i.e., the maximum allowed PCI burst).
CONFIG_RD	2h	Configuration Read. This is a request to copy the configuration of CX28500's internal registers into shared memory. The configuration located at the address specified by the CX28500 register Map Base Address Offset is read and copied to the address specified by the shared memory address. The number of dwords copied is specified in the LENGTH bit field. The user needs to instruct CX28500 to perform the correct number of reads so that when data is written in shared memory, no data overlapping occurs. The Service Request Descriptor used for this command is Device Configuration Descriptor.
CH_ACT	3h	Channel Activation. This is a request to activate a single channel. CX28500 assumes that the channel was already configured. If the channel is currently active, this command results in a destructive termination of the current message being processed, as well as flushing any other messages residing in the channel's FIFO. CX28500 fetches the channel's new head pointer from shared memory (actually from the Transmit Head Pointer Table (THPT) or Receive Head Pointer Table (RHPT)), before it internally activates the channel. The Service Request Descriptor used for this command is Channel Configuration Descriptor.
CH_DEACT	4h	Channel Deactivation. This is a request to deactivate a channel. This command results in a destructive termination of the current message being processed, and flushing of any other messages residing in the channel's FIFO. The SRD used for this command is Channel Configuration Descriptor.
CH_JMP	5h	Channel Jump. This is a request to jump to a new head pointer buffer descriptor. If there is an active transfer of a message from or to shared memory, the channel jump service request command (CH_JMP) is not executed until an EOM indication is received. When an EOM indication is received (or if there is no active transfer), CX28500 fetches a new head pointer buffer descriptor for this channel from shared memory (actually from the Transmit Head Pointer Table (THPT) or Receive Head Pointer Table (RHPT)).
EBUS_WR	6h	EBUS Write. This is a request to execute write transaction(s) over the EBUS. Data is copied from Host memory to the EBUS.

Table 6-5. Service Request Descriptor—OPCODE Description (2 of 2)

OPCODE	Value	Description
EBUS_RD	7h	EBUS Read. This is a request to execute read transaction(s) over the EBUS. Data is copied from the EBUS Address specified in the 3rd dword of EBUS Configuration Descriptor to the shared memory location specified in the 2nd dword of EBUS Configuration Descriptor. The data length copied from one location to another location is specified by LENGTH bit field in EBUS Configuration Descriptor. Note: The EBUS_RD and EBUS_WR Service Request mechanism allow a maximum of 16 K dwords transfer to/from the EBUS. The transaction is split to bursts of 32 dwords over the PCI.
RSVD	8h-1Fh	Reserved

6.2.1.2 Service Request Descriptors

Each SRD is 4 dwords. The SRDs used by CX28500 are as follows:

- Device Configuration Descriptor (DCD)
- EBUS Configuration Descriptor (ECD)
- Channel Configuration Descriptor (CCD)

Device Configuration Descriptor (DCD)

Table 6-6 presents the structure of DCD.

Table 6-6. Device Configuration Descriptor

Dword Number	Bit 31			Bit 0
dword 0	OPCODE[31:27]	SACKIEN[26]	Reserved[25:14](1)	Length[13:0]
dword 1	Shared Memory Pointer [31:2](2)			
dword 2	Reserved [31:19](1)	Indirect Register Map Address [18:2](2)		
dword 3	Reserved(1)			

FOOTNOTE:
(1) All reserved bits must be written to 0 for forward compatibility.
(2) Bits [1:0] must be equal to zero for dword alignment.

Table 6-7 describes these fields.

Table 6-7. Device Configuration Field Descriptions

Dword Number	Descriptor Field	Size (bits)	Value	Description
dword 0	OPCODE	5	1	CONFIG_WR
			2	CONFIG_RD
	SACKIEN	1	0	SACK interrupt disabled
			1	SACK interrupt enabled. An appropriate interrupt is generated after the command execution is completed.
	Reserved	12	0	A read from this field returns all zeros. A write to this field does nothing.
	Length	14	—	Number of dword memory commands. Zero length equals 16 K, but valid maximal length is [16 K–1]. Therefore, a length of 0 is invalid; do not use.
dword 1	Shared Memory Pointer	32	—	Address in shared memory where the Host service request descriptor table exists. The two LSBs must equal 0 for dword alignment.
dword 2	Indirect Register Map Address	19	—	Register Map Address to the specified register that needs to be configured. This address is a dword-aligned address, meaning the 2 LSB must equal to 0. Bits [18:2] contain the map address while bits [31:19] are reserved.

GENERAL NOTE:
1. In general, reserved fields should be written with zeros.

EBUS Configuration Descriptor (ECD)

Table 6-8 presents the EBUS Configuration Service Request Descriptor.

Table 6-8. EBUS Configuration Service Request Descriptor

Dword Number	Bit 31					Bit 0
dword 0	OPCODE[31:27]	SACKIEN[26]	Reserved [25:19]	FIFO_BURST[18]	EBUS Byte Enable [17:14]	Length[13:0]
dword 1	Shared Memory Pointer[31:2](1)					
dword 2	(3)	EBUS Base Address Offset[30:0]				
dword 3	Reserved(2)					

FOOTNOTE:
(1) Bits [1:0] must equal 0 for dword alignment.
(2) All reserved bits must be written to zeros for forward compatibility.
(3) Bit 31 must be set to 1 for all transactions.

Table 6-9 describes the ECD fields.

Table 6-9. Field Descriptions of ECD

Dword Number	Descriptor Field	Size (bits)	Value	Description
dword 0	OPCODE	5	6	EBUS_WR
			7	EBUS_RD
	SACKIEN	1	0	SACK interrupt disabled.
			1	SACK interrupt enabled.
	Reserved	7	0	A read from this field returns all zeros. A write to this field does nothing.
	FIFO-BURST	1	0	Do increment address by 1 after each EBUS access.
			1	Do not increment EBUS address for this access. This feature allows same location access (i.e., FIFO read/write).
	EBUS Byte Enable	4	—	Determinates which byte lanes carry meaningful data in EBUS transactions. The BE [0] applies to byte 0 (LSB).
	Length	14	—	Number of dwords to transfer over EBUS (up to 16 K dwords, in bursts of up to 32 dwords per burst). LENGTH = 0 is not allowed.
dword 1	Pointer Shared Memory	32	—	The address of shared memory EBUS base address, where the configuration of local devices exists. This pointer is dword-aligned, hence bits [1:0] must equal to 0.
dword 2	EBUS Base Address Offset(1)	31	—	EBUS base (byte-aligned) address for an EBUS transaction.

FOOTNOTE:
(1) Bit 31 must be set to 1 for all transactions.

Channel Configuration Descriptor (CCD)

Table 6-10 presents the structure of CCD.

Table 6-10. Channel Configuration Service Request Descriptor

Dword Number	Bit 31				Bit 0
dword 0	OPCODE[31:27]	SACKIEN[26]	Reserved[25:11](1)	Direction[10]	Channel[9:0]
dword 1	Reserved(1)				
dword 2	Reserved(1)				
dword 3	Reserved(1)				

FOOTNOTE:
(1) All reserved bits must be written to zeros for forward compatibility.

Table 6-11 describes the CCD fields.

Table 6-11. Fields Description of CCD

Dword Number	Descriptor Field	Size (bits)	Value	Description
dword 0	OPCODE	5	0	NOP
			3	CH_ACTIV
			4	CH_DEACT
			5	CH_JUMP
	SACKIEN	1	0	SACK interrupt disabled.
			1	SACK interrupt enabled.
	Reserved	—	0	A read from this field returns all zeros. A write to this field does nothing.
	Direction	1	0	The command is for the Receive channel.
			1	The command is for the Transmit channel.
	Channel	10	—	Channel number. This field is interpreted as a channel number for the CH_ACT, CH_DEACT and CH_JUMP command. The field is interpreted as reserved for the NOP command.

GENERAL NOTE: In general, reserved fields should be written to with zeros.

6.2.2 Port Alive Registers

The Receive and Transmit Port Alive registers are read-only registers. These registers can only be accessed via direct PCI read.

Each bit of the Receive and Transmit Port Alive register represents the device port number. Refer to Tables 6-12 and 6-13 for these registers.

Table 6-12. Receive Port Alive Register

Bit	Field Name	Value	Type	Description
31:0	RPA [31:0]	0	RO	This register controls the access to the Receive Port Configuration register. If one of 32 bits is set to 1, then the Receive Port Configuration for that specific port is allowed.

Table 6-13. Transmit Port Alive Register

Bit	Field Name	Value	Type	Description
31:0	TPA [31:0]	0	RO	This register controls the access to the Transmit Port Configuration register. If one of 32 bits is set to 1, then the Transmit Port Configuration for that specific port is allowed.

These registers operate as a gate which enables or disables the access to the Port Configuration register. If the corresponding bit of the Receive and Transmit Port Alive register is set, a new port configuration for the specified port is allowed.

After a PCI reset or Software Chip reset, all 32 bits of the Receive and Transmit Port Alive register are cleared (set to 0). Each bit is automatically set to 1 after 8 or 16 serial clock cycle occur on that specific port. After the corresponding bit is set to 1, the Host can write to the Port Configuration register. In addition, Port Alive can also be reset by writing to the Port Configuration register. If the Host writes to a dead port, the SRQ completes but the register is not modified.

The Host cannot program a new port configuration until the corresponding bit/port is set to 1 in the Port Alive register depending upon the direction of receive or transmit.

A proper configuration sequence for accessing the Port Configuration register is as follows:

1. Host polls the Port Alive register for the specific port/direction and waits (at an interval of 8–16 line clocks) until the corresponding bit in the Port Alive register is set.
2. Host issues a Service Request (SRQ) Port Configuration command and waits for a Service Request Acknowledge (SACK), or polls the SR length to determine when the table entry is completed.
3. Host gets the SACK.

NOTE: Writing to the Port Configuration register causes the corresponding bit from the Port Alive register to be cleared. This bit is automatically set to 1 after 8 or 16 serial clocks occur; therefore a new port configuration is allowed.

4. Host checks if a new port configuration is allowed by checking he corresponding bit in the Port Alive register. Go to 1.

6.2.3 Soft Chip Reset Register

Any write of any value to a Soft Chip Reset (SCR) generates a soft reset for CX28500. An SCR write affects CX28500 exactly as PCI Reset, except that the PCI block is not reset. No PCI configuration is performed after a SCR.

6.2.4 General PCI Note

While addressing CX28500 in slave mode, every PCI access must have all four byte enables active. Any PCI accesses without all four bytes enabled is treated as if all four byte enables were inactive.

6.3 Interrupt Level Descriptors

CX28500 generates interrupts for a variety of reasons. Interrupts are events or errors detected by CX28500 during internal processing. Interrupts are generated by CX28500 and forwarded to the Host for servicing.

CX28500 gathers the many events and errors (generated by all units such as RxDMA and TxDMA, RSLP and TSLP, and SIU) and notifies the Host by asserting PCI interrupts. Interrupt descriptors are generated by CX28500 and forwarded to the Host for servicing. Individual types of interrupts can be masked from being generated by setting the appropriate interrupt mask or interrupt disable bit fields in various descriptors. The interrupt mechanism, each individual interrupt, and interrupt controlling mechanisms are discussed in this section.

6.3.1 Interrupt Queue Descriptor

CX28500 employs a single Interrupt Queue Descriptor to communicate interrupt information to the Host. This descriptor is stored within CX28500 in an internal register. The descriptor in this register stores the location and the size of an interrupt queue in allocated shared memory where the interrupt descriptors is directly pushed by CX28500 while acting as a PCI bus master. CX28500 requires this information to transfer interrupt descriptors to shared memory. All the interrupts are processed by the Host, in an Interrupt Service Routine (ISR). CX28500 directly writes Interrupt Descriptors into the shared memory Interrupt Queue using PCI bus master mode. CX28500's PCI interface must be configured to allow bus mastering.

The Interrupt Queue Descriptor (i.e., Interrupt Queue Pointer and Interrupt Queue Length) is initialized by the Host via a direct PCI write transaction. After a PCI Reset or Software Chip Reset (SCR), the Interrupt Queue Pointer is the first register that needs to be initialized. A typical initialization procedure is as follows:

1. The Host writes in the Interrupt Queue Pointer register allocated by performing a direct write to the address of the Interrupt Queue in shared memory.
2. The Host writes the Interrupt Queue Length register by performing a direct write to this location, the value of the interrupt queue length allocated in shared memory.

NOTE: The user can change, at any time, the length of the Interrupt Queue (IQLEN field in the Interrupt Queue Length register) or the pointer value of the Interrupt Queue Pointer (IQPTR field in the Interrupt Queue Pointer register). However, writing to these registers while the chip is operating may result in flushing the interrupts held in the internal FIFO.

Tables 6-14 through 6-16 list the details of the Interrupt Queue descriptor.

Table 6-14. Interrupt Queue Descriptor

Byte Offset in CX28500 Register Map	Field Name	dwords
000Ch	Interrupt Queue Pointer	1
0010h	Interrupt Queue Length	1

Table 6-15. Interrupt Queue Pointer

Bit	Field Name	Value	Description
31:3	IQPTR [31:3]	—	These 29 bits are appended with 000b to form a 64-bit aligned address. This address points to the first entry (Quadword) of the Interrupt Queue buffer. The Host can change this field while the chip is operating. However, this results in flushing all interrupts residing in the internal interrupts FIFO.
2:0	IQPTR [2:0]	0	Ensures 64-bit alignment.

Table 6-16. Interrupt Queue Length

Bit	Field Name	Value	Description
31:15	RSVD	0	Reserved.
14:0	IQLEN[14:0]	—	This 15-bit number specifies the number of interrupt descriptors. The maximum size for an interrupt queue is 32,768 descriptors. This is a 0-based number. A value of 1 indicates that the queue length is 2 descriptors long, the required minimum. The Host can change this field while the chip is operating. However, this results in flushing all interrupts residing in the internal interrupts FIFO. After reset, IQLEN is set to 0. This has the effect of blocking all the interrupt processing by CX28500. Similarly, writing a 0 to IQLEN forces CX28500 to stop writing interrupts. This feature can be used to switch interrupt queues. To switch interrupt queues, first write a 0 to IQLEN. Next, write the base address of the new interrupt queue into IQPTR in the Interrupt Queue Pointer. Finally, write the new interrupt queue length into IQLEN.
Note(s): Since CX28500 must work with 64-bit alignment, there must be an even number of entries in the buffer.			

6.3.1.1 Interrupt Descriptors

The interrupt descriptor describes the format of data transferred into the queue. There are two different types of interrupt descriptor. The first type represents DMA's block-related interrupts, and the second type represents other interrupts. Both types are 64-bit fields. Generically, the interrupt descriptor includes fields for the following:

- Identifying the source of interrupt from within the CX28500 channel causing the interrupt (0–1023) and direction (receive or transmit)
- Events assisting the Host in synchronization channel activities
- Errors and unexpected conditions resulting in lost data, discontinued message processing, or prevented successful completion of a service request
- Number of bytes transferred to or from shared memory

All the interrupts are associated with a channel or direction with the following exceptions:

1. When an OOF or COFA condition is detected on a serial port, only one interrupt is generated for the port until the condition is cleared and the condition reoccurs.
2. The ILOST interrupt bit indicates that an interrupt has been lost internally when CX28500 generates more interrupt descriptors than can be stored in the internal Interrupt Queue. The latency of Host processing of the Interrupt Queue (handling the interrupts in the IRS) in shared memory may cause an ILOST condition. This condition is conveyed by CX28500 overwriting the ILOST bit field in the last interrupt descriptor in the internal queue prior to being transferred to shared memory. The ILOST field is not specific to or associated with the Interrupt Descriptor being overwritten. Only the ILOST bit is overwritten, and the integrity of the original descriptor is maintained.

3. The PERR interrupt bit indicates that a parity error was detected by CX28500 during a PCI access cycle. This condition is conveyed by CX28500 overwriting the PERR bit field in the last interrupt descriptor in the internal queue prior to being transferred out to shared memory. The bit field is not specific to or associated with the interrupt descriptor being overwritten. Only one bit is overwritten and the integrity of the original descriptor is maintained.

The CX28500 has two types of interrupt descriptor. One is the DMA Interrupt Descriptor, and the other is the Non-DMA Interrupt Descriptor.

The following items describe the errors/events reported in the DMA Interrupt Descriptor:

- ILOST (Interrupt Lost)
- RxEOB/TxEOB (Receive or Transmit End Of Buffer)
- RxONR/TxONR (Receive or Transmit Owner bit error)
- RxEOM (Receive End Of Message)
- RxEOCE/TxEOCE (Receive or Transmit End Of Command, i.e., Activate, Deactivate or Jump Execution)
- If EOM (i.e., the message is an end of message), the other errors/events reported in the DMA Interrupt descriptor are as follows:
  - RxBUFF Overflow
  - RxCOFA Change Of Frame Alignment
  - RxOOF Out Of Frame
  - RxABT Abort Frame
  - RxLNG Long Message
  - RxALIGN Byte Alignment Error
  - RxFCS Frame Check Sequence Error

The following items describe the errors/events reported in the Non-DMA Interrupt Descriptor.

- RxBUFF/TxBUFF (Receive and Transmit Buffer errors)
- RxSHT/TxEOM (Receive Message Too Short, Transmit End Of Buffer)
- RxCHABT (Receive Change to Abort)
- RxCHIC (Receive Change to Idle Codes)
- RxCOFA/TxCOFA (Receive and Transmit Change Of Frame Alignment)
- RxOFF (Receive Out Of Frame)
- ProgERR (Programming Error, an attempt to access an illegal address within CX28500) valid only when SACK = 1, ignore Prog ERR otherwise
- RxFREC/RxSPORT (Receive Frame Recovery, Receive Serial Port Interrupt)
- RxCREC/TxCREC (Receive COFA Recovery, Transmit COFA Recovery)

DMA Interrupt Descriptor Format

The DMA interrupt descriptor is 69 bits wide, and the detailed description of its field is provided in Table 6-17. The most significant bit in the DMA interrupt descriptor is always read as 0.

Table 6-17. DMA Interrupt Descriptors Format *(1 of 2)*

Bit	Field Name	Value	Description
63	TYP	0	Interrupt descriptor type 0.
62:58	RSVD	—	Reserved
57:48	CH [9:0]	—	The channel number causing the interrupt.
47:43	RSVD	—	Reserved
42	DIR	—	0: Direction—receive 1: Direction—transmit
41:40	RSVD	—	Reserved
39:38	INT[1:0]	—	0: RxEOB/TxEOB, (Receive or Transmit End Of Buffer) This event is generated when current data buffer has been completely processed, and the EOBIEN bit field is set in the associated Receive/ Transmit Buffer Descriptor. The EOB interrupt reports the correct number of received transmitted bytes in BLEN field. 1: RxONR/TxONR–Generated when the next Buffer Descriptor is not available to CX28500 when expected, the NP bit field in the buffer descriptor is set and the DMA is in the middle of a message, and the ownership interrupt is enabled (bit field ONR in RDMA Channel Configuration register or TDMA Channel Configuration register). 2: RxEOM–Generated when End Of Message occurs (even if errors were detected). If EOMIEN (bit field in the Receive Buffer Descriptor) is enabled, an RxEOM is generated regardless the value of EOBIEN. In other words, the EOM gets higher priority than an EOB event. An EOM interrupt reports the correct number of received bytes in BLEN field. Only when RxEOM = 1, which means an end of messages has been reported for that particular message and the RxERR field is relevant. 3: RxEOCE/TxEOCE–End Of Command Execution generated after service request channel activation, deactivation or jump.
37:36	RSVD	—	Reserved

Table 6-17. DMA Interrupt Descriptors Format *(2 of 2)*

Bit	Field Name	Value	Description
35:33	RxERR/ EOCE[2:0]	—	When this is an RxEOM interrupt then this field specifies the error type: 0: Receiver error (decoded)—no error. 1: RxBUFF—Overflow. 2: RxCOFA—Change Of Frame Alignment. 3: RxOOF—Out Of Frame. 4: RxABT—Abort Termination. It is generated when the received message is terminated with an abort sequence (seven consecutive 1s) instead of the specific closing flag, 7Eh. 5: RxLNG—Long Message. Generated when received message length (after zero extraction) is greater than the selected maximum message size. The message reception is terminated and transfer to shared memory is not performed. 6: RxALIGN—Byte Alignment Error. It is generated when the message payload size (after zero extraction), is not a multiple of 8 bits. This generally occurs with an FCS error. The FCS interrupt is not generated if the ALIGN interrupt was issued. 7: RxFCS—Frame Check Sequence Error. It is generated when received HDLC frame is terminated with a byte aligned 7Eh flag but the computed FCS does not match the received FCS. When this is an RxEOCE or TxEOCE interrupt then this field specifies the command type that was executed: 0: Activate 1: Deactivate 2: Jump 3–7: Reserved
32:28	RSVD	—	Reserved
27:14	BLEN[13:0]	—	Receiver—Number of received bytes. Transmitter—Buffer length.
13:1	RSVD	—	Reserved
0	ILOST	—	0: No interrupts have been lost. 1: Interrupt Lost. Generated when internal interrupt queue is full and more interrupt conditions are detected. Because CX28500 cannot store the newest interrupt descriptors, it discards the new interrupts and overwrites this bit in the last interrupt in an internal queue prior to that interrupt being transferred out to shared memory. The integrity of the descriptor being overwritten is maintained completely.

Non-DMA Interrupt Descriptor Format

The Non-DMA Interrupt Descriptor is 64 bits wide, and the detailed description of its field is provided in Table 6-18. The most significant bit in the Non-DMA Interrupt Descriptor is always read as 1.

Table 6-18. Non-DMA Interrupt Descriptors Format *(1 of 2)*

Bit	Field Name	Value	Description
63	TYP	1	Interrupt Descriptor Type 1.
62:58	RSVD	—	Reserved.
57:48	CH [9:0]	—	The channel number causing the interrupt.
47:43	RSVD	—	Reserved.
42	CHDIR	—	0: Receive direction. 1: Transmit direction.
41	RxBUFF/ TxBUFF	—	Receive or Transmit Buffer Errors The data is lost. CX28500 has no place to read or write data internally. If reported as TxBUFF, the internal buffer underruns. If reported as RxBUFF, the internal buffer overflows.
40:38	RSVD	—	Reserved.
37	RxSHT/ TxEOM	—	RxSHT—Receive direction message too short or other error causes a truncated message. The error is generated when the received message length (after zero extraction in HDLC mode) is less than or equal to the number of bits in FCS field. The message data is not transferred to shared memory. TxEOM—Transmit direction End Of Message.
36:34	RSVD	—	Reserved.
33	RxCHABT/ TxRSVD	—	RxCHABT—Receive direction Change to Abort code. The event is generated when a received pad fill code changes from 7Eh to 7Fh (15 consecutive 1s are detected). Transmit: reserved
32:30	RSVD	—	Reserved.
29	RxCHIC/ TxRSVD	—	RxCHIC—Receive direction Change to Idle Code (RxCHIC). The event is generated when a received pad fill code changes from 7Fh to 7Eh, or upon the detection of the first flag after channel activation. Transmit: reserved.
28:25	RSVD	—	Reserved.
24:20	PRT [4:0]	—	The port number causing the interrupt.
19:18	RSVD	—	Reserved.
17	PRTDIR	—	0: Receive direction of port causing the interrupt. 1: Transmit direction of port causing the interrupt.
16	RxCOFA/ TxCOFA	—	Receive and Transmit Change Of Frame Alignment.
15	RxOOF/ TxRSVD	—	RxOOF—Receive direction Out Of Frame. Generated when serial port is configured in channelized mode and Receiver-Out-Of-Frame (ROOF) input signal assertion is detected. Transmit: reserved.

Table 6-18. Non-DMA Interrupt Descriptors Format *(2 of 2)*

Bit	Field Name	Value	Description
14	RxFREC/ RxSPORT/ TxRSVD	—	RxFREC—Receive direction. When ROOF signal deasserts, frame recovery (FREC) interrupt is generated. The serial port transitions from Out-Of-Frame (OOF) back to in-frame. RxSPORT—Receive Direction. When ROOF signal is used as a general serial port interrupt line (SPORT). If OOFABT=0 and OOFIEN=1 bit fields in RSIU Port Configuration register and ROFF transitions from high-to-low, the SPORT Interrupt is generated. The data stream processing is not affected. Transmit–Reserved.
13	CREC	—	COFA recovery—Generated when the internal COFA signal deasserts. The serial port transitions from COFA back to in-frame.
12:4	RSVD	—	Reserved.
3	PROGERR	—	Reports a programming error: an attempt was made to perform a SRQ access to an illegal address within CX28500. An illegal address is any address which is not defined in Table 6-2. Reports status of SACK, only valid when SACK = 1.
2	SACK	—	DMA service request acknowledge. Generated to conclude successfully a service request command of Host service.
1	PERR	—	0: No PCI parity errors have been detected. 1: PCI Bus Parity Error. Generated when CX28500 detects a parity error on data being transferred into/from CX28500, either from another PCI agent that writes into CX28500 registers or from CX28500 that reads data from shared memory. This error is specific to the data phase of a PCI transfer while CX28500 is receiving data. PCI system error signal, SERR*, is ignored by CX28500. To mask the PERR interrupt—in CX28500's PCI Configuration Space, Function 0, Register 1—Parity Error Response field must be set to 0.
0	ILOST	—	0: No interrupts have been lost. 1: Interrupt Lost. Generated when CX28500's (internal) interrupt queue is full and more interrupt conditions are detected. As CX28500 has no way to store the newest interrupt descriptors, it discards the new interrupts and overwrites this bit in the last interrupt in an internal queue prior to that interrupt being transferred out to shared memory. The integrity of the descriptor being overwritten is maintained completely.

6.3.1.2 Interrupt Status Descriptor Register

The interrupt status descriptor is located in a fixed position in CX28500's internal register. CX28500 updates this descriptor after each transfer of interrupt descriptors from its internal queue to the Interrupt Queue in shared memory. The Host is required to read this descriptor from CX28500's registers before it processes any interrupts. The contents of the interrupt status descriptor are reset on hardware reset or soft chip reset or whenever any field in the Interrupt Queue Descriptor is modified.

NOTE: This internal register is directly accessed by the Host.

Table 6-19 lists the details of the Interrupt Status Descriptor.

Table 6-19. Interrupt Status Descriptor

Bit	Field Name	Host Access	Value	Description
31	MSTRABT	R	—	When CX28500 encounters a PCI abort while operating as a PCI master, it does not attempt to recover from this error. In this case CX28500 asserts the SERR* signal, and the MSTRABT bit and waits for the Host to reset (i.e., PCI reset or Soft reset). This bit is asserted when the target does not assert DEVSEL within a specific PCLK cycles or when the target terminates a transaction in which CX28500 is the master, with an abort (i.e., assertion of STOP# with a deassertion of DEVSEL) sequence.
30:16	WRPTR[14:0]	R	—	Write Interrupt Pointer. 15-bit Quadword index from start of Interrupt Queue up to where CX28500 is going to insert the next Interrupt Descriptors. The Host may read this value to get the location of the last descriptor, which was not served yet, in the queue. As the queue is circular, care must be taken to ensure roll over at beginning and end of queue. Only CX28500 updates this value. The WRPTR is a read only bit field.
15	INTFULL	R	—	0: Interrupt Queue Not Full–shared memory. 1: Interrupt Queue Full–shared memory. The Host writing ANY value to the ReadPtr clears the INTFULL status bit.
14:0	RDPTR[14:0]	R/W	—	Read Interrupt Pointer. 15-bit Quadword index from start of Interrupt Queue up to where The Host first unread Interrupt Descriptor resides. The Host may read this value to get the location of the first descriptor, which was not served yet, in the queue. As the queue is circular, care must be taken to ensure roll over at beginning and end of queue. Only The Host updates this value. The ReadPtr is a read/write bit field. Note: Writing the value of the ReadPtr automatically resets the INTFULL status bit. Therefore, if the value written into RDPTR is the same value as was read from this field, it is assumed that the Host did read all the interrupt descriptors.

6.3.1.3 Interrupt Handling

Initialization

Interrupt management resources are automatically reset upon the following:

- Hardware reset
- Soft reset
- Write to Interrupt Queue Pointer by a direct PCI write
- Write to Interrupt Queue Length by a direct PCI write

CX28500 uses two interrupt queues. One is internal to CX28500 and is controlled exclusively by the DMA block. The other is the Interrupt Queue in shared memory, which is allocated and administered by the Host, and written to (filled) by CX28500.

Upon initialization, the data in the status descriptor is reset to all 0s, indicating the first location for next descriptor, the queue is not full, and no descriptors are currently in the queue. Any existing descriptors in the internal queue are discarded.

The Host must allocate sufficient shared memory space for the Interrupt Queue. Up to 64 K dwords of queue space are accessible by CX28500, setting the upper limit for the queue size. CX28500 requires a minimum of two Quadwords of queue space. This sets the lower limit for the queue size.

The Host must store the pointer to the queue and the length in Quadwords of the queue in CX28500 within the Interrupt Queue Descriptor registers. Issuing the appropriate Host service to CX28500 can do this. As CX28500 takes in the new values, it automatically resets the controller logic as indicated above. This mechanism can also be used to switch interrupt queues while CX28500 is in full operation.

Interrupt Descriptor Generation

Interrupt conditions are detected in both error and non-error cases. CX28500 makes a determination based on channel and device configuration whether reporting of the condition is to be masked or whether an Interrupt Descriptor is to be sent to the Host. If the interrupt is not masked, CX28500 generates a descriptor and stores it internally prior to transferring it to the Interrupt Queue in shared memory.

The internal queue is capable of holding 512 descriptors while CX28500 arbitrates to master the PCI bus and transfer the descriptors into the Interrupt Queue in shared memory.

As the PCI bus is mastered and after descriptors are transferred out to the shared memory, CX28500 updates the Interrupt Status Descriptor. When CX28500 updates the WRPTR field in the Interrupt Status Descriptor, it asserts the PCI INTA# signal line.

If during the transfer of descriptors, the Interrupt Queue in shared memory becomes full, CX28500 stops transferring descriptors until the Host indicates more descriptors can be written out. CX28500 indicates that it cannot transfer more descriptors into shared memory by setting the bit field INTFULL in the Interrupt Status Descriptor.

In cases where the internal queue is full (either because the Host queue is full or there was not enough PCI bandwidth) and new descriptors are generated, the new descriptors are discarded. CX28500 indicates it has lost interrupts internally by overwriting the bit field ILOST in the last Interrupt Descriptor in the internal queue. The ILOST indication represents one or more lost descriptors.

INTA# Signal Line

The Host must monitor the INTA# signal line at all times. An assertion of this line signifies the updating of the WRPTR field in the Interrupt Status Descriptor, indicating that Interrupt Descriptors have been transferred to the Interrupt Queue in shared memory from the internal interrupt queue.

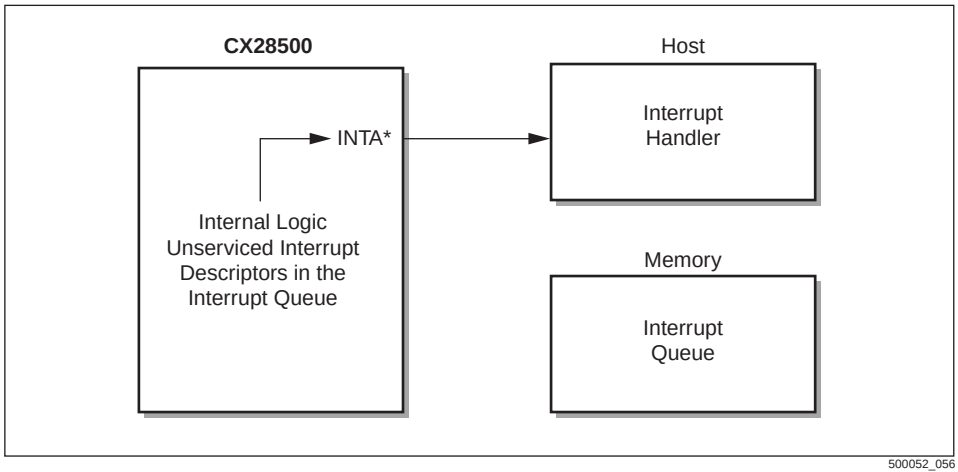
Upon detection of the INTA# assertion, the Host must perform a direct read of the Interrupt Status Descriptor from within CX28500. This descriptor provides the offset to the location of the first descriptor in the Host queue that has not been served, the offset to the location of the last descriptor serviced by the Host, and the determination if the queue is full. The INTA* signal is deasserted on each read of the Interrupt Status Descriptor.

The Host applies its interrupt service routines to service each of the descriptors. As the Host finishes servicing a number of descriptors, it must write the offset to the location of the last serviced descriptor back into the RDPTR field of the Interrupt Status Descriptor. A write to this field indicates to CX28500 that the descriptor locations, which were waiting to be serviced, have been serviced and new descriptors can be written.

NOTE: CX28500 continues to write to available space regardless of whether the Host updates the RDPTR field. The difference between the two interrupt queue pointers RDPTR and WRPTR indicates the number of interrupts still need to be serviced. When calculating the number of outstanding interrupts, please make sure to take care of offsets, or pointers, wraparound.

Figure 6-1 illustrates the operation of INTA*.

Figure 6-1. Interrupt Notification to Host



6.4 Global Configuration Register

The Global Configuration Descriptor specifies configuration information applying to the entire device. This register must be programmed before any channel is activated. The only two fields in this register that can be changed while the chip is operating (i.e., not immediately after reset) are the INTRS field and the PCI_EN field.

The components and their descriptors are given in Table 6-20.

***Table 6-20. Global Configuration Descriptor** (1 of 2)*

Bit	Field Name	Value	Description
31:15	RSVD	0	Reserved
14	PCHMODE	0	Normal operating mode (i.e., not Preserve Channel Mode)
		1	Preserve Channel Mode. In this mode requires ECCMODE = 1: 1. CX28500 transfers in the Rx direction the channel number least significant 8 bits in bits [25:8] in the first dword at Status Buffer Descriptor and the channel most significant 2 bits in bits 1:0 of the second dword of the Status Buffer Descriptor (i.e., the data pointer). 2. In the Tx direction, CX28500 assumes that the field PADCOUNT in the Buffer Descriptor is 0. This enables the Host to write any value that helps to better handle buffers circulation into this location knowing that CX28500 does not use PADCOUNT and preserve this value in the Status Buffer Descriptor. Note that PCHMODE can be set only when ECCMODE bit is set. Doing otherwise causes unpredictable behavior.
13	ECCMODE	0	Normal operating mode (i.e., not ECCMODE)
		1	ECCMODE: CX28500 supports a 64-bit wide ECC memory as the shared memory. The special behavior includes the following: 1. When updating Buffer Descriptors in the shared memory, the whole Buffer Descriptor is written (i.e., 64 bits) rather than the status dword only. 2. When transferring incoming data to shared memory, any transfer which does not include the end of message is ended in 64-bit alignment. 3. Since this mode makes sense only if the data buffers are 64-bit aligned, CX28500 assumes that the three least significant bits in the data pointers of all (i.e., Tx and Rx) buffer descriptors are 0 regardless of their actual value. CX28500 preserves the actual value when updating the buffer descriptor with the status. Start of receive message indication—informs host via SOM-bit in the Receive Buffer Status Descriptor (bit 2 of second dword) that this data buffer contains the beginning of a message.
12	TARGET_64	—	0: Target (any target that CX28500 can access over the PCI) is not guaranteed to allow 64-bit transfers when CX28500 is the master. In this case, a single 64-bit transaction is executed in two 32-bit accesses even when operating in 64-bit PCI (for any transaction greater than a 64-bit transfer, CX28500 always attempts to transfer 64 bits regardless of the setting of this bit). 1: Target (any target that CX28500 can access over the PCI) is guaranteed to allow 64-bit transfer. In this case CX28500 always transfers 64 bits whenever possible.

Table 6-20. Global Configuration Descriptor *(2 of 2)*

Bit	Field Name	Value	Description
11	TARGET_FBTB	—	0: Use the fast back-to-back feature as configured in the PCI configuration settings. 1: CX28500 as PCI master attempts to fast back-to-back the PCI transaction to other targets regardless of PCI configuration settings. This bit is defined to force CX28500's fast back-to-back capability regardless of the PCI configuration. The PCI specification states that if there is a single device in the system that does not support a fast back-to-back transaction as a target, the fast back-to-back mode is disabled. Setting this bit to 1 instructs CX28500 to ignore the PCI configuration settings and execute fast back-to-back transactions when appropriate. The Host can set this bit only if CX28500 is always accessing the same target which is capable of fast back to back transactions. This is not a violation of the PCI specification, rather it is an implementation of allowed behavior.
10	BR	—	0: Little-Endian Storage Convention (Intel-style). The least significant byte to be stored in and retrieved from the lowest memory address. 1: Big-Endian Storage Convention (Motorola-style). An example of little-big Endian byte ordering is shown in Appendix E.
9:1	INTTRS[8:0]	—	Threshold of internal interrupt queue Service request. This bit field contains the configuration parameter of DMA internal interrupt service request queue. Once the internal queue contains at least one interrupt vector, a low priority DMA request is generated toward the internal PCI arbiter. This request is removed when the internal queue is empty. If the internal queue contains greater than or equal INTTRS number of descriptors, a high priority request is generated toward the internal PCI arbiter. This request is removed when the internal queue contains less than INTTRS descriptors. See the description in DMA Internal Arbiter. This field must never equal 0.
0	PCI_EN	—	0: PCI Interrupt disabled–global interrupt mask. 1: PCI Interrupt enabled.

6.5 EBUS Configuration Register

The EBUS Configuration Descriptor, defined in Table 6-21, specifies the configuration parameters for EBUS transactions. The Host must configure this register before any attempt to access the EBUS.

Table 6-21. EBUS Configuration Register

Bit	Field Name	Value	Description
31:14	RSVD	0	Reserved.
13	ECLKDIV(1)	0	EBUS clock (ECLK) has the same frequency as the PCI clock (PCLK).
		1	EBUS clock (ECLK) has half the frequency of the PCI clock (PCLK).
12	MPUSEL	0	Expansion Bus Microprocessor Selection–Motorola-style. Expansion bus supports the Motorola-style microprocessor interface and uses Motorola signals: Bus Request (BR*), Bus Grant (BG*), Address Strobe (AS*), Read/Write (R/WR*), and Data Strobe (DS*).
		1	Expansion Bus Microprocessor Selection–Intel-style. Expansion bus supports the Intel-style microprocessor interface and uses Intel signals: Hold Request (HOLD), Hold Acknowledge (HLDA), Address Latch Enable (ALE*), Write Strobe (WR*), and Read Strobe (RD*).
11	ECKEN	0	Expansion Bus Clock Disabled. ECLK output is three-stated.
		1	Expansion Bus Clock Enabled. CX28500 re-drives and inverts PCLK input onto ECLK output pin.
10:8	ALAPSE[2:0]	—	Expansion Bus Address Duration. CX28500 extends the duration of valid address bits during an EBUS address phase to ALAPSE+1 number of ECLK periods. The control lines ALE* (Intel) or AS* (Motorola) indicate that the address bits have had the desired setup time.
7:4	BLAPSE[3:0]	—	Expansion Bus Access Interval. CX28500 waits BLAPSE number of ECLK periods immediately after relinquishing the bus. This wait ensures that all the bus grant signals driven by the bus arbiter have sufficient time to be deasserted as a result of bus request signals being deasserted by CX28500.
3:0	ELAPSE[3:0]	—	Expansion Bus Data Duration. CX28500 extends the duration of valid data bits during an EBUS data phase to ELAPSE+1 number of ECLK periods. The control lines RD* and WR* (Intel) or DS* and R/WR* (Motorola) indicate the data bits have had the desired setup time.

FOOTNOTE:

(1) After reset, the value of EBUS Configuration register is 0, except ECLKDIV bit field, which is 1. The user can change the clock division at any time but the EBUS clock must be first reset (disabled), i.e., reset ECKEN bit, and after that a new value of ECLKDIV must be written with ECKEN set.

6.6 Receive Path Registers

Receive path registers contain the information necessary to configure the receive direction. This configuration includes registers that are related to the DMA block, Host interface, registers that control the RSLP line processor, and RSIU.

6.6.1 RSLP Channel Status Register

The RSLP Channel Status register is a Read Only (RO) register.

It provides information from RSLP block regarding the channel state and status. There is one RSLP Channel status register for each of CX28500's channels (i.e., 1024 registers).

Table 6-22. RSLP Channel Status Register

Bit	Field Name	Host	Value	Description
31:4	RSVD	R	0	Reserved.
3:1	RSVD	R	Don't care	Reserved.
0	RACTIVE	R	0	Channel Inactive. The channel has been deactivated due to either a service request channel deactivation or Reset (PCI Reset or Soft Chip Reset)
			1	Channel Active. The channel has been activated by service request channel activation.

6.6.2 RSLP Channel Configuration Register

The Receive Channel Configuration register contains configuration bits applying to the logical channels within CX28500. There are 1024 such registers, one for each channel. The RSLP Channel Configuration Descriptor configures aspects of the channel common to all messages passing through the channel. One descriptor exists for each logical channel direction. Table 6-23 lists the values and descriptions of each channel configuration descriptor.

Table 6-23. RSLP Channel Configuration Register

Bit	Field Name	Value	Description
31:30	RPROTCOL[1:0]	0	TRANSPARENT.
		1	HDLC with no FCS. Used in ISLP or full packet forwarding and/or channel monitoring application. For this mode the SHT message detection is disabled. Any number of bytes can be transmitted and received within any single message including messages of only one byte.
		2	HDLC with FCS16 (FCS—2 bytes).
		3	HDLC with FCS32 (FCS—4 bytes).
29	RINV	0	Data Inversion disabled.
		1	Data Inversion enabled. Message is received from SIU with polarity change (the inversion is done to all received bits).
28:21	RMASK[7:0]	—	Data Mask. Only bits with a value of 1 contain relevant data (e.g., Mask = 10000001, then only bits 0 and 7 contain channel's data). Enables the subchanneling feature.
20:5	RSVD	0	Reserved.
4:3	MAXSEL[1:0]	0	Message Length–Disable maximum message length check.
		1	Message Length–Use MAXFRM1 bit field in the message length descriptor for maximum receive message length limit.
		2	Message Length–Use MAXFRM2 bit field in the message length descriptors maximum receive message length limit.
		3	Message Length–Use MAXFRM3 bit field in the message length descriptor for maximum receive message length limit.
2	FCSTRANS	0	FCS Transfer Normal. Do not transfer received FCS into shared memory along with data message.
		1	Non-FCS Mode. Transfer received FCS into shared memory along with data message. In a Non-FCS Mode a SHT message detection is disabled.
1	BUFFIEN	0	Overflow Interrupt disabled.
		1	Overflow Interrupt enabled.
0	IDLEIEN	0	CHABT, CHIC, SHT Interrupt disabled.
		1	CHABT, CHIC, SHT Interrupt enabled. Receive Only. When receiver detects change to abort code, change to idle code, or too-short message, this bit generates an interrupt to indicate condition.

6.6.3 RDMA Buffer Allocation Register

The Buffer Allocation register configures the internal receive memory. There is one RDMA Buffer Allocation register for each logical channel (i.e., 1024 channels).

CX28500's internal Rx memory is a 32-KB dual-port RAM, which can be split into 1024 parts, one part for each channel. The allocation granularity is two dwords. For each active channel it is required to specify the following:

- The starting location of internal data buffer
- The ending location of internal buffer
- A threshold. This value is triggered when a request from DMA needs to be generated to the internal PCI arbiter. As soon as the channel's internal FIFO contains more data bytes than this threshold, a request to the RDMA to serve this channel is generated, meaning transferring data in the FIFO into shared memory. A request to serve this channel is also generated, regardless of the value of the threshold, if a complete message, or an End Of Message (EOM), resides in the channel's FIFO.

The Host can issue a Service Request to change the value of this register only when the affected channel is inactive. Additionally, internal data buffer allocation must conform to the following criteria:

1. Host must set the buffers so there is no overlap between buffers belonging to different channels.
2. Allocated memory segments should not have wraparounds. That is, ending addresses must be greater than starting addresses.
3. The Host cannot allocate all of the FIFO to one channel. The maximal allocation to one channel is (all FIFO) – 1 quad dwords.

Each receive channel must be allocated buffer space before the channel can be activated. Other important considerations for allocation of internal data buffers include the channel's data rate and PCI latency tolerance. This architecture of configured buffer allocation is completely flexible and allows the Host to assign larger FIFO buffers to channels that operate at higher rates. For applications operating high-speed channels (i.e., hyperchanneling) the Host can increase the FIFO allocation per channel. PCI latency tolerance equals the maximum length of time a particular channel can operate normally between PCI bus transactions without reaching an internal overflow or underflow condition. PCI latency tolerance is primarily dependent on each channel FIFO's buffer size. Table 6.6.3 describes the bit fields in the RDMA Buffer Allocation register. There are 1024 RDMA Buffer Allocation registers, one for each channel.

Table 6-24. RDMA Buffer Allocation Register

Bit	Field Name	Value	Description
31:29	RSVD	0	Reserved
28:16	RDMA_ENDAD[12:0](1)	—	Ending address of internal channel data buffer pointing to dword.
15:13	RSVD	0	Reserved
12:0	RDMA_STARTAD[12:0](2)	—	Starting address of internal channel data buffer

GENERAL NOTE:

1. One channel maximum must be 32 KB–2 bytes (i.e., ENDAD = FFFh and STARTAD = 000h).
2. ENDAD must be greater than STARTAD (i.e., no rollover).
3. Minimal allocation to any channel ≥ 4 dwords (i.e., ENDAD – STARTAD ≥ 3).

FOOTNOTE:

(1) RDMA_ENDAD must be an odd address (i.e., LSB must be 1).
(2) RDMA_STARTAD must be an even address (i.e., LSB must be 0).

6.6.4 RDMA Configuration Register

This register, defined in Table 6-25, controls the per channel RxDMA operational mode. There are 1024 such registers, one for each channel.

Table 6-25. RDMA Channel Configuration Register

Bit	Field Name	Value	Description
31:18	RSVD	0	Reserved
17	EOCEIEN	0	End Of Command Execution Interrupt disabled.
		1	End Of Command Execution Interrupt enabled. Interrupt generated when a command (Activate, Deactivate or Jump) execution was completed.
16	INHRBSD(1)	0	Inhibit Receive Buffer Status Descriptor disabled. At the end of each Receive Data Buffer, overwrite Rx Buffer Descriptor with Rx Buffer Status Descriptor.
		1	Inhibit Receive Buffer Status Descriptor enabled. At the end of each Receive Data Buffer, do not overwrite Rx Buffer Descriptor with Rx Buffer Status Descriptor.
15	ONRIEN	0	ONR Interrupt disabled.
		1	ONR Interrupt enabled. Interrupt generated when a new buffer descriptor is read from the Host memory and this buffer is Host-owned and NP bit field is 1.
14	ERRIEN	0	Error Interrupt disabled.
		1	Error Interrupt enabled. Interrupt generated when End Of Message detected and error occurred (such as: too-long message, FCS error, and message alignment error or abort condition).
13	EOMIEN	0	EOM Interrupt disabled.
		1	EOM Interrupt enabled. Interrupt generated when End Of Message detected and no error occurred (such as too-long message, FCS error, and message alignment error or abort condition).
12:0	RDMA_ TRSHOLD[12:0]	—	Threshold value. As soon as the channel's internal FIFO contains more than or equal number of data dwords than the threshold, a request to the RDMA to serve this channel is generated. The value programmed into this field must not exceed the channel's buffer size, and the threshold must not equal 0.

FOOTNOTE:

(1) The normal mode of operation of the CX28500 is to overwrite the Rx Buffer Descriptor with Rx Buffer Status Descriptor. This bit is used to inhibit this behavior, so that when it is set, the Rx Buffer Descriptor is not overwritten and the Host must rely on interrupts to find out when the CX28500 relinquishes ownership of the buffers.

6.6.5 RSIU Time Slot Configuration Register

6.6.5.1 Time Slot Map

The receive time slot map consists of 4096 time slot descriptors. The entire receive map contains configuration information for 4096 separate time slots. The CX28500's serial ports support 4096 time slots, which can be configured among the CX28500's 32 receive serial ports by setting the RSIU Time Slot Pointer Assignment.

Numerous mappings of time slots are possible, multiple time slots can be mapped to a single channel; however each time slot can map to only one channel at a time. For each serial port two time slot maps are required, one for transmit functions and one for receive functions. The two separate maps are configured independently for transmit (TSIU Time Slot Configuration Descriptor and TSIU Time Slot Pointer Assignment) and receive direction (RSIU Time Slot Configuration Descriptor and RSIU Time Slot Pointer Assignment).

6.6.5.2 RSIU Time Slot Configuration Descriptor

For each time slot there is an RSIU Time Slot Configuration Descriptor.

There are 4096 entries in memory that set the translation between time slots and logical channels for each of CX28500's 32 ports. The actual mapping of these time slot descriptors to the 32 ports is done by 32 sets of pointer pairs (receive and transmit), one pair set for each port, which indicates the start and the end address of the memory location that belongs to the configured port. Time Slot pointer allocation is described in RSIU Time Slot Pointer Allocation.

The bit fields of RSIU Time Slot Configuration Descriptor include information:

- Time slot is enabled or disabled
- Time slot is a full DS0, or subchanneling enabled so that only a part of 64 Kbps transports information
- Indicates if it is the first time slot assigned to the logical channel
- Logical channel number

NOTE: For polling to work properly and efficiently, it is mandatory that the FIRST (receive time slot)/LAST (transmit time slot) indications be configured for each channel regardless of the operational mode (i.e., HDLC, transparent, etc.). The polling mechanism in the CX28500 uses these markings as triggering points. When configured correctly, CX28500 polls on a channel once at the selected poll throttle rate instead of polling at every time slot allocated to that channel, as in the case of hyperchannels. When a channel consists of only one time slot, as in a DSO channel, its corresponding receive time slot should have the FIRST_TS bit set. Likewise, its corresponding transmit time slot should have the LAST_TS bit set.

Table 6-26 specifies the content of each receive time slot configuration descriptor.

Table 6-26. RSIU Time Slot Configuration Descriptor

Bit	Field Name	Value	Description
31:13	RSVD	0	Reserved.
12:3	RCHANNEL[9:0]	—	Logical channel number assigned to the time slot.
2	RTS_ENABLE	0	Time Slot Disabled.
		1	Time Slot Enabled.
1	RMASKEN_SB	0	The RMASK_SB bit field (RSLP Channel Configuration Descriptor) is ignored. All the 8 bits of the time slot are processed.
		1	Allow data mask for the specified time slot. The bits specified by RMASK_SB bit field (RSLP Channel Configuration Descriptor) are processed.
0	RFIRST_TS	0	This bit field indicates that the specified time slot is not the first time slot of the logical channel.
		1	This bit field indicates that the specified time slot is the first time slot of the logical channel. If a serial port is configured to operate in channelized mode, each channel defined to operate over the serial port must have one time slot assigned to that logical channel that is the first time slot for that channel. In unchannelized mode, this bit must be set in the single time slot assigned.

GENERAL NOTE:

1. The timeslot map registers have no default values and may be active after reset, so they must be configured before activating the port.

6.6.6 RSIU Time Slot Pointer Allocation Register

There is one RSIU Time Slot Pointer Allocation Descriptor for each of CX28500's 32 serial ports. This register sets the start and end time slot address for the specific configured port. The difference between the configured end and start address specifies the number of time slots allocated for the specified serial port.

6.6.6.1 Time Slot Allocation Rules

1. If both pointers point to the same location, this port should be configured to operate in unchannelized mode. This is done by setting the RPORT_TYPE field in RSIU Port Configuration register to 0.
2. If there are two, three, or four time slots, the RPORT_TYPE field in RSIU Port Configuration register must be set to 2, 3, or 4 respectively.
3. If there are more than four time slots, the RPORT_TYPE field in TSIU Port Configuration register must be set to either 5, if it is not T1 framing, or 1 if it is.
4. If serial port is configured to channelized TSBUS mode, RSIU Time Slot Pointer Allocation Descriptor is configured to support more than eight time slots and the RPORT_TYPE bit field in RSIU Port Configuration register must be set to channelized TSBUS mode.

In the case of unchannelized mode (i.e., the RPORT_TYPE field in RSIU Port Configuration register is programmed to 0), CX28500 assumes that only one entry (the one pointed to by STARTAD) is used for this port. This frees the ENDAD pointer to point to any location in the RSIU time slot memory. The differences in these pointers now define the number of time slots to count for polling purposes as described in section Descriptor Polling.

Table 6-27 describes the bit fields in RSIU Time Slot Pointer Allocation Descriptor.

Table 6-27. RSIU Time Slot Pointer Allocation Register

Bit	Field Name	Value	Description
31:28	RSVD	0	Reserved.
27:16	RENDAD_TS[11:0]	—	Ending location in the Receive Time Slot Map of the last time slot assigned to this port.
15:12	RSVD	0	Reserved.
11:0	RSTARTAD_TS[11:0]	—	Starting location in the Receive Time Slot Map of the first time slot assigned to this port.

6.6.7 RSIU Port Configuration Register

There is a Receive Port Configuration register for each serial port. It defines how CX28500 interprets and synchronizes the received bit streams associated with the serial port.

Table 6.6.7 describes the bit fields in RSIU Port Configuration register.

Table 6-28. RSIU Port Configuration Register *(1 of 2)*

Bit	Field Name	Value	Description
31:14	RSVD	0	Reserved.
13	RXENBL[1]	0	Receive Port Disabled. Logically resets the serial port, regardless of RTS_ENABLE bit field in RSIU Time Slot Configuration Descriptor. This does not affect the bit values in any time slot descriptor. When the serial port becomes inactive, no data is transferred to the SLP.
		1	Receive Port Enabled. This bit field acts as a logical AND between RTS_ENABLE bit field in RSIU Time Slot Configuration Descriptor and time slot. Logically, if RTS_ENABLE bit field in RSIU Time Slot Configuration Descriptor is enabled, it allows all channels with time slot enable bits set to start processing data. This does not affect the bit values in any time slot descriptor.
12	RSVD	0	Reserved.
GENERAL NOTE: Writing RXENBL from 0 to 1 forces the port to realign itself to the incoming sync (if channelized) and generate RCOFA. In unchannelized port mode, the logical channel must be deactivated.			
11:9	RPORT_TYPE	0	Unchannelized mode. The user must configure time slot to contain one time slot.
		1	T1 mode. This mode includes some number of time slots and one bit of overhead coincident with frame or flywheel sync, and that one bit can be masked out.
		2	Nx64—2 TS. The user must configure the time slot map to contain two time slots.
		3	Nx64—3 TS. The user must configure the time slot map to contain three time slots.
		4	Nx64—4 TS mode. The user must configure the time slot map to contain four time slots.
		5	Nx64—The user must configure more than 4 time slots.
		6	Channelized TSBUS mode. The user must configure the time slot map to contain at least eight time slots.
		7	Reserved

Table 6-28. RSIU Port Configuration Register *(2 of 2)*

Bit	Field Name	Value	Description
8:6	RPOLLTH[2:0]	0	Poll Throttle. Poll at every frame synchronization event. This mechanism can be used to avoid bus saturation when multiple logical channels are active and idle and waiting for buffers to service.
		1	Poll at every 2nd frame synchronization event.
		2	Poll at every 4th frame synchronization event.
		3	Poll at every 8th frame synchronization event.
		4	Poll at every 16th frame synchronization event.
		5	Poll at every 64th frame synchronization event.
		6	Poll at every 128th frame synchronization event.
		7	Poll at every 256th frame synchronization event.
5	RSYNC_EDGE/ RSTUFF_EDGE	0	Receiver Frame Synchronization/ Receive Stuff Indication—Falling Edge. RSYNC/RSTUFF input sampled in on falling edge of RCLK.
		1	Receiver Frame Synchronization/ Receive Stuff Indication—Rising Edge. RSYNC/RSTUFF input sampled in on rising edge of RCLK.
4	RDAT_EDGE	0	Receiver Data—Falling Edge. RDAT input sampled in on falling edge of RCLK.
		1	Receiver Data—Rising Edge.
3	ROOF_EDGE/RTSTB_EDGE	0	Receiver Out Of Frame/ TSBUS Strobe—Falling Edge. ROOF/TSTB input sampled in on falling edge of RCLK.
		1	Receiver Out Of Frame/ TSBUS Strobe—Rising Edge. ROOF/TSTB input sampled in on rising edge of RCLK.
2	OOFABT	0	OOF Message Processing Enabled. When OOF condition is detected, continue processing incoming data. SIU should not report about the OOF.
		1	OOF Message Processing Disabled.
1	OOFIEN	0	Out Of Frame/Frame Recovery Interrupt Disabled.
		1	Out Of Frame/Frame Recovery Interrupt Enabled. If OOF/FREC is detected, generate Interrupt indicating OOF/FREC.
0	RCOFAIEN	0	Change Of Frame Alignment/Recovery from Change Of Frame Alignment Interrupts Disabled.
		1	Change Of Frame Alignment/Recovery from Change Of Frame Alignment Interrupts Enabled. If COFA is detected, generate Interrupt indicating COFA. When COFA is recovered from, generate a Recovery From COFA Interrupt indication.

FOOTNOTE:

(1) It is not allowed to change configuration of the port while the port is enabled. Therefore, it is not allowed to disable the port and change configuation in the same operation. It is also not allowed to change configuation and enable the port in the same operation. Both actions need to be split to 2 seperate operations. If the port is active one must first clear the RXENBL bit, and only then write new configuration. If the port is inactive, one must first write new configuration with the RXENBL bit cleared and then re-write the register with the same configuration but with the RXENBL bit set.

The bit fields OOFIEN and OOFABT are related to each other in the following manner:

1. If both are 0: the signal ROOF is ignored by CX28500's Rx side. This is the correct programming for TSBUS mode or when the ROOF pin is used as CTS or not connected at all.
2. If both are 1: this is the case of OOF functionality with OOF interrupt. CX28500 interrupts the Host and executes the appropriate behavior in the channels affected by the OOF. In addition, an OOF interrupt is generated and later an OOF recovery interrupt when the OOF condition ends.
3. If OOFABT is 1 and OOFIEN is 0: CX28500 as the case of both 1s except that no interrupt is generated.
4. If OOFIEN is 1 but OOFABT is 0: This is the case where the ROOF pin is used as a general purpose interrupt pin. CX28500 Rx blocks ignore the ROOF signal but an interrupt to the Host is generated when the ROOF transitions from high to low.

NOTE: For polling to work properly and efficiently, it is mandatory that the FIRST (receive time slot)/LAST (transmit time slot) indications be configured for each channel regardless of the operational mode (i.e., HDLC, transparent, etc.). The polling mechanism in the CX28500 uses these markings as triggering points. When configured correctly, CX28500 polls on a channel once at the selected poll throttle rate instead of polling at every time slot allocated to that channel, as in the case of hyperchannels. When a channel consists of only one time slot, as in a DSO channel, its corresponding receive time slot should have the FIRST_TS bit set. Likewise, its corresponding transmit time slot should have the LAST_TS bit set.

6.6.8 RSLP Maximum Message Length Register

The RSLP Maximum Message Length register, defined in Table 6-29, can have three separate values for maximum message length: MAXFRM1, MAXFRM2, and MAXFRM 3. Their structure is shown in the RSLP Channel Configuration register. The minimum message length is either 1, 3, or 5 depending on protocol mode: no FCS, 16-bit FCS, or 32-bit FCS, respectively. In the case of a short message, data is not transferred into shared memory and is discarded. In addition, an interrupt descriptor is generated toward the Host indicating the same error condition.

Each receive channel either selects one of these message length values or disables message length checking altogether.

The MAXSEL bit field (see Table 6-23, *RSLP Channel Configuration Register*) selects which (if any) register is used for received message length checking. If CX28500 receives a message exceeding the allowed maximum, the current message processing is discontinued and terminates further transfer of data to shared memory. In addition, a Receive Buffer Status Descriptor, corresponding to the partially received message, indicates a Long Message error condition, and an interrupt descriptor is generated toward the Host indicating the same error condition.

Table 6-29. Maximum Message Length Register

Bit	Field Name	Value	Description
31:14	RSVD	0	Reserved.
13:0	MAXFRM[13:0]	—	Defines a limit for the maximum number of bytes allowed in a received HDLC message. Valid values for the register range from 0 to 16 K-1. The formula to set MAXFRM is MAXFRM = **Max Allowed Message Length** (bytes) + FCS (bytes)–1. where FCS = 0 for Non-FCS Mode FCS= 2 byte for HDLC-16 Mode FCS = 4 byte for HDLC-32 Mode A Too Long Message interrupt is generated when the number of bytes in the processed message exceeds **Max Allowed Message Length**.
GENERAL NOTE: The Host may change the value of Maximum Message Length register only if the channel that uses its value (according to MAXSEL-bit in the configuration memory) is inactive.			

6.7 Transmit Path Registers

Transmit path registers contain the information necessary to configure the transmit direction. This configuration includes registers that are related to the DMA block, Host interface, registers that control the TSLP line processor, and TSIU.

6.7.1 TSLP Channel Status Register

The TSLP Channel Status register, defined in Table 6-30, is a Read Only (RO) register.

It provides information from the TSLP block regarding the channel state and status. There is one TSLP Channel status register for each of CX28500's channels (i.e., 1024 registers).

Table 6-30. TSLP Channel Status Register

Bit	Field Name	Host	Value	Description
31:4	RSVD	R	0	Reserved.
3:1	RSVD	R	Don't care	Reserved.
0	TACTIVE	R	0	Channel Inactive. The channel has been deactivated due to either a PCI Reset or Soft Chip Reset, or a Service Request Channel Deactivation or one of the following transmit errors: TxBUFF, TxCOFA.
			1	Channel Active.

6.7.2 TSLP Channel Configuration Register

The Transmit Channel Configuration register contains configuration bits applying to the logical channels within CX28500. There are 1024 such registers, one for each channel. The TSLP Channel Configuration Descriptor configures aspects of the channel common to all messages passing through the channel. One descriptor exists for each logical channel direction. Table 6-31 lists the values and descriptions of each channel configuration descriptor.

Table 6-31. TSLP Channel Configuration Register

Bit	Field Name	Value	Description
31:30	TPROTOCOL[1:0]	0	TRANSPARENT.
		1	HDLC with no FCS. Used in ISLP or full packet forwarding and/or channel monitoring application. For this mode, the SHT message detection is disabled. Any number of bytes can be transmitted and received within any single message including messages of only one byte.
		2	HDLC with FCS16 (FCS–2 bytes).
		3	HDLC with FCS32 (FCS–4 bytes).
29	TINV	0	Data Inversion disabled.
		1	Data Inversion enabled. Message is transferred to the SIU with polarity change (the inversion is done to all bits passed).
28:21	TMASK[7:0]	—	An 8-bit data mask. Each bit with the value of 1 should contain data when the relevant time slot is transmitted (e.g., Mask = 10000001, message data is transmitted only on bits 0 and 7, the other bits are discarded by the receiver).
20:3	RSVD	0	Reserved.
2	PADJ	0	Pad Count Adjustment disabled. No adjustment is made to the number of pad fill bytes if Zero Insertions is detected.
		1	Pad Count Adjustment enabled.
1	BUFFIEN	0	Underrun Interrupt disabled.
		1	Underrun Interrupt enabled.
0	EOMIEN	0	EOM Interrupt disabled.
		1	EOM Interrupt enabled. Interrupt generated upon End Of Message detection when transferring data from the TSLP to the TSIU.

6.7.3 TDMA Buffer Allocation Register

The Buffer Allocation register configures the internal receive memory. There is one TDMA Buffer Allocation register for each logical channel (i.e., 1024 channel).

CX28500's internal Tx memory is a 32-KB dual RAM, which can be split into 1024 parts, one part for each channel. The allocation granularity is two dword. For each active channel it is required to specify the following:

- The starting location of internal data buffer
- The ending location of internal buffer
- A threshold. This value is triggered when a request from DMA must be generated to the internal PCI arbiter. As soon as the channel's internal FIFO contains less data bytes than the threshold, an attempt to read data from shared memory is made. The threshold indicates and determines when the TxDMA begins transfer of data to the TxSLP. Hence, it indicates when a new transmission of a new message can start. If the buffer contains less data than the threshold, no transfer is generated unless there is already an EOM, meaning a whole message, in the internal FIFO.

The Host can issue a Service Request to change the value of this register only when the affected channel is inactive. Additionally, internal data buffer allocation must conform to the following criteria:

1. Host must set the buffers so there is no overlap between buffers belonging to different channels.
2. Allocated memory segments should not have wraparounds. That is, ending addresses must be greater than starting addresses.
3. The Host cannot allocate all of the FIFO to one channel. The maximal allocation to one channel is (all FIFO) – 1 quad dwords.

Each transmit channel must be allocated buffer space before the channel can be activated. Other important considerations for allocation of internal data buffers include the channel's data rate and PCI latency tolerance. This architecture of configured buffer allocation is completely flexible and allows the Host to assign larger FIFO buffers to channels that operate at higher rates. For applications operating high-speed channels (i.e., hyperchanneling) the Host can increase the FIFO allocation per channel. PCI latency tolerance equals the maximum length of time a particular channel can operate normally between PCI bus transactions without reaching an internal overflow or underflow condition. PCI latency tolerance is primarily dependent on each channel FIFO's buffer size.

Table 6-32 describes the bit fields in TDMA Buffer Allocation register. There are 1024 TDMA Buffer Allocation registers, one for each channel.

Table 6-32. TDMA Buffer Allocation Register

Bit	Field Name	Value	Description
31:29	RSVD	0	Reserved.
28:16	TDMA_ENDAD[12:0](1)	—	Ending address of internal channel data buffer.
15:13	RSVD	0	Reserved.
12:0	TDMA_STARTAD[12:0](2)	—	Starting address of internal channel data buffer.

GENERAL NOTE:

1. One channel maximum must be 32 KB-1 quad-words (i.e., ENDAD = 1FFEh and STARTAD = 0000h).
2. ENDAD must be greater than STARTAD (i.e., no rollover).
3. Minimal allocation to any channel ≥ 4 dwords (i.e., ENDAD – STARTAD ≥ 3).

FOOTNOTE:

(1) TDMA_ENDAD must be an odd address (i.e., LSB must be 1).

(2) TDMA_STARTAD must be an even address (i.e., LSB must be 0).

6.7.4 TDMA Configuration Register

This register, defined in Table 6-33, controls per channel the TxDMA operational mode. There are 1024 such registers, one for each channel.

Table 6-33. TDMA Channel Configuration Register

Bit	Field Name	Value	Description
31:17	RSVD	0	Reserved.
16	EOCEIEN	0	End Of Command Execution Interrupt disabled.
		1	End Of Command Execution Interrupt enabled. Interrupt generated when a command (Activate, Deactivate or Jump) execution was completed.
15	INHTBSD(1)	0	Inhibit Transmit Buffer Status Descriptor disabled. At the end of each Transmit Data Buffer, overwrite Tx Buffer Descriptor with Tx Buffer Status Descriptor.
		1	Inhibit Transmit Buffer Status Descriptor enabled. At the end of each Transmit Data Buffer, do not overwrite Tx Buffer Descriptor with Tx Buffer Status Descriptor.
14	ONRIEN	0	ONR Interrupt disabled.
		1	ONR Interrupt enabled. Interrupt generated when a new buffer descriptor is read from the Host memory, the ownership bit state is owned by the Host, and the NP bit field is 1, and the DMA is in mid-message.
13	AUTOENABLE	0	Automatic fetch feature disabled.
		1	Automatic fetch feature enabled. Attempts fetching of more Shared memory as soon as it has 32 free dwords in the channel buffer.
12:0	TDMA_TRSHOLD[12:0]	—	Threshold value. The value programmed into this field must not exceed the channel's buffer size. The threshold value indicates the following: 1. When the buffer contains less data than the threshold, a request to the TDMA to serve this channel is generated. An attempt to read more data from the Host memory is made. 2. When a transmission of a new message can start. When the buffer contains less data than the threshold, no new transmission starts unless there is a whole message already in the internal FIFO.

FOOTNOTE:

(1) The normal mode of operation of the CX28500 is to overwrite the Tx Buffer Descriptor with Tx Buffer Status Descriptor. This bit is used to inhibit this behavior, so that when it is set, the Tx Buffer Descriptor is not overwritten and the Host must rely on interrupts to find out when the CX28500 relinquishes ownership of the buffers.

6.7.5 TSIU Time Slot Configuration Register

6.7.5.1 Time Slot Map

The transmit time slot map consists of 4096 time slot descriptors. The entire transmit map contains configuration information for 4096 separate time slots. The CX28500's serial ports support 4096 time slots, which can be configured among the CX28500's 32 transmit serial ports by setting the TSIU Time Slot Pointer Assignment.

Numerous mappings of time slots are possible, and multiple time slots can be mapped to a single channel; however each time slot can map to only one channel at a time. For each serial port, two time slot maps are required: one for transmit functions and one for receive functions. The two separate maps are configured independently for transmit direction (TSIU Time Slot Configuration Descriptor and TSIU Time Slot Pointer Assignment) and receive direction (RSIU Time Slot Configuration Descriptor and RSIU Time Slot Pointer Assignment). These pointers are described in Table 6-34.

Table 6-35 specifies the content of each entry.

6.7.5.2 TSIU Time Slot Configuration Descriptor

There is an TSIU Time Slot Configuration Descriptor for each time slot (refer to Table 6-34).

There are 4096 entries in memory that set the translation between time slots and logical channels for each of CX28500's 32 ports. The actual mapping of these time slot descriptors to the 32 ports is done by 32 sets of pointer pairs (receive and transmit), one pair set for each port, which indicates the start and the end address of the memory location that belongs to the configured port. Time Slot pointer allocation is described in TSIU Time Slot Pointer Allocation.

The bit fields of TSIU Time Slot Configuration Descriptor include information:

- Time slot is enabled or disabled
- Time slot is a full DS0, or subchanneling enabled so that only a part of 64 Kbps transports information
- Indicates if it is the last time slot assigned to the logical channel
- Logical channel number (1024)

NOTE: For polling to work properly and efficiently, it is mandatory that the FIRST (receive time slot)/LAST (transmit time slot) indications be configured for each channel regardless of the operational mode (i.e., HDLC, transparent, etc.). The polling mechanism in the CX28500 uses these markings as triggering points. When configured correctly, CX28500 polls on a channel once at the selected poll throttle rate instead of polling at every time slot allocated to that channel, as in the case of hyperchannels. When a channel consists of only one time slot, as in a DSO channel, its corresponding receive time slot should have the FIRST_TS bit set. Likewise, its corresponding transmit time slot should have the LAST_TS bit set.

Table 6-34. TSIU Time Slot Configuration Descriptor

Bit	Field Name	Value	Description
31:13	RSVD	0	Reserved.
12:3	TCHANNEL [9:0]	—	Channel assigned to the time slot.
2	TTS_ENABLE	0	Time Slot Disabled.
		1	Time Slot Enabled.
1	TMASKEN_SB	0	Do not use the data mask for this time slot for this channel. This means all 8 bits are processed. (Subchanneling disabled)
		1	Allow using the data mask for this time slot for this channel. This means only the bits specified by the data mask are processed. (Subchanneling enabled)
0	TLAST_TS	0	Indicates that this is not the last time slot where this channel appears in this frame.
		1	Indicates that this is the last time slot where this channel appears in the frame. If a serial port is configured to operate in channelized mode, each channel defined to operate over the serial port must have one time slot assigned to that logical channel that is defined as the last time slot for that channel. In unchannelized mode, this bit must be set in the single time slot assigned.

GENERAL NOTE:

1. The timeslot map registers have no default values and may be active after reset, so they must be configured before activating the port.

6.7.6 TSIU Time Slot Pointer Assignment Register

There is one TSIU Time Slot Pointer Allocation Descriptor for each of CX28500's 32 serial ports. This register, defined in Table 6-35, sets the start and end time slot address for the specific configured port. The difference between the configured end and start address specifies the number of time slots allocated for the specified serial port.

6.7.6.1 Time Slot Allocation Rules

1. If both pointers point to the same location, this port should be configured to operate in unchannelized mode. This is done by setting the PROTTYP field in TSIU Port Configuration register to 0.
2. If there are two, three, or four time slots, the PORTTYP field in TSIU Port Configuration register must be set to 2, 3, or 4, respectively.
3. If there are more than four time slots, the PORTTYP field in RSIU Port Configuration register must be set to either 5, if it is not T1 framing, or 1 if it is.
4. If serial port is configured to channelized TSBUS mode, RSIU Time Slot Pointer Allocation Descriptor is configured to support more than eight time slots and the TPROT_TYPE bit field in RSIU Port Configuration register must be set to channelized TSBUS mode.

In the case of unchannelized mode (i.e., the PORTTYP field in TSIU Port Configuration register is programmed to 0), CX28500 assumes that only one entry (the one pointed to by STARTAD) is used for this port. This frees the ENDAD pointer to point to any location in the TSIU time slot memory. The differences in these pointers now define the number of time slots to count for polling purposes as described in section Descriptor Polling.

Table 6-35. TSIU Time Slot Pointers Register

Bit	Field Name	Value	Description
31:28	RSVD	0	Reserved.
27:16	TENDAD_TS[11:0]	—	Ending location in the Transmit Time Slot Map of the last time slot assigned to this port.
15:12	RSVD	0	Reserved.
11:0	TSTARTAD_TS[11:0]	—	Starting location in the Transmit Time Slot Map of the first time slot assigned to this port.
GENERAL NOTE: ENDAD must be ≥ STARTAD (meaning, no wraparound).			

6.7.7 TSIU Port Configuration Register

There is one TSIU Port Configuration register per port (see Table 6.7.7). This register defines how CX28500 generates and synchronizes the transmit bit streams associated with the port. There are 32 such registers, one for each port.

Table 6-36. TSIU Port Configuration Register *(1 of 2)*

Bit	Field Name	Value	Description
31:14	RSVD	0	Reserved.
13	TXENBL(1)	0	Transmit Port Disabled. Logically resets the time slot, regardless of TTS_ENABLE bit field in TSIU Time Slot Configuration Descriptor. This does not affect the bit values in any time slot descriptor.
		1	Transmit Port Enabled. This bit field acts as a logical AND between TTS_ENABLE bit field in TSIU Time Slot Configuration Descriptor and time slot. Logically, if TTS_ENABLE bit field in TSIU Time Slot Configuration Descriptor is enabled, it allows all channels with time slot enable bits set to start processing data. This does not affect the bit values in any time slot descriptor.
12	RSVD	0	Reserved.
11:9	TPORT_TYPE	0	Unchannelized mode. The user must configure time slot to contain one time slot.
		1	T1 mode. This mode implies 24 time slots and T1 signalling. The user must configure the time slot map to contain exactly 24 time slots.
		2	Nx64—2 TS. The user must configure the time slot map to contain two time slots.
		3	Nx64—3 TS. The user must configure the time slot map to contain three time slots.
		4	Nx64—4 TS mode. The user must configure the time slot map to contain four time slots.
		5	Nx64—The user must configure more than 4 time slots.
		6	Channelized TSBUS mode. The user must configure the time slot map to contain at least eight time slots.
		7	Reserved
8:6	TPOLLTH[2:0]	0	Poll Throttle. Poll at every frame synchronization event. This mechanism can be used to avoid bus saturation when multiple logical channels are active and idle and waiting for buffers to service.
		1	Poll at every 2nd frame synchronization event.
		2	Poll at every 4th frame synchronization event.
		3	Poll at every 8th frame synchronization event.
		4	Poll at every 16th frame synchronization event.
		5	Poll at every 64th frame synchronization event.
		6	Poll at every 128th frame synchronization event.
		7	Poll at every 256th frame synchronization event.
5	TSYNC_EDGE/ TSTUFF_EDGE	0	Transmitter Frame Synchronization/Receive Stuff Indication—Falling Edge. RSYNC/RSTUFF input sampled in on falling edge of RCLK.
		1	Transmitter Frame Synchronization/Receive Stuff Indication—Rising Edge. RSYNC/RSTUFF input sampled in on rising edge of RCLK.

Table 6-36. TSIU Port Configuration Register *(2 of 2)*

Bit	Field Name	Value	Description
4	TDAT_EDGE	0	Transmitter Data—Falling Edge. TDAT input sampled in on falling edge of RCLK.
		1	Transmitter Data—Rising Edge.
3	CTS_EDGE/ STB_EDGE	0	Transmitter CTS/TSBUS Strobe—falling edge. CTS/STB input is sampled on the falling edge of TCLK.
		1	Transitter CTS/TSBUS Strobe—rising edge. CTS/STB input is sampled on the rising edge of TCLK.
2	CTSENB	0	CTS disabled. This bit is ignored when the port is operating in TSBUS mode.
		1	CTS enabled. This bit is ignored when the port is operating in TSBUS mode.
1	TRITX	0	Transmit Three-state Disabled. When a channel port is enabled, but a time slot within the port is not mapped via the Time Slot Map, the transmitter outputs a logic 1 on the output data signal.
		1	Transmit Three-state Enabled. When a port is enabled, but a time slot within the port is not mapped via the Time Slot Map, the transmitter three-states the output data signal.
0	TCOFAIEN	0	Change of Frame Alignment/Recovery from Change Of Frame Alignment Interrupts Disabled.
		1	Change Of Frame Alignment/Recovery from Change Of Frame Alignment Interrupts Enabled. If COFA is detected, generates an interrupt indicating COFA. When COFA is recovered from, generates a Recovery From COFA Interrupt indication.

FOOTNOTE:

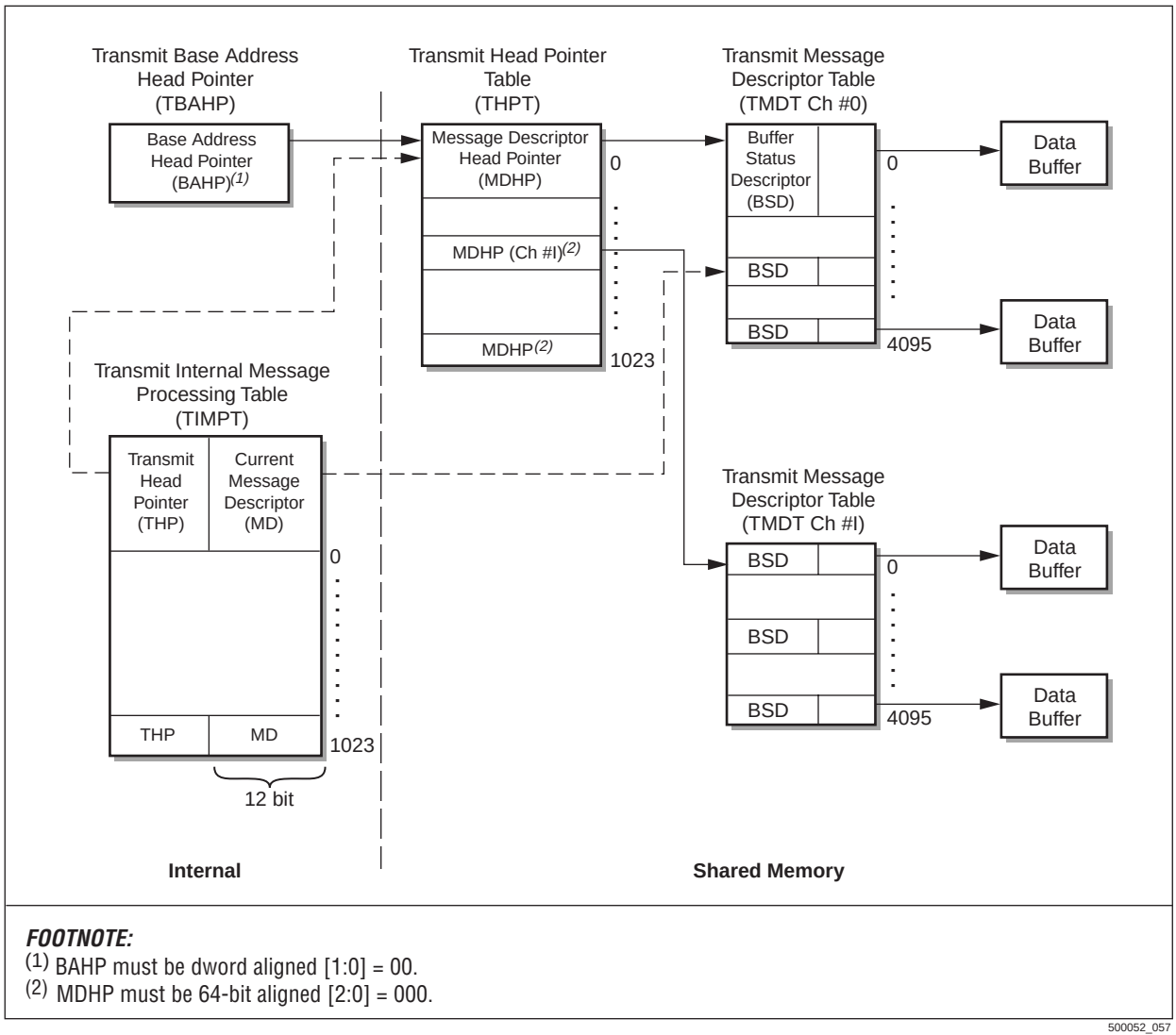
(1) It is not allowed to change configuration of the port while the port is enabled. Therefore, it is not allowed to disable the port and change configuation in the same operation. It is also not allowed to change configuation and enable the port in the same operation. Both actions need to be split to 2 seperate operations. If the port is active one must first clear the TXENBL bit, and only then write new configuration. If the port is inactive, one must first write new configuration with the TXENBL bit cleared and then re-write the register with the same configuration but with the TXENBL bit set.

NOTE: For polling to work properly and efficiently, it is mandatory that the FIRST (receive time slot)/LAST (transmit time slot) indications be configured for each channel regardless of the operational mode (i.e., HDLC, transparent, etc.). The polling mechanism in the CX28500 uses these markings as triggering points. When configured correctly, CX28500 polls on a channel once at the selected poll throttle rate instead of polling at every time slot allocated to that channel, as in the case of hyperchannels. When a channel consists of only one time slot, as in a DSO channel, its corresponding receive time slot should have the FIRST_TS bit set. Likewise, its corresponding transmit time slot should have the LAST_TS bit set.

6.8 Receive and Transmit Data Structures

Figures 6-2 and 6-3 illustrate the receive and transmit data structures used for data communication between CX28500 and the Host.

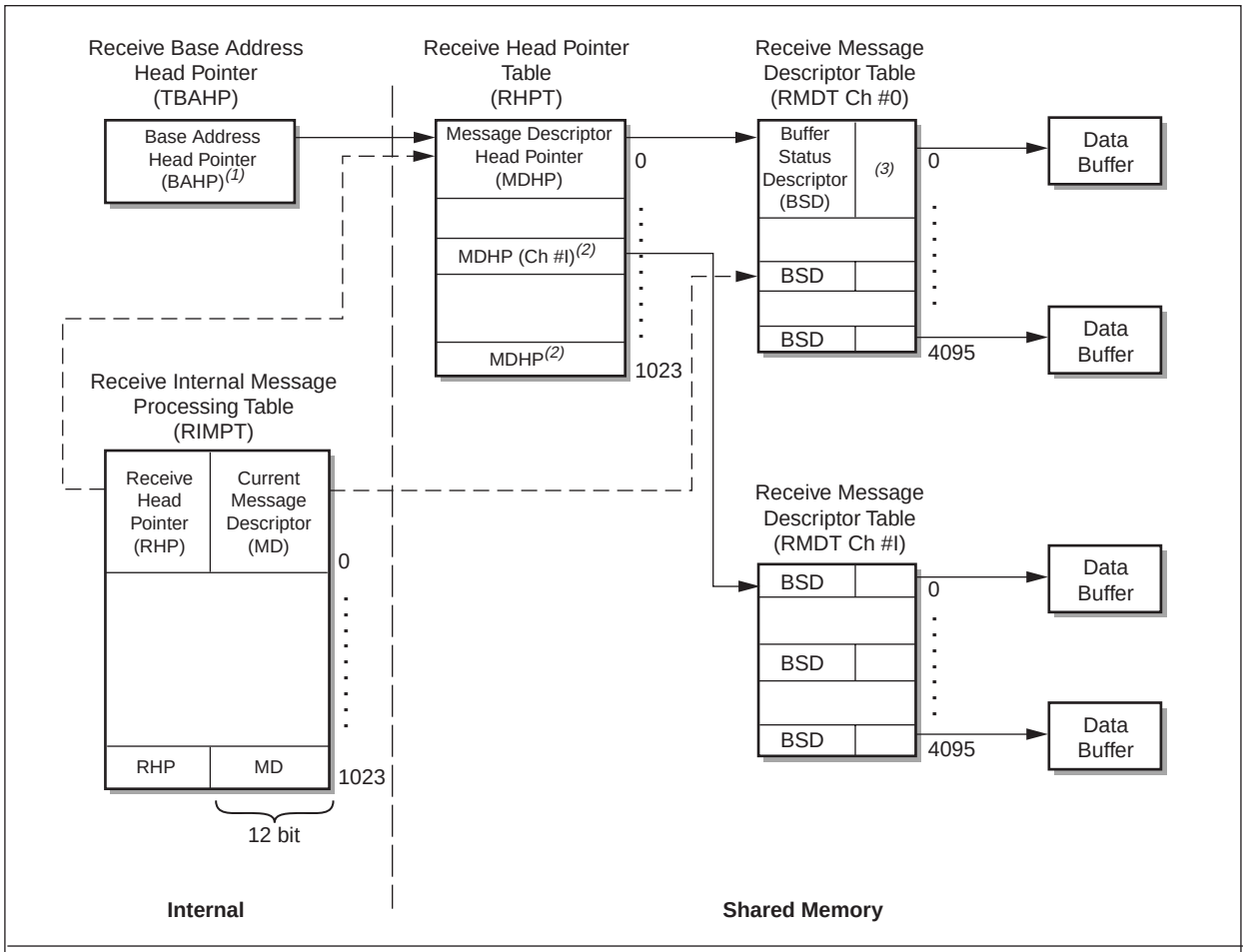
Figure 6-2. Transmit Data Structures



FOOTNOTE:
(1) BAHP must be dword aligned [1:0] = 00.
(2) MDHP must be 64-bit aligned [2:0] = 000.

500052_057

Figure 6-3. Receive Message Data Structures



FOOTNOTE:

(1) BAHP must be dword aligned [1:0] = 00.

(2) MDHP must be 64-bit aligned [2:0] = 000.

(3) For both Tx and Rx directions, ECCMODE = 1 requires 64-bit aligned data buffer pointers. Additionally, for the Rx direction, receive data buffer length must be an integer multiple of 8 bytes. For both Tx and Rx directions, ECCMODE = 0 allows byte-aligned data buffer pointers and data buffer length.

500052_058

6.8.1 Transmit Message Path

6.8.1.1 Shared Memory

Transmit Head Pointer Table (THPT)

This table is allocated and initialized by the Host. This table is an array of 1024 entries (one for each channel) which contain the message descriptor head pointer address for each logical channel configured in the application. Each message descriptor head pointer points to the starting address of the transmit message descriptor table associated with the respective channel.

Transmit Message Descriptor Table (TMDT)

For each channel, the Host allocates the TMDT table. This table is an array which contains up to 4096 entries. Each entry contains the buffer status descriptor, plus a pointer to the data buffer. Both TMDT and Data Buffer are allocated in shared memory and initialized by the Host during the driving initialization process.

After channel activation service request, the Buffer Status Descriptor (BSD) is updated by CX28500, after processing the current buffer.

Data Buffer

Location in Shared memory where data is stored.

6.8.2 Receive Message Path

6.8.2.1 Shared Memory

Receive Head Pointer Table (RHPT)

This table is allocated and initialized by the Host. This table is an array of 1024 entries (one for each channel) which contain the message descriptor head pointer address for each logical channel configured in the application. Each message descriptor head pointer points to the starting address of the receive message descriptor table associated with the respective channel.

Receive Message Descriptor Table (RMDT)

The Host allocates the RMDT table for each channel. This table is an array which contains up to 4096 entries. Each entry contains the buffer status descriptor, plus pointer to the data buffer. Both RMDT and Data Buffer are allocated in shared memory and initialized by the Host during the driving initialization process.

After channel activation service request, the Buffer Status Descriptor (BSD) is updated by CX28500, after processing the current message.

Data Buffer

Location in shared memory where data is stored.

6.8.3 Internal Memory

6.8.3.1 Transmit Path

Transmit Base Address Head Pointer (TBAHP) (CONFIG_WR)

This register contains the Base Address of the Transmit Head Pointer Table update during the CONFIG_WR SRQ.

Transmit Internal Message Processing Table (TIMPT)

For each channel's message descriptor (MD) table, CX28500 maintains a pointer to its first MD copied from the head pointer table and a pointer (actually the offset from the starting address of the current MD being processed). The length of Transmit Message Descriptor table is given by the fact that the MD is a 12-bit field in Transmit Internal Message Processing Table (TIMPT).

6.8.3.2 Receive Path

Receive Base Address Head Pointer (RBAHP) (CONFIG_WR)

This register contains the Base Address of the Receive Head Pointer Table update during the CONFIG_WR SRQ.

Receive Internal Message Processing Table (RIMPT)

For each message descriptor (MD) table, CX28500 maintains a pointer to its first MD copied from the head pointer table and a pointer (actually the offset from the starting address of the current MD being processed). The length of Receive Message Descriptor table is given by the fact that the MD is a 12-bit field in Receive Internal Message Processing Table (RIMPT).

6.8.4 Head Pointer Table and Its Content

There are two CX28500 internal registers that contain the address of the Transmit and Receive Head Pointer Table (i.e., THPT and RHPT).

These registers are updated with the values of THPT and RHPT during CONFIG_WR service request cycle. The content of TBAHP or RBAHP register is described in Table 6-37.

Table 6-37. Transmit and Receive Base Address Head Pointer (TBAHP and RBAHP) Content

Bit	Field Name	Value	Description
31:2	BAHPTBL[29:0]	—	These 30 bits are appended with 00b to form a dword-aligned 32-bit address. This address points to the first Head Pointer descriptor in the table of Head Pointers Descriptors in the Host Shared Memory.
1:0	BAHPTBL[1:0]	00	Ensures dword alignment

The content of each register (TBAHP or RBAHP) contains the starting address of the Transmit Head Pointer Table (THPT) or Receive Head Pointer Table (RHPT), respectively.

The content of each THPT or RHPT table contains the address of the Message Descriptor Head Pointer Table, for each independent channel (see Table 6-38).

Table 6-38. Transmit and Receive Head Pointer Table (THPT and RHPT) Content

Bit	Field Name	Value	Description
31:3	MDHP[28:0]	—	These 29 bits are appended with 000b to form a 64-bit aligned address. This address points to the first Message Descriptor in the table of message descriptors associated with a specific channel as illustrated in Figure 6-3, *Receive Message Data Structures*.
2:0	MDHP[2:0]	000	Ensures 64-bit alignment

6.8.5 Message Descriptor (MD)

A message descriptor defines one data buffer where all or part of a message is stored in shared memory.

By allocating a number of message descriptors to the message descriptor table, numerous data buffers can be linked together to support high-speed data links or large messages spread across a number of smaller data buffers.

Depending on the transmission and reception rate of individual channels, the numbers and sizes of message buffers can vary between channels and/or applications. For high-speed channels, more and larger buffers can be employed to provide ample data storage while the Host processes each message in the table of messages. For low-speed channels, fewer and smaller buffers can be employed, because the Host may be able to process each message faster, and the need to store messages is lessened.

Multiple smaller data buffers can store one large message. In utilizing multiple buffers, the importance of keeping the sequence of data buffers in order is obvious.

CX28500's operation allows for the following:

- Multiple message descriptor tables
- Multiple and variable size buffers within a message descriptor table
- Multiple buffers storing a single message
- Sequencing of individual data buffers for a multi-buffer message

A message descriptor is intentionally designed to be usable by both the transmit and receive functions in CX28500. In providing this symmetry, a mechanism known as self-servicing buffers is available. This mechanism allows the reuse of a single descriptor for both the transmit and receive portions of a channel, and is designed for diagnostics and loop-back capabilities. For details, see *Self-Servicing Buffers*.

There are up to 4096 entries in Transmit or Receive Message Descriptor Table (TMDT and RMDT) each of 2 dwords. Each entry in this table contains either a Buffer Descriptor (BD) which is written by the Host or a Buffer Status Descriptor (BSD) which is written by CX28500. The BD and BSD represent one dword from the content of TMDT or RMDT; the other dword contains the address of the data buffer (see Table 6-39).

Table 6-39. Transmit or Receive Message Descriptor Table (TMDT) or (RMDT) Content

Byte Offset	Field Name	dwords	Bytes
00h	Buffer Descriptor (Host Writes) Buffer Status Descriptor (CX28500 Writes)	1	4
04h	Data Pointer	1	4
Total		2	8

CX28500 checks certain data from a message descriptor before processing the associated data buffer. When a data buffer is completely processed (either transmitted or received), CX28500 overwrites the buffer descriptor field (the first dword in a message descriptor) with a buffer status descriptor, if this is allowed for the related channel. For details see Table 6-25, *RDMA Channel Configuration Register* and Table 6-33, *TDMA Channel Configuration Register*.

NOTE: When operating in ECC mode, if configured to write Buffer Status Descriptor, CX28500 writes both the status descriptor and its attached data buffer pointer. This is necessary in order to perform a 64-bit write transaction. The pointer value is the same value that was read by CX28500.

The buffer status descriptor specifies the number of bytes transferred, an End Of Message indicator, and a buffer ONR-bit indicator that assigns control of associated buffers back to the Host.

The ONR-bit mechanism transfers control of a data buffer between CX28500 and the Host. The message descriptor can be assigned before an associated data buffer is allocated in memory. In this case, CX28500 is instructed to poll the contents of the buffer descriptor until the Host grants ownership of a data buffer to CX28500. After CX28500 processes the data buffer, it grants the ownership back to the Host.

If the ONR-bit indicates that CX28500 does not own the next buffer and if NP = 1 (polling is not enabled) and in mid-message, then this is an error condition. CX28500 generates ONR interrupt and all the DMA handling is suspended for the channel. The Host must use the Host service mechanism (channel activation or channel jump commands) in order to continue DMA handling for the channel.

The ONR-bit mechanism prevents CX28500 from processing the same buffer twice without intervention from the Host.

Additionally, the Host can append additional information beyond the end of a data buffer as long as the longest message length can be fitted first into the data buffer. In the case of additional information, it is ignored.

For simplicity, the following discussion is made in reference to one channel. Each channel is serviced independently of another channel, and separate message descriptor tables are maintained for each supported channel.

Similarly, both transmit and receive sections of a channel service the descriptor tables identically, and separate message descriptor tables are maintained for each section. Also, the size of data fields in the descriptors is identical; however, the layout of fields between receive and transmit descriptors is different.

6.8.6 Buffer Descriptors

The Buffer Descriptor (BD) resides in the shared memory and is fetched by CX28500 each time a new data buffer is required. All Buffer Descriptors include the following fields:

- Owner Indicator Bit (ONR)
- No Poll/Poll Indicator (NP)
- End Of Buffer Interrupt Enable (EOBIEN)
- Last MD in the table (LAST)
- Buffer Length (BLEN)

The ONR bit is a generic term for any descriptor. In a Transmit Buffer Descriptor it is generically called CX28500-owned. In a Receive Buffer Descriptor it is generically called Host-owned. The names are different to indicate that the active sense of the ONR bit is different between transmit and receive functions.

In addition to the above list of fields, Transmit Buffer Descriptors include the following fields:

- End Of Message Indicator (EOM)
- Idle Code Selection (IC)
- Pad Count (PADCNT)

IC and PADCNT are valid only when EOM = 1.

Tables 6-40 and 6-41 list the transmit and receive buffer descriptors and definitions.

Table 6-40. Transmit Buffer Descriptor *(1 of 2)*

Bit	Field Name	Value	Description
31	ONR	0	HOST Owns Buffer. Channel is to remain in idle mode while polling this bit periodically (if NP = 0) until Host relinquishes control to CX28500 by setting ONR = 1.
		1	CX28500 Owns Buffer. Continue processing data buffer normally.
30	NP	0	Poll Enabled. If ONR = 0, Host-owned, CX28500 polls the message descriptor periodically while in idle mode until ONR = 1.
		1	Poll Disabled. If ONR = 0, then enter suspend mode and wait for a Channel Activate or Jump Host Service from Host.
29	EOM	0	End Of Message Indicator clear—this is not the last buffer for the current message.
		1	End Of Message Indicator set—this is the last buffer for the current message.
28	EOBIEN	0	End Of Buffer Interrupt Disabled. When the last data byte was taken from this buffer, an EOB interrupt is not generated.
		1	End Of Buffer Interrupt Enabled. When the last data byte was taken from this buffer, or the BSD is written, an EOB interrupt is generated.
27:26	IC[1:0](1)	0	Idle Code Select—If the protocol is HDLC, the Idle Code Select is 7Eh. If the protocol is transparent, the Idle Code Select is FFh.
		1	Idle Code Select—If the protocol is HDLC, the Idle Code Select is FFh. If the protocol selection is transparent, the Idle Code Select is 7Eh.
		2	Idle Code Select—00h.
		3	Reserved.

Table 6-40. Transmit Buffer Descriptor *(2 of 2)*

Bit	Field Name	Value	Description
25:18	PADCNT[7:0][(2)] RSVD [7:0]	—	Pad Count/Reserved. When operating in normal mode (i.e., not in Preserve Channel Mode) this field is treated as PADCOUNT. PADCNT indicates the minimum number of idle codes to be inserted between the closing flags and the next opening flag (7Eh). If PADCNT = 2 and IC = 1, for example, CX28500 outputs the bit pattern 7Eh..FFh..FFh..7Eh. There is no indication by CX28500 if more than PADCNT number of idle codes are inserted. When operating in Preserve Channel Mode, this field is treated as a reserved field whose bits are preserved in the Transmit Buffer Descriptor.
17:15	ABORT	0	Packet should be transmitted and end correctly.
		1-7	Packet transmission should end with ABORT sequence when EOM = 1.
14	TxLAST	0	This is not the last MD in this message descriptor table.
		1	This is the last MD in this message descriptor table.
13:0	BLEN[13:0][(3)]	—	Buffer Length. The number of bytes in data buffer to be transmitted. In general, this would equal the allocated buffer size. If EOM = 0 and BLEN = 0, the DMA ignores this BD and skips to the next one. If EOM = 1, BLEN = 0, ABORT = 0, and prior NOT-EOM data buffer, the DMA completes transmission of the prior buffer.

GENERAL NOTE:

1. ABORT is ignored if EOP = 0.
2. BLEN is ignored if ABORT = 0 and EOM = 1. Unhandled Exception: EOM = 1, BLEN = 0, ABORT = 0, and prior EOM buffer results in unpredictable behavior.

FOOTNOTE:

(1) IC field is processed and used by CX28500 only when EOM is set (i.e., only in the last buffer descriptor of a message).
(2) PADCNT field is processed and used by CX28500 only when EOM is set (i.e., only in the last buffer descriptor of a message).
(3) The combination of BLEN = 0 and EOM = 0 in the middle of a message is not allowed.

Table 6-41. Receive Buffer Descriptor *(1 of 2)*

Bit	Field Name	Value	Description
31	ONR	0	CX28500 Owns Buffer. Continue processing data buffer normally.
		1	HOST Owns Buffer. Channel is to remain in idle mode while polling this bit periodically (if NP = 0) until Host relinquishes control to CX28500 by setting ONR = 0.
30	NP	0	Poll Enabled. If ONR = 1, Host-owned, CX28500 polls message descriptor periodically until ONR = 0.
		1	Poll Disabled. If ONR = 1, then enter suspend mode and wait for a Channel Activate or Jump Host Service from Host.
29	RSVD	0	Reserved.
28	EOBIEN	0	End Of Buffer Interrupt Disabled. When the last data byte is put into this buffer, an EOB interrupt is generated.
		1	End Of Buffer Interrupt Enabled. When the last data byte is put into this buffer, an EOB interrupt is generated.
27:26	RSVD	0	Reserved.

Table 6-41. Receive Buffer Descriptor *(2 of 2)*

Bit	Field Name	Value	Description
25:18	RSVD/ CHAN [7.0]	0	When operating in normal mode (i.e., not Preserve Channel Mode as indicated by bit PCHMODE in Global Configuration Descriptor) this field is Reserved. When operating in Preserve Channel Mode, this field is used by CX28500 to transfer the least significant 8 bits of the channel number. The most significant bits are transferred to the data pointer.
17:15	RSVD	0	Reserved
14	RxLAST	0	This is not the last MD in this message descriptor table.
		1	This is the last MD in this message descriptor table.
13:0	BLEN[13:0]	—	Buffer Length. Actual number of received data bytes might be less than this. This number indicates how many will fit into the data buffer. If BLEN=0, the RDMA ignores this MD and skips to the next one.

6.8.7 Buffer Status Descriptor

The Buffer Status Descriptor (BSD) contains information regarding the data buffer processing. If BSD updates are allowed, the information is written over the Buffer Descriptor. The BSD includes the following bit fields:

- Owner Indicator Bit (ONR)
- End Of Message Indicator (EOM)
- Last MD in the table (LAST)
- Data Length (DLEN)

The Receive Buffer Status Descriptor contains an Error Status (ERROR) bit field.

The design of BSD and BD in the same location allows CX28500 to self-service the buffers without Host intervention. See the Self Servicing Mechanism chapter.

NOTE: While operating in ECC mode, if configured to update the Buffer Status Descriptor, the CX28500 writes both the BSD and the data buffer pointer. This is required in order to perform a 64-bit PCI write transaction.

Tables 6-42 and 6-43 list the transmit and receive buffer status descriptors and their descriptions.

Table 6-42. Transmit Buffer Status Descriptor

Bit	Field Name	Value	Description
31	ONR	0	HOST Owns Buffer. Channel is to remain in idle mode while polling this bit periodically (if NP = 0) until Host relinquishes control to CX28500 by setting ONR = 1.
		1	CX28500 Owns Buffer. Continue processing data buffer normally.
30	NP	0	Poll Enabled. If ONR = 0, Host-owned, CX28500 polls the message descriptor periodically while in idle mode until ONR = 1.
		1	Poll Disabled. If ONR = 0, then enter suspend mode and wait for a Channel Activate or Jump Host Service from Host.
29	EOM	0	End Of Message Indicator clear–this is not the last buffer for the current message.
		1	End Of Message Indicator set –this is the last buffer for the current message.
28	EOBIEN	0	End Of Buffer Interrupt Disabled. When the last data byte is taken from this buffer, an EOB interrupt is not generated.
		1	End Of Buffer Interrupt Enabled. When the last data byte is taken from this buffer, an EOB interrupt is generated.
27:26	IC[1:0]	0	Idle Code Select—If the protocol is HDLC, the Idle Code Select is 7Eh. If the protocol selection is transparent, the Idle Code Select is FFh.
		1	Idle Code Select—If the protocol is HDLC, the Idle Code Select is FFh. If the protocol selection is transparent, the Idle Code Select is 7Eh.
		2	Idle Code Select—00h.
		3	Reserved.
25:18	PADCNT[7:0] RSVD [7:0]	—	Pad Count/Reserved. When operating in normal mode (i.e., not in Preserve Channel Mode) this field is treated as PADCOUNT. PADCNT indicates the minimum number of idle codes to be inserted between the closing flags and the next opening flag (7Eh). If PADCNT = 2 and IC = 1, for example, CX28500 outputs the bit pattern 7Eh..FFh..FFh..7Eh. There is no indication by CX28500 if more than PADCNT number of idle codes are inserted. When operating in Preserve Channel Mode, this field is treated as a reserved field whose bits are preserved in the Transmit Buffer Status Descriptor.
17:15	ABORT	0	Packet should be transmitted and end correctly.
		1-7	Packet transmission should end with ABORT sequence.
14	TxLAST	0	This is NOT the last MD in this message descriptor table.
		1	This is the last MD in this message descriptor table.
13:0	BLEN[13:0]	—	Buffer Length. The number of bytes in data buffer to be transmitted. In general, this would equal the allocated buffer size. If EOM = 1 and BLEN = 0, CX28500 is requested to generate HDLC abort sequence. If EOM = 0 and BLEN = 0, the DMA ignores this BD and skips to the next one.

Table 6-43. Receive Buffer Status Descriptor

Bit	Field Name	Value	Description
31	ONR	0	CX28500 Owns Buffer. Until CX28500 relinquishes control, the data in this descriptor is being used by CX28500.
		1	Host Owns Buffer. CX28500 has relinquished control of buffer back to Host. CX28500 is done processing buffer.
30	NP		No Poll. Copied from Receive Buffer Descriptor.
29	EOM	0	End Of Message Indicator. The last byte for this message is not in this buffer.
		1	End Of Message indicator. The last byte for this data message is in this buffer either because a valid closing flag (7Eh) was detected or the receiver terminated due to an error condition.
28	EOBIEN		EOB Interrupt Enabled. Copied from Receive Buffer Descriptor.
27:26	RSVD	0	Reserved
25:18	RSVD/ CHAN [7.0]	0	When operating in normal mode (i.e., not in Preserve Channel Mode as indicated by bit PCHMODE in Global Configuration Descriptor) this field is Reserved. When operating in Preserve Channel Mode (i.e., bit PCHMODE in Global Configuration Descriptor is set0, this field is used by CX28500 to transfer in it the channel number least 8 significant bits.
17:15	ERROR[2:0]	0	OK: No errors detected in this receive buffer.
		1	BUFF: Buffer Error. Data is lost due to Internal data buffer overflow.
		2	COFA: Change Of Frame Alignment. RSYNC signal is misaligned with the flywheel in the serial interface.
		3	OOF: Out Of Frame. ROOF signal is asserted.
		4	ABT: Abort Flag Termination. Received message is terminated with abort sequence, seven sequential ones, instead of a closing flag (7Eh).
		5	LNG: Long Message. Message payload size greater than selected limit was received. Message processing is terminated, and transfer to shared memory is discontinued. Channel resumes scanning for HDLC flags or idle codes.
		6	ALIGN: Octet Alignment Error. Message payload size, after zero extraction, is not multiple of 8 bits. This error takes precedence over a FCS error.
		7	FCS: Frame Check Sequence Error. Received HDLC frame is terminated with proper 7Eh flag, but computed FCS does not match received FCS.
14	TxLAST	—	Last buffer indicator. Copied from Receive Buffer Descriptor.
13:0	DLEN[13:0]	—	Number of received bytes stored by CX28500 in this buffer.
GENERAL NOTE: In ECCMODE, Tx/Rx buffer descriptors are 64 bits in length. Therefore, both the BSD and DATAPTR are written after the buffer is completely processed.			

6.8.8 Data Buffer Pointer

The Data Buffer Pointer, defined in Table 6-44, is a 32-bit address to the first byte of a data message in shared memory.

NOTE: While CX28500 does support byte addresses for the data, when operating in ECC mode, all Rx and Tx buffers must be 64-bit aligned. CX28500 assumes the three least significant bits of the data pointers are 0 regardless of the actual value. However, when updating the buffer desriptor status, CX28500 preserves the actual value.

Table 6-44. Data Buffer Pointer

Bit	Field Name	Value	Description
31:3	DATAPTR[31:3]	—	This address points to the first byte of a data buffer.
2:0	Don't Care	—	

While operating in Preserve Channel Mode (i.e., bit PCHMODE in Global Configuration Descriptor is set), CX28500 transfers in DATAPTR[1:0] the two MSBs of the channel number and uses bit DATAPTR[2] to signal if this buffer is the first buffer of a received message (i.e., DATAPTR[2] is set) or not (i.e., DATAPTR[2] is clear). This is illustrated in Table 6-45.

Table 6-45. Data Buffer Pointer in PCH Mode

Bit	Field Name	Value	Description
31:3	DATAPTR[31:3]	—	Write back DATAPTR value.
2	SOM_STATUS	0	Start of message status: not start.
		1	Start of message status: start of message contained in current buffer.
1:0	CHAN_MSBs	—	Two most significant bits of the logical channel number CHAN[9:8].

7 Functional Description

7.1 Initialization

7.1.1 Reset

There is one level of reset:

1. Hard PCI Reset
2. Soft Chip Reset

There are two ways to assert a reset:

1. Assert the PCI reset signal pin, PRST*.
2. Assert a service request through the Host interface to perform the soft chip reset.

After reset, the Host needs to configure CX28500 for it to operate. This configuration includes several stages that should be performed in the following order:

- PCI Configuration—needs to be performed only after Hard PCI Reset
- Interrupt Queue Configuration
- Global Configuration
- Channels and Ports Configuration

NOTE: The Interrupt Queue needs to be configured before other registers. If the Interrupt Queue is not configured with the correct value of Shared Memory Interrupt Queue Pointer and Interrupt Queue Length, it may result in writes to location 0, since the Service Request Acknowledge (SACK) is written to a zero-address location.

7.1.1.1 Hard PCI Reset

The PCI reset is the most thorough level of reset in CX28500. All subsystems enter into their initial states, including the PCI interface. PCI reset is accomplished by asserting the PCI signal, PRST*.

The PRST* signal is an asynchronous signal on the PCI bus. The reset signal can be activated in several ways. The system must always assert the reset signal on power-up. Also, a Host bus to PCI bus bridging device should provide a way for software to assert the reset signal. Additionally, software-controlled circuitry can be included in the system design to specifically assert the reset signal on demand.

Asserting PRST* towards CX28500 guarantees that data transfer operations and PCI device operations do not commence until after CX28500 has been properly initialized for operation. Upon entering PCI reset state, CX28500 outputs a three-stated signal on all output pins and halts activity on all subsystems including the Host interface, serial interface, and expansion bus.

The effects of a PCI reset signal within CX28500 takes ten PCI clock cycles to complete after deasserting the reset signal. After this time, the Host can communicate with CX28500 using the PCI configuration cycles.

After the PCI configuration, the device is not ready to start communication with the Host via service request mechanism until the SRQ_LEN bit field in Service Request register is set to 0.

7.1.1.2 Soft Chip Reset

A soft chip reset is a device-wide reset without the Host interface's PCI state being reset. Serial interface operations and EBUS operations are halted. The soft chip reset state is entered in one of two ways:

- As a result of the PCI reset
- As a result of a soft chip reset Host Service request

A soft chip reset causes the following:

- Transmit data signals, TDAT, to be three-stated
- All EBUS address lines to be three-stated and read enable and write enable outputs to be deasserted, halting all memory operations on EBUS
- All active channels to enter the channel deactivated state
- DMA controllers to be reset, halting all PCI transactions
- All registers reset to default values

The Host acts as if this was a PCI reset, except that the PCI configuration does not need to be repeated (it is kept unchanged).

The Host can assume that the reset was completed by CX28500 and can start configuration of registers when the field SRQ_LEN is zero. After any kind of reset, the user must reconfigure all aspects of CX28500.

7.1.2 Configuration

A sequence of hierarchical initialization must occur after resets. The levels of hierarchy are as follows:

- PCI Configuration—only after hardware reset
- Interrupt Queue Configuration
- Global Configuration
- Channel and Port Configuration

Channel and port configuration involves programming many registers and must be done to comply with its own hierarchy, as explained below.

7.1.2.1 PCI Configuration

After power-up or a PCI reset sequence, CX28500 enters a holding pattern. It waits for PCI configuration cycles directed specifically for CX28500. They are actually directed at the PCI bus and PCI slot where CX28500 resides.

PCI configuration involves PCI read and write cycles. These cycles are initiated by the Host and performed by a Host-bus-to-PCI-bus bridge device. The cycles are executed at the hardware signal level by the bridge device. The bridge device polls all possible slots on the bus it controls for a PCI device and then iteratively reads the configuration space for all supported functions on each device. All information from the basic configuration sequence is forwarded to the system controller or Host processor controlling the bridge device.

During PCI configuration, the Host can perform the following configuration for CX28500:

- Read PCI configuration space (Device Identification, Vendor Identification, Class Code, and Revision Identification)
- Allocate 1 MB system memory range and assign the Base Address register using this memory range
- Allow fast back-to-back transactions
- Enable PCI system error signal line, SERR*
- Allow response for PCI parity error detection
- Allow PCI bus-master mode
- Allow PCI bus-slave mode
- Assign latency
- Assign interrupt line routing

7.1.2.2 Service Request Mechanism

After PCI configuration is complete, a set of hierarchical configuration sequences must be executed to begin operation at the channel level.

The Service Request mechanism is the main communication channel between CX28500 and the Host. It is used to configure CX28500's registers, read status registers, execute transactions over the EBUS, and activate ports and channels.

The mechanism is fully described in Section 6.2.1.

7.1.2.3 Global and EBUS Configuration

Global configuration is initiated by the Host issuing service requests. Global configuration specifies information used across the entire device including all ports, all channels, and the EBUS.

For more information, refer to:

- Section 6.4
- Section 6.5

NOTE: Device identification at the PCI Configuration Level must be used to identify the number of supported ports and channels in CX28500, which in turn affects CX28500's configuration.

7.1.2.4 Interrupt Queue Configuration

Part of the global configuration involves interrupt queue configuration. For more information, refer to Section 6.3.1.

7.1.2.5 Channel and Port Configuration

After the global configuration, a specific channel and port configuration must be performed for each supported channel and port.

- Table 6-23, *RSLP Channel Configuration Register*
- Table 6-24, *RDMA Buffer Allocation Register*
- Table 6-25, *RDMA Channel Configuration Register*
- Table 6-26, *RSIU Time Slot Configuration Descriptor*
- Table 6-27, *RSIU Time Slot Pointer Allocation Register*
- Table 6-28, *RSIU Port Configuration Register* (the Rx PortAlive register needs to be read and verified before writing to TSIU/RSIU Port Configuration register).
- Table 6-29, *Maximum Message Length Register*
- Table 6-31, *TSLP Channel Configuration Register*
- Table 6-32, *TDMA Buffer Allocation Register*
- Table 6-33, *TDMA Channel Configuration Register*
- Table 6-34, *TSIU Time Slot Configuration Descriptor*
- Table 6-35, *TSIU Time Slot Pointers Register*
- Table 6-36, *TSIU Port Configuration Register* (the Tx PortAlive register needs to be read and verified before writing to TSIU/RSIU Port Configuration register).

Notice that the order of configurations follows their internal addresses in CX28500. This enables the Host to configure all of them using a single write command

(CONFIG_WR) after setting the appropriate values in contiguous locations in its memory. For details see the Section 6.2.1.

Channel operations service request commands are:

- CH_ACT: Channel Activate
- CH_DEACT: Channel De-Activate
- CH_JMP: Channel Jump

7.1.2.6 Typical Initialization Procedure

This section depicts a typical initialization procedure.

1. PCI Configuration
2. PCI Reset or Soft Chip Reset (a Soft Chip Reset is performed by a direct write to the CX28500 register map—in the Soft Chip Reset register)

NOTE: After performing a Soft Chip Reset, it is not necessary to reconfigure the PCI.

3. Allocate areas in the share memory for:
   - Interrupt Queue
   - Service Request Table
   - CX28500 configuration registers (global and local per channel/port/TS basis)
4. Initialize values in allocated space for: XXXXXX
5. Loop and wait for the Service Request Length register to be ready. This step confirms that the CX28500 completed its internal initialization.
   a. Read the SRQ_LEN through the PCI slave access and check if it is 0.
   b. If true, go to the next step.
   c. Otherwise continue to check.
6. Initialize the Interrupt Queue Pointer register and Interrupt Length register by performing a direct write to the CX28500 registers with the address of the Interrupt Queue located in the shared memory and respectively its length.
7. Check the port alive availability (i.e., TxPortAlive and RxPortAlive) register by performing direct reads. For each active port the correspondent bit in TxPortAlive and RxPortAlive registers needs to be set to 1.
   a. While port is not alive (this is equivalent with the correspondent bit not set) wait 8–16 serial line clocks.
   b. If port is not alive, poll until port is alive.
   c. Otherwise go to the next step.

NOTE: If the port is not alive in 16 line clocks then there are no serial clocks applied specific port.

8. Initialize the Service Request Pointer (SRP) and Service Request Length (SRL) registers by performing a direct write to the CX28500's Service Request Pointer and Service Request Length register and update the value with the address all the SRP table and its length in shared memory.

9. Perform a CONFIG_WR Service Request and wait for the SACK which copies the content of the register in shared memory in CX28500 internal register. The Host can performe one CONFIG_WR Service request given that all the registers have been initialized in the shared memory prior to the CONFIG_WR Service Request or can perform CONFIG_WR Service Request for each register individually. A detail typical configuration write request procedure is [13.1]. Allocate the Service Request Table in the shared memory.

NOTE: This allocation can be done in the very beginning (see step 5 or in the configuration write request procedure) [13.2]. Initialize the content of the Service Request Table. [13.3]. Initialize the Service Request Pointer (SRP) with the address of Service Request Table by performing a direct write to the Service Request Pointer register. [13.3] Start the execution by writing the table length into to the Service Request Length register by performing a direct write. [13.4]. If other Service Request Table is required, the Host must poll the Service Request Length register by performing a direct read and check the SRQ_LEN field. If this field is not 0, then CX28500 did not complete the execution of the last Service Request Table. The number written in the SRQ_LEN indicates how many Configuration Write Commands (i.e., table entries) are pending for execution. While processing these commands, CX28500 generates SACK interrupt for each command in which the SACKIEN bit was set. When SRQ_LEN becomes 0, the Host can start from [13.1], whereas prior to a new execution either frees the memory which was allocated for the prior Service Request Table or uses the same memory as a pool memory. The registers which are initialized through Service Request Mechanism are as follows:

1. Global Configuration [1] (one per chip)
2. RSLP Channel Configuration [1024] (for each channel to be activated)
3. RDMA Buffer Allocation [1024] (for each channel to be activated)
4. RDMA Configuration [1024] (for each channel to be activated)
5. RSIU Time slot configuration [4096] (for each time slot to be used)
6. RSIU Time slot Pointer [32] (for each port to be activated)
7. RSIU Port Configuration [32] (for each port which should operate, this command activates the port)
8. RSLP Max. Message Length [3] (three registers)
9. Receive Base Address Head Pointer [1] (one per chip)
10. Transmit Base Address Head Pointer [1] (one per chip)
11. EBUS configuration [1] (one per chip)
12. TSLP Channel Configuration [1024] (for each channel to be activated)
13. TDMA Buffer Allocation [1024] (for each channel to be activated)
14. TDMA Configuration [1024] (for each channel to be activated)
15. TSIU Time slot Configuration [4096] (for each time slot to be used)
16. TSIU Time slot Pointer [32] (for each port to be activated)
17. TSIU Port Configuration [32](for each port which should operate, this command activates the port)

NOTE: The steps [1] to [17] can be performed in one single step by setting the Service Request Table with two entries and waiting for only one SACK or ECI. This increases the performance over the PCI bus.

18. Perform a CH_ACT Service Request and wait for SACK when the SACKIEN bit is set. For each channel that needs to be activated, the Host prepares a CH_ACT Service Request and inserts it into the Service Request Table. The Host may decide to activate all channels by writing the Service Request queries into one single Service Request table or by splitting the service request commands into one or more tables. For each CH_ACT Service Request the Host follows the same steps as were specified at [13.1] through [13.4]

7.2 Channel Operations

7.2.1 Channel Activation

Channel Activation is an asynchronous command from the Host interface to a transmit or receive section of a channel to jump to a new message. Message Descriptors in shared memory describe the attributes of the new message, what to do between messages, and identify the location of message data buffers in memory to use for transmit data or receive data.

After the previous levels of configuration are completed, individual channels are ready to be activated. Service requests are used to activate channels.

Each channel consists of a transmitter and a receiver section. Each section is independent of the other and maintains its own state machine, configuration registers, and internal resources. To activate both transmitter and receiver sections, two separate service requests are required, one directed to the transmitter and one to the receiver. CX28500 responds to each service request with the SACK Interrupt Descriptor, which notifies the Host that the task was initiated. The notification to the Host that the task was completed is an EOCE interrupt. This acknowledges the Host that the SRQ was completed. Upon channel activation, CX28500 flushes out all internal FIFO's, SLP's, and DMA's, then, updates their head pointer tables.

7.2.1.1 Transmit Channel Activation

The following describes what CX28500 does when the transmit channel is activated.

1. CX28500 reads Tx Head Pointer for channel from shared memory and stores it in the internal channel descriptor map, and generates an EOCE interrupt if it's enabled.
2. CX28500 reads Message Descriptor (buffer descriptor and data pointer) pointed to by the fetched Tx Head Pointer and stores it in internal channel descriptor memory. The actual read of the Message Descriptor depends on crossing a threshold in the internal transmit FIFO. In this case, however, because the activation command empties the internal transmit FIFO, which causes it to cross the FIFO threshold, it is guaranteed that the message descriptor is read.
3. CX28500 checks bit field OWNER and NP in Buffer Descriptor.
   If OWNER = 1, CX28500 is buffer owner. Go to step 4.
   If OWNER = 0, CX28500 is not the buffer owner and TDMA buffer processing for this channel is temporarily suspended. There are several ways of TDMA to exit the channel suspended state:
   a. Channel is instructed to jump to a new MD list (see CH-JUMP).
   b. Channel is instructed to reactivate to a new MD list, go to step 1.
   c. While NP is zero, TDMA polls current BD until OWNER = 0, go to step 4.
   d. If NP is 1, then TDMA does not poll the current BD, and this channel remains suspended until one of conditions a. through b. are satisfied. If CX28500 is in the middle of transmitting a message and the NP bit becomes 1, the TDMA generates a TxONR interrupt (if ONREIM is unmasked).
4. CX28500 reads DATA from the memory buffer to internal FIFO until either an End Of Message (EOM) is fetched or the entire memory buffer (BLEN) is read. This is not true for transmission. In the transmit direction, CX28500 reads all valid data buffers regardless of EOM.
5. CX28500 transmits the read data as an HDLC frame, or as a transparent frame depending on the type of protocol selected for the channel.
6. CX28500 checks INHTBSD bit field:
   If INHTBSD = 0, CX28500 overwrites the Buffer Descriptor (BD) with a Buffer Status Descriptor (BSD) file, do not write Buffer Status Descriptor (BSD).
   If the buffer contains an EOM, the TSLP posts a TxEOM interrupt (if EOMIEM unmasked).

NOTE: Because transmit EOB and EOM interrupts are independent events if the buffer contains an EOM, two interrupts are generated, an EOB interrupt when the lost data byte is read into internal memory, and an EOM interrupt when the last data byte is transferred from the internal memory to the Serial Interface Unit.

If the buffer is not EOM, CX28500 posts a TxEOB interrupt (if EOBIEN are masked).

7. CX28500 checks last bit field:
   - If Last = 0, CX28500 advances the current MD pointer.
   - If Last = 1, CX28500 sets current MD pointer equal to Head Pointer.
8. CX28500 reads the next message descriptor. Go to step 3.

7.2.1.2 Receive Channel Activation

The following describes what CX28500 does when receive channel is activated.

1. All internal FIFOs are flushed CX28500 reads Rx Head Pointer for channel from shared memory and stores it in internal channel descriptor map.
2. Simultaneously, the receiver is configured and data is sampled in from serial port using control lines. CX28500 does not pass the control to the Host memory (i.e., it does not move to step 3) until enough data is accumulated in its internal buffers (i.e., the threshold programmed for this channel gets crossed). If HDLC mode is selected, the RSIU will scan for an opening flag.
3. CX28500 reads Message Descriptor (Receive Buffer Descriptor and Data Pointer) pointed to by the fetched Rx Head Pointer and stores in internal channel descriptor memory.
4. CX28500 checks bit field OWNER and NP in Receive Buffer Descriptor. If OWNER = 0, CX28500 is the buffer owner, go to step 5. If OWNER = 1, CX28500 is not the buffer owner and RDMA buffer processing for this channel is temporarily suspended. There are several ways for RDMA to exit the channel suspend state:
   a. Channel is instructed to jump to a new MD list (see Channel Jump).
   b. Channel is instructed to reactivate to a new MD list, go to step 1.
   c. While NP is 0, RDMA polls current BD until OWNER = 0, go to step 5.
   d. If NP is 1, then RDMA does not poll the current BD, and this channel goes to suspended until one of conditions a through b are satisfied). If CX28500 is in the middle of receiving a message and the NP bit becomes 1, the RDMA generates a RxONR interrupt (if ONRIEN is unmasked).
5. CX28500 writes DATA from its FIFO to the memory buffer until either an End Of Message (EOM) is stored or the entire memory buffer (BLEN) is filled.
6. CX28500 checks INHTBSD bit field in Section 6.6.4. If INHTBSD = 0 (i.e., CX28500 is allowed to overwrite Buffer Descriptor with a Buffer Status Descriptor), it overwrites the Receive Buffer Descriptor. If the message ended (i.e., EOM is set in the Receive Buffer Status Descriptor), go to step 7. If the EOM is not set in Receive Buffer Status, it might generate an EOB interrupt depending on EOBIEN bit setting in the Receive Buffer Descriptor. Go to step 8.
7. If the Receive Buffer Descriptor contains the End Of Message (EOM), an interrupt descriptor is written in the shared memory depending on the mask interrupt status.
8. Depending on the LAST bit value set in the Receive Buffer Descriptor, CX28500 reads the next Message Descriptor. Go to step 4.

7.2.2 Channel Deactivation

A Channel Deactivation is an asynchronous command from the Host interface to a transmit or receive section of a channel to suspend processing and halt memory transfers into shared memory.

After the channel has been activated, channel deactivation via a service request suspends activity on an individual channel direction. Each channel consists of a transmitter and a receiver section. Each section is independent of the other and maintains its own state machine and configuration registers. To deactivate both transmitter and receiver sections, two separate service requests are required, one directed to the transmitter and one to the receiver. CX28500 may respond to each service request with the SACK Interrupt Descriptor, which notifies the Host that the task was initiated. The notification to the Host that the task was completed is an EOCE interrupt. This acknowledges the Host that the SRQ was completed.

7.2.2.1 Transmit Channel Deactivation

The following describes what CX28500 does when transmit channel is deactivated:

1. Current message processing is terminated destructively. That is, data can be lost and messages prematurely aborted. CX28500 does not give any indication of a lost message. If a Message Descriptor is in the middle of processing, it is left as it is.
2. Internal FIFOs are flushed and the data is lost.
3. The TSLP is responsible for handling outbound bits when the serial port is asynchronously disabled. The data output pin, TDAT, is held at logic 1. Data transfers from shared memory are halted.
4. The transmit channel remains in the suspended state until the channel is activated. The current channel direction configuration is maintained.

7.2.2.2 Receive Channel Deactivation

The following describes what CX28500 does when receive channel is deactivated:

1. Current message processing is terminated destructively. That is, data can be lost and messages prematurely aborted. CX28500 gives no indication of the lost messages. If a Message Descriptor is in the middle of processing, it is left as it is.
2. Internal FIFOs are flushed, and all data is lost.
3. The RSLP is responsible for handling inbound bits when the serial port is asynchronously disabled. Data transfers to shared memory are halted.
4. The receive channel remains in the suspended state until the channel is activated. The current channel direction configuration is maintained.

7.2.3 Channel Jump

A jump request is issued by the Host via a service request (CH_JMP). Notice that the jump service request acknowledge (via the SACK interrupt descriptor) is output toward the Host immediately. The EOCE Interrupt acknowledges that the CH_JMP service request was performed, and the next Head Pointer was read. However, this does not mean that CX28500 starts working on the new message descriptors table pointed by the new Head Pointer immediately. Channel jump takes effect after an End Of Message (EOM) is fetched from or stored to memory buffer or while the channel is in suspended state.

7.2.3.1 Receive Channel Jump

For a receiver, channel jumps do not affect the current message being transferred to the Host (if there is such a message). The command is executed only after the current message transfer to the Host is completed. No internal FIFO flushing happens; the next messages are transferred using the new message descriptors provided by the Jump command. The channel state is not reset as in the channel activate sequence. For a receiver, channel jumps provide a non-destructive way of storing incoming data using a new message descriptors table.

7.2.3.2 Transmit Channel Jump

For a transmitter, channel jumps are non-destructive to currently serviced messages. The channel state is not reset as in the channel activate sequence. Hence, a transmitter channel must be activated first, then subsequent jump requests can be made using the channel jump service request. For a transmitter, channel jumps provide a nondestructive way to start transmitting the new message list. CX28500 waits until the completion of the current message before jumping to the new message array pointed to by a new Head Pointer.

7.2.4 Channel Reactivation

A channel reactivation happens when an activate command is issued to an active channel. CX28500 behavior is as if the channel was deactivated and then activated again. This means that the sequence described in Section 7.2.2 happens followed by the sequence described in Section 7.2.1. However, only one SACK interrupt and one EOCE interrupt are generated. SACK is generated to indicate the initiation of the operation, and EOCE is generated to indicate the completion of channel reactivation.

7.2.5 Unmapped Time Slots

The Host can stop CX28500 from processing certain time slots regardless of the channel activation/deactivation/jump/reactivation commands. This can be performed by programming time slots in RSIU Time Slot Configuration and TSIU Time Slot Configuration to indicate that the specific time slots are not mapped (see *RTS_ENABLE* and *TTS_ENABLE bit fields in Table 6-26, RSIU Time Slot Configuration Descriptor* and *Table 6-34, TSIU Time Slot Configuration Descriptor*, respectively).

NOTE: The TDAT signal is either set to logic 1 or three-state according to bit TRITx in Section 6.7.5.

8 Basic Operations

The two main channel protocols, HDLC and transparent mode, are described in subsequent sections of this chapter. HDLC and transparent mode operations perform protocol-specific processing of their respective input and output serial bit streams and behave differently in their treatment of those bit streams during abnormal conditions.

8.1 Protocol-Independent Operations

From a functional viewpoint, many CX28500 operations are protocol-independent, though some behaviors may differ between the transmitter and receiver. The protocol-independent operations described below apply to all event and error handling:

- During DMA shared memory, SLP channel protocol and SIU serial port operations, an event or error may occur that indicates the status of the message transfer process or that affects the outcome of the overall message transfer process. Unless masked, all such events and errors cause CX28500 to write an Interrupt Descriptor to the shared memory interrupt queue. Interrupt Descriptors identify the error or event condition, the transmit or receive direction, and the affected channel or port number.
- If CX28500 suspends a channel's operation, the Host must perform a channel reactivation by issuing either a Channel Activation Service Request or a Channel Jump Service Request. This is referred to as "requiring reactivation." On the receiving side one scenario that would suspend a channel is when a message descriptor is Host owned and NP = 1 is encountered. On the transmission side, several occurrences of COFA's will cause the TSLP to stop transmission, hence suspending all active channels.
- If CX28500 deactivates a channel, the Host must perform a channel reactivation by issuing a Channel Activation Service Request. This is referred to as "requiring complete reactivation."
- The bit fields INHTBSD and INHRBSD in the RDMA/TDMA Channel Configuration registers specify whether CX28500 writes a Buffer Status Descriptor into the Message Descriptor to indicate that CX28500 has completed servicing both the message descriptor and its associated data buffer.
- During the normal course of shared memory buffer processing, the DMA calculates the position of the next Message Descriptor within the Message Descriptor Table and reads that Buffer Descriptor to determine ownership. The Buffer Descriptor's ONR bit field indicates whether CX28500 or the Host owns that particular message descriptor and its associated data buffer. CX28500 never writes to a Host-owned descriptor nor processes its associated data buffer.
- Whenever both EOB and EOM events happen together, an EOM interrupt is generated and an EOB interrupt is not generated for the receive direction. That is, unless the EOM interrupt is disabled (masked) and the EOB interrupt is enabled (unmasked), in which case an EOB interrupt is generated. This is true even if the cause of the EOM interrupt was due to an error condition. For the transmit direction, since EOB and EOM are independent, both interrupts will be reported if they are enabled.

8.1.1 Transmit

CX28500 initiates data transfer to the serial interface only if the following conditions are true:

- TxENBL bit set to 1 in Table 6-36, *TSIU Port Configuration Register*.
- Transmit channel is mapped to time slots, which are enabled in the port's Section 6.7.5.
- Transmit channel has been activated by a Host service request.
- If not in unchannelized mode, then CX28500 waits until the first detection of a sync pulse or strobe to get out of three-state.

If TxENBL bit is set to 0 (transmit port disabled), the serial data output signal is placed in high-impedance three-state. If TxENBL = 1 (port enabled) and a time slot is disabled, the corresponding time slot's transmitter output is either a three-state or all 1s depending on the state of the TRITx bit field in the port's Table 6-36, *TSIU Port Configuration Register*.

NOTE: If TxENBL = 1 and the port is configured in any channelized mode (i.e., not unchannelized), *until the first TSYNC/STB pulse is detected*, that port outputs either a three-state signal or all 1s depending on the state of the TRITx bit field.

8.1.2 Receive

The receiver processes data from the serial interface only if all of the following conditions are true:

- RxENBL bit is set to 1 in the port's Table 6-28, *RSIU Port Configuration Register*.
- Receive channel is mapped to time slot(s), that are enabled in the port's Section 6.6.5.
- Receive channel has been activated by a Host via service request.

If any one of the above conditions is not true, the receiver ignores the incoming data stream.

Data transfer consists of CX28500 first seeking the (next) message descriptor from the Message Descriptor in shared memory for each active channel. The buffer descriptor in each message descriptor, plus the protocol mode set for the channel, dictates the treatment of the incoming bit stream.

8.2 HDLC Mode

CX28500 supports three HDLC modes. The modes are assigned on a per-channel and direction basis by setting the PROTOCOL bit field within the RSLP/TSLP Channel Configuration registers. The HDLC modes are as follows:

- HDLC_NOCRC: HDLC support, no CRC
- HDLC-16CRC: HDLC support, 16-bit CRC
- HDLC-32CRC: HDLC support, 32-bit CRC

HDLC protocol-specific support in the transmitter includes the following:

- Generate opening/closing/shared flags
- Zero bit insertion after five consecutive 1s are transmitted
- Generate pad fill between frames and adjust for zero insertions
- Generate 0-, 16- or 32-bit CRC (i.e., FCS)
- Generate abort sequences upon FIFO underflow condition or as instructed on a per-message basis by the ABORT field in the message descriptor
- Data polarity inversion of all bits (including flags and padfill characters)

HDLC protocol-specific support in the receiver includes the following:

- Detection and extraction of opening/closing/shared flags
- Detection of shared-0 between successive flags
- Zero bit extraction after five consecutive 1s are received
- Detect changes in pad fill idle codes
- Check and extract 0-, 16- or 32-bit FCS
- Check frame length
- Check for octet alignment. Failures result in EOM and an error code in the buffer/interrupt descriptor
- Check for abort sequence reception
- After channel activation, check for the first flag character to be received and generate a CHIC interrupt

The Transmit Buffer Descriptor specifies inter-message bit-level operations. Specifically, when the EOM bit field is set to 1 within a Message Descriptor by the Host, it signifies that the descriptor represents the last buffer for the current message being transmitted and the bit fields IC and PADCNT take effect. These bits are described in Section 8.2.5.

Additionally, the NP bit field in both Receive and Transmit Buffer Descriptors selects whether CX28500 polls a Host-owned message descriptor.

8.2.1 Frame Check Sequence

CX28500 is configurable to calculate and insert either a 16- or 32-bit Frame Check Sequence (FCS) for HDLC packets, provided the packet length contains a minimum of 2 octets. The FCS is always calculated over the entire packet length.

For all HDLC modes that require FCS calculation, the polynomials used to calculate FCS are according to ITU-T Q.921 and ISO 3309-1984.

- CRC-16

$$x^{16} + x^{12} + x^5 + 1$$

- CRC-32

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

8.2.2 Opening/Closing Flags

For HDLC modes only, CX28500 supports the use of opening and closing message flags. The 7Eh (01111110b) flag is the opening and closing flag. An HDLC message is always bounded by this flag at the beginning and the end of the message.

CX28500 supports receiving a shared flag where the closing flag of one message can act as the opening of the next message. CX28500 also supports receiving a shared-zero bit between two flags—that is, the last zero bit of one flag is used as the first zero bit of the next flag. Receiving a shared zero between the FCS and the closing flag is not supported.

CX28500 can be configured to transmit a shared flag between successive messages by configuring the bit field PADCNT in each transmit buffer descriptor. CX28500 does not transmit shared-zero bits between successive flags.

8.2.3 Abort Codes

At least seven consecutive 1s constitute an abort code. Receiving the abort code causes the current frame processing to be aborted, and further data transfer into shared memory for that message is terminated. After detecting the abort code, CX28500 enters a scan mode, which searches for a new opening flag character.

Notification of this detected condition is provided by the receive buffer status descriptor and/or an interrupt descriptor indicating the error condition Abort Flag Termination.

In cases where received idle codes transition to an abort code, an interrupt descriptor is generated toward the Host (if enabled in Section 6.6.2), indicating the informational event Change To Abort Code. All received abort codes are discarded.

NOTE: Seven ones are the abort condition CX28500 checks for while receiving a message, but the criteria for detection and generation of a Change To Abort Code interrupt is equal to 14 consecutive ones.

In the transmission direction, an ABORT code's handling is depended on what the user tells CX28500 to do. An ABORT code could just abort the transmission of the current message, or it can mean aborting the current message transmission and then starting transmitting the next message, if it exists.

8.2.4 Zero-Bit Insertion/Deletion

CX28500 provides zero-bit insertion and deletion when it encounters five consecutive 1s within a frame. In the receiver, the zero bit is removed (discarded). In the transmitter, the zero bit is inserted after each sequence of five ones. Zero-bit insertion, or bit-stuffing, is only done in the message section and not in the pad fill section.

8.2.5 Message Configuration Bits–HDLC Mode

The Transmit Buffer Descriptor contains message configuration bits to specify what data pattern is transmitted after the end of a current message and its respective closing flag have been transmitted. The bits are specified as follows:

- Idle Code specification, IC
- Inter-message Pad Fill Count, PADCNT
- Send an Abort Sequence, ABORT

NOTE: Message configuration bits are also used in Transparent mode with slightly different meanings. For details see Section 8.3.1.

8.2.5.1 Idle Code

The Idle Code (IC) specification allows one of a set of idle codes to be chosen to be transmitted after the current message in case the next message is not available to be transmitted or inter-message pad fill is requested via PADCNT. The default Idle Code for HDLC mode is flag, or 7Eh.

8.2.5.2 Inter-Message Pad Fill

The Pad Count (PADCNT) specification allows pad fill octets (a sequence of one or more specified idle codes) to be transmitted between messages. PADCNT is the minimum number of fill octets to be transmitted between closing flag of one message and the opening flag of the next message in the following manner:

1. PADCNT = 0: Shared open/close flag
2. PADCNT = 1: Separate open/close flags, no idle code
3. PADCNT = 2: Separate open/close flags, at least one idle code

This pad fill feature enables rate adaptation applications (i.e., ISDN Rate Adaptation), as defined in ITU Standards V.110 and V.120. Furthermore, CX28500 also allows the reduction of the number of pad fill octets to compensate for the number of zero-bit insertion. One pad fill octet is omitted from transmission for every eight zero-bit insertions. The number of zero-bit insertions is rounded up to the nearest number of octets for pad fill adjustment. Pad Count Adjustment can be enabled in the TLSP Channel Configuration Register. For details, please see Section 6.7.2

8.2.5.3 Ending a Message with an Abort or Sending an Abort Sequence

If BLEN = 0 in any transmit buffer descriptor, CX28500 interprets it as a request to end an in-progress message with the abort sequence. If the previous buffer descriptor (before the BLEN = 0 buffer) contained an End-Of-Message (EOM) indication, the abort request is simply ignored and CX28500 moves on to the next message descriptor. If the previous buffer descriptor was not EOM (i.e., a transmit message was in-progress), an abort code sequence is transmitted to end that partially sent message. Transmission of an abort code sequence is defined as 16 consecutive 1s.

8.2.6 Transmit Events

Transmit events are informational in nature and do not require channel recovery actions.

8.2.6.1 End Of Buffer [EOB]

Reason:

- TDMA reached the end of a shared memory buffer by servicing the number of bytes equal to BLEN in the Transmit Buffer Descriptor. Note that transmit EOB and transmit EOM events are not coincident in time, so they result in two separate events being generated.

Effects:

- TxEOB interrupt (if EOBIEN = 1 in Transmit Buffer Descriptor).
- CX28500 continues with normal message processing. If the TDMA does not get more data from shared memory before the TSLP needs to output the next data bit (assuming the message did not end), the TSLP enters underflow state, described below.

8.2.6.2 End Of Message [EOM]

Reason:

- TSLP has transmitted (actually, transferred to the TSIU) the last bit of a data buffer (excluding the FCS and closing flag) and the Transmit Buffer Descriptor signifies that buffer contained an end of a message (EOM = 1).

NOTE: EOB and EOM are not coincident, resulting in separate events.

Effects:

- TxEOM interrupt (if EOMIEN = 1 in Section 6.7.2).
- TSLP and TDMA continue normal processing. If the TDMA does not get more data from shared memory before the TSLP needs to output the next data bit, TSLP outputs another octet of flag or idle code.

8.2.6.3 ABORT Termination

Reasons:

- The user wants to abort the transmission of the current message. For details, please see Section 6.8.6.
- To provide a mechanism for the self-served transmitter to recover from a receiver error. For details, please see Chapter 9.

Effects:

- CX28500 stops transmitting the current message, sends out an ABORT sequence, and goes on to process the next message, if it exists.

8.2.7 Receive Events

Receive events are informational in nature and do not require channel recovery actions.

8.2.7.1 End Of Buffer [EOB]

Reason:

- One message is split across multiple shared memory buffers. CX28500 reached the end of a buffer by servicing (writing) the number of bytes equal to BLEN specified in the Receive Buffer Descriptor.

Effects:

- EOB Interrupt (if EOBIEN = 1 in Receive Buffer Descriptor).
- RDMA and RSLP continue normal processing.

8.2.7.2 End Of Message (EOM)

Reason:

- RSLP has detected the end of a message (closing flag or an error condition). Error conditions include: Overflow, COFA, OOF, Abort, Too Long, Alignment, and FCS error.

Effects:

- If there were no errors, RxEOM interrupt (if EOMIEN = 1 in Section 6.6.5). If there were errors, RxEOM interrupt (if ERRIEN = 1 in Section 6.6.4 and Section 6.7.4)
- RDMA sets EOM = 1 in Receive Buffer Status Descriptor (if INHRBSD = 0 in Section 6.6.4 and Section 6.7.4).
- RDMA and RSLP continue normal processing.

8.2.7.3 Change to Abort Code (CHABT)

Reason:

- RSLP detected received data changed from flag (7Eh) octets to abort code (zero followed by fourteen consecutive ones) in the idle code section of the bit stream and not in the message payload section. An abort code detection in the message payload results in an ABORT Termination Error.

Effects:

- CHABT Interrupt (if IDLEIEN = 1 in Section 6.6.2).
- RSLP and RDMA continue normal processing.

8.2.7.4 Change to Idle Code (CHIC)

Reason:

- RSLP detects received data changed to flag (7Eh) octets. CX28500 requires detection of three consecutive flags and the previous idle code is all 1s before a CHIC event is generated.

Effects:

- CHIC interrupt (if IDLEIEN = 1 in Section 6.6.2).
- RSLP and RDMA continue normal processing.

NOTE: After channel activation/reactivation, the first flags detected on the line generate a CHIC interrupt.

8.2.7.5 Frame Recovery (FREC) or Generic Serial PORT (SPORT) Interrupt

Reason:

- RSIU detects the serial interface ROOF signal transition from an out-of-frame (ROOF = 1) to an in-frame (ROOF = 0) condition. If the ROOF signal is programmed for use as an out-of-frame indicator, this frame recovery event (ROOF returning low) generates a FREC interrupt after one full frame without ROOF assertion. If the ROOF signal is used as a general-purpose interrupt input, this event generates a SPORT (Serial PORT) interrupt. Since both FREC and SPORT interrupt events are reported in the same field in the non-DMA interrupt descriptor, the user must interpret the event according to how the ROOF signal is used.

Effects:

- FREC/SPORT Interrupt (if OOFIEN = 1 in Table 6-28, *RSIU Port Configuration Register*).
- RSLP and RDMA continue normal processing.

8.2.7.6 Receive COFA Recovery (RCREC)

Reason:

- RSIU terminates the internal COFA condition due to the arrival of a RSYNC/TSTB pulse followed by at least the assigned number of time slots for this port without another unexpected RSYNC/TSTB pulse.

Effects:

- RCREC Interrupt (if COFAIEN = 1 in Table 6-28, *RSIU Port Configuration Register*).
- RSLP and RDMA continue normal processing.

8.2.8 Transmit Errors

Transmit errors are service-affecting and require a corrective action by a controlling device (i.e., the Host) to resume normal channel processing.

8.2.8.1 Transmit Underrun [BUFF]

CX28500 needs to send more data toward the TSIU for an in-progress transmit message, but the internal channel FIFO is empty.

Reasons:

- Degradation of the Host subsystem or application software.
- Buffer descriptor containing the continuation of a message is Host-owned.
- PCI bus congestion.

Effects:

- TxBUFF Interrupt (if BUFFIEN = 1 in Section 6.7.2).
- Transmit channel enters deactivate state where the TSLP transmits a repetitive abort sequence of 16 consecutive 1s.
- TDMA may read more buffers to refill the internal channel FIFO (since this process is asynchronous to the TSLP), but eventually the TDMA stops servicing this channel since the TSLP has stopped sending data.
- Transmit output is all 1s, or an ABORT code, to indicate a termination event.

Channel Level Recovery Actions:

- Transmit channel complete reactivation is required.

NOTE: Since CX28500 completely separates the two processes of loading data from the Host and transmitting data to the serial interface, it is irrelevant to CX28500 how the underrun condition was created. To be specific, there is no distinction between a BUFF error created due to a Host-owned buffer descriptor or due to a latency-induced empty FIFO condition. CX28500 behavior when encountering a buffer descriptor owned by the Host is described in Table 6-40, *Transmit Buffer Descriptor*. The behavior is the same regardless of the presence or absence of an underrun condition. The effects of an underrun condition, once detected, are as described above, regardless of current buffer ownership.

8.2.8.2 Transmit Change Of Frame Alignment (COFA)

The TSYNC or STB input signal transitions from low to high, but at an unexpected time in comparison to the internal frame synchronization flywheel mechanism. COFA errors are only applicable to channelized ports (i.e., unchannelized ports ignore the TSYNC input). Frame synchronization indicates the expected location of the first bit of time slot 0 on the transmit serial data output. Lacking frame synchronization, the transmitter cannot map or align time slots. This error affects all active channels on the respective port.

Reason:

- Signal failure, glitch or realignment caused by the physical interface sourcing the TSYNC/STB input signal.

Effects:

- Causes serial interface to enter COFA condition until a TSYNC/STB pulse arrives and is followed by at least the assigned number of time slots for this port, without another unexpected TSYNC/STB pulse.
- For every active channel on the respective port, TSLP places channels into the deactivate state. Wherein, TSLP sends a repetitive abort sequence of 16 consecutive ones.
- Transmit COFA Interrupt (if COFAIEN = 1 in Section 6.7.5).

NOTE: COFA interrupt is generated immediately. To synchronize the Host's response to a COFA condition, a COFA Recovery interrupt is also provided.

- TDMA can read more buffers to refill the internal channel FIFO (since this process is asynchronous to the TSLP), but eventually the TDMA stops servicing this channel since the TSLP has stopped sending data.
- Transmit output is three-stated.

Channel Level Recovery Action:

- Transmit channel complete reactivation is required.

8.2.8.3 Transmit COFA Recovery (TCREC)

Reason:

- TSIU terminates the internal COFA condition due to the arrival of a TSYNC/STB pulse followed by at least one frame for this port without another unexpected TSYNC/STB pulse.

Effects:

- TCREC Interrupt (if COFAIEN = 1 in Table 6-36, *TSIU Port Configuration Register*).

Channel Level Recovery Actions:

- Transmit channel complete reactivation is required. Must wait until COFA recovery interrupt is generated, then reactivate the channel. If the COFA event is not flushed via a channel reactivation, then the port is three-stated, again.

8.2.9 Receive Errors

Receive errors are service-affecting, but do not require a corrective action by the Host to resume normal processing, except when CX28500 fetches a HOST owned NP = 1 message buffer descriptor. Then a jump service request is required to resume normal processing.

NOTE: In all the cases where a message is received and its buffer descriptor is closed with any of the errors listed below, the ABORT field of the receive status descriptor is set to non-zero. This ensures a transmitter operating in the self-servicing buffer mode (see below) ends each incomplete message with an appropriate abort code.

8.2.9.1 Receive Overflow [BUFF]

The RxDMA receives a signal from the RSLP that more data bits are available to be stored, but the RxDMA channel FIFO is already full.

Reasons:

- Degradation of Host subsystem performance.
- Shortage of shared memory buffers. The receive buffer CX28500 needs to fill is presently Host-owned.
- PCI bus congestion.

Effects:

- RxBUFF Interrupt (if BUFFIEN = 1 in Section 6.6.2).
- If a receive message was in-progress, that message is marked as errored. RSLP scans for the opening flag of the next HDLC message and any subsequent receive messages are discarded until the internal FIFO has room to accept more RSIU data. Notice the channel remains active, and channel recovery is automatic.
- When the in-progress message reaches the top of the internal FIFO, the entire HDLC message (before the overflow occurred) is copied to shared memory buffers and their last Receive Buffer Status Descriptors are written with ONR = HOST, EOM = 1, ERROR = BUFF (if INHRBSD = 0 in Section 6.6.4 and Section 6.7.4).
- RxERR interrupt is generated, if ERRIEN is set in Section 6.6.4 and Section 6.7.4, indicating a RxBUFF error overflow.
- RDMA is not affected and continues shared memory buffer processing.
- If a consecutive overflow condition exists, only the first overflow triggers an overflow interrupt while all succeeding overflows are discarded and not reported.

Channel Level Recovery Actions:

- If possible, increase internal FIFO size assigned to this channel. For this action, the channel must first be deactivated.
- If necessary, alleviate PCI bus congestion.
- Notice that channel reactivation is not required.

NOTE: Since CX28500 completely separates the two processes of storing data in shared memory buffers and receiving data from the serial interface, it is irrelevant to CX28500 how the overflow condition was created. To be specific, there is no distinction between a BUFF error created due to a Host-owned buffer descriptor or due to a latency-induced full FIFO condition. CX28500 behavior when encountering a buffer descriptor owned by the Host is described in Table 6-41, *Receive Buffer Descriptor*. The behavior is the same regardless of the presence or absence of an overflow condition. The effects of an overflow condition, once detected, are as described above, regardless of the current buffer ownership.

8.2.9.2 Receive Change Of Frame Alignment (COFA)

RSYNC or STB input signal transitions from low to high, but at an unexpected time in comparison to the frame synchronization flywheel mechanism. COFA errors are only applicable to channelized ports (i.e., unchannelized ports ignore the RSYNC/STB input). Frame synchronization indicates the expected location of the first bit of time slot 0 on the receive serial data input. Lacking frame synchronization, the receiver cannot map or align time slots. This error affects all active channels on the respective port, but does not require a Host recovery action.

Reason:

- Signal failure, glitch, or realignment caused by the physical interface sourcing the RSYNC or STB input signal.

Effects:

- Causes serial interface to enter COFA condition until the RSYNC/STB pulse is followed by at least the assigned number of time slots for this port, without another unexpected RSYNC/STB pulse.
- If a receive message was in-progress, that message is marked as errored. RSLP scans for the opening flag of the next HDLC message and any subsequent receive messages are discarded until the internal COFA condition has ended.
- When the in-progress message reaches the top of the internal FIFO, the entire HDLC message is copied to shared memory buffers, and Receive Buffer Status Descriptors are written with ONR = HOST and ERROR = COFA (if INHRBSD = 0 in Section 6.6.4 and Section 6.7.4).
- Receive COFA Interrupt is generated (if COFAIEN = 1 in Table 6-28, *RSIU Port Configuration Register*). Note that a TSTB change of alignment causes both a receive and a transmit COFA interrupt, since TSTB applies to both transmit and receive directions simultaneously.
- Normal operations continue after the COFA condition ends.
- RDMA is not affected and continues shared memory buffer processing.

Channel Level Recovery Actions:

- None required.

8.2.9.3 Out Of Frame (OOF)

Out-of-frame or loss-of-frame indicates the entire, or partial, receive serial data stream is invalid and only time slots marked with OOF from that port should be ignored.

Reason:

- ROOF input pin is asserted (high) because the attached physical layer device is unable to recover a valid, framed signal.

Effects:

- OOF Interrupt (if OOFIEN = 1 and OOFABT = 1 in Table 6-28, *RSIU Port Configuration Register*).
- If bit field OOFABT = 0, RSLP and RDMA continue as if no errors and transfer received data into shared memory buffers normally.
- If bit field OOFABT = 1 and a receive message is in-progress, the current message is ended with OOF status and RSLP scans for the opening flag of the next HDLC message. When the in-progress message reaches the top of the internal FIFO, the entire message is copied to shared memory buffers and the Buffer Status Descriptor is written (if INHRBSD = 0 in Section 6.6.5) with ONR = HOST and ERROR = OOF.
- RDMA is not affected and continues shared memory buffer processing.
- Receive channels recover automatically when the ROOF input pin is deasserted (low), indicating the OOF condition has ended.

Channel Level Recovery Actions:

- None required.

8.2.9.4 Frame Check Sequence (FCS) Error

In this case, the Frame Check Sequence (FCS) which CX28500 calculated for the received HDLC message does not match the FCS located within the message.

Reason:

- Bit errors during transmission.

Effects:

- When the message reaches the top of the internal FIFO, the entire HDLC message is copied to shared memory buffers and the Buffer Status Descriptor is written (if INHRBSD = 0 in Section 6.6.4 and Section 6.7.4) with ONR = HOST, ERROR = FCS.
- EOM Interrupt with RxFCS error status, (if ERRIEN = 1 in Section 6.6.4 and Section 6.7.4).
- The RSLP scans for the opening flag of the next HDLC message.
- RDMA is not affected and continues shared memory buffer processing.

Channel Level Recovery Actions:

- None required.

8.2.9.5 Octet Alignment Error (ALIGN)

The HDLC message size after zero-bit extraction was not a multiple of 8 bits.

Reasons:

- Bit errors during transmission.
- Incorrect message transmission from distant end.

Effects:

- When the message reaches the top of the internal FIFO, the entire HDLC message is copied to shared memory buffers and the Buffer Status Descriptor is written (if INHRBSD = 0 in Section 6.6.4 and Section 6.7.4) with ONR = HOST, ERROR = ALIGN.
- EOM Interrupt with RxALIGN error status, (if ERRIEN = 1 in Section 6.6.4 and Section 6.7.4).
- The RSLP scans for the opening flag of the next HDLC message.
- RDMA is not affected and continues shared memory buffer processing.

Channel Level Recovery Actions:

- None required.

8.2.9.6 Abort Termination (ABT)

The receiver detects an abort sequence in the middle of the message payload. An abort sequence is defined as any zero followed by 15 consecutive 1s.

Reasons:

- Distant end failed to complete transmission of the HDLC message.
- Path conditioning has replaced the normal channel content with an all ones pattern, due to a network alarm condition.

Effects:

- When the message reaches the top of the internal FIFO, the entire HDLC message is copied to shared memory buffers and the Buffer Status Descriptor is written (if INHRBSD = 0 in Section 6.6.4 and Section 6.7.4) with ONR = HOST, ERROR = ABT.
- EOM Interrupt with RxABT error status, (if ERRIEN = 1 in Section 6.6.4 and Section 6.7.4).
- The RSLP scans for the opening flag of the next HDLC message.
- RDMA is not affected and continues shared memory buffer processing.

Channel Level Recovery Actions:

- None required.

8.2.9.7 Long Message (LNG)

The received HDLC message length is determined to be greater than the maximum allowable message size per the MAXSEL bit field in Section 6.6.8.

Reason:

- Incorrect message transmission from distant end.

Effects:

- When the message reaches the top of the internal FIFO, the HDLC message up to the maximum legal length is copied to shared memory buffers and the Buffer Status Descriptor is written (if INHRBSD = 0 in Section 6.6.4 and Section 6.7.4) with ONR = HOST, ERROR = LNG.
- EOM Interrupt with RxLNG error status (if ERRIEN = 1 in Section 6.6.4 and Section 6.7.4).
- The RSLP scans for the opening flag of the next HDLC message.
- RDMA is not affected and continues shared memory buffer processing.

Channel Level Recovery Actions:

- None required.

8.2.9.8 Short Message (SHT)

The total received HDLC message size (between open/close flags) is determined to be less than the number of FCS bits specified for that channel plus one octet. For example, a channel configured for 16-bit FCS must receive a minimum of three octets, one octet of payload and two octets of FCS, to avoid a short message error. In this example, receiving only two octets is considered a short message.

NOTE: Any message that ends with an error (any error except for an overflow) and for which the entire message (regardless of its length) still resides in the internal SLP buffer (meaning no data has yet been transferred to the internal channel FIFO), CX28500 generates a SHT interrupt and does not transfer any of that message to a shared memory buffer. In this case, no other indication is given for the errored message, thus saving PCI bandwidth as well as shared memory buffer space.

Since the RxSHT interrupt in this case is reported immediately, its interrupt descriptor can arrive in the shared memory interrupt queue before an earlier message that remains queued in the internal DMA channel FIFO. Hence, interrupts from these two messages may appear out of sequence with respect to their actual order of arrival.

Reasons:

- Bit errors during transmission.
- Incorrect message transmission from distant end.

Effects:

- RxSHT Interrupt (if IDLEIEN = 1 in Section 6.6.2).
- RSLP resumes scanning for opening flag of the next HDLC message.
- RDMA is not affected and continues shared memory buffer processing. Notice that a short message's contents are not transferred to memory, therefore no message descriptor or data buffer is consumed by a short message.

Channel Level Recovery Actions:

- None required.

8.3 Transparent Mode

CX28500 supports a completely transparent mode where no distinction is made between information and non-information bits in the channel bit stream. This mode is assigned on a per-channel and per-direction basis by the PROTOCOL bit field in Section 6.6.2 and Section 6.7.2.

8.3.1 Message Configuration Bits–Transparent Mode

The transmit Buffer Descriptor contains a group of bits that specify the data to be transmitted after the end of a transparent mode message. The bits are specified as follows:

- Idle Code specification, IC
- Inter-message Pad Fill Count, PADCNT
- Send an Abort Sequence, ABORT

NOTE: Message configuration bits are also used in HDLC mode, but their meaning is slightly different. Refer to Section 8.2.5.

8.3.1.1 Idle Code

Idle Code (IC) bit field selects one of a set of idle pad fill octets to be sent after the current message is transmitted in the event the next message descriptor is unavailable (i.e., Host-owned) or inter-message pad fill is requested via PADCNT. The default idle code in transparent mode is all 1s, a silent signal.

8.3.1.2 Inter-Message Pad Fill

Pad Count (PADCNT) bit field specifies how many pad fill octets (selected by IC) are transmitted between messages. PADCNT specifies the minimum number of pad fill octets plus one, as follows:

1. PADCNT = 0: One IC
2. PADCNT = 1: Two ICs
3. PADCNT = 2: Three ICs
4. etc...

8.3.1.3 Ending a Message with an Abort or Sending an Abort Sequence

When the ABORT field is non-zero in any CX28500-owned transmit buffer descriptor, CX28500 interprets this as a request to end an in-progress message with the abort sequence. Abort sequence for Transparent mode is defined to be a sequence of all 1s. The abort sequence is terminated only when a new CX28500-owned message descriptor with a non-zero BLEN becomes available. In this case, CX28500 resynchronizes the start of the next message transmission to the time slot marked as the first time slot on that channel.

If ABORT ≠ 0, and the prior buffer descriptor on that channel contained EOM = 1, (i.e., the previous buffer descriptor ended a message), the abort command is simply ignored and CX28500 moves onto the next message descriptor.

The Host can set EOM = 1 in any transmit buffer descriptor to separate this transparent mode “message” from the next message, according to the IC and PADCNT bit fields. Unlike HDLC mode, the number of pad fill octets transmitted equals PADCNT plus one and no flag characters are inserted.

8.3.2 Transmit Events

Transmit events are informational in nature and require no recovery actions.

8.3.2.1 End Of Buffer [EOB]

Reason:

- TDMA reached the end of a buffer by servicing a number of octets equal to the BLEN bit field in the Transmit Buffer Descriptor. Note that EOB and EOM are not coincident and thus generate two separate events.

Effects:

- TxEOB interrupt (if EOBIEN = 1 in Transmit Buffer Descriptor).
- CX28500 continues with normal message processing. If the TDMA does not get more data from shared memory before the internal channel FIFO becomes empty and the TSLP needs to output another data bit, TSLP generates an underflow error.

8.3.2.2 End Of Message [EOM]

Reason:

- TSLP has transmitted (actually, transferred to the TSIU) the last bit of a data buffer and the Transmit Buffer Descriptor signified the end of a message with bit field EOM = 1. Note that EOB and EOM are not coincident and thus generate two separate events.

Effects:

- TxEOM interrupt (if EOMIEN = 1 in Section 6.7.2).
- TSLP and TDMA continue normal message processing. If the TDMA does not get more data from shared memory before the internal channel FIFO becomes empty and the TSLP needs to output another data bit, TSLP outputs pad fill octets until more data is available.

8.3.3 Receive Events

Receive events are informational in nature and require no recovery actions.

8.3.3.1 End Of Buffer [EOB]

Reason:

- Transparent mode receive channels have no concept of a “message” boundary. Hence, all received data is simply written to a shared memory data buffer until that buffer is filled according to the number of octets specified by the bit field BLEN in the Receive Buffer Descriptor.

Effects:

- EOB Interrupt (if EOBIEN = 1 in Receive Buffer Descriptor).
- RDMA and RSLP continue normal processing.

8.3.3.2 End Of Message (EOM)

Reason:

- RSLP must force an end of a message due to a receive error condition. Error conditions include Overflow, COFA, or OOF.

Effects:

- RDMA sets bit field EOM = 1 in Receive Buffer Status Descriptor (if INHRBSD = 0 in Section 6.6.5).
- RxEOM interrupt (if ERRIEN = 1 in Section 6.6.5) with the appropriate RxERR status.
- RSLP continues normal processing after the error condition has ended.
- RDMA is not affected and continues shared memory buffer processing.

8.3.3.3 Frame Recovery (FREC)

Reason:

- SIU detects the serial interface has transitioned from an out-of-frame to an in-frame condition. If the ROOF pin is used as an out-of-frame indication, a FREC interrupt is generated only after one full frame without any OOF assertion is detected. If the ROOF pin is used as a general purpose interrupt input, a SPORT (Serial PORT) interrupt is generated.

Effects:

- FREC/SPORT Interrupt (if OOFIEN = 1 in Table 6-28, *RSIU Port Configuration Register*).
- RSLP and RDMA continue normal processing.

8.3.3.4 Receive COFA Recovery (RCREC)

Reason:

- SIU terminates the internal COFA condition due to a RSYNC/STB pulse followed by at least one frame for this port without another unexpected RSYNC/STB pulse.

Effects:

- RCREC Interrupt (if COFAIEN = 1 in Table 6-28, *RSIU Port Configuration Register*).
- RSLP and RDMA continue normal processing.

8.3.4 Transmit Errors

Transmit Errors are service-affecting and require a corrective action by the Host to resume normal processing.

8.3.4.1 Transmit Underrun [BUFF]

Same as HDLC mode.

Reasons:

- Degradation of the Host subsystem or application software.
- Buffer descriptor containing the continuation of a message is Host-owned.
- PCI bus congestion.

Effects:

- TxBUFF Interrupt (if BUFFIEN = 1 in Section 6.7.2).
- Transmit channel enters deactivate state, wherein TSLP sends a repetitive all 1s sequence.
- TDMA may read more buffers to refill the internal channel FIFO (because this process is asynchronous to the TSLP), but eventually the TDMA stops servicing this channel because the TSLP has stopped sending data.

Channel Level Recovery Actions:

- Transmit channel complete reactivation is required.

8.3.4.2 Transmit Change Of Frame Alignment (COFA)

Reason:

- Signal failure, glitch, or realignment caused by the physical interface sourcing the TSYNC/STB input signal.

Effects:

- Causes serial interface to enter COFA condition until a TSYNC/STB pulse arrives and is followed by at least one frame for this port, without another unexpected TSYNC/STB pulse.
- For every active channel on the respective port, TSLP places channels into the deactivate state, wherein TSLP sends a repetitive all ones sequence.
- Transmit output is three-stated.

Channel Level Recovery Actions:

- Transmit channel complete reactivation is required.

8.3.5 Receive Errors

Receive errors are service-affecting and may require a corrective action by the Host to resume normal processing.

8.3.5.1 Receive Overflow [BUFF]

Same as HDLC mode.

Reasons:

- Degradation of Host subsystem performance.
- Shortage of shared memory buffers. The receive buffer CX28500 needs to fill is presently Host-owned.
- PCI bus congestion.

Effects:

- RxBUFF Interrupt (if BUFFIEN = 1 in Section 6.6.2).
- Data received during an overflow condition is discarded.
- Data in the internal FIFO is copied to shared memory, the Receive Buffer Status Descriptor is written with ONR = HOST, EOM = 1, ERROR = BUFF (if INHRBSD = 0 in Section 6.6.4 and Section 6.7.4).
- If ERRIEN is set in Section 6.6.4 and Section 6.7.4, an RxERR interrupt is generated, indicating an RxBUFF overflow.
- When the overflow condition ends (i.e., space becomes available in the channel FIFO), RSLP automatically restarts data processing. However, RSLP ignores all time slots until reaching the time slot marked “first.”
- RDMA is not affected and continues shared memory buffer processing.

Channel Level Recovery Actions:

- If possible, increase internal FIFO size assigned to this channel. For this action, the channel must first be deactivated.
- If necessary, alleviate PCI bus congestion.
- Notice that channel reactivation is not required.

8.3.5.2 Receive Change Of Frame Alignment (COFA)

Same as HDLC mode.

Reason:

- Signal failure, glitch, or realignment caused by the physical interface sourcing the RSYNC or TSTB input signal.

Effects:

- Causes serial interface to enter COFA condition until the RSYNC/TSTB pulse is followed by at least one frame for this port, without another unexpected RSYNC/TSTB pulse.
- Current message processing is ended for every active channel on this port. All data received prior to the COFA condition is copied to shared memory buffers, and the Receive Buffer Status Descriptor is written with ONR = HOST and ERROR = COFA (if INHRBSD = 0 in Section 6.6.4 and Section 6.7.4). The only exception to this description happens when the COFA condition is detected within the first few bytes after channel activation or after the channel suffered an overflow or another COFA, as described in Section 8.3.5.4.
- Receive COFA Interrupt (if COFAIEN = 1 in Table 6-28, *RSIU Port Configuration Register*). When the COFA condition ends, RSLP restarts data processing automatically; however, all time slots are ignored until reaching the time slot marked “first.”
- RDMA is not affected and continues shared memory buffer processing.

Channel Level Recovery Actions:

- None required.

8.3.5.3 Out Of Frame (OOF)

Same as HDLC mode.

Reason:

- ROOF input pin is asserted (high) because the attached physical layer device is unable to recover a valid, framed signal. Only time slots marked with an OOF assertion are affected, and not the whole port.

Effects:

- OOF Interrupt (if OOFIEN = 1 and OOFABT = 1 in Table 6-28, *RSIU Port Configuration Register*).
- If bit field OOFABT = 0, RSLP and RDMA continue as if there are no errors and transfer received data into shared memory buffers normally.
- If bit field OOFABT = 1, all incoming data is replaced by all ones (0xFF) data sequence. Normal data processing resumes when the ROOF input pin is deasserted (low), indicating the OOF condition has ended.
- RDMA is not affected and continues shared memory buffer processing.

Channel Level Recovery Actions:

- None required.

8.3.5.4 Short COFA (SHT COFA)

A short COFA interrupt is generated for any transparent mode message whose reception is ended due to a COFA error and for which no data was transferred from RSLP to RDMA or to shared memory. In this case, no other indication is provided for this errored message, thus saving PCI bandwidth and Host buffers.

NOTE: Only transparent mode COFA creates such a scenario. The exact scenario is as follows: a COFA condition happens within the next few bytes after an abnormal message termination (i.e., a prior COFA or overflow error) or after a channel activation.

Reason:

- Signal failure, glitch or realignment caused by the physical interface sourcing the RSYNC or STB input signal.

Effects:

- RxSHT Interrupt (if IDLEIEN = 1 in Section 6.6.2).
- RSLP restarts channel operation as soon as the COFA condition is recovered and the channel reaches its first assigned time slot.
- RDMA is not affected and continues shared memory buffer processing. Notice that no shared memory buffer descriptors are consumed.

Channel Level Recovery Actions:

- None required.

9 Self-Servicing Buffers

The transmit and receive Buffer Descriptors and Buffer Status Descriptors are designed to facilitate a mechanism known as self-servicing buffers. This mechanism allows the Host to configure CX28500 to fill a table of data buffers as it receives a complete message through a receive channel, then empty the same list of data buffers through a transmit channel without any further Host intervention.

The self-servicing buffer mechanism works as follows:

1. Host initializes message descriptor table in shared memory.
2. Host configures receive channel head pointer to point to first message descriptor.
3. Host configures transmit channel head pointer to point to the same message descriptor.
4. The OWNER bit field in the buffer descriptor is set to 0. For the transmitter, this means the buffer is owned by the Host. For the receiver, this means the buffer is owned by CX28500.
5. The NP bit field in the transmit buffer descriptor is set to 0. This allows the CX28500 to poll the OWNER bit field of the buffer descriptors.
6. Both receive and transmit channels are activated.
7. As the receiver detects an incoming message and begins filling the first data buffer, the transmitter remains idle and polls the OWNER bit in the buffer descriptor.
8. When the receiver fills the first buffer, it writes the Buffer Status Descriptor (setting the OWNER bit field to 1) and then moves on to the next message descriptor. Although the data buffer is not going to be returned to the Host, the Inhibit Receive Buffer Status descriptor option should not be enabled (INHRBSD = 0 in RDMA channel configuration register). In other words, the CX28500 must be allowed to overwrite the Buffer Descriptor with the Buffer Status Descriptor.
9. During the next poll transaction, the transmit channel detects if the OWNER bit field is set to 1 in the first buffer descriptor and assumes ownership. The transmitter then begins emptying the first data buffer and moving data to the serial port. The Inhibit Transmit Buffer Status Descriptor option must be disabled for CX28500 to return ownership to the HOST (i.e., INHTBSD = 0 in TDMA Channel Configuration register). In other words, the CX28500 must be allowed to overwrite the Buffer Descriptor with the Buffer Status Descriptor.
10. Upon detecting an End Of Message, the receiver writes the Buffer Status Descriptor, marking the buffer as an End Of Message and setting the buffer length (BLEN) to indicate the amount of data contained in this buffer.
11. When the transmitter reads an End Of Message buffer, it sends the BLEN amount of data out the serial port and writes the buffer status descriptor (setting the OWNER bit field to 0), then moves onto the next buffer or enters the idle state (if the next buffer is still owned by the receiver).

12. Steps 6-10 are repeated continuously, until the Host deactivates either the receive or transmit channel.

It is important to note that for self-servicing buffers, the Host does not need to write to any descriptors for receive or transmit operations. CX28500 writes the Receive Buffer Status Descriptor, which is subsequently used as the Transmit Buffer Descriptor.

To provide a mechanism for the self-served transmitter to recover from a receiver error, RDMA automatically sets ABORT ≠ 0 in the buffer descriptor upon detection of any receive error (EOM is also set to one). In HDLC mode, ABORT ≠ 0 in the buffer descriptor causes the transmitter to send an abort sequence. In Transparent mode, ABORT ≠ 0 in the buffer descriptor causes pad fill transmission until the next message is available. In either case, subsequent message transmission remains unaffected.

Two things are important to notice:

1. If the transmitter is slower than the receiver (even by a tiny fraction), it will not empty the buffers fast enough and the receiver will eventually overflow the internal buffers.
2. If the receiver marks End Of Message (EOM) in a Buffer Status Descriptor, it also changes the BLEN field to a value that is necessarily lower than the original value. The transmitter sends the correct number of bytes and relinquishes ownership, but it does not return the BLEN to the original value. The receiver, upon wrapping around and reaching this buffer will see the new (lower) value of BLEN instead of the older (higher) value. Over a long period of time, this can cause problems as buffers become shorter and shorter. The solution is to use buffers that are all the same length, and periodically test the Buffer Descriptor Table to return the BLEN fields to the original value (be careful, though, not to overwrite the BLEN field of a buffer that the transmitter has not yet begun sending).

10 Electrical and Mechanical Specification

10.1 Electrical and Environmental Specifications

10.1.1 Absolute Maximum Ratings

Stressing the device parameters beyond absolute maximum ratings may cause permanent damage to the device. This is a stress rating only. Functional operation of the device at these or any other conditions beyond those listed in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Table 10-1. Absolute Maximum Ratings

Parameter	Symbol	Value		Unit
		Minimum	Maximum	
Core power supply	VDD_c	–0.5	3.3	V
I/O Power Supply	VDD_io	–0.5	4.6	V
5 V Tolerant Power Supply	VGG	VDD_io–0.5	6.0	V
Core Supply Current	IDD_c	—	1	A
I/O Supply Current	IDD_io	—	0.3	A
Continuous Power Dissipation	P_d	—	3.7[(1)]	W
Constant Voltage on any Signal Pin	V_i	–1.0	VDD_io + 0.5	
Constant Current on any Signal Pin	I_i	–10	10	mA
Transient Current on any Signal Pin	Latchup	150 mA @ 125 °C	300 mA @ 25 °C	mA
Operating Junction Temperature	T_j	–40	125	°C
Storage Temperature	T_s	–55	125	°C
Vapor phase soldering temperature (1 min.)	T_{vsol}	—	220	°C

FOOTNOTE:

(1) The 3.7 W maximum continuous power dissipation is observed for an application utilizing the complete potential of the device, with voltages that deviate 5% higher than nominal, and with 60 pF capacitance load on the I/Os. Any real-world application will see much lower figures.

10.1.2 Recommended Operating Conditions

Table 10-2. Recommended 3.3 V Operating Conditions

Parameter	Symbol	Value		Unit
		Minimum	Maximum	
Power Supply	VDD_c	2.375	2.625	V
I/O Power Supply	VDD_io	3.135	3.465	V
5 V Tolerant Power Supply	VGG	VDD_io	5.25	V
Ambient Operating Temperature KPF EPF	T_{ac}	 0 –40	 +70 +85	 °C °C
High-Level Input Voltage	V_{ih} (1)	2.0	VDD_io + 0.5	V
Low-Level Input Voltage	V_{il} (1)	0	0.8	V
High-Level Output Current Source	I_{oh}	200	400	µA
Low Level Output Current Sink	I_{ol}	2	4	mA
Output Capacitive Loading	C_{ld}	60	85	pF

FOOTNOTE:
(1) Apply to all pins, except the PCI interface, which is defined in Table 10-4.

10.1.3 Electrical Characteristics

Table 10-3. DC Characteristics for 3.3 V Operation

Parameter	Symbol	Value	Units
High-Level Output Voltage	V_{oh}	2.4	V
Low-Level Output Voltage	V_{ol}	0.4	V
Input Leakage Current	I_{I}	–10 to 10	µA
Three-state Leakage Current	I_{oz}	–10 to 10	µA
Resistive Pullup Current	I_{pr}	40 to 500	µA

10.1.4 Power-up Sequencing

Power up sequencing involves the order of powering up the I/O and core supplies and the period of time between powering up these supplies. When the I/O supply (VDD_io) is powered up first, the output drivers can be in an indeterminate state until the core supply (VDD_c) is powered up. If the delay in the power sequence is too long (several ms or more), the unknown state of the output drivers can cause system problems. The recommended power up sequence is first 5 V (VGG), then 3.3 V (VDD_io), and then 2.5 V (VDD_c).

The I/O supply must be powered up before the core supply. Otherwise, a high current will be drawn until the I/O supply is powered up. The time from the detection of I/O power to the disabling of output drivers and the time from the detection of core power to the enabling of output drivers will not exceed 100 ns. Therefore, it is recommended that the core is powered up more than 100 ns after I/O. If core is powered up during the first 100 ns after I/O power-up, the device may draw more current until the first 100 ns are elapsed.

10.2 Timing and Switching Specifications

10.2.1 Overview

This section defines the timing and switching characteristics of CX28500. The major subsystems include the Host interface, the expansion bus interface, and the serial interface. The Host interface is Peripheral Component Interface (PCI) compliant. For other references to PCI, see the *PCI Local Bus Specification*, Revision 2.1, June 1, 1995. The expansion bus and serial bus interfaces are similar to the Host interface timing characteristics; the differences and specific characteristics common to either interface are further defined.

10.2.2 Host Interface (PCI) Timing and Switching Characteristic

Reference the *PCI Local Bus Specification, Revision 2.1*, June 1, 1995 for information the following:

- Indeterminate inputs and metastability
- Power requirements, sequencing, decoupling
- PCI DC specifications
- PCI AC specifications
- PCI V/I curves
- Maximum AC ratings and device protection

Table 10-4. PCI Interface DC Specifications

Symbol	Parameter	Condition	Min	Max	Units
VDD_io	Supply Voltage	—	3	3.6	V
V_{ih}	Input High Voltage	—	0.5 VDD_io	VDD_io + 0.5	V
V_{il}	Input Low Voltage	—	–0.5	0.3 VDD_io	V
I_{il}	Input Leakage Current(1)	$0 < V_{in} <$ VDD_io	—	+/-10	µA
V_{oh}	Output High Voltage	I_{out} = –500 µA	0.9 VDD_io	—	V
V_{ol}	Output Low Voltage(2)	I_{out} = 1500 µA	—	0.1 VDD_io	V
$C_{out}/C_{in}/C_{io}$	Output, Input, and I/O Pin Capacitance	—	—	10	pF
C_{clk}	PCLK Pin Capacitance	—	5	12	pF
C_{idsel}	IDSEL Pin Capacitance(3)	—	—	8	pF
L_{pin}	Pin Inductance	—	—	20	nH

FOOTNOTE:

(1) Input leakage currents include hi-Z output leakage for all bidirectional buffers with three-state outputs.

(2) Signals without pullup resistors must have 3 mA low output current. Signals requiring pullup must have 6 mA; the latter include FRAME*, TRDY*, IRDY*, DEVSEL*, STOP*, SERR*, and PERR*.

(3) Lower capacitance on this input-only pin allows for non-resistive coupling to AD[xx].

Table 10-5. PCI Clock (PCLK) Waveform Parameters, 3.3 V Clock

Symbol	Parameter	Min 33 MHz	Max 33 MHz	Min 66 MHz	Max 66 MHz	Units
T_{cyc}	Clock Cycle Time(1)	30	Infinite	15	30	ns
T_{high}	Clock High Time	11	—	6	—	ns
T_{low}	Clock Low Time	11	—	6	—	ns
—	Clock Slew Rate(2)	1	4	1.5	4	V/ns
V_{ptp}	Peak-to-Peak Voltage	0.4 VDD_io	—	0.4 VDD_io	—	V

FOOTNOTE:

(1) CX28500 works with any clock frequency between DC and 66 MHz, nominally. The clock frequency may be changed at any time during operation of the system as long as clock edges remain monotonic, and minimum cycle and high and low times are not violated. The clock may only be stopped in a low state.

(2) Rise and fall times are specified in terms of the edge rate measured in V/ns. This slew rate must be met across the minimum peak-to-peak portion of the clock waveform.

Figure 10-1. PCI Clock (PCLK) Waveform, 3.3 V Clock

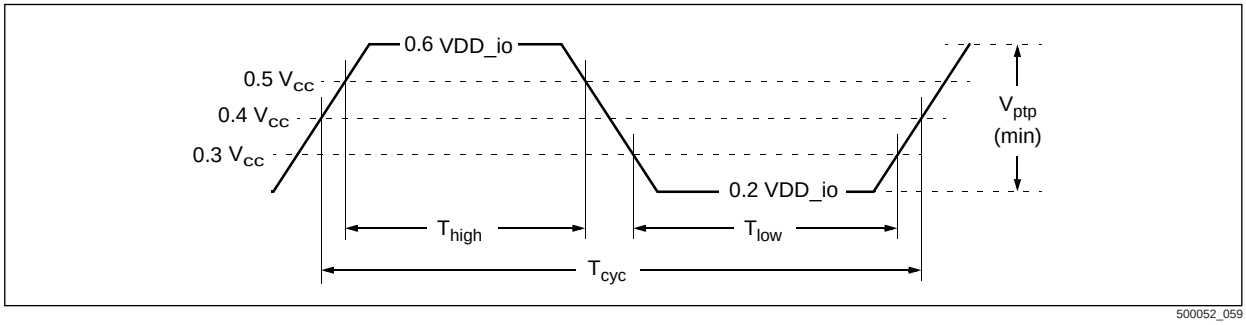


Table 10-6. PCI Reset Parameters

Symbol	Parameter	Min	Max	Units
T_{rst}	Reset Active Time after Power Stable	1	—	ms
T_{rst_clk}	Reset Active Time after Clock Stable	100	—	µs
$T_{rst\text{-}off}$	Reset Active to Float Delay	—	40	ns
T_{rrsu}	REQ64 to RST setup time	10 T_{cyc}	—	ns
T_{rrh}	RST to REQ64 hold time	0	50	ns

Table 10-7. PCI Input/Output Timing Parameters

Symbol	Parameter	Min 33 MHz	Max 33 MHz	Min 66 MHz	Max 66 MHz	Units
T_{val}	PCLK to Signal Valid Delay–Bused Signal(1, 2)	2	11	2	10	ns
T_{val} (ptp)	PCLK to Signal Valid Delay–Point To Point(1, 2)	2	12	2	10	ns
T_{val} (o.d.)	PCLK to Signal Valid Delay–Open Drain(5)	2	11	2	10	ns
T_{on}	Float to Active Delay(3)	2	—	2	—	ns
T_{off}	Active to Float Delay(3)	—	28	—	14	ns
T_{ds}	Input Setup Time to Clock–Bused Signal(2)	7	—	4	—	ns
T_{su} (ptp)	Input Setup Time to Clock–Point To Point(2)	10, 12	—	5	—	ns
T_{dh}	Input Hold Time from Clock	0(4)	—	0(4)	—	ns

FOOTNOTE:

(1) Minimum and maximum times are evaluated at 80 pF equivalent load. Actual test capacitance may vary, and results should be correlated to these specifications.

(2) REQ* and GNT* are the only point-to-point signals, and have different output valid delay and input setup times than do bused signals. GNT* has a setup of 10 ns; REQ* has a setup of 12 ns for 33 MHz.

(3) For purposes of active/float timing measurements, the hi-Z or off state is defined to be when the total current delivered through the component pin is less than or equal to the leakage current specification at 80 pF equivalent load.

(4) Actual measurements were done with 0.5 ns for test equipment guardband.

(5) The only open-drain outputs in the PCI interface are the INTA# signal and the SERR# signal.

Table 10-8. PCI I/O Measure Conditions

Symbol	Parameter	Value	Unit
V_{th}	Voltage Threshold High(1)	0.6 VDD_c	V
V_{tl}	Voltage Threshold Low(1)	0.2 VDD_c	V
V_{test}	Voltage Test Point	0.4 VDD_c	V
V_{max}	Maximum Peak-to-Peak(2)	0.4 VDD_c	V
—	Input Signal Edge Rate	1	V/ns

FOOTNOTE:

(1) The input test is done with 0.1 VDD_c of overdrive (over V_{ih} and V_{il}). Timing parameters must be met with no more overdrive than this. Production testing can use different voltage values, but must correlate results back to these parameters.

(2) V_{max} specifies the maximum peak-to-peak voltage waveform allowed for measuring input timing. Production testing can use different voltage values, but must correlate results back to these parameters.

Figure 10-2. PCI Output Timing Waveform

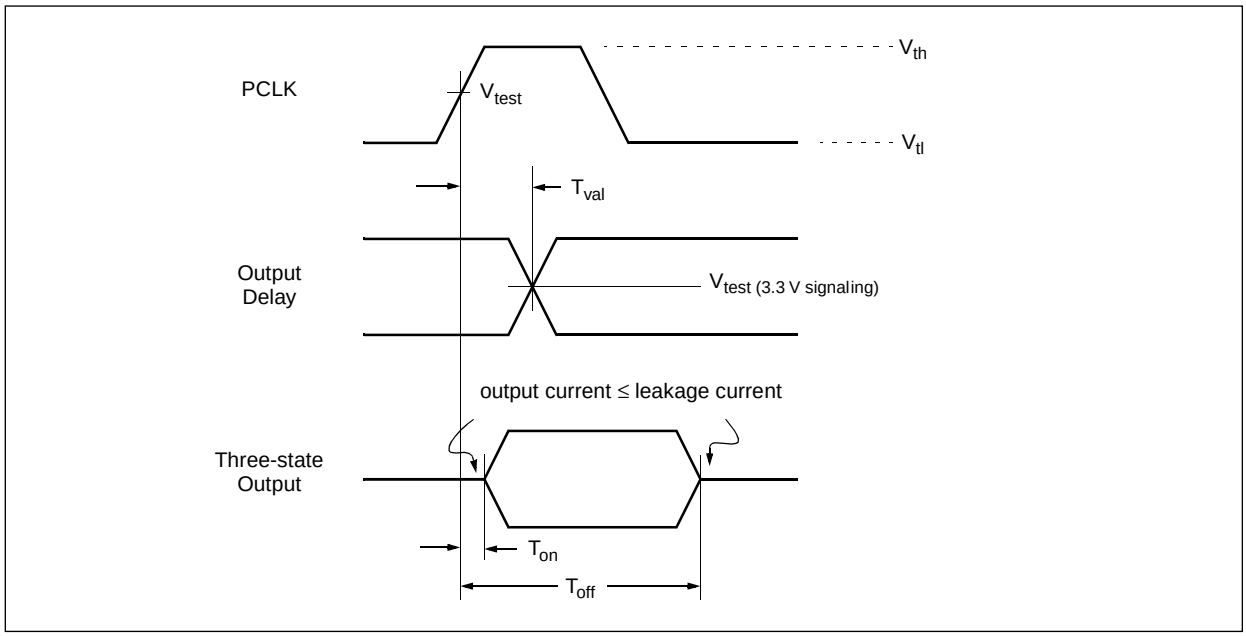
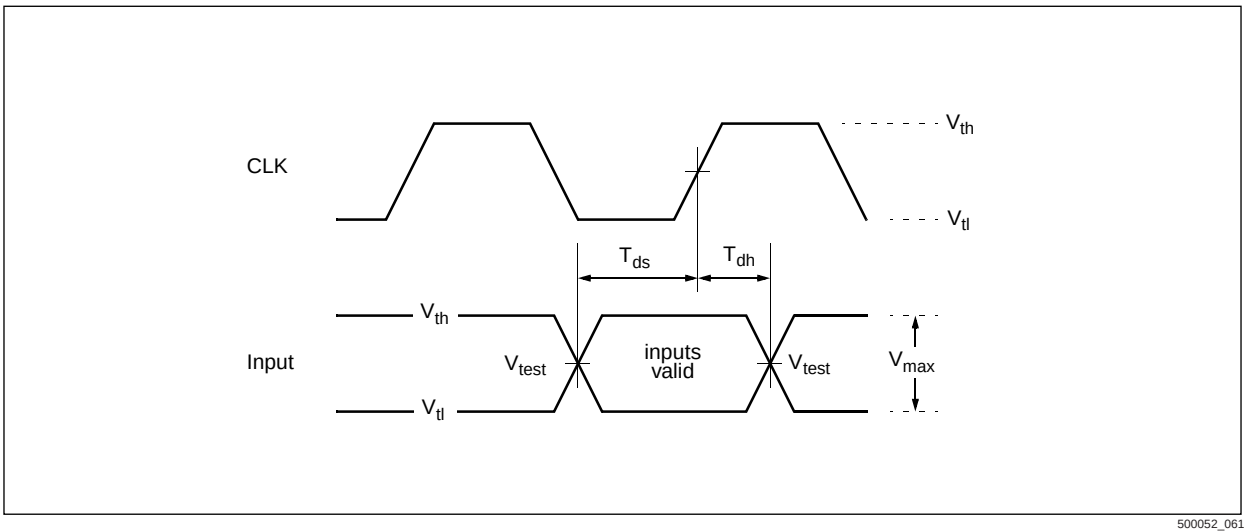


Figure 10-3. PCI Input Timing Waveform



10.2.3 Expansion Bus (EBUS) Timing and Switching Characteristics

The EBUS timing is derived directly from the PCI clock (PCLK) input into CX28500.

The EBUS clock can have the same frequency as the PCI clock, or it can have half the frequency of the PCI clock.

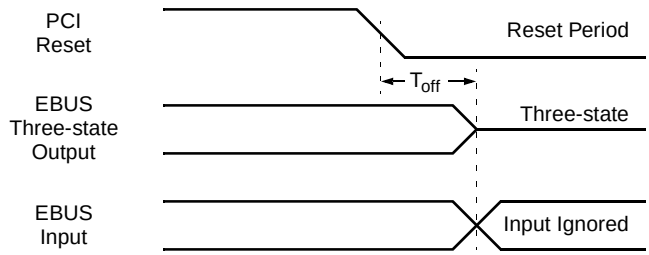
This option is configured in the EBUS Configuration register. Please see Table 6-12 for more detail.

Table 10-9. EBUS Reset Parameters

Symbol	Parameter	Min	Max	Units
T_{off}	Active to Inactive Delay(1)	—	28	ns

FOOTNOTE:
(1) For purposes of active/float timing measurements, the hi-Z or off state is defined to be when the total current delivered through the component pin is less than or equal to the leakage current specification.

Figure 10-4. EBUS Reset Timing



GENERAL NOTE: The EBUS reset is dependent on the PRST* (PCI Reset) signal being asserted low.

500052_062

Table 10-10. EBUS Input/Output Timing Parameters

Symbol	Parameter	Min	Max	Units
T_{val}	ECLK to Signal Valid Delay—Bused Signal(1)	–9	1	ns
T_{val} (ptp)	ECLK to Signal Valid Delay—Point To Point(1)	2	12	ns
T_{on}	Float to Active Delay(2)	–5	—	ns
T_{off}	Active to Float Delay(2)	—	20	ns
T_{ds}	Input Setup Time to Clock – Bused Signal	18	—	ns
T_{ds} (ptp)	Input Setup Time to Clock – Point To Point	5	—	ns
T_{dh}	Input Hold Time from Clock	0	—	ns

FOOTNOTE:
(1) Minimum and maximum times are evaluated at 80 pF equivalent load. Actual test capacitance may vary, and results should be correlated to these specifications.
(2) For purposes of active/float timing measurements, the hi-Z or off state is defined to be when the total current delivered through the component pin is less than or equal to the leakage current specification at 80 pF equivalent load.

Table 10-11. EBUS Input/Output Measure Conditions

Symbol	Parameter	Value	Units
V_{th}	Voltage Threshold High(1)	0.6 VDD_io	V
V_{tl}	Voltage Threshold Low(1)	0.2 VDD_io	V
V_{test}	Voltage Test Point	0.4 VDD_io	V
V_{max}	Maximum Peak-to-Peak(2)	0.4 VDD_io	V
—	Input Signal Slew Rate	1.5	V/ns

FOOTNOTE:

(1) The input test for the 3.3 V environment is done with 0.1*VDD_io of overdrive. Timing parameters must be met with no more overdrive than this. Production testing may use different voltage values, but must correlate results back to these parameters.

(2) V_{max} specifies the maximum peak-to-peak voltage waveform allowed for measuring input timing. Production testing may use different voltage values, but must correlate results back to these parameters.

Figure 10-5. EBUS Output Timing Waveform

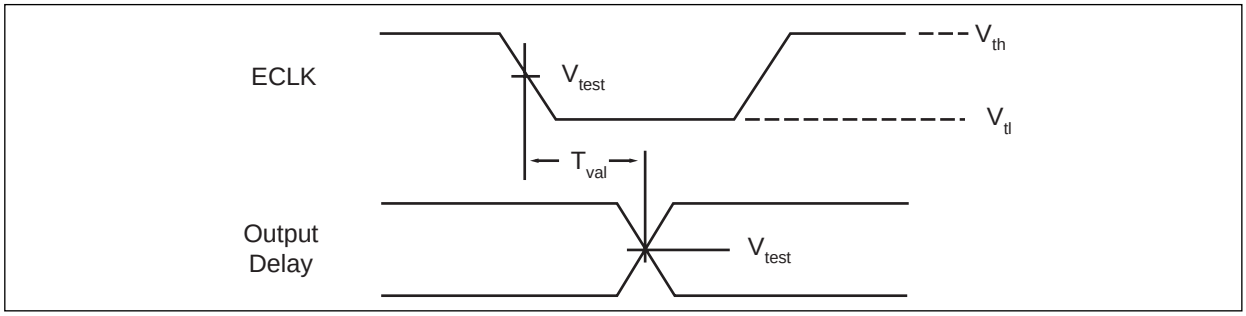
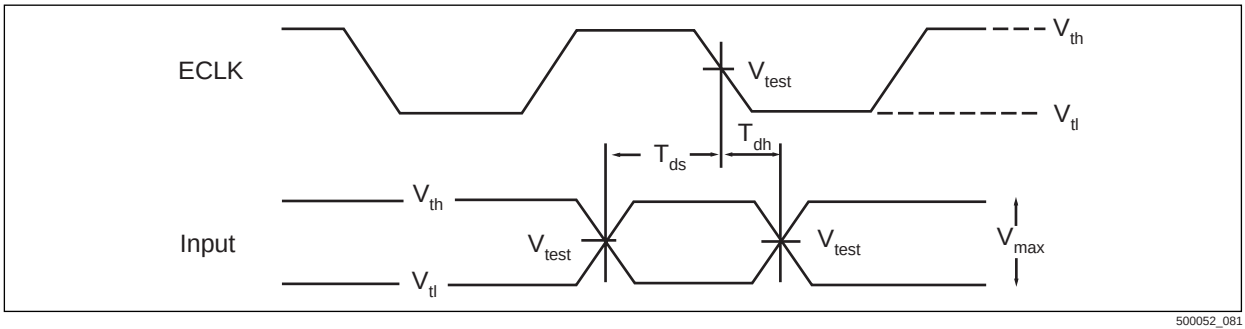
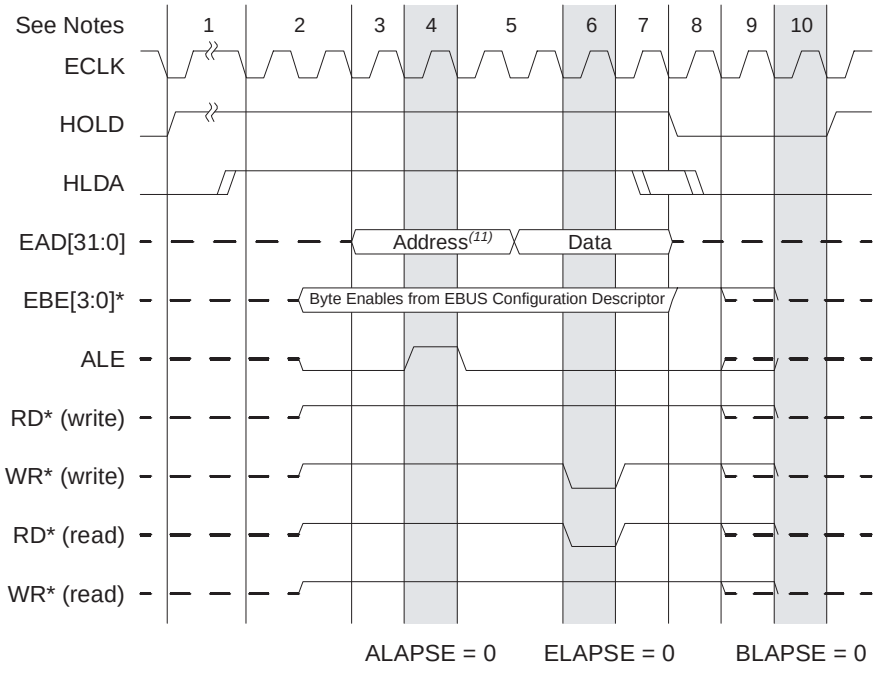


Figure 10-6. EBUS Input Timing Waveform



10.2.4 EBUS Arbitration Timing Specification

Figure 10-7. EBUS Write/Read Cycle, Intel-Style

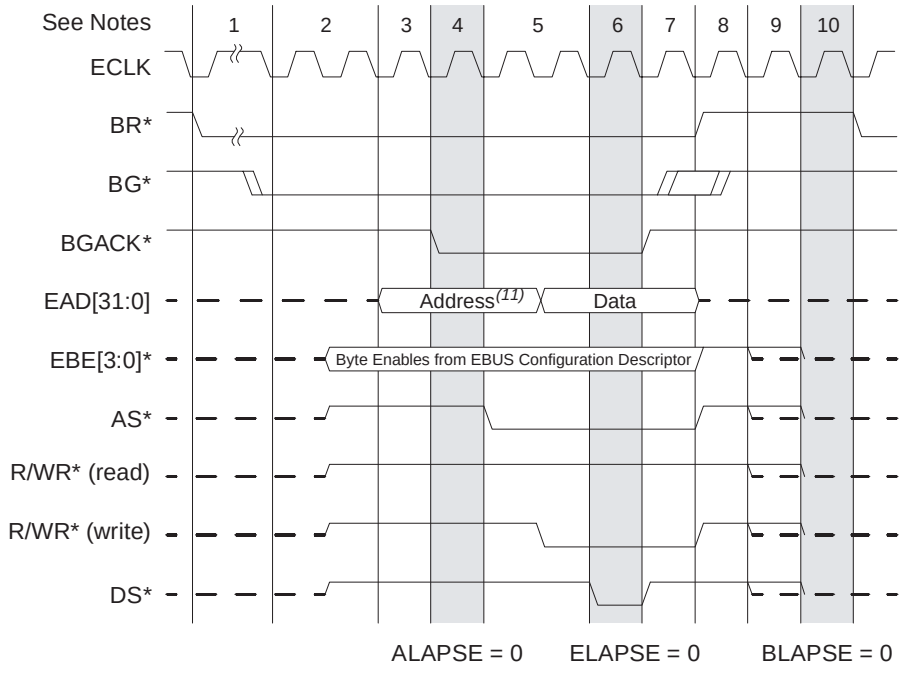


GENERAL NOTE:

1. HLDA assertion depends on the external bus arbiter. While HOLD and HLDA are both deasserted, MUSYCC places shared EBUS signals in high impedance (three-state, shown as dashed lines).
2. One ECLK cycle after HLDA assertion, MUSYCC outputs valid command bus signals: EBE, ALE, RD*, and WR*.
3. Two ECLK cycles after HLDA assertion, MUSYCC outputs valid EAD address signals.
4. ALE assertion occurs 3 ECLK cycles after HOLD and HLDA are both asserted. ALAPSE inserts a variable number of ECLK cycles to extend ALE high pulse width and EAD address interval.
5. EAD address remains valid for one ECLK cycle after ALE falling edge. During a write transaction, MUSYCC outputs valid EAD write data one ECLK prior to WR* assertion. During a read transaction, EAD data lines are inputs.
6. ELAPSE inserts a variable number of ECLK cycles to extend RD*/WR* low pulse width and EAD data intervals. Read data inputs are sampled on ECLK rising edge coincident with RD* deassertion.
7. EAD write data and EBE byte enables remain valid for one ECLK cycle after RD*/WR* deassertion.
8. One ECLK after RD* or WR* deassertion, HOLD is deasserted and the bus is parked (command bus deasserted, EAD three-state). The bus parked state ends when HLDA is deasserted.
9. Command bus is unparked (three-stated) one ECLK after HLDA deassertion; two different unpark phases are shown, indicating the dependence on HLDA deassertion. If HLDA remained asserted until the next bus request, then command bus remains parked until one ECLK cycle following the next HOLD assertion. Caution: Whenever HLDA is deasserted, all shared EBUS signals are forced to three-state after one ECLK cycle, regardless of whether the EBUS transaction was completed. MUSYCC does not reissue or repeat such an aborted transaction.
10. BLAPSE inserts a variable number of ECLK cycles to extend HOLD deassertion interval until the next bus request.
11. The address line A31 must be asserted in all transactions.

500052_067

Figure 10-8. EBUS Write/Read Cycle, Motorola-Style



GENERAL NOTE:

1. BG* assertion depends on the external bus arbiter. While BG* and BR* are both deasserted, MUSYCC places shared EBUS signals in high impedance (three-state, shown as dashed lines).
2. One ECLK cycle after BG* assertion, MUSYCC outputs valid command bus signals: EBE, AS*, R/WR*, and DS*.
3. Two ECLK cycles after BG* assertion, MUSYCC outputs valid EAD address signals. BGACK* assertion occurs three ECLK cycles after BG* and BR* are both asserted.
4. ALAPSE inserts a variable number of ECLK cycles to extend AS* high pulse width and EAD address interval.
5. EAD address remains valid for one ECLK cycle after AS* falling edge. During a write transaction, MUSYCC asserts R/WR* and outputs valid EAD write data one ECLK prior to DS* assertion. During a read transaction, EAD data lines are inputs.
6. ELAPSE inserts a variable number of ECLK cycles to extend DS* low pulse width and EAD data interval. Read data inputs are sampled on ECLK rising edge coincident with DS* deassertion.
7. EAD write data, EBE, R/WR*, and AS* signals remain valid for one ECLK cycle after BGACK* and DS* are deasserted.
8. One ECLK after BGACK* deassertion, the BR* output is deasserted and the bus is parked (command bus deasserted, EAD three-state). The bus parked state ends when the external bus arbiter deasserts BG*.
9. Command bus is unparked (three-stated) one ECLK after BG* deassertion; two different unpark phases are shown, indicating the dependence on BG* deassertion. If BG* remained asserted until the next bus request, then command bus remains parked until one ECLK cycle following the next BR* assertion. Caution: Whenever BG* is deasserted, all shared EBUS signals are forced to three-state after one ECLK cycle, regardless of whether the EBUS transaction was completed. MUSYCC does not reissue or repeat such an aborted transaction.
10. BLAPSE inserts a variable number of ECLK cycles to extend BR* deassertion interval until the next bus request.
11. The address line A31 must be asserted in all transactions.

500052_068

10.2.5 Serial Interface Timing and Switching Characteristics

Table 10-12. Serial Interface Clock (RCLK, TCLK) Parameters

Symbol	Parameter	Min	Max	Units
F_c	Clock Frequency	DC	52	MHz
T_r	Clock Rise Time for non-HSSI ports (6–31)	—	20	ns
	Clock Rise Time for HSSI ports (0–5)	—	3	ns
T_f	Clock Fall Time for non-HSSI ports (6–31)	—	20	ns
	Clock Fall Time for HSSI ports (0–5)	—	3	ns
—	Clock Duty Cycle	40	60	%

Figure 10-9. Serial Interface Clock (RCLK,TCLK) Waveform

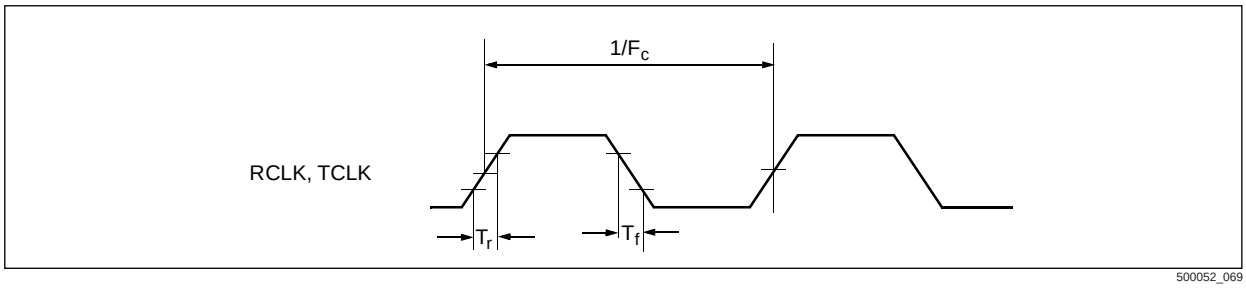


Table 10-13. Serial Interface Input/Output Timing Parameters

Symbol	Parameter	Min	Max	Units
T^{2}_{val}	Clock to Signal Valid Delay for non-HSSI ports (6–31)	2	30	ns
	Clock to Signal Valid Delay for HSSI ports (0–5)	2	8	ns
T^{3}_{ds}	Data Setup Time for non-HSSI ports (6–31)	15	—	ns
	Data Setup Time for HSSI ports (0–5)	3	—	ns
T^{3}_{dh}	Data Hold Time for non-HSSI ports (6–31)	15	—	ns
	Data Hold Time for HSSI ports (0–5)	4	—	ns

GENERAL NOTE:
1. Parameters were characterized with C load = 70 pF
2. Output Delay
3. Input Signals

Table 10-14. Serial Interface Input/Output Measure Conditions

Symbol	Parameter	Value	Units
V_{th}	Voltage Threshold High(1)	0.6 VDD_io	V
V_{tl}	Voltage Threshold Low(1)	0.2 VDD_io	V
V_{test}	Voltage Test Point	0.4 VDD_io	V
V_{max}	Maximum Peak-to-Peak(2)	0.4 VDD_io	V
—	Input Signal Slew Rate	1.5	V/ns
C_{ld}	Maximum Load capacitance–output and I/0	70	pF

FOOTNOTE:

(1) The input test for the 3.3 V environment is done with 0.1*VDD_io of overdrive. Timing parameters must be met with no more overdrive than this. Production testing may use different voltage values, but must correlate results back to these parameters.

(2) V_{max} specifies the maximum peak-to-peak voltage waveform allowed for measuring input timing. Production testing may use different voltage values, but must correlate results back to these parameters.

Figure 10-10.Serial Interface Data Input Waveform

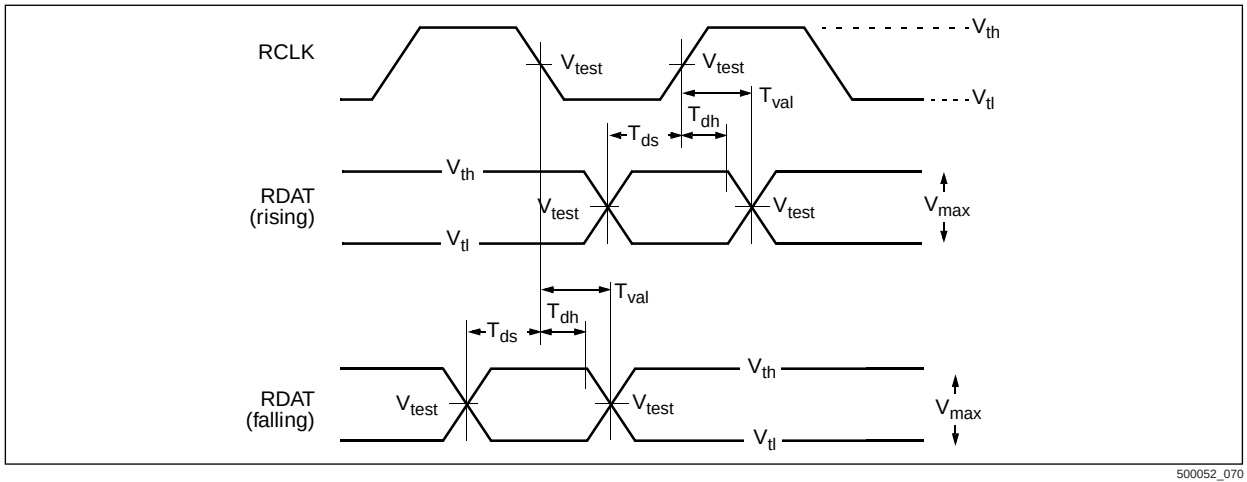


Figure 10-11.Serial Interface Data Delay Output Waveform

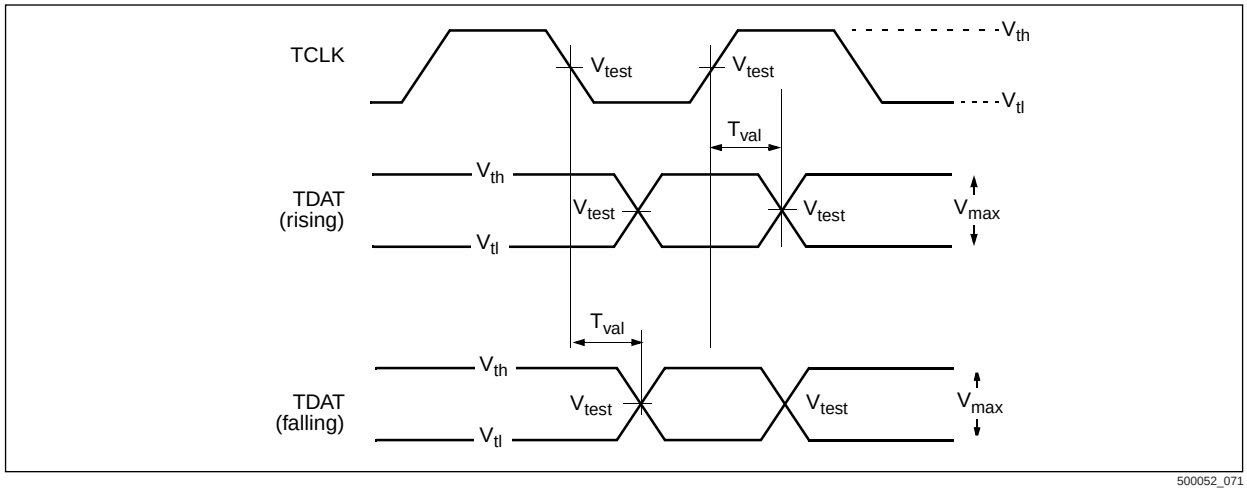
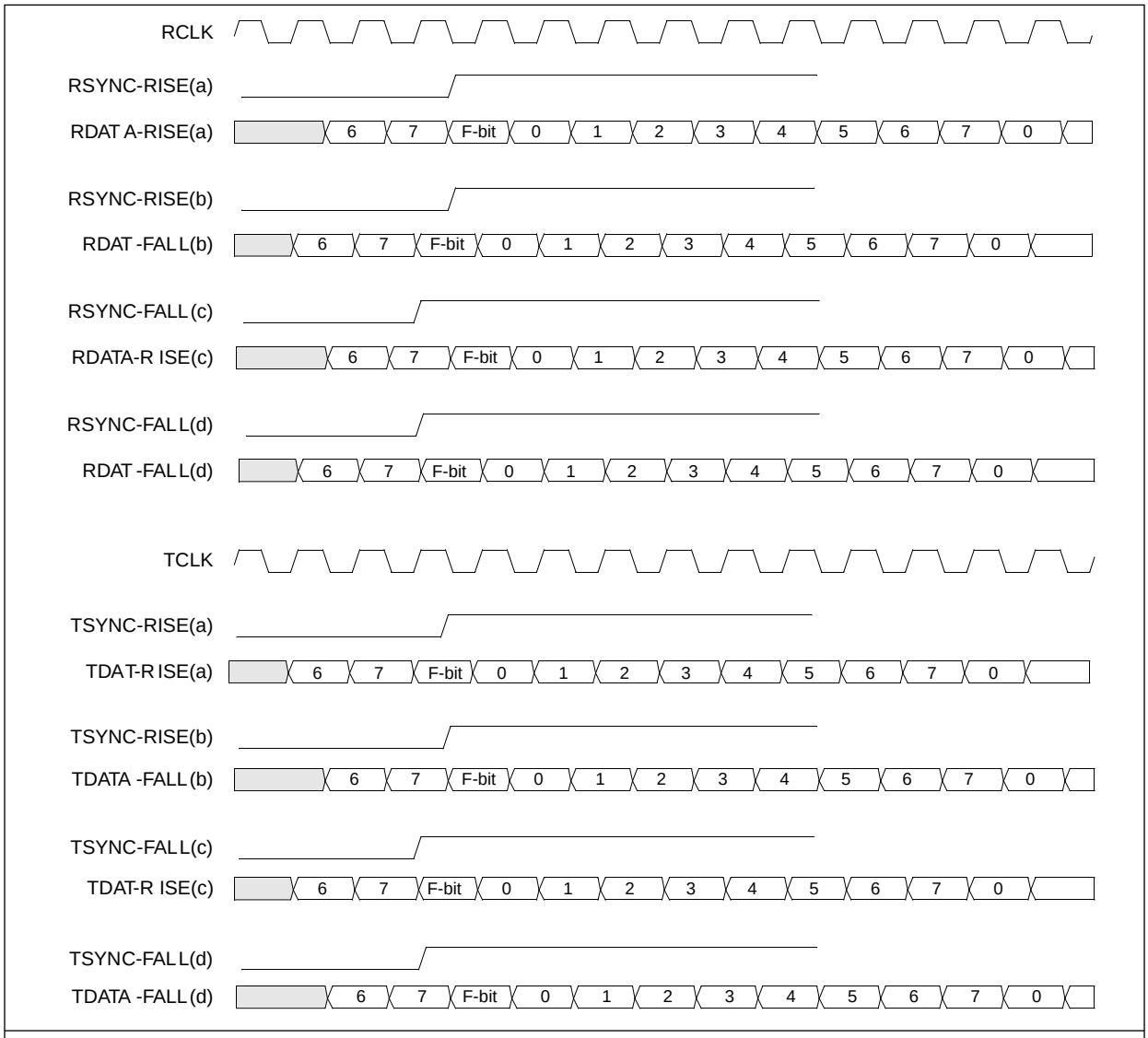


Figure 10-12. Transmit and Receive T1 Mode

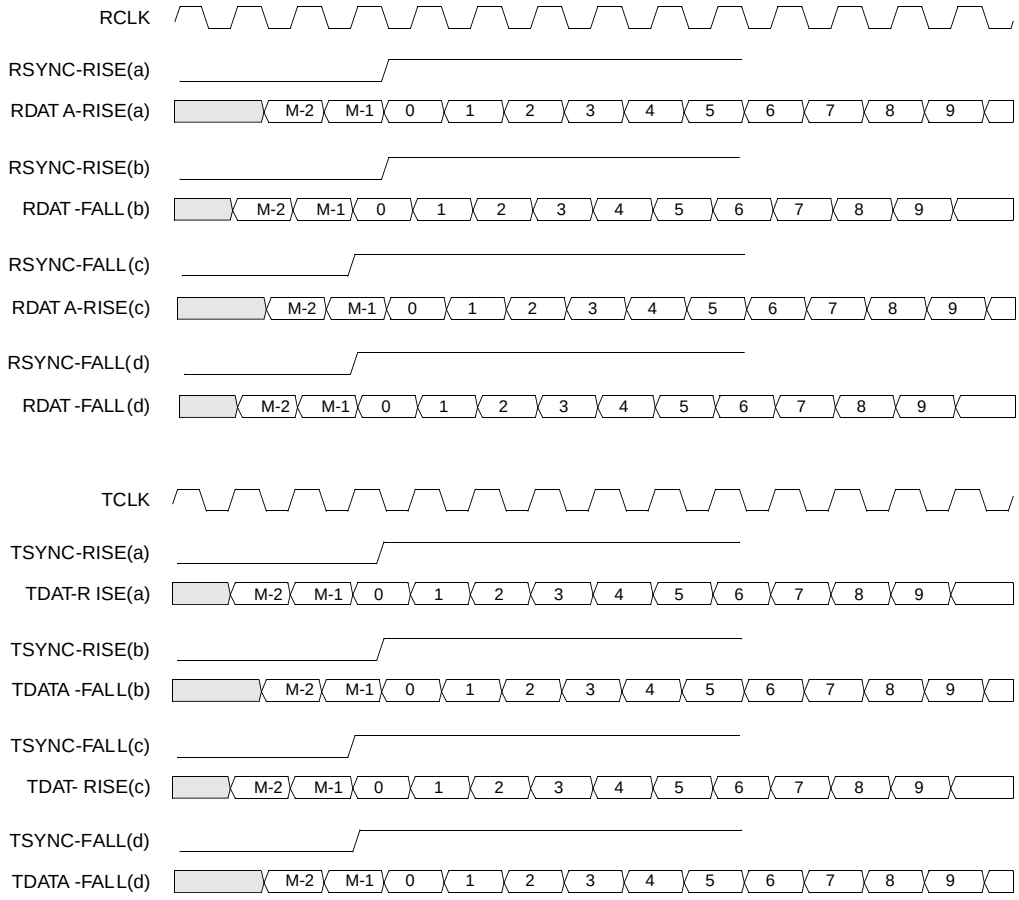


GENERAL NOTE:

1. T1 Mode employs 24 time slots (0–23) with 8 bits per time slot (0–7) and 1 Frame-bit every 193 clock periods. One frame of 193 bits occurs every 125 µs (1.544 MHz).
2. RSYNC and TSYNC must be asserted for a minimum of 1 CLK period.
3. CX28500 can be configured to sample RSYNC, TSYNC, RDAT, and TDAT on either a rising or falling clock edge independently of any other signal sampling configuration.
4. Relationships between the various configurations of active edges for the synchronization signal and the data signal are shown using a common clock signal for receive and transmit operations. Note the relationship between the frame bit (within RDAT, TDAT) and the frame synchronization signal (e.g., RSYNC, TSYNC).
5. All received signals (e.g., RSYNC, RDAT, TSYNC) are “sampled” in on the specified clock edge (e.g., RCLK, TCLK). All transmit data signals (TDAT) are latched on the specified clock edge.
6. In configuration (a), synchronization and data signals are sampled/latched on a rising clock edge.
7. In configuration (b), synchronization signal is sampled on a rising clock edge and the data signal is sampled/latched on a falling clock edge.
8. In configuration (c), synchronization signal is sampled on a falling clock edge and the data signal is sampled/latched on a rising clock edge.
9. In configuration (d), synchronization and data signals are sampled/latched on a falling clock edge.

500052_072

Figure 10-13. Transmit and Receive Channelized non T1 (i.e., N x 64) Mode



LEGEND: M = Nx8 bits, where M = number of time slots.

GENERAL NOTE:

1. E1 Mode employs 32 time slots (0–31) with 8 bits per time slot (0–7) and 256 bits per frame and one frame every 125 µs (2.048 MHz).
2. 2xE1 Mode employs 64 time slots (0–63) with 8 bits per time slot (0–7) and 512 bits per frame and one frame every 125 µs (4.096 MHz).
3. 4xE1 Mode employs 128 time slots (0–127) with 8 bits per time slot (0–7) and 1024 bits per frame and one frame every 125 µs (8.192 MHz).
4. RSYNC and TSYNC must be asserted for a minimum of 1 CLK period.
5. CX28500 can be configured to sample RSYNC, TSYNC, RDAT, and TDAT on either a rising or falling clock edge independently of any other signal sampling configuration.
6. Relationships between the various configurations of active edges for the synchronization signal and the data signal are shown using a common clock signal for receive and transmit operations. Note the relationship between the frame bit (within RDAT, TDAT) and the frame synchronization signal (e.g. RSYNC, TSYNC).
7. All received signals (e.g., RSYNC, RDAT, TSYNC) are sampled in on the specified clock edge (e.g. RCLK, TCLK). All transmit data signals (TDAT) are latched on the specified clock edge.
8. In configuration (a), synchronization and data signals are sampled/latched on a rising clock edge.
9. In configuration (b), synchronization signal is sampled on a rising clock edge and the data signal is sampled/latched on a falling clock edge.
10. In configuration (c), synchronization signal is sampled on a falling clock edge and the data signal is sampled/latched on a rising clock edge.
11. In configuration (d), synchronization and data signals are sampled/latched on a falling clock edge.

500052_073

10.2.6 Test and Diagnostic Interface Timing

Table 10-15. Test and Diagnostic Interface Timing Requirements

Symbol	Parameter	Minimum	Maximum	Units
1	TCK Pulse-Width High	80	—	ns
2	TCK Pulse-Width Low	80	—	ns
3	TMS, TDI Setup Prior to TCK Rising Edge(1)	15	—	ns
4	TMS, TDI Hold after TCK High(1)	20	—	ns

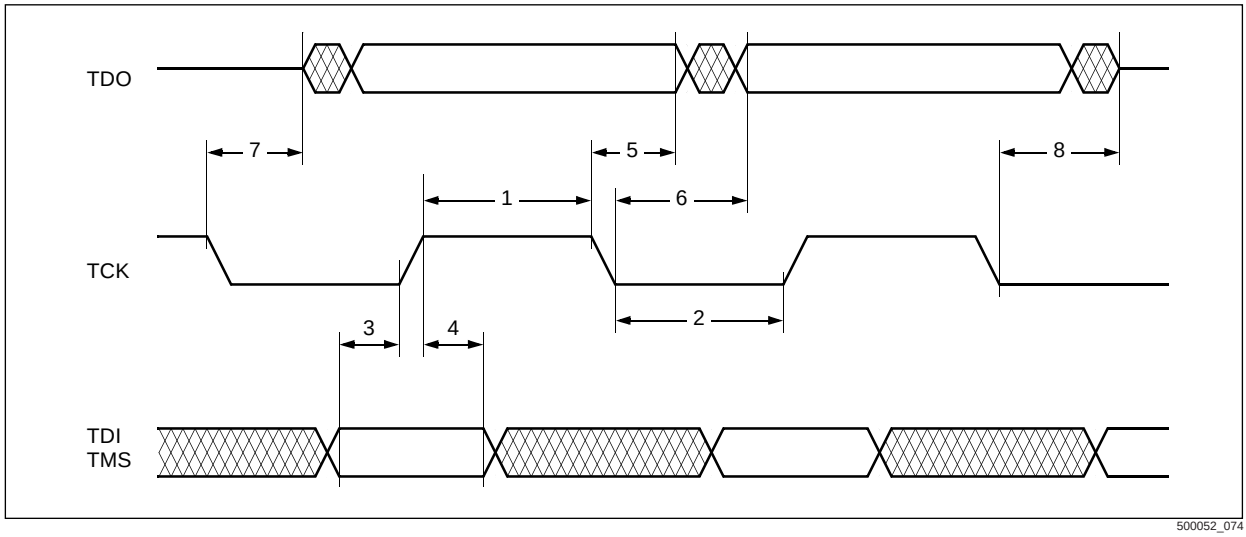
FOOTNOTE:
(1) Also applies to functional inputs for SAMPLE/PRELOAD and EXTEST instructions.

Table 10-16. Test and Diagnostic Interface Switching Characteristics

Symbol	Parameter	Minimum	Maximum	Units
5	TDO Hold after TCK Falling Edge	0	—	ns
6	TDO Delay after TCK Low	—	50	ns
7	TDO Enable (Low Z) after TCK Falling Edge	2	15	ns
8	TDO Disable (High Z) after TCK Low	—	25	ns

GENERAL NOTE: Also applies to functional outputs for the EXTEST instruction.

Figure 10-14.JTAG Interface Timing



GENERAL NOTE: Please refer to Tables 10-14 and 10-15 for numerical symbol reference.

10.3 Package Thermal Specification

Table 10-17. CX28500 Package Thermal Resistance Characteristics

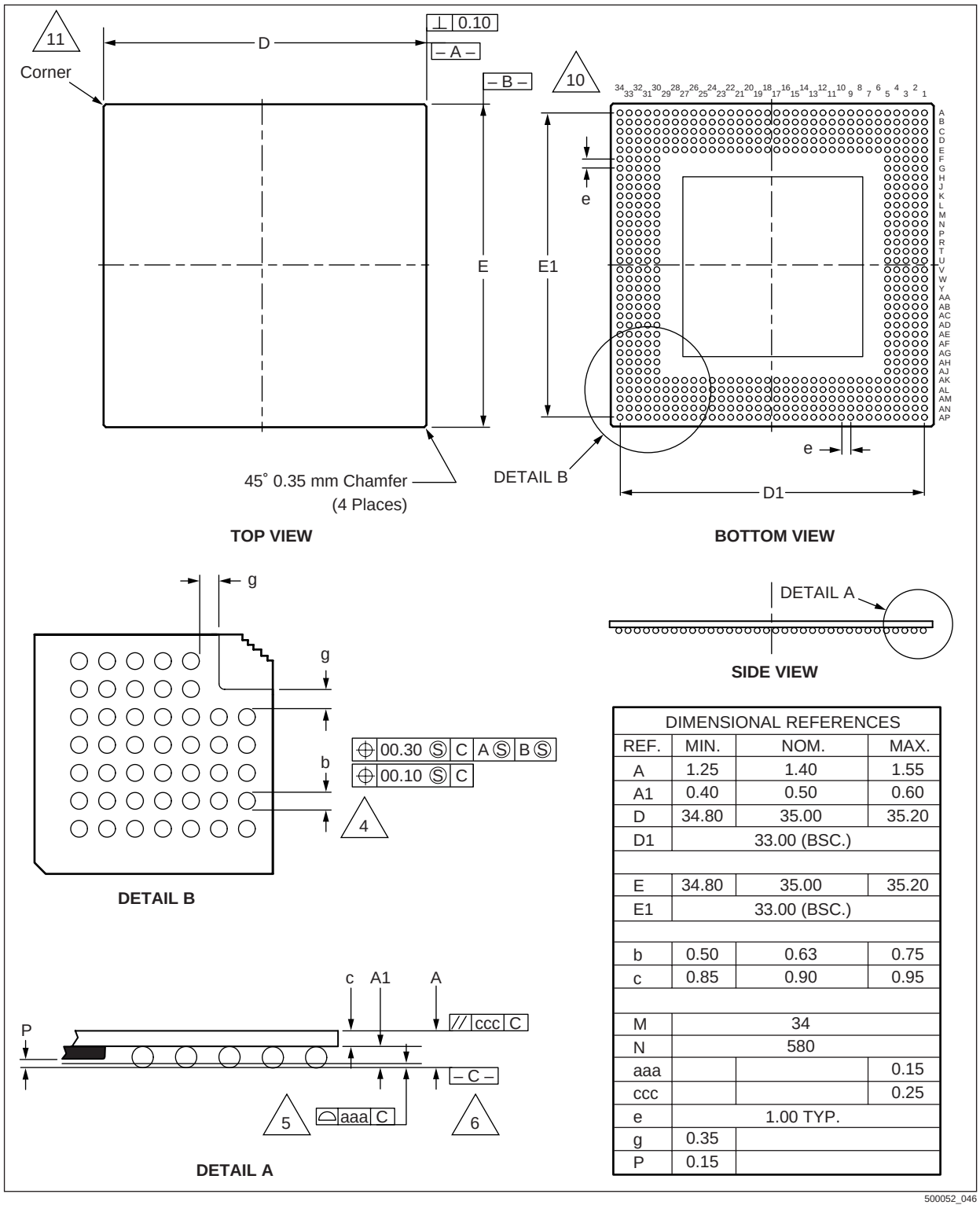
Package	Mounting Conditions	Airflow-LFM(1) (LMS(2))	Thermal Resistance (junction to ambient) = **C/W	Max. Power(3)	
				Tac = 85 °C	Tac = 70 °C
580-pin TBGA	Board-Mounted	0	10	4 W	5.5 W
		200 (1.02)	8.7	4.5 W	6.3 W
		500 (2.54)	7.5	5.3 W	7.3 W

GENERAL NOTE:

1. LFM-linear feet per minute.
2. LMS-linear meters per second.
3. Junction to case temperature (°C): Tjc =Tac + (θja x Pd).
   Tjc = θja x Pd (measured) + Tac (measured)
   Where Tjc = Junction Temperature (see Table 10-1)
   θja = Thermal Resistance
   Tac = Ambient Case Temperature (see Table 10-2)
   Pd = Power Dissipation = VDD_c x IDD (see Table 10-1)

10.4 Mechanical Specification

Figure 10-15.580-Pin BGA Package Diagram



DIMENSIONAL REFERENCES			
REF.	MIN.	NOM.	MAX.
A	1.25	1.40	1.55
A1	0.40	0.50	0.60
D	34.80	35.00	35.20
D1		33.00 (BSC.)	
E	34.80	35.00	35.20
E1		33.00 (BSC.)	
b	0.50	0.63	0.75
c	0.85	0.90	0.95
M		34	
N		580	
aaa			0.15
ccc			0.25
e		1.00 TYP.	
g	0.35		
P	0.15		

500052_046

A CX28500 PCI Bus Latency and Utilization Analysis

A.1 Objective

To check whether CX28500 internal FIFOs can withstand the time delays to their being serviced that are caused by a combination of PCI latency and other channels requesting service without experiencing an underflow or overflow. Further to analyze PCI bus utilization.

A.2 Definitions

A transmitter Underflow (TxBUFF) is defined as the condition that exists when an output FIFO for a specific channel is emptied before transmission of a complete HDLC frame.

A receiver Overflow (RxBUFF) is caused when CX28500 does not service a channel in time, which in turn is caused by either excessive PCI bus latency or other channel demands on the PCI. An Overflow is defined as the condition that exists when the input FIFO for a specific channel is completely full and that channel receives more input data.

MaxData is the maximum number of data cycles that can be transferred across the PCI bus during one PCI transaction.

NumCH is the number of channels configured.

A PCI Data Transaction is defined as a PCI transaction that involves a read or write burst of message data to or from the Host memory. This does not include a read or write burst of a Buffer Descriptor (BD) nor of status bits.

f_{pci} is the PCI clock rate in Hz.

f_{ch} is the internal channel bit rate in bits per second. This value takes into account the overhead caused by HDLC framing and of storing the status in the SLP (Serial Line Processor) FIFO.

The threshold of a receive channel RSLP FIFO (Thr_{rx}) is the amount of data in bits that, when the amount of data in the FIFO crosses this level, causes the channel to request service from the DMA (Direct Memory Access) controller.

The threshold of a transmit channel TSLP FIFO (Thr_{tx}) is the amount of data in bits that, when the amount of data in the FIFO crosses this level, causes the channel to request service from the DMA.

The BuffLen (BUFFLEN) of a channel is the internal FIFO length allocated to that channel for use between the SLP and the DMA.

A.3 Assumptions/Modes of Operation

- Each servicing of a channel is initiated by its channel number reaching the front of the Service Queue and is terminated after precisely one PCI transaction regardless of whether that transaction moves data or buffer descriptor overhead. If a channel requires further servicing, the DMA Controller automatically replaces its request at the back of the Service Queue.
- The DMA Aging-Period is not relevant to PCI latency when considering the worst case scenario.
- An internal, fairly weighted, round-robin scheme is used to decide whether the next PCI transaction is receive-based or transmit-based.
- Regardless of whether the PCI bus works in 32-bit or 64-bit mode, MaxData has a maximum value of 32 dwords (to limit the time of any one data transaction).
- The PCI clock frequency is either 33 or 66 MHz.
- The Host causes no bottleneck in the operation of CX28500 and Host access latency (TRDY delay) is considered to be zero. Hence, this model also assumes that no PCI read transaction is ever retried due to target unavailability and the PCI bridge FIFO is large enough to accept continuous CX28500 transactions without becoming full.
- An unchannelized port is considered for the purpose of calculations to be one channel. For the case of a TSBUS interface, each Virtual Serial Port (VSP) is considered to be one channel.
- Each packet is contained in one buffer in the Host memory. If this is not the case, extra overhead per packet is associated with buffer descriptor management.
- Interrupt service is not taken into consideration since using status for every BD is more demanding on the PCI.

A.4 PCI Transaction Timing

Each PCI transaction is accompanied by an associated overhead. The two types of PCI transaction and their associated overheads are outlined below.

A.4.1 Read

A read transaction of X dwords of data including a Host access latency of r cycles takes (3 + X + r) cycles for a 32-bit mode PCI or (3 + [(X/2)] + r) cycles for a 64-bit mode PCI. These include an address cycle and two bus turnaround cycles.

NOTE: With usage of the fast back-to-back feature of the PCI, the number of turn around cycles can be reduced. However in the following calculations the worst case has been considered so the maximum number of cycles has been used.

A.4.2 Write

A write transaction of X dwords of data, including a Host access latency of w cycles, takes (2 + X + w) cycles for a 32 bit mode PCI or (2 + [(X/2)] + w) cycles for a 64-bit mode PCI. These include one address cycle and one bus turnaround cycle.

NOTE: With usage of the fast back-to-back feature of the PCI, the number of turn around cycles can be reduced. However in the following calculations the worst case has been considered so the maximum number of cycles has been used.

A.5 Receive Messages

When dealing with received messages, the full set of PCI transactions processed by the DMA controller, per channel, is as follows:

- Read BD: CX28500 performs a burst read of 2 dwords from Host memory. This transaction takes (3 + 2 + r) cycles during 32-bit mode, or (3 + 1 + r) cycles during 64-bit mode.
- Data Transfer: Frame information is written to Host memory until either the end of the memory buffer, the end of the message, or the PCI bus is lost. In the general case, this transaction takes (2 + X + w) cycles for 32-bit mode or (2 + [(X/2)] + w) cycles for 64-bit mode. Where X is the number of dwords transferred. The longest possible value of this interval is the remaining length of the PCI latency timer or MaxData cycles. Hence, the transaction may take (2 + MaxData + w).
- Write Buffer Status: This transaction takes (2 + 1 + w) cycles regardless of the 32- or 64-bit PCI mode, if the ECCMODE bit of the Global Configuration Register is clear (i.e., 0). When ECCMODE is set to 1, this transaction takes (2 + 2 + w) cycles in 32-bit PCI mode, and (2 + 1 + w) cycles in 64-bit PCI mode. This chapter was written with the former in mind (i.e., this transaction is always calculated as (2 + 1 + w) cycles, as this is the prevalent usage mode of the device).

A.6 Transmit Messages

Considering the transmit data path, the full set of PCI transactions processed by the DMA controller, per channel, is as follows:

- Read BD: CX28500 performs a burst read of 2 dwords from Host memory. This transaction takes (3 + 2 + r) cycles during 32-bit mode, or (3 + 1 + r) cycles during 64-bit mode.
- Data Transfer: Frame information is read from Host memory to CX28500 until either the end of the memory buffer, the end of the message, or the PCI bus is lost. In the general case, this transaction takes (3 + X + r) cycles for 32-bit mode or (3 + [(X/2)] + r) cycles for 64-bit mode. Where X is the number of dwords transferred. The largest value is the remaining length of the PCI latency timer or MaxData cycles. Hence, the transaction may take (3 + MaxData + r).
- Write Buffer Status: This transaction takes (2 + 1 + w) cycles regardless of the 32- or 64-bit PCI mode.

A.7 Allocation of Internal SLP Buffer (FIFO) Space

The total internal buffer space for SLP usage is 32 KB in each direction—total of 64 KB full-duplex. In general, this memory should be allocated in ratio to the channels' bit rate. For example, one logical channel (unchannelized) working at 52 Mbps should be allocated a buffer approximately five times the size of one logical channel working at 10 Mbps.

A.8 Maximum Feasible PCI Latency

The Maximum Feasible PCI Latency for a given receive channel (max $L_{pci\text{-}rx}$) is defined as the maximum length of time in seconds that a receive channel must wait before the first data transaction is performed from its internal SLP buffer. This can be considered as the amount of time required to service all the transmit channels plus the amount of time required to service all but one of the receive channels. This involves updating status and reading new buffer descriptors for all channels in both directions, transferring data from the Host memory for all transmit channels, and transferring data to the Host memory from the SLP internal buffer for all but one of the receive channels.

Hence, max $L_{pci\text{-}rx}$ can be represented by the equations:

$$\frac{1}{f_{pci}}\left\{\sum_{NumCH}(2 \bullet Read\ BD + Read\ DATA + 2 \bullet Write\ STATUS) + \sum_{NumCH-1}(Write\ DATA)\right\}$$

or,

$$\frac{1}{f_{pci}}\left\{\sum_{NumCH}\left(6 + 2 \bullet w + 2 \bullet \left(3 + \left(\frac{64}{PCI\ Mode}\right) + r\right) + 3 + DATA + r\right) + \sum_{NumCH-1}(2 + DATA + w)\right\}$$

Where DATA is the maximum amount, in dwords, that can be transferred from a channel during one PCI transaction. PCI Mode is the number of bits transferred by the PCI in one clock.

The Maximum Feasible PCI Latency for a given transmit channel (max $L_{pci\text{-}tx}$) is defined to be the maximum length of time in seconds that a transmit channel must wait until its first data transaction. This can be considered as the amount of time required to service all the receive channels plus the amount of time required to service all but one of the transmit channels. This involves updating status and reading new buffer descriptors for all channels in both directions, transferring data to the Host memory for all receive channels, and transferring data from the Host memory to the SLP internal buffer for all but one of the transmit channels.

Hence max $L_{pci\text{-}tx}$ can be represented by the equations:

$$\frac{1}{f_{ocu}}\left\{\sum_{NumCH}(2 \bullet Read\ BD + Write\ DATA + 2 \bullet Write\ STATUS) + \sum_{Num\ CH-1}(Read\ DATA)\right\}$$

or,

$$\frac{1}{f_{pci}}\left\{\sum_{NumCH}\left(2 \bullet \left(3 + \left(\frac{64}{PCIMode}\right) + r\right) + 2 + DATA + 6 + 3 \bullet w\right) + \sum_{NumCH-1}(3 + DATA + r)\right\}$$

Where DATA is the maximum amount, in dwords, that can be transferred from a channel during one PCI transaction. PCI Mode is the number of bits transferred by the PCI in one clock.

NOTE: Both of these maximum feasible PCI latency times are internal latencies and do not include any external influences such as PCI arbitration.

A.9 Maximum Endurable Latency

The Maximum Endurable Latency of a channel (L_{ch}) is the amount of time that a specific channel can endure until the first data transaction.

The timing of the Endurable Latency for a given receive channel ($L_{ch\text{-}rx}$) starts when that channel first requests DMA service. The interval ends on the clock that the first dword of data is removed from the channel's SLP buffer by the DMA for transferal to Host memory, or at worst when its buffer is full.

The timing of the Endurable Latency for a given transmit channel ($L_{ch\text{-}tx}$) starts when that channel first requests DMA service (i.e., when the FIFO empties to threshold level). The interval ends on the clock that the first dword of data is moved from Host memory and is transferred to that channel's SLP buffer by the DMA, or at worst when its buffer is empty.

$$L_{ch\text{-}rx} = \frac{1}{f_{ch}}\ (BuffLen - Thr_{rx})$$

$$L_{ch\text{-}tx} = \frac{1}{f_{ch}}\ (BuffLen - Thr_{tx})$$

If Endurable Latency of a channel is less than Maximum Feasible PCI Latency in both the receive and transmit directions, no overflow or underflow occurs.

A.10 PCI Bus Utilization

PCI bus utilization is defined to be the ratio of the amount of time CX28500 uses the bus to the total amount of time that could be utilized by all components on the bus, including the Host. Utilization is calculated by comparing the time required to transfer one bit of receive and one bit of transmit information across the PCI and the time required to fill one bit of internal buffer space.

The time required per bit to transfer across the PCI (Y_{bit}) is calculated as an average over one packet, by dividing the amount of time required for one packet's transactions by the number of bits in the packet. Giving the average time to transfer one bit:

$$Time\ for\ a\ bit\ of\ RX\ Data = \frac{1}{P \bullet f_{pci}}\{Read\ BD + Write\ Packet\ Data + Write\ Status\}$$

Where P is packet length, measured in bits. Hence $Y_{bit\text{-}rx}$ can be represented by:

$$\frac{1}{P \bullet f_{pci}}\left\{3 + w + \left(3 + \left(\frac{64}{PCI\ Mode}\right) + r\right) + \left(\frac{P}{PCI\ Mode}\right) + (2 + w) \bullet \left(ROUNDUP\left(\frac{P}{MIN(TH(R, Z, P))}\right)\right)\right\}$$

Z is the amount of data transferred in 32 PCI cycles. ROUNDUP is a simple rounding up function. P is packet length in bits. PCI Mode is the number of bits transferred by one PCI clock and THR is the FIFO threshold for that channel.

Similarly for the transmit direction:

$$Time\ for\ a\ bit\ of\ TX\ Data = \frac{1}{P \bullet f_{pci}}\{Read\ BD + Read\ Packet\ Data + Write\ Status\}$$

Where P is packet length, in bits. Hence $Y_{bit\text{-}tx}$ can be represented by:

$$\frac{1}{P \bullet f_{pci}}\left\{3 + w + \left(3 + \left(\frac{64}{PCI\ Mode}\right) + r\right) + \left(\frac{P}{PCI\ Mode}\right) + (3 + r) \bullet \left(ROUNDUP\left(\frac{P}{MIN(TH(R, Z, P))}\right)\right)\right\}$$

Z is the amount of data transferred in 32 PCI cycles. ROUNDUP is a simple rounding up function. P is packet length in bits. PCI Mode is the number of bits transferred by one PCI clock, and THR is the FIFO threshold for that channel.

So the utilization, U, of the PCI bus due to one CX28500 is:

$$U = \sum_{NumCh} \frac{Amount\ time\ to\ transfer\ one\ bit\ across\ the\ PCI}{Amount\ time\ to\ transfer\ one\ bit\ of\ RX\ and\ TX\ in/out\ of\ (CX28500)}$$

$$U = \sum_{NumCh} (Y_{bit-rx} \bullet f_{ch-rx} + Y_{bit-tx} \bullet f_{ch-tx})$$

NOTE: To avoid overflow or underflow of an internal SLP buffer, U must be less than one and Maximum Feasible PCI Latency of each channel must be less than the Maximum Endurable Latency of that channel.

A.11 Maximum Tolerable Delay

The filling of an SLP buffer can be divided into 3 sections:

1. Up to the point where the amount of data crosses the threshold;
2. The amount of data filled in max L_{pci};
3. The amount of time left to fill the rest of the buffer.

Diagrammatically:

- Spare
- Amount filled in max L_{pci}
- Threshold Data

In the case of the receive direction, the amount of time that an SLP internal buffer takes to fill the Spare space in the buffer is the Maximum Tolerable Delay. This time period is usable by the Host (or other component) once per number of extra cycles it takes to completely empty the amount of data filled in that time. Hence if the channel buffer is exactly in data equilibrium, this spare can only be used once until the same amount of cycles has been returned to CX28500.

In the case of the transmit direction, the amount of time that an SLP internal buffer takes to empty the Spare space in the buffer is the Maximum Tolerable Delay. This time period is usable by the Host (or other component) once per number of extra cycles it takes to completely fill the amount of data emptied in that time. Hence if the channel buffer is exactly in data equilibrium, this spare can only be used once until the same amount of cycles has been "returned" to CX28500.

The value of the Maximum Tolerable Delay can be calculated and is represented by the following equation:

$$\max L_{ch} - \max L_{pci}$$

A.12 Other Considerations

1. An overflow occurs if one of the following is false:
   a. Utilization <1;
   b. $L_{pci} < L_{ch}$;
   c. Amount of data transferred > Amount of data transferred in/out of the PCI during L_{pci} CX28500 during the same time
2. PCI bus utilization allows an estimation of the number of CX28500 devices that can share one PCI bus. However, the relationship between utilization and the number of CX28500 devices is not linear.
3. The amount of data filled during the usage of spare cycles plus the amount of data already in the buffer can be used as the figure for the maximum allowable threshold.

A.13 Summary and Explanation of Terms Used in Calculations

For each device configuration, there are three sets of standard variables and two sets of statistics. One of the statistics sets includes a calculation of spare time which takes into account the Host using the Maximum Tolerable Delay, and afterwards allowing CX28500 to take the remaining bus time to achieve 100% utilization. The second statistics set does not include spare time and indicates average utilization.

Below, as in the full breakdown of figures, the explanation is in 5 sections:

1. Configuration–configurable variables;
2. Suggested–configurable variables that have suggested values;
3. Calculated–calculated variables common to both statistics sets;
4. Including Spare Time–calculations include Host /others usage of spare time;
5. Not Including Spare Time–calculations not including usage of spare time.

Table A-1. Configuration

Variable Name	Description	Calculated
Number of Ch	Number of active channels	Configurable
Mem Available (Kb)	Memory available (KB) for all channels	Configurable
Packet Length (Bits)	Length of packets, receive and transmit	Configurable
Ext. Ch Rate (Kbps)	Channel rate of serial lines (pure) without taking into account the HDLC overhead	Configurable
PCI Freq. (MHz)	PCI clock frequency in MHz	Configurable
Read Latency	Latency incurred by the PCI due to a read transaction	Configurable
Write Latency	Latency incurred by the PCI due to a write transaction	Configurable
PCI Bit Mode	Number of bits transferred in one PCI cycle	Configurable

Table A-2. Suggestions

Variable Name	Description	Calculated
Buffer Len (Bits)	Size of buffer available to each channel in each direction	Total of 32 bytes apportioned equally across Number of Channels
Rx Threshold	FIFO Threshold of RX channels	Set as half the buffer size
Tx Threshold	FIFO Threshold of TX channels	Set as half the buffer size

Table A-3. Calculated *(1 of 2)*

Variable Name	Description	Calculated
Int. Ch Rate (Kbps)—RX	Actual rate at which the internal SLP buffer fills taking into account HDLC framing and status bits.	From actual channels rate, packet length and HDLC overhead.
Int. Ch Rate (Kbps)—TX	Actual rate at which the internal SLP buffer empties taking into account HDLC framing.	From actual channels rate, packet length and HDLC overhead.
PCI Freq (kHz)	Frequency of the PCI in kHz.	From configured PCI frequency.
Time to read BD (average)	Average amount of time required to read a buffer descriptor.	Dependent on PCI bit mode and the value of PCI read latency.

Table A-3. Calculated *(2 of 2)*

Variable Name	Description	Calculated
Time to read BD (max)	Maximum amount of time required to read a buffer descriptor.	Dependent on PCI bit mode and the value of PCI read latency.
Time to update status	Number of PCI cycles required to update a packet status.	Dependent on PCI write latency only.

Table A-4. Including Spare Time

Variable Name	Description	Calculated
MAXDATA	Maximum amount of data that can be transferred across the PCI to or from the internal SLP buffer.	Calculated (by iteration) taking into account amount filled in spare time, amount filled in while a channel waits to be serviced, and the channel's threshold; separate for receive and transmit channels.
Max L_{ch} (ms) RX	Maximum amount of time a channel can endure without a single data transaction before an overflow occurs.	Calculated from threshold, buffer length, and internal channel rate.
Max L_{pci} (ms) RX	Maximum amount of time the DMA will take from the end of spare time until the first data transaction from a specific channel.	Calculated (by iteration) taking into account PCI bit mode, PCI bit rate, MAXDATA, and packet transactions of all channels.
Max L_{ch} (ms) TX	Maximum amount of time a channel can wait without a single data transaction before its internal SLP buffer is empty.	Calculated from threshold, buffer length, and internal channel rate.
Max L_{pci} (ms) TX	Maximum amount of time the DMA will take from the end of spare time until the first data transaction to that channel.	Calculated (by iteration) taking into account PCI bit mode, PCI bit rate, MAXDATA, and packet transactions of all channels.
Amount Data Filled in L_{pci} RX	Amount of data in bits that is filled into the SLP internal buffer during the maximum time the DMA takes to service a specific channel with a data transaction.	Calculated from channel rate and L_{pci}.
Amount Data Emptied in L_{pci} TX	Amount of data in bits that is transferred from the SLP internal buffer during the maximum time the DMA takes to service a specific channel with a data transaction.	Calculated from channel rate and L_{pci}.
Spare time (RX/TX)	Amount of time the (RX/TX) FIFO can endure before 100% PCI bus utilization.	Calculated from L_{pci} and L_{ch}
Spare time (Total)	Minimum spare time available after considering both RX/TX channels.	—
In spare time amount filled/ emptied	Self explanatory.	—
Max total in SLP	Maximum amount of data in SLP internal buffers at any one time.	Calculated from buffer length, threshold, amount filled in spare time, and amount filled in L_{pci}

Table A-5. Non –"Spare Time" Calculations

Variable Name	Description	Calculated
MAXDATA	Maximum amount of data that can be transferred across the PCI from the internal FIFO.	Calculated (by iteration) taking into account amount filled in while a channel waits to be serviced, and the channel's threshold; separate for receive and transmit channels.
Max L_{ch} (ms) RX	Maximum amount of time a channel can endure without a single data transaction before an overflow will occur.	Calculated from threshold, buffer length, and internal channel rate.
Max L_{pci} (ms) RX	Maximum amount of time the DMA will take from the end of spare time until the first data transaction from a specific channel.	Calculated (by iteration) taking into account PCI bit mode, PCI bit rate, MAXDATA, and packet transactions of all channels.
Max L_{ch} (ms) TX	Maximum amount of time a channel can wait without a single data transaction before its internal SLP buffer will be empty.	Calculated from threshold, buffer length, and internal channel rate.
Max L_{pci} (ms) TX	Maximum amount of time the DMA will take from the end of spare time until the first data transaction to that channel.	Calculated (by iteration) taking into account PCI bit mode, PCI bit rate, MAXDATA, and packet transactions of all channels.
Utilization–RX	Average ratio of Host to CX28500 RX utilization of the PCI.	Amount of time taken to transfer one bit of RX data out of CX28500 divided by the amount of time to transfer one bit of receive data to the Host memory.
Utilization–TOTAL	Average ratio of Host to CX28500 utilization of the PCI.	CX28500 Receive utilization plus CX28500 Transmit utilization.
Amnt data filled in $L_{pci\text{-}rx}$ RX	Amount of data added to receive internal SLP buffers during the maximum amount of time taken to reach it.	Calculated from internal channel rate and L_{pci}.
Amnt data filled in $L_{pci\text{-}tx}$ TX	Amount of data removed from transmit internal SLP buffers during the maximum amount of time taken to reach it.	Calculated from internal channel rate and L_{pci}.

A.14 Examples

Tables A-6 through A-8 show calculated results for four different types of channel configuration. Tables A-6 and A-7 show similar results for three types of configuration, where all channels within each type are configured exactly the same, per the column heading. The only difference between Tables A-6 and A-7 is the read latency (r value), which increases from zero to two. This highlights the impact of a small increase in PCI latency. Table A-8 gives results for a fourth system configuration that combines two different channel configurations (T1 payload and overhead) operating simultaneously.

Since Tables A-6 through A-8 are calculated for only one CX28500 device, total PCI system utilization for a two-device system is approximately double what is shown on the following pages.

A.15 Differences in the Combined T1 Payload and Overhead Table

Refer to Tables A-6 through Tables A-8.

1. Memory Available is apportioned according to the rate at which the bits arrive on that channel. Hence a combination of high-speed channels and low-speed channels yields a different amount of buffering allocated to channels in the third example.
2. Throughout the calculation, each column relates only to that specific set of channels with one exception. The Utilization ABS Total represents the overall utilization of the CX28500.

Table A-6. Example One *(1 of 2)*

66 MHz PCI		44-Byte Messages				12-Byte Messages	
		168 x 1536K Channels		6 x 44.21M Channels		168 x 4K Channels	
Configuration		**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**
Number of Ch	N	168	168	6	6	168	168
Mem Available (KB)	M	64	64	64	64	64	64
Packet Length (bits)	P	352	352	352	352	96	96
Ext. Ch Rate (Kbps)	R	1536	1536	44210	44210	4	4
PCI Freq. (MHz)	fpci	66	66	66	66	66	66
Read Latency	r	0	0	0	0	0	0
Write Latency	w	0	0	0	0	0	0
PCI Bit Mode	PBIT	64	32	64	32	64	32
Suggested							
Buffer Size	BUFFLEN	1592	1592	43723	43723	1592	1592
Rx Threshold	THR – RX	796	796	21861	21861	796	796
Tx Threshold	THR – TX	796	796	21861	21861	796	796
Calculated							
Int. Ch Rate (Kbps)	Fch – RX	1568.68	1568.68	45150.64	45150.64	4.27	4.27
Int. Ch Rate (Kbps)	Fch – TX	1437.96	1437.96	41388.09	41388.09	3.20	3.20
PCI Freq. (kHz)	fpci	66000	66000	66000	66000	66000	66000
Time to Read BD	Average	4	5	4	5	4	5
Time to Read BD	Actual	4	5	4	5	4	5
Time to Update Status	—	3	3	3	3	3	3
Including Spare Time							
MAXDATA	RX	352	352	352	352	96	96
MAXDATA	TX	352	352	352	352	96	96

Table A-6. Example One *(2 of 2)*

66 MHz PCI		44-Byte Messages				12-Byte Messages	
		168 x 1536K Channels		6 x 44.21M Channels		168 x 4K Channels	
Configuration		**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**
Max L_{ch} (ms)	RX	0.508	0.508	0.484	0.484	186.607	186.607
Max L_{pci} (ms)	RX	0.076	0.109	0.003	0.004	0.056	0.069
Max L_{ch} (ms)	TX	0.554	0.554	0.528	0.528	248.810	248.810
Max L_{pci} (ms)	TX	0.076	0.109	0.003	0.004	0.056	0.069
Amnt Data Filled in L_{pci}	RX	120	171	118	168	0	0
Amnt Data Emptied on L_{pci}	TX	110	157	108	153	0	0
Spare Time (ms)	RX	0.431	0.398	0.482	0.480	186.551	186.538
Spare Time (ms)	TX	0.477	0.444	0.526	0.525	248.754	248.741
Spare Time (ms)	TOTAL	0.431	0.398	0.482	0.480	186.551	186.538
Spare Time (cycles)	TOTAL	28466	26287	31783	31711	12312378	12311540
In Spare Time Amnt Filled	RX	677	625	21743	21694	796	796
In Spare Time Amnt Emptied	TX	620	573	19931	19886	597	597
Max Total in SLP Buffer	RX	1592	1592	43723	43723	1592	1592
Max Total in SLP Buffer	TX	1526	1526	41900	41900	1393	1393
Not Including Spare Time							
MAXDATA	RX	352	352	352	352	96	96
MAXDATA	TX	352	352	352	352	96	96
Max L_{ch} (ms)	RX	0.508	0.508	0.484	0.484	186.607	186.607
Max L_{pci} (ms)	RX	0.076	0.109	0.003	0.004	0.056	0.069
Max L_{ch} (ms)	TX	0.554	0.554	0.528	0.528	248.810	248.810
Max L_{pci} (ms)	TX	0.076	0.109	0.003	0.004	0.056	0.069
Utilization	RX	0.170	0.238	0.175	0.245	0.001	0.001
Utilization	TX	0.156	0.218	0.160	0.224	0.001	0.001
Utilization	**TOTAL**	**0.326**	**0.457**	**0.335**	**0.469**	**0.002**	**0.003**
Amnt Data Filled in L_{pci}	RX	120	171	118	168	0	0
Amnt Data Emptied in L_{pci}	TX	110	157	108	153	0	0

Table A-7. Example Two *(1 of 2)*

66 MHz PCI		44-Byte Messages				12-Byte Messages	
		168 x 1536K Channels		6 x 44.21M Channels		168 x 4K Channels	
Configuration		**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**
Number of Ch	N	168	168	6	6	168	168
Mem Available (KB)	M	64	64	64	64	64	64
Packet Length (bits)	P	352	352	352	352	96	96
Ext. Ch Rate (Kbps)	R	1536	1536	44210	44210	4	4
PCI Freq. (MHz)	fpci	66	66	66	66	66	66
Read Latency	r	2	2	2	2	2	2
Write Latency	w	0	0	0	0	0	0
PCI Bit Mode	PBIT	64	32	64	32	64	32
Suggested							
Buffer Size	BUFFLEN	1592	1592	43723	43723	1592	1592
Rx Threshold	THR – RX	796	796	21861	21861	796	796
Tx Threshold	THR – TX	796	796	21861	21861	796	796
Calculated							
Int. Ch Rate (Kbps)	Fch – RX	1568.68	1568.68	45150.64	45150.64	4.27	4.27
Int. Ch Rate (Kbps)	Fch – TX	1437.96	1437.96	41388.09	41388.09	3.20	3.20
PCI Freq. (kHz)	fpci	66000	66000	66000	66000	66000	66000
Time to Read BD	Average	6	7	6	7	6	7
Time to Read BD	Actual	6	7	6	7	6	7
Time to Update Status	—	3	3	3	3	3	3
Including Spare Time							
MAXDATA	RX	352	352	352	352	96	96
MAXDATA	TX	352	352	352	352	96	96
Max L_{ch} (ms)	RX	0.508	0.508	0.484	0.484	186.607	186.607
Max L_{pci} (ms)	RX	0.092	0.125	0.003	0.004	0.071	0.084
Max L_{ch} (ms)	TX	0.554	0.554	0.528	0.528	248.810	248.810
Max L_{pci} (ms)	TX	0.091	0.124	0.003	0.004	0.071	0.084
Amnt Data Filled in L_{pci}	RX	144	195	143	192	0	0
Amnt Data Emptied in L_{pci}	TX	132	179	129	174	0	0
Spare Time (ms)	RX	0.416	0.383	0.481	0.480	186.536	186.523
Spare Time (ms)	TX	0.462	0.429	0.525	0.524	248.738	248.726
Spare Time (ms)	TOTAL	0.416	0.383	0.481	0.480	186.536	186.523

Table A-7. Example Two *(2 of 2)*

66 MHz PCI		44-Byte Messages				12-Byte Messages	
		168 x 1536K Channels		6 x 44.21M Channels		168 x 4K Channels	
Configuration		**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**
Spare Time (cycles)	TOTAL	27458	25279	31747	31675	12311370	12310532
In Spare Time Amnt Filled	RX	653	601	21719	21669	796	796
In Spare Time Amnt Emptied	TX	598	551	19909	19863	597	597
Max Total in SLP Buffer	RX	1592	1592	43723	43723	1592	1592
Max Total in SLP Buffer	TX	1526	1526	41899	41899	1393	1393
Not Including Spare Time							
MAXDATA	RX	352	352	352	352	96	96
MAXDATA	TX	352	352	352	352	96	96
Max L_{ch} (ms)	RX	0.508	0.508	0.484	0.484	186.607	186.607
Max L_{pci} (ms)	RX	0.092	0.125	0.003	0.004	0.071	0.084
Max L_{ch} (ms)	TX	0.554	0.554	0.528	0.528	248.810	248.810
Max L_{pci} (ms)	TX	0.091	0.124	0.003	0.004	0.071	0.084
Utilization	RX	0.193	0.261	0.198	0.268	0.001	0.002
Utilization	TX	0.177	0.239	0.182	0.246	0.001	0.001
Utilization	**TOTAL**	**0.370**	**0.500**	**0.380**	**0.514**	**0.003**	**0.003**
Amnt Data Filled in	RX	144	195	143	192	0	0
Amnt Data Emptied in L_{pci}	TX	132	179	129	174	0	0

Table A-8. Example Three *(1 of 2)*

66 MHz PCI		44-Byte Messages				12-Byte Messages	
		168 x 1536K Channels		6 x 44.21M Channels		168 x 4K Channels	
Configuration		**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**
Number of Ch	N	168	168	—	—	—	—
Total Mem Available (KB)	—	64	64	—	—	—	—
Mem Available (KB)	M	63.83	0.17	—	—	—	—
Packet Length (bits)	R	352	96	—	—	—	—
Ext. Ch Rate (Kbps)	R	1536	4	—	—	—	—
PCI Freq. (MHz)	fpci	66	66	—	—	—	—
Read Latency	r	0	0	—	—	—	—
Write Latency	w	0	0	—	—	—	—
PCI Bit Mode	PBIT	64	64	—	—	—	—
Suggested							
Buffer Size	BUFFLEN	1588	36	—	—	—	—
Rx Threshold	THR – RX	794	18	—	—	—	—
Tx Threshold	THR – TX	794	18	—	—	—	—
Calculated							
Int. Ch Rate (Kbps)	Fch – RX	1569	4	—	—	—	—
Int. Ch Rate (Kbps)	Fch – TX	1438	3	—	—	—	—
PCI Freq. (kHz)	fpci	66000	66000	—	—	—	—
Time to Read BD	Average	4	4	—	—	—	—
Time to Read BD	Actual	4	4	—	—	—	—
Time to Update Status	—	3	3	—	—	—	—
Including Spare Time							
MAXDATA	RX	352	0	—	—	—	—
MAXDATA	TX	352	0	—	—	—	—
Max L_{ch} (ms)	RX	0.506	4.219	—	—	—	—
Max L_{pci} (ms)	RX	0.125	0.125	—	—	—	—
Max L_{ch} (ms)	TX	0.552	0.625	—	—	—	—
Max L_{pci} (ms)	TX	0.125	0.125	—	—	—	—
Amnt Data Filled in L_{pci}	RX	195	1	—	—	—	—
Amnt Data Emptied in L_{pci}	TX	179	0	—	—	—	—
Spare Time (ms)	RX	0.382	4.094	—	—	—	—
Spare Time (ms)	TX	0.428	5.500	—	—	—	—

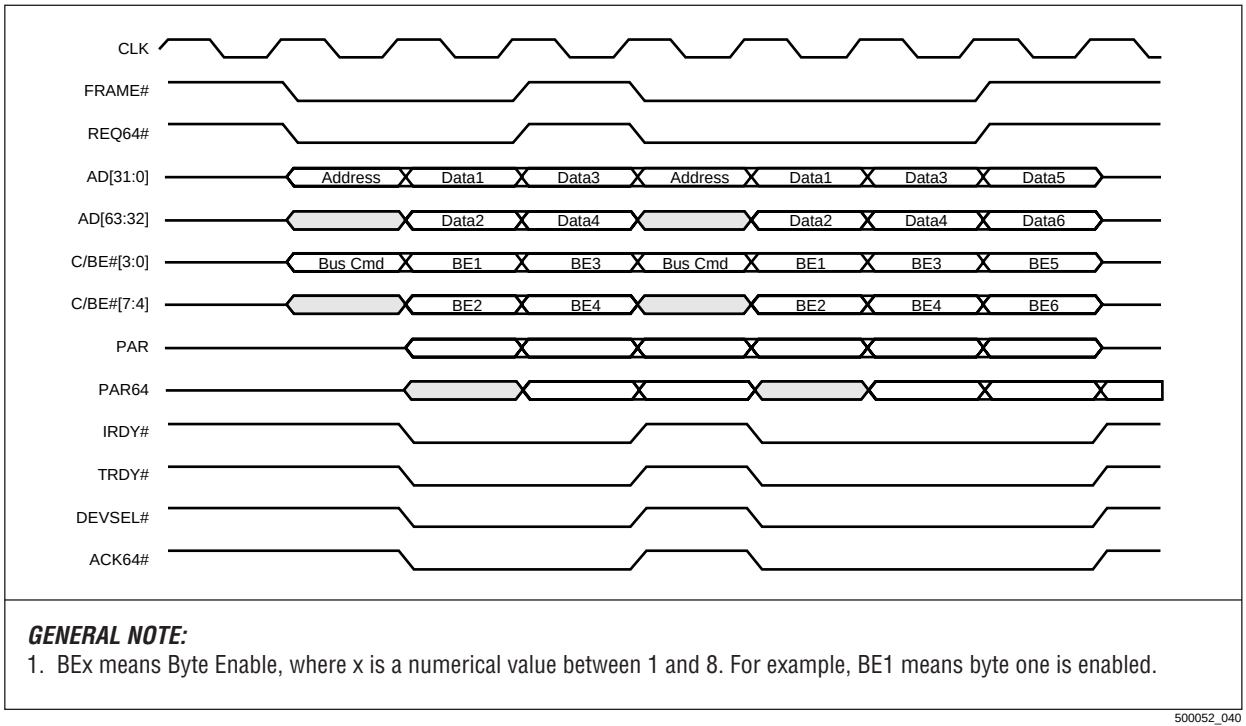
Table A-8. Example Three *(2 of 2)*

66 MHz PCI		44-Byte Messages				12-Byte Messages	
		168 x 1536K Channels		6 x 44.21M Channels		168 x 4K Channels	
Configuration		**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**	**64-Bit PCI**	**32-Bit PCI**
Spare Time (ms)	TOTAL	0.382	0.382	—	—	—	—
Spare Time (cycles)	TOTAL	25181	25181	—	—	—	—
In Spare Time Amnt Filled	RX	599	2	—	—	—	—
In Spare Time Amnt Emptied	TX	549	1	—	—	—	—
Max Total in SLP Buffer	RX	1588	20	—	—	—	—
Max Total in SLP Buffer	TX	1522	20	—	—	—	—
Not Including Spare Time							
MAXDATA	RX	352	18	—	—	—	—
MAXDATA	TX	352	18	—	—	—	—
Max L_{ch} (ms)	RX	0.506	4.219	—	—	—	—
Max L_{pci} (ms)	RX	0.126	0.126	—	—	—	—
Max L_{ch} (ms)	TX	0.552	5.625	—	—	—	—
Max L_{pci} (ms)	TX	0.126	0.126	—	—	—	—
Utilization	RX	0.170	0.002	—	—	—	—
Utilization	TX	0.156	0.002	—	—	—	—
Utilization	TOTAL	0.326	0.004	—	—	—	—
Utilization	**ABS TOTAL**	**0.330**	**—**	**—**	**—**	**—**	**—**
Amnt Data Filled in L_{pci}	RX	198	1	—	—	—	—
Amnt Data Emptied in L_{pci}	TX	181	0	—	—	—	—

B Example of an Arbitration for Fast Back-to-Back and Non-Fast Back-to-Back Transactions

Figure B-1 illustrates, in a specific scenario, CX28500's PCI transactions while operating as a master and fast back-to-back feature enabled. CX28500 performs as a master while operating at 64-bit address-data, a burst write of 4 dwords, which are transferred during the first cycle and a burst write of 6 dwords which are transferred during the second cycle. It is seen that both transaction cycles require 4 PCLK cycles.

Figure B-1. PCI Burst Write: Two 64-bit Fast Back-to-Back Transactions to Same Target



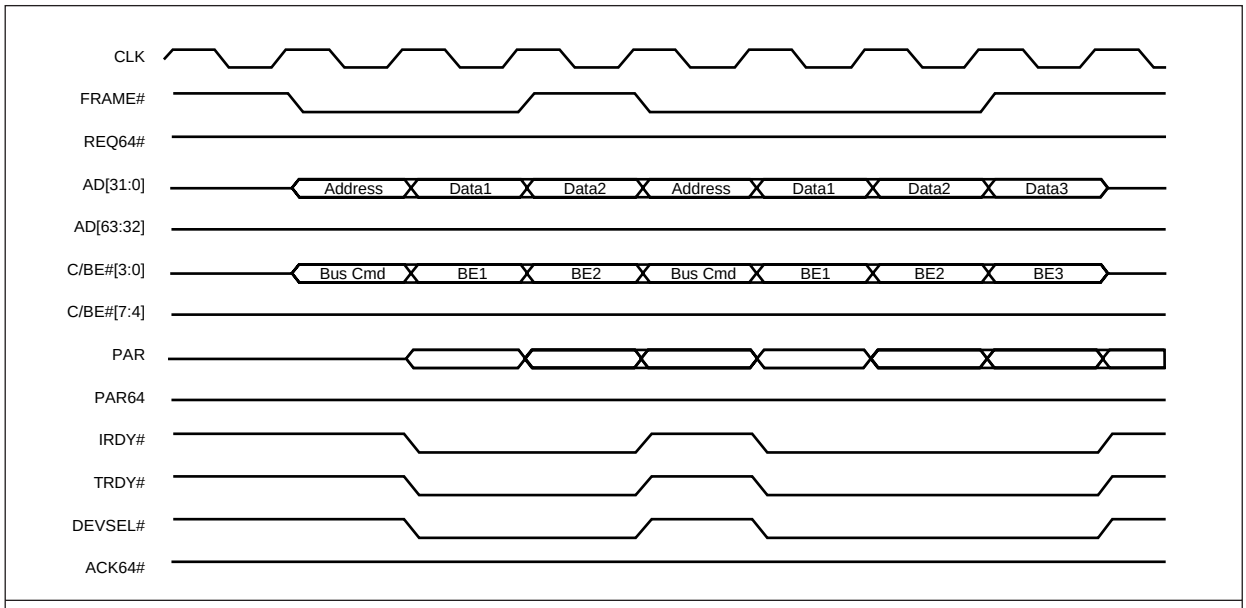
GENERAL NOTE:

1. BEx means Byte Enable, where x is a numerical value between 1 and 8. For example, BE1 means byte one is enabled.

500052_040

Figure B-2 illustrates, in a specific configuration, CX28500's PCI transactions while operating as a master, and fast back-to-back feature enabled. CX28500 performs as a master while operating at 32-bit address-data, a burst write of 2 dwords, which are transferred during the first cycle and a burst write of 3 dwords, which are transferred during the second cycle. Both transaction cycles require 4 PCLK cycles.

Figure B-2. PCI Burst Write: Two 32-bit Fast Back-to-Back Transactions to Same Target



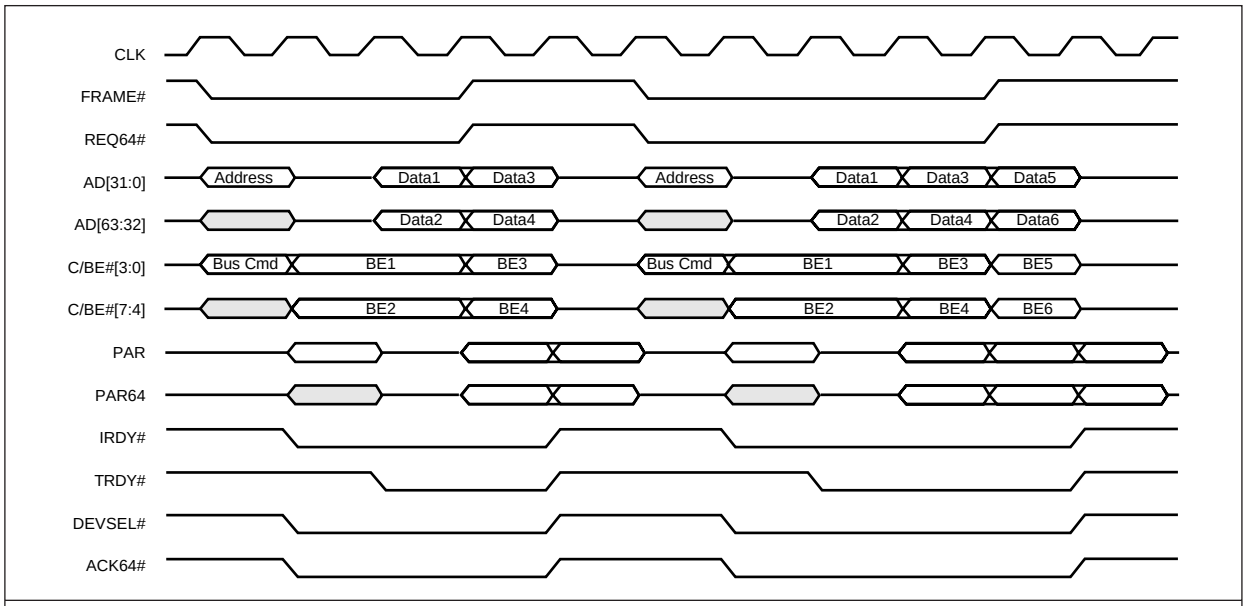
GENERAL NOTE:

1. BEx means Byte Enable, where x is a numerical value between 1 and 8. For example, BE1 means byte one is enabled.

500052_041

Figure B-3 illustrates how CX28500 operates at 64-bit address-data and performs a burst read of 4 dwords followed by a burst read of 6 dwords. The fast back-to-back is disabled. It can be observed that the first cycle takes 5 PCLK cycles (with one PCLK post-data phase) and the second cycle of transferring 6 dwords requires 6 PCLK cycles.

Figure B-3. PCI Burst Read: Two 64-bit Transactions



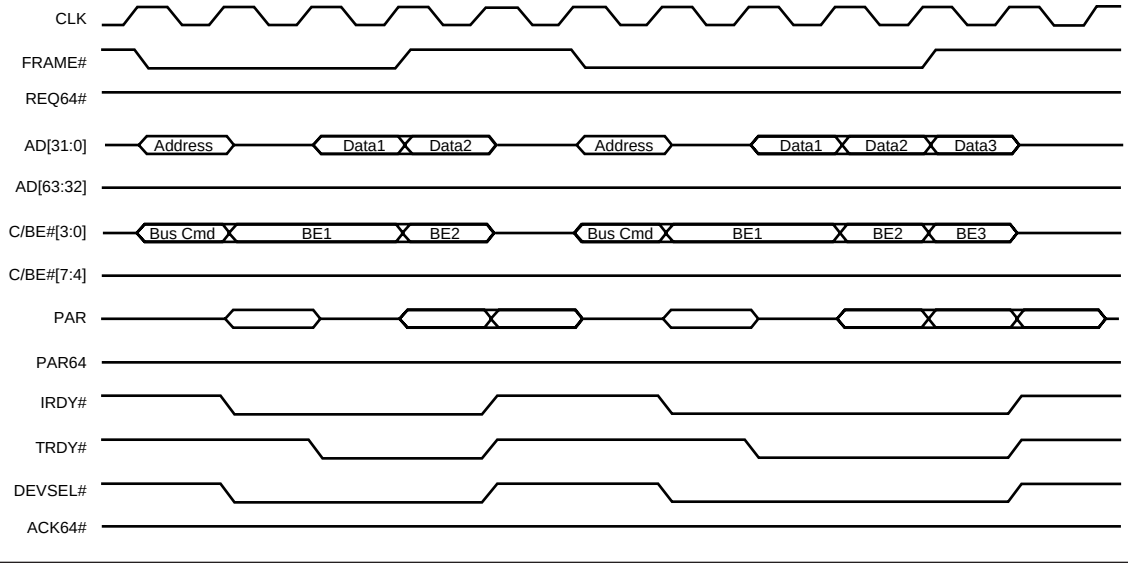
GENERAL NOTE:

1. BEx means Byte Enable, where x is a numerical value between 1 and 8. For example, BE1 means byte one is enabled.

500052_042

Figure B-4 illustrates how CX28500 operates at 32-bit address-data and performs a burst read of 2 dwords transfer during the first cycle and 3 dwords transfer during the second cycle. The fast back-to-back feature is disabled. It can be observed that the first cycle takes 5 PCLK cycles (with one PCLK post-data phase) and the second cycle of transferring 3 dwords requires 6 PCLK cycles.

Figure B-4. PCI Burst: Two 32-bit Transactions



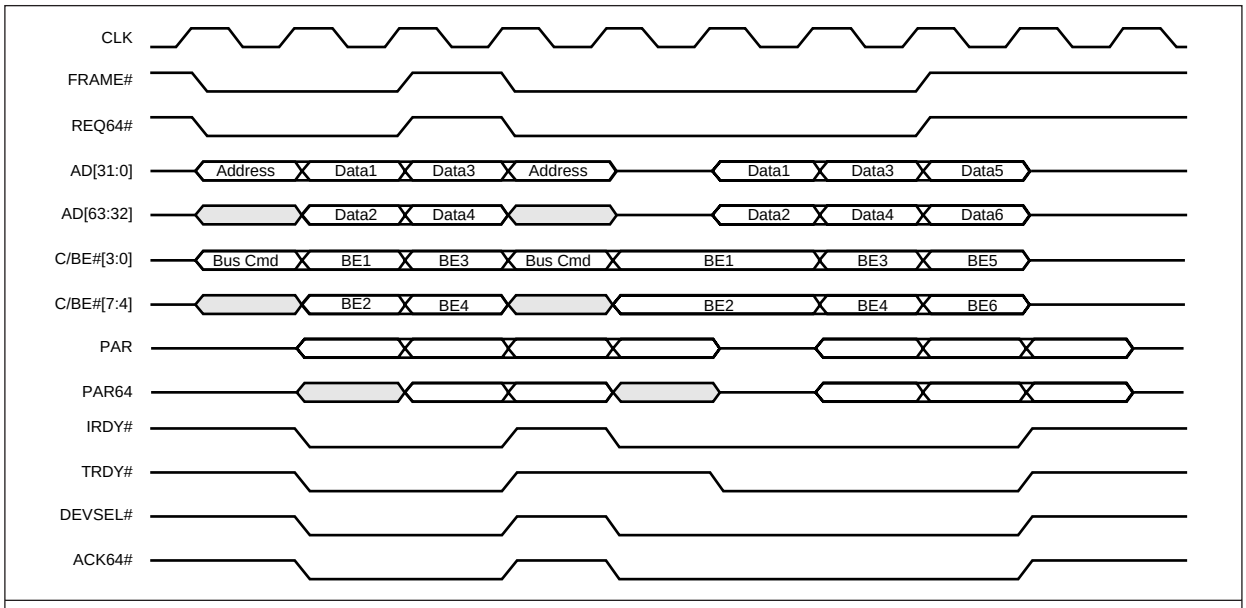
GENERAL NOTE:

1. BEx means Byte Enable, where x is a numerical value between 1 and 8. For example, BE1 means byte one is enabled.

500052_045

In Figure B-5, CX28500 performs PCI transactions as a master while fast back-to-back enabled and it operates at 64-bit address-data. A burst write of 4 dwords are transferred during the first cycle and a burst read of 6 dwords are transferred during the second cycle. It can be observed that the first cycle requires 3 PCLK cycles and the second cycle requires 6 PCLK cycles.

Figure B-5. PCI Burst Write Followed by Burst Read: Fast Back-to-Back to Same Target



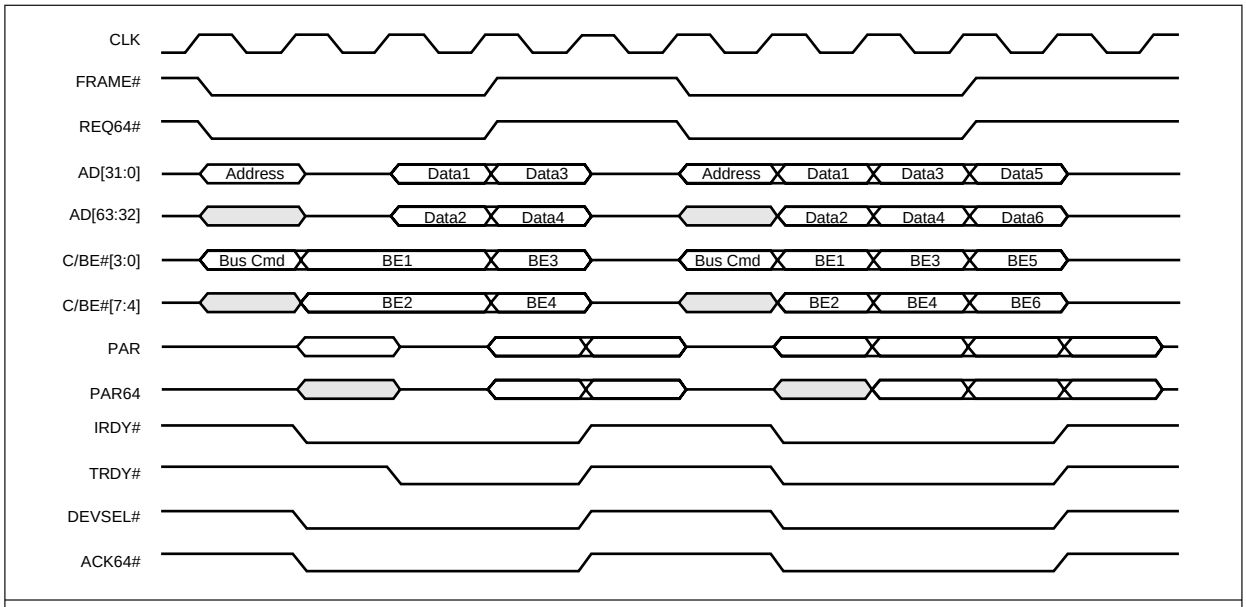
GENERAL NOTE:

1. BEx means Byte Enable, where x is a numerical value between 1 and 8. For example, BE1 means byte one is enabled.

500052_043

In Figure B-6, CX28500 operates at 64-bit address-data and performs a burst read of 4 dwords followed by a burst write of 6 dwords. The fast back-to-back is disabled. It can be observed that the first cycle requires 3 PCLK cycles and the second cycle requires 6 PCLK cycles.

Figure B-6. PCI Burst Read Followed by Burst Write



GENERAL NOTE:

1. BEx means Byte Enable, where x is a numerical value between 1 and 8. For example, BE1 means byte one is enabled.

500052_044

C

T3 Frame Relay Switch Application

This appendix illustrates the detailed description of the CX2833x, CX2834x, and CX28500 device communications.

Figure C-1. High-End Router Application

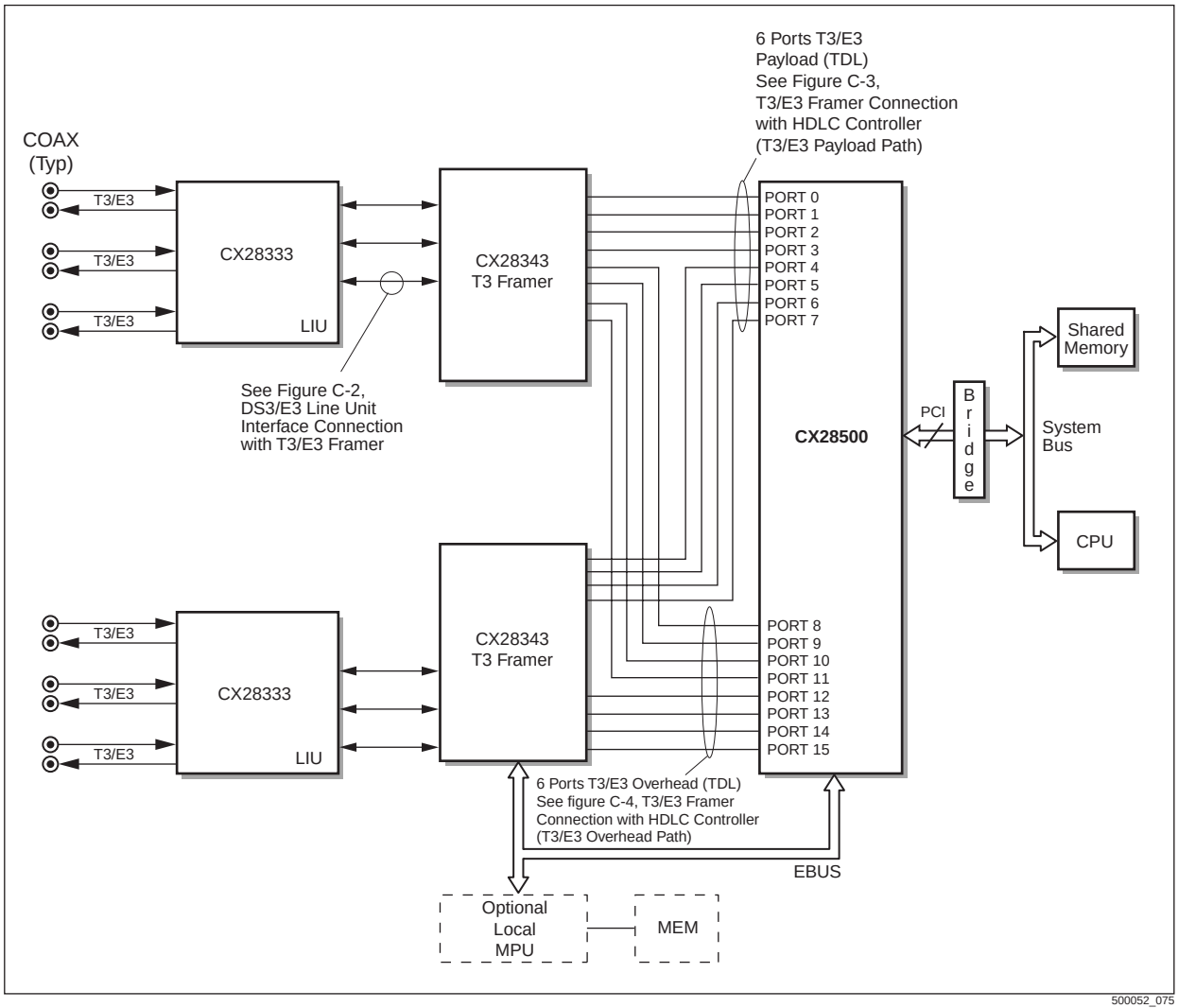
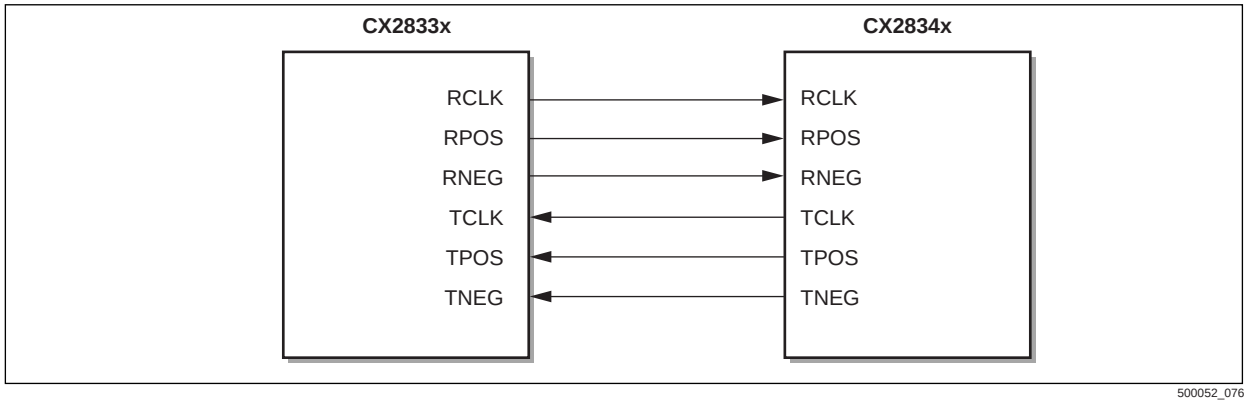


Figure C-2. DS3/E3 Line Unit Interface Connection with T3/E3 Framer



RCLK Recovered clock for each channel receiver, intended for storing the corresponding RDAT* into the following framer or logic.

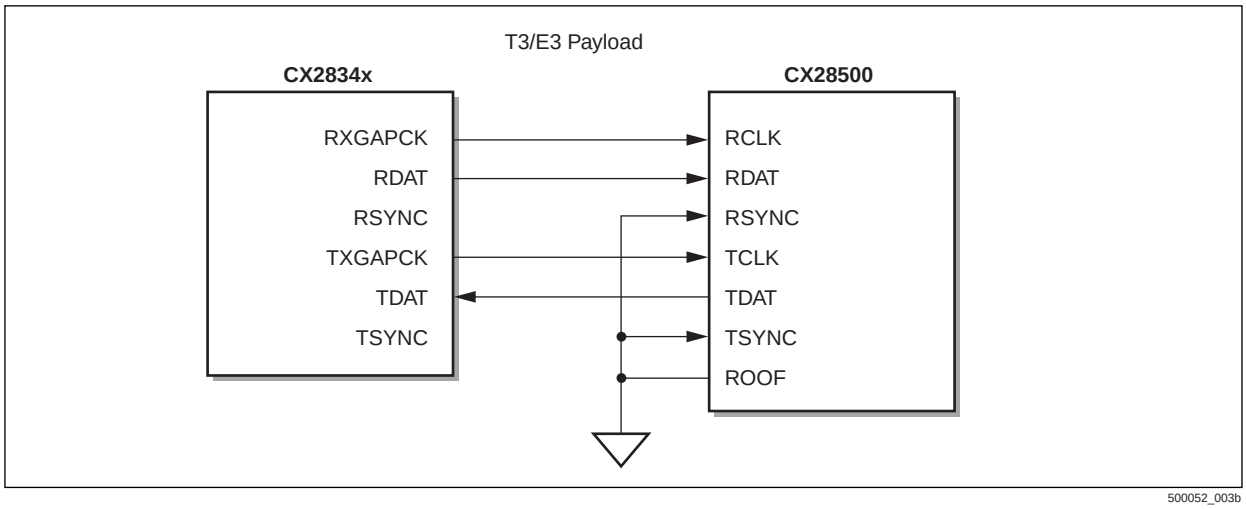
RCLK Transmit bit clock input for storing with transmit data.

RPOS Resynchronized review data intended to store in the corresponding RCLK*.

TPOS Synchronized transmit data intended to store in the corresponding TCLK*.

(For details see CX28332/CX28333 Data Sheet.)

Figure C-3. T3/E3 Framer Connection with HDLC Controller (T3/E3 Payload Path)



TSYNC, RSYNC AS3 Frame Synchronization Specification

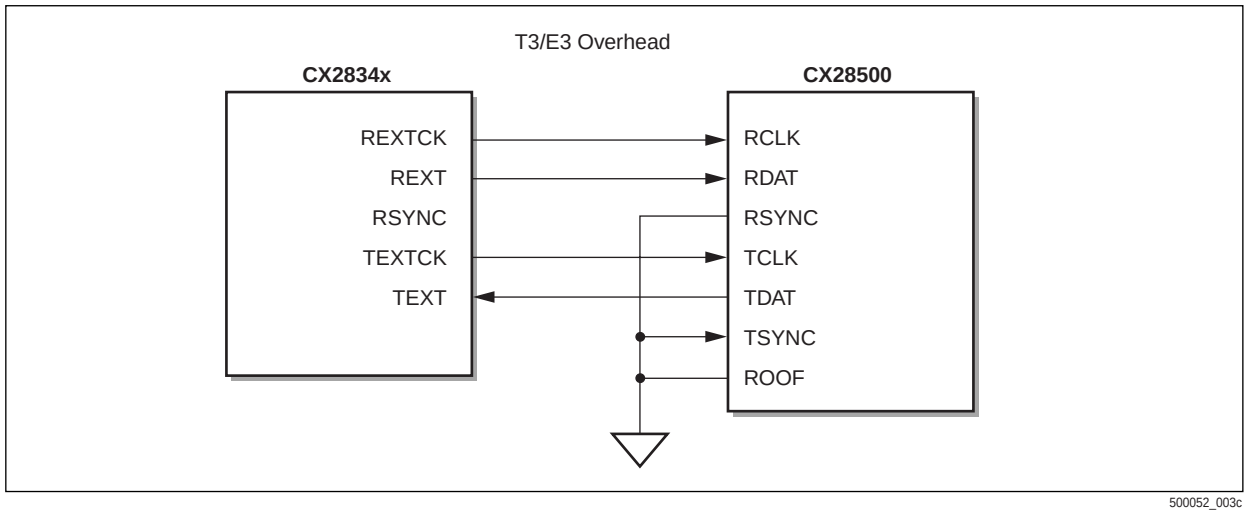
TDAT/RDAT DS3/E3 Transmits Receive Data

RxGAPC Transmit, Receive Clock

TxGAPCK

(For details see CX28313 ASIC Specification.)

Figure C-4. T3/E3 Framer Connection with HDLC Controller (T3/E3 Overhead Path)



TSYNC, RSYNC	AS3 Frame Synchronization Specification
TDAT/RDAT	DS3/E3 Transmits Receive Data
RxGAPCK	Transmit, Receive Clock
TxGAPCK	

(For details see CX28313 ASIC Specification.)

D Example of Little-Big Endian Byte Ordering

An example of Little-Big-Endian byte ordering is shown in the next table. For the example a 32-bit dword was used—76543210h:

Table D-1. Little Endian

Address	x+3	x+2	x+1	x
Data	76h	54h	32h	10h

Table D-2. Big Endian

Address	x+3	x+2	x+1	x
Data	10h	32h	54h	76h

NOTE: When Little-Big-Endian byte ordering is used, this only refers to the data portion of the interface to the Host, meaning that only data transfers are affected. This is useful mainly for applications that use 32-bit values, rather than for plain stream applications.

CAUTION

Using big-endian mode in combination with buffers that are not 32-bit aligned is not recommended.

E TSBUS

E.1 Connection Between CX28500 and Other TSBUS Device

This section details the signals required to implement the TSBUS[1] Interface. The CX28500 implements a subset of this bus. Figure E-1 illustrates the TSBUS connections between the other device and CX28500. The signals required are summarized in Tables E-1 and E-2. The TSBUS consists of the Payload and the Overhead bus. Each bus has a Transmit and Receive path. The receive path is defined from the other device to CX28500, and the transmit path is defined from CX28500 to the other device.

Note that while the bus defines the TSB_RSYNCI, TSB_TSYNCI and TSB_TSYNCO pins, these are absent from the CX28500, and thus a CX28500 implements only a subset of the TSBUS standard. See the documentation of the CX29503 or CX28560 parts for more complete explanation of the TSBUS standard.

Figure E-1 illustrates the CX28500 pin definitions in TSBUS mode.

1.Time Slot Bus

Figure E-1. CX28500 Time Slot Interface Pins

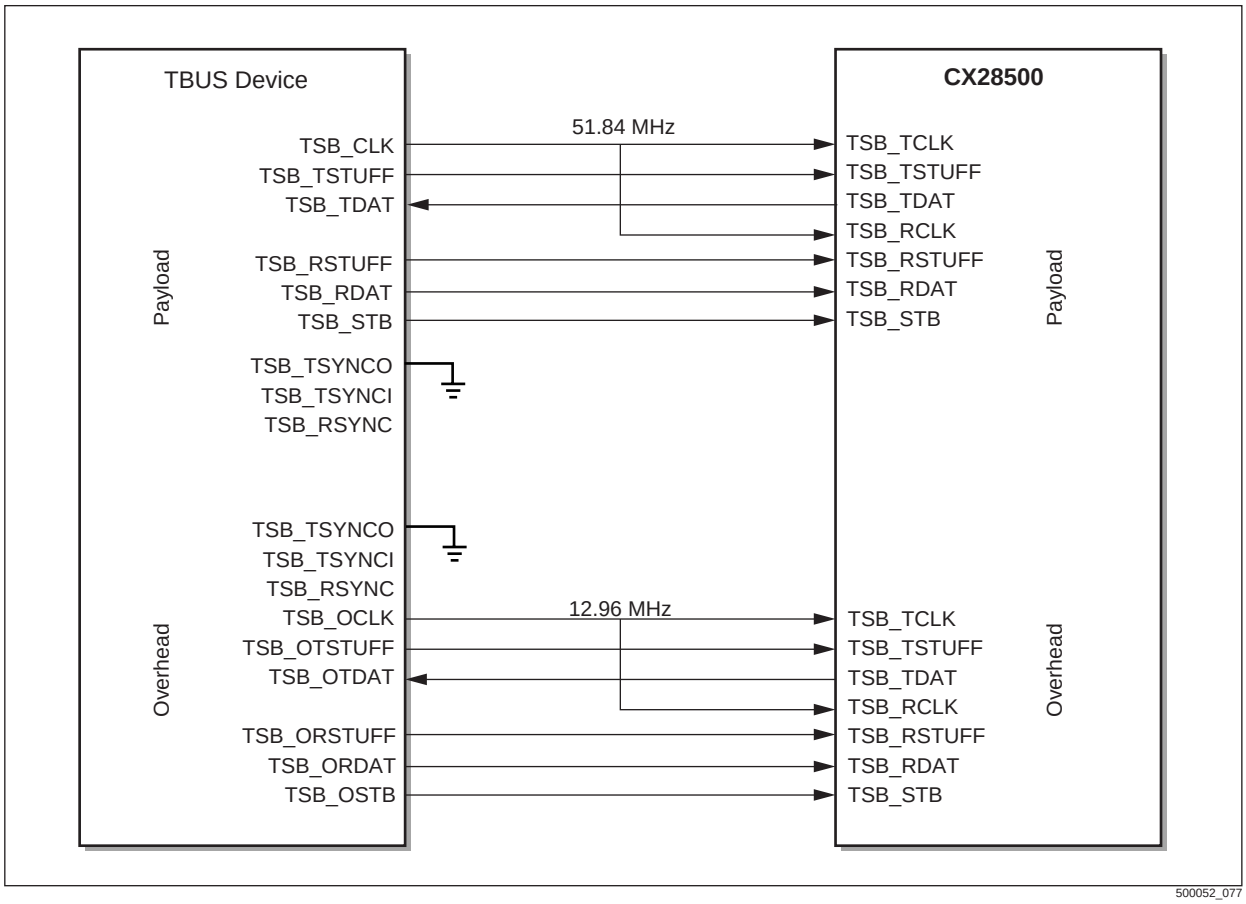


Table E-1. System Side Interface: Payload Time Slot Bus

Symbol	Reset Behavior	I/O	Definition
TSB_CLK	Low	IN	Payload Time Slot Bus Clock: This clock is usually based on SIB_TXHSCLK[1]. It is used for all timing on the Payload Time Slot Bus. Clock Rate is 51.84 MHz (±20 ppm)
TSB_STB	Low	IN	Payload Time Slot Bus Strobe: A strobe signal that indicates the start of a frame with 84 time slots carrying payload data. The Strobe indicates the beginning of each Payload time slot Frame.
TSB_TDAT	—	OUT	Payload Time Slot Bus Transmit Data: This is the serial payload data sent from the CX28500 to the other device. This signal may be driven on the rising edge of TSB_CLK or on the falling edge, according to the TDAT_EDGE bit in the TSIU Port Configuration Register.
TSB_TSTUFF	High	IN	Payload Time Slot Bus Transmit Stuff Indication: When high, indicates a stuff byte must be transmitted in place of the data byte arriving 8 time slots later.
TSB_RDAT	Low	IN	Payload Time Slot Bus Receive Data: This is the received serial payload data. It may be sampled on the rising edge of TSB_CLK or on the falling edge, according to the RDAT_EDGE bit in the RSIU Port Configuration Register.
TSB_RSTUFF	High	IN	Payload Time Slot Bus Receive Stuff Indication: When high, indicates that data on TSB_RDAT is not valid data. TSB_RDAT is stuffed with all 1s.

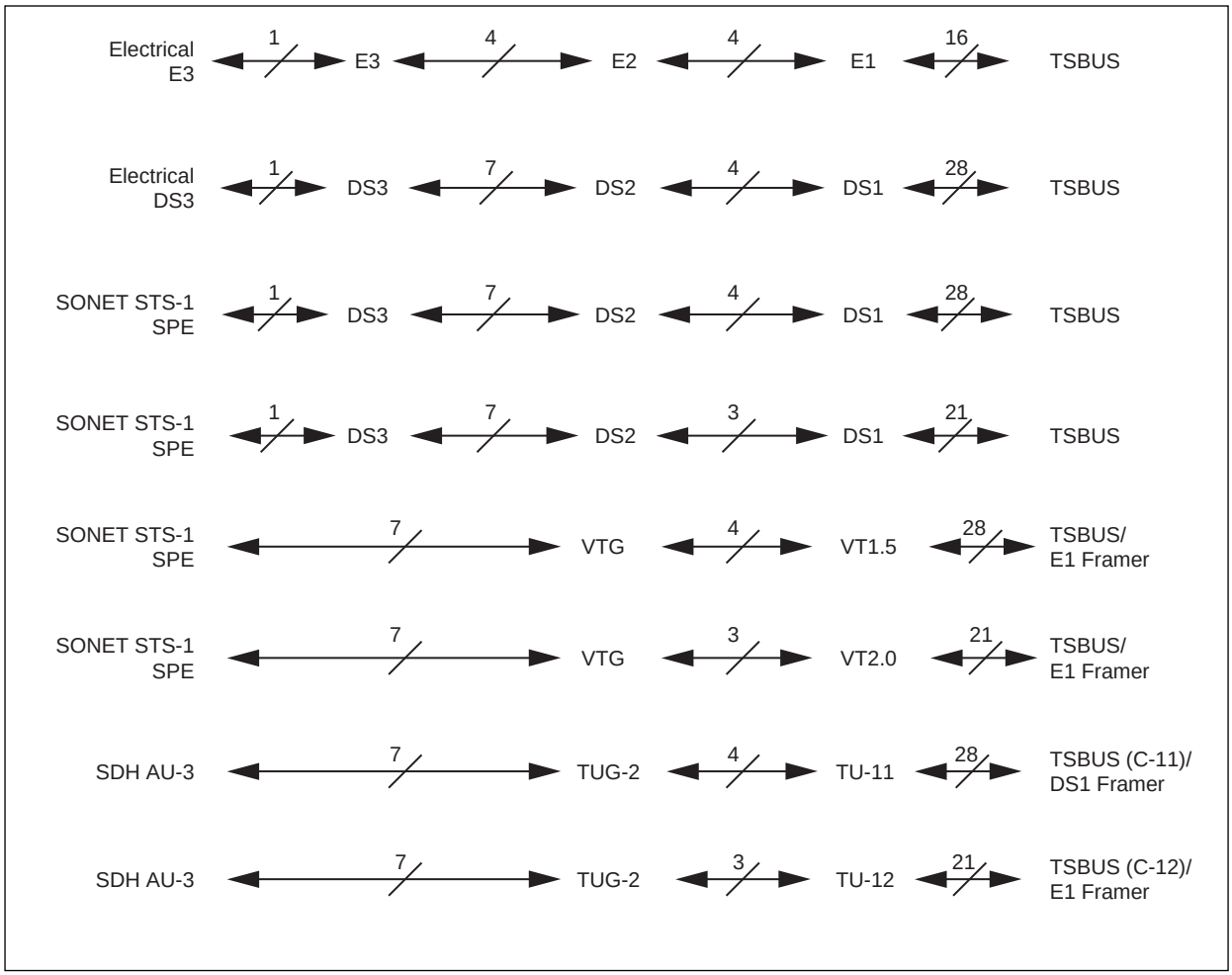
[1] This is a signal that is output from a CX29610, and is used by the CX29503 as a reference 51.84MHz clock. When CX29610 is not used, it may be derived from another reference clock.

Table E-2. System Side Interface: Overhead Time Slot Bus

Symbol	Reset Behavior	I/O	Definition
TSB_OCLK	Low	IN	Overhead Time Slot Bus Clock: This clock is based on SIB_TXHSCLK[1]. It is used for all timing on the Overhead Time Slot Bus. Clock Rate is 12.96 / 11.184 / 8.592 MHz.
TSB_OSTB	Low	IN	Overhead Time Slot Bus Strobe: A strobe signal that indicates the start of a frame with 84 time slots carrying Overhead data. The Strobe indicates the beginning of each Overhead time slot Frame.
TSB_OTDAT	—	OUT	Overhead Time Slot Bus Transmit Data: This is the serial overhead data sent from the CX28500 to the other device. This signal may be driven on the rising edge of TSB_OCLK or on the falling edge, according to the TDAT_EDGE bit in the TSIU Port Configuration Register.
TSB_OTSTUFF	High	IN	Overhead Time Slot Bus Transmit Stuff Indication: When high, indicates a stuff byte must be transmitted in place of the data byte arriving 8 time slots later.
TSB_ORDAT	Low	IN	Overhead Time Slot Bus Receive Data: This is the received serial overhead data. It may be sampled on the rising edge of TSB_OCLK or on the falling edge, according to the RDAT_EDGE bit in the RSIU Port Configuration Register.
TSB_ORSTUFF	High	IN	Overhead Time Slot Bus Receive Stuff Indication: When high, indicates that data on TSB_RDAT is not valid data. TSB_RDAT is stuffed with all 1s.

[1] This is a signal that is output from a CX29610, and is used by the CX29503 as a reference 51.84MHz clock. When CX29610 is not used, it may be derived from another reference clock.

Figure E-2. Source/Destination of TSBUS Block Line-Side Signals



500052_031

Table E-3. System Side Interface: Overhead Time Slot Bus Frame

TSBUS Source/Destination	Overhead Data Communication Channel Mapped to Virtual Serial Port (VSP)	
	Description	Data Rate
DS1/E1 Framer No. 1-28	F-bit Data Link/Sa4 Bit Data Link	112 Kbps
STS-12/STS-3/STM-1 Mapper	Regenerator Section Data Communication Channel (DCCR) Bytes 1–3	194 Kbps
STS-12/STS-3/STM-1 Mapper	Multiplex Section (Line) Data Communication Channel (DCCM) Byte 1–9	583 Kbps
SONET/SDH STS-1/AU-3 Mapper	Path User Channel: F2	64 Kbps
SONET/SDH STS-1/AU-3 Mapper	Path User Channel: F3	64 Kbps
SONET/SDH STS-1/AU-3 Mapper	SPE/AU Path Overhead Nibble N1 (4 LSBs) Path Data Channel/Bit Oriented or LAPD Tandem Connection	32 Kbps
Unused Communication Time Slots	Future Use	3.564 Mbps
Command Status Processor (CSP)	CSP Channel. Used to control/monitor CX29503 device.	6.48 Mbps

E.1.1 VSP Mapping of Intermixed Digital Level 2 Signals

The information in this subsection relates to the way a CX29503 device presents channels of various types of mappings to the CX28500 on the TSBUS. It is reproduced from the CX29503 datasheet in order to make the explanation of the timeslot assignments complete.

The following Digital Level 2 signals can transport either DS1 or E1 signals: VTG, TUG-2, and DS2. SONET, SDH, and PDH transport their respective Level 2 signals in sets of seven Level 2 signals. This set of seven Level 2 signals can operate in mixed mode where a portion of the seven Level 2 multiplexed signals transport DS1 signals and the remainder transport E1 signals. Any given Level 2 signal in mixed mode can only transport DS1 signals or E1 signals. It cannot transport both signals.

Table E-4 defines the mapping of DS1 and E1 signals when they are extracted from a mixed set of seven VTG's, a mixed set of seven TUG-2s, or a mixed set of seven DS2s.

Each level 2 signal has a set of 3 or 4 related framers. All framers within a set must be configured for the same type of signal. This prevents framers for different data paths from multiplexing data into the same time slot.

There are four framers in a set for DS1, VT1.5, and VC-11 signals. There are three framers in a set for E1, VT2.0, and VC-12 signals.

The types of Level 2 signals that can be mixed together are limited to the following combinations:

1. DS2 signals containing DS1 signals and the DS2 signals containing E1 signals.
2. VTG signals containing VT1.5, which contain DS1 signals and VTG signals containing VT2.0, which contain E1 signals.
3. TUG-2 signals containing VC-11, which contain DS1 signals and VTG-2 signals containing VC-12, which contain E1 signals.

Table E-4. VSP Mapping of Intermixed Digital Level 2 Signals Containing Either DS1 or E1 Signals *(1 of 2)*

Framer Set No.	Framer No.	Concatenated Time Slot Numbers		VSP No.
		Framer Configured to Extract DS1 Signal	Framer Configured to Extract E1 Signal	
1	1	1, 29, 57	1, 22, 43 64	1
2	2	2, 30, 58	2, 23, 44, 65	2
3	3	3, 31, 59	3, 24, 45, 66	3
4	4	4, 32, 60	4, 25, 46, 67	4
5	5	5, 33, 61	5, 26, 47, 68	5
6	6	6, 34, 62	6, 27, 48, 69	6
7	7	7, 35, 63	7, 28, 49, 70	7
1	8	8, 36, 64	8, 29, 50, 71	8
2	9	9, 37, 65	9, 30, 51, 72	9
3	10	10, 38, 66	10, 31, 52, 73	10
4	11	11, 39, 67	11, 32, 53, 74	11

Table E-4. VSP Mapping of Intermixed Digital Level 2 Signals Containing Either DS1 or E1 Signals *(2 of 2)*

Framer Set No.	Framer No.	Concatenated Time Slot Numbers		VSP No.
		Framer Configured to Extract DS1 Signal	Framer Configured to Extract E1 Signal	
5	12	12, 40, 68	12, 33, 54, 75	12
6	13	13, 41, 69	13, 34, 55, 76	13
7	14	14, 42, 70	14, 35, 56, 77	14
1	15	15, 43, 71	15, 37, 57, 78	15
2	16	16, 44, 72	16, 37, 58, 79	16
3	17	17, 45, 73	17, 38, 59, 80	17
4	18	18, 46, 74	18, 39, 60, 81	18
5	19	19, 47, 75	19, 40, 61, 82	19
6	20	20, 48, 76	20, 41, 62, 83	20
7	21	21, 49, 77	21, 42, 63, 84	21
1	22	22, 50, 78	NA	22
2	23	23, 51, 79	NA	23
3	24	24, 52, 80	NA	24
4	25	25, 53, 81	NA	25
5	26	26, 54, 82	NA	26
6	27	27, 55, 83	NA	27
7	28	28, 56, 84	NA	28

GENERAL NOTE: Framers with the same Set Number must be configured for the same data signal (i.e., all framers within a set must be configured for DS1 or E1 signals but not both).

E.2 Timing Details

E.2.1 Payload Bus, AC Characteristics

The CX28500 (HDLC Controller) device operates as the Slave on the Time-Slot bus, and the other device (usually CX29503) is the Master. The Master generates TSBUS clocks and control signals and the CX28500 device responds by transmitting data to or receiving data from the Master device. The Master generates a TSBUS Frame Strobe (TSB_STB) on the rising edge of TSB_CLK. The Time Slot Bus frame strobe TSB_STB indicates the start of an N time slot Frame carrying payload data, where the standard value of N is 84 (but this is configurable).

The Time Slot bus exchanges data over two I/O chip boundaries so care must be taken in ensuring that the data is exchanged on the right phase of the master TSBUS clock TSB_CLK. A possible solution for ensuring correct data exchange is for the Slave (CX28500) to transmit data on the Rising edge of TSB_CLK, and sample the Received data on the falling edge of TSB_CLK.

There is only one Time Slot Frame strobe used (TSB_STB) for transmit and receive direction. There is also only one clock (TSB_CLK) used in the definition of bit boundaries for transmit and receive. This results in the Time Slot Frame alignment of the receive and transmit payload (illustrated in Figure E-3). Each time slot in the Time Slot Bus consists of eight serial data bits. The MSB bit for each Time Slot is transmitted first.

E.2.2 Transmit Timing

The Master generates clock, Frame sync Strobe signal, and Stuff signal. CX28500 will generate Transmit data (TSB_TDAT) or generate an all-1s Stuff pattern eight time slots after receiving an active Stuff signal (TSB_TSTUFF). The Master will generate a Frame sync Strobe (TSB_STB) output synchronously with the rising edge of TSB_CLK. Figure E-3 illustrates the timing requirements for the Transmit. Figure E-4 illustrates the Transmit Stuff signal (TSB_TSTUFF). The timing values are illustrated in Table E-4, but see Section 10.2.5 Serial Interface Timing and Switching Characteristics.

Figure E-3. Payload Time Slot Bus Transmit Data (TSB_TDAT)

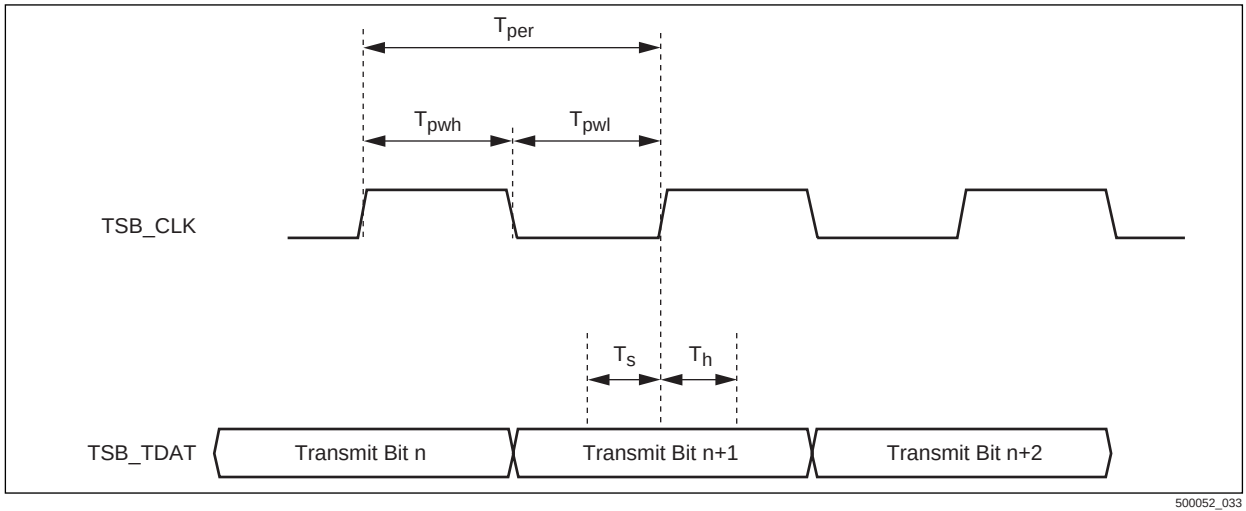


Figure E-4. Payload Time Slot Bus Transmit Stuff Indicator (TSB_TSTUFF)

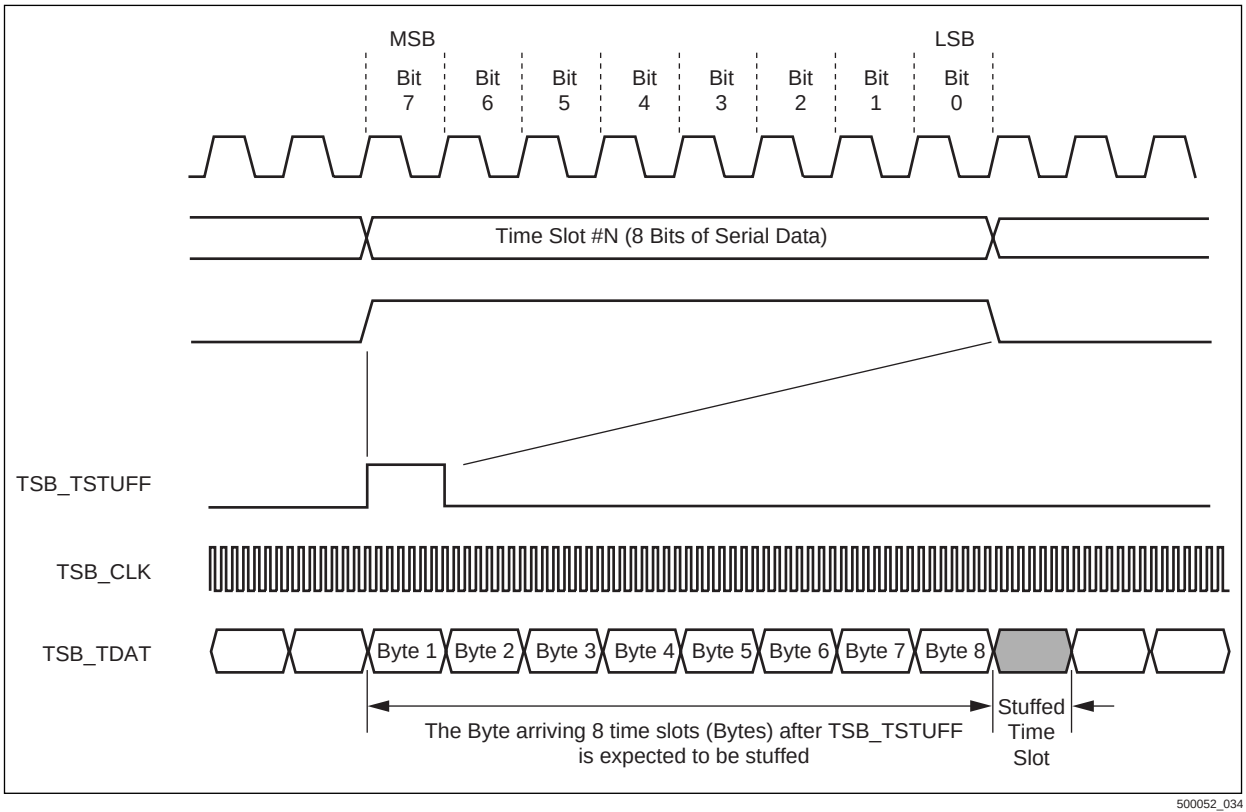


Table E-5. Transmit Timing Values

Label	Description	Min	Max	Unit
t_{fc}	Clock Frequency	t_{dc}/52 MHz	t_{dc}/52 MHz	ns
t_{dc}	Duty Cycle	40%	60%	—

Table E-5. Transmit Timing Values

Label	Description	Min	Max	Unit
t_s	Setup, TSB_TDAT to rising edge of TSB_CLK	15[1] 2[2]	—	ns
t_h	Hold, TSB_TDAT from rising edge of TSB_CLK	15[1] 3[2]	—	ns

FOOTNOTE:
[1] If port frequency is less than 13 MHz.
[2] If port frequency is more than 13 MHz.

E.2.3 Receive Timing

The Master generates clock, data, Frame sync Strobe signal, and the Stuff signal (TSB_RSTUFF). The Master generates an all ones stuff pattern in place of the payload data during the same time slot that the Receive Stuff signal (TSB_RSTUFF) is active. The Master generates control and data outputs synchronously with the rising edge of TSB_CLK. The nominal clock frequency is 51.84 MHz. Figure E-5 shows the timing requirements for the receive interface. See Figure E-6 for the Receive Stuff signal (TSB_RSTUFF). The timing values are illustrated in Table E-5, but see Section 10.2.5 Serial Interface Timing and Switching Characteristics.

Figure E-5. Payload Time Slot Bus Receive Data (TSB_RDAT)

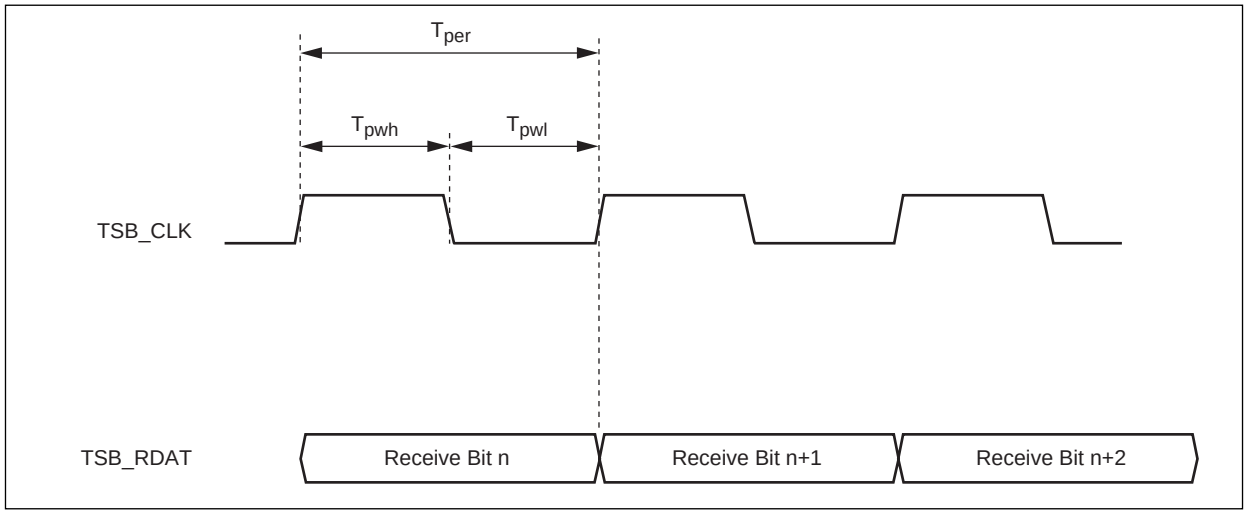
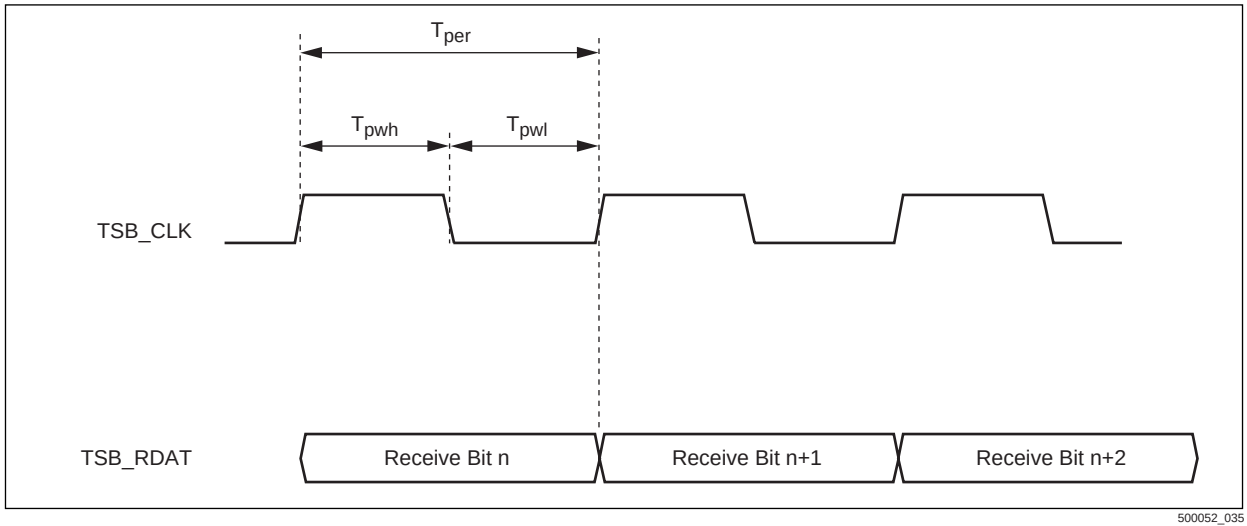


Figure E-6. Payload Time Slot Bus Receive Stuff Indicator (TSB_RSTUFF)



E.3 Overhead Bus, AC Characteristics

The Overhead TSBUS has the same operation as the Payload TSBUS. The only difference is that the TSB_OCLK clock can be run at 12.96/11.184/8.592 MHz compared to the Payload TSBUS frequency of 51.84 MHz.

E.3.1 Transmit Timing

See Section E.2.2.

E.3.2 Receive Timing

See Section E.2.3.

E.4 DC Characteristics

See Chapter 10 of this document.

F Notation, Acronyms, Abbreviations, and Definitions

This appendix serves as a reference regarding notational conventions used throughout the specification.

F.1 Radix Notation

The general representation for numbers is as follows:

	Suffix	Example
Binary	b	001011b
Hex	h	6Fh, 01Bh
Decimal	d	01d, 750d

The suffix is often dropped for clarity when the context makes the intended radix obvious. Note that this suffix convention requires that letters [ABCDEF] used in hex numbers always be capitalized.

F.2 Bit Stream Convention

By historical convention, digital voice data represented in octet samples are transmitted serially starting at the leftmost bit (i.e., MSB = bit 1) and proceeding down to the rightmost bit of the sample (i.e., LSB = bit 8).

On the other hand, data organized in n-bit words is transmitted serially starting with the rightmost bit (i.e., LSB = bit 0) and proceeding along to the left most bit (i.e., MSB = bit n-1).

CX28500 is designed to support the data transmission convention. That is, on the receive side, serial data is placed in the LSB of a word first up to the MSB. On the transmit side, the LSB (i.e., bit 0) is transmitted first, then on up to the MSB.

F.3 Acronyms, Abbreviations, and Definitions

F.3.1 Acronyms and Abbreviations

The following is a list of acronyms used in this specification, and their expanded meanings.

23B+D	Twenty-three B Channels plus one D Channel
2B+D	Two B Channels plus one D Channel
2B1Q	Two Bits in One Quaternary
30B+D	Thirty B Channels plus one D Channel
ANSI	American National Standards Institute
ATM	Asynchronous Transfer Mode
BAM	Broadband Access Multiplexer
ChAct	Channel Activation
ChDeact	Channel Deactivation
CMOS	Complementary Metal-Oxide Semiconductor
COFA	Change Of Frame Alignment
CPU	Central Processing Unit
CRC	Cyclic Redundancy Code
CTS	Clear To Send
DCE	Data Communication Equipment
DMA	Direct Memory Access
DMI	Digital Multiplexed Interface
DS0	Digital Signal–Level Zero (64 Kbps)
DS1	Digital Signal–Level One (1.544 Mbps)
DS1C	Digital Signal–Level One C (3.152 Mbps)
DS2	Digital Signal–Level Two (6.312 Mbps)
DS3	Digital Signal–Level Three (44.736 Mbps)
DSLAM	Digital Subscriber Live Access Multiplexed
DSX-1	Digital Signal Cross-connect–Level 1 (DS1)
DSX-3	Digital Signal Cross-connect–Level 3 (DS3)
DTE	Data Terminal Equipment
DXI	Data Exchange Interface
E1	European transmission service at the E1 rate of 2.048 Mbps
ECC	Error Correcting Code

EOM	End Of Message
EOS	End Of Segment
FCS	Frame Check Sequence
FDL	Facilities Data Link
FDM	Frequency Division Multiplexing
FDMA	Frequency Division Multiple Access
FEP	Front-End Processor
FIFO	First-In First-Out (memory)
FT-1	Fractional T-1
HDLC	High-level Data Link Control
IC	Idle Code
IEEE	Institute of Electrical and Electronic Engineers
ISDN	Integrated Services Digital Network
ISLP	Intersystem Link Protocol
ITF	Interframe Time Fill
ITU	International Telecommunications Union
JTAG	Joint Test Action Group
LAN	Local Area Network
LAP-B	Link Access Protocol-Balanced
LAP-D	Link Access Protocol-D
LEC	Local Exchange Carrier
LSB	Least Significant Bit, or Least Significant Byte
M13	Multiplexer between 28 DS1s and a DS3
Mbps	Megabit per Second
MSB	Most Significant Bit, or Most Significant Byte
MUX	Multiplexer
OC	Optical Carrier
OC-1	Optical Carrier–Level 1 (51.84 Mbps)
OC-12	Optical Carrier–Level 12 (622.08 Mbps)
OC-24	Optical Carrier–Level 24 (1.244 Gbps)
OC-3	Optical Carrier–Level 3 (155.52 Mbps)
OC-48	Optical Carrier–Level 48 (2.488 Gbps)
ODSX	Optical Digital Signal Cross-connect
OOF	Out Of Frame

OSI	Open System Interconnection
PBX	Private Branch Exchange
PCI	Peripheral Component Interface (bus)
PCM	Pulse Code Modulation
PM	Performance Monitoring
PPP	Point-to-Point Protocol
PRI	Primary Rate Interface
PSN	Packet Switched Network
RSIU	Receive Serial Interface Unit
RT	Remote Terminal
RX	Receive
SDH	Synchronous Digital Hierarchy
SLP	Serial Line Processor
SMDS	Switched Multi-megabit Data Service
SONET	Synchronous Optical Network
SONET Frame	Payload (data) plus overhead (control)
SPE	Synchronous Payload Envelope
SS7	Signalling System 7
STE	Section Terminating Equipment
STM-1	Synchronous Transport Multiplex (155.52 Mbps)
STS	Synchronous Transport Signal
STS-1	Synchronous Transport Signal – Level 1 (51.84 Mbps)
STS-3	Synchronous Transport Signal – Level 3 (155.52 Mbps)
STS-3c	Synchronous Transport Signal – Level 3 concatenated (155.52 Mbps)
T1	Transmission service at the DS1 rate of 1.544 Mbps
T3	Transmission service at the DS3 rate of 44.736 Mbps
TDM	Time Division Multiplexing
TE	Terminal Equipment
TS	Time Slot
TSBUS	Time Slot Bus
TSIU	Transmit Serial Interface Unit
TX	Transmit
VPN	Virtual Private Network

VSP	Virtual Serial Port
VT	Virtual Tributary
VT-1.5	Virtual Tributary – Level 1.5 (1.728 Mbps)
VT-2	Virtual Tributary – level 2 (2.304 Mbps)
VT-3	Virtual Tributary – Level 3 (3.456 Mbps)
VT-6	Virtual Tributary – Level 6 (6.912 Mbps)
VT6-N	Virtual Tributary – Level 6 (N x 6.9 Mbps)
WAN	Wide Area Network
ZBTSI	Zero Byte Time-Slot Interchange
ZCS	Zero Code Suppression

F.3.2 Definitions

The following is the list of technical definitions used in this document:

Byte	A group of 8 binary bits. A byte is exactly synonymous with an octet.
Channel	A logical bit stream passed through CX28500. The transmit direction is defined as the path from Host interface to serial port; the receive direction is defined as the path from serial port to Host interface. The channel rate is configurable.
Channelized	Refers to a serial port configuration whereby the bit stream is logically partitioned into 8-bit time slots. Frame synchronization is required and allows mapping of individual bits, time slots, or Virtual Serial Ports (VSP) into logical channels. This mode is synonymous to PCM Highway.
Data Path	A data path is one communications channel (i.e., DS1, E1, VT1.5, VT2.0, C-11, C-12, Unchannelized DS3, or Unchannelized STS-1 signals). One data path occupies a fixed number of time slots at fixed locations in a Time slot Bus Frame.
Descriptor	A single dword (i.e., 32-bit) control structure that describes some attributes of a data block.
Digital Level 1 Signals	The following Digital Level 1 signals are the channelized data paths transported over the Payload TSBUS: DS1, E1, VT1.5, VT2.0, C-11, and C-12.
Digital Level 2 Signals	The following Digital Level 2 signals contain the channelized data paths that are extracted and then transported over the receive Payload TSBUS: DS2, E2, VTG, and TUG-2.

DWord	A group of 32 bits. CX28500 assumes that memory is organized as 32-bit words, as viewed through the PCI interface. This term is equivalent to a dword which is used for historical reasons to refer to double 16-bit words.
Flag	As defined in HDLC, an octet with the value 7Eh.
HDLC Frame	In the context of an HDLC bit stream, a frame is a packet of information delimited with 7E flags. This term can be used interchangeably with message or packet. The term frame in this context is different than a T1 or E1 frame.
Hyperchannel	Refers to the concatenation of multiple time slots into a single logical channel.
Idle Code	Octet pattern used to fill the time between the closing flag of one message and the opening flag of the next message. CX28500 supports the following patterns: 7Eh, FFh, and 00h.
Logical Channel	Also called a Virtual Serial Port (see VSP, below).
Message	Refers to an HDLC frame or packet as delimited by opening and closing 7Eh flags.
Octet	Synonymous with byte. Refers to an association of 8 bits.
Pointer	A single word (i.e., 32-bit) control structure that serves as an address to another word (i.e., word-aligned pointer) or byte (i.e., byte-aligned pointer).
Port	One of the 32 full-duplex serial interfaces supported by CX28500.
SPE	Synchronous Payload Envelope. This is the envelope used within an STS frame structure to carry the path layer overhead and payload data in the SONET system.
Structure	A general term referring to one or more data structures stored in shared memory.
SubChannel	A portion of a 64 bps time slot. That is, when a time slot is split and utilized as one sub-64 bps channel, it is referred to as subchannel.
Time Slot	An 8-bit portion of a T1 or E1 frame that repeats every 125 µs, for a total of 64 Kbps.
TSBUS	Time Slot (TS) Bus. The time slot bus is a time division multiplexed serial (one bit wide) connection for passing digital communication signals (data paths) between TBUS Device (broadband access multiplexer) and CX28500 (HDLC Controller).
TSBUS Frame	A TSBUS contains 84 time slots.

Unchannelized	Refers to a serial port configuration whereby the bit stream is considered a continuous stream delimited only by occasional overhead bits. No frame synchronization is required, only overhead bit identification. This port can be utilized only as a single logical channel.
VC	Virtual Container. This is the SDH equivalent term to SPE. However, it applies to all digital signal levels. It contains payload bytes plus path overhead bytes. The VC contains one C-11 signal (which has 27 bytes and carries a DS1 signal or a non-DS1 signal) plus path overhead or one C-12 signal (which has 36 bytes and carries and E1 signal or a non-E1 signal) plus path overhead.
VSP	Virtual serial port. A VSP is the multiple number of time slots that carry one data path in the transmit or in the receive TSBUS Frames.
VT	Virtual Tributary. A virtual tributary contains payload bytes plus pointer bytes in SONET systems. The two virtual tributaries referred to in this specification are: VT1.5 which has 27 bytes and carries a DS1 signal or a non-DS1 signal; VT2.0 which has 36 bytes and carries an E1 signal or a non-E1 signal.
VTG	Virtual Tributary Group. A virtual tributary group consists of 108 bytes. One VTG carries 4 time multiplexed VT1.5's signals or it carries 3 time multiplexed VT2.0 signals. One VTG does not carry both signals.
Word	A word is a unit of data in general. Usually it refers to 16-bits, but many times it is used more loosely to mean a single unit of data.

G

Scope of Specification

This document specifies the CX28500 input/output interfaces and the basic functionality of the device.

G.1 Applicable Specifications

The following is a list of specifications that provide backup information, or definition of standards that apply to CX28500.

- PCI Local Bus Specification, Revision 2.1, Production Version, June 1, 1995
- ANSI T1.408-1990
- CCITT Recommendation G.704
- IEEE Standard 1149.1-1990
- CX28478/8474A/8472A Data Sheet
- Topaz Specification
- Pandora Specification
- CX28398 Data Sheet

MINDSPEED™

www.mindspeed.com

General Information:
U.S. and Canada: (800) 854-8099
International: (949) 483-6996
Headquarters - Newport Beach
4311 Jamboree Rd. P.O. Box C
Newport Beach, CA. 92658-8902