PacketTrunk-4™ Device
TDMoIP/MPLS Gateway Device
TXC-05870

DATA SHEET

FEATURES

- Four T1/E1/Serial or one T3/E3 TDM interfaces
- One 10/100 Ethernet IEEE 802.3 MAC, interface via MII/RMII/SMII/SSMII; HDX or FDX
- VLAN support per IEEE 802.1 p & Q
- Four independent advanced clock recovery blocks
- Recovered clock jitter and wander compliant to ITU-T G.823, G.824 (E1, T1 Jitter/Wander Control)
- 64 bundles with independent Tx/Rx queues, configurable jitter buffer depth, and optional redundancy via traffic duplication
- 128 DS0 cross connect
- Support for HDLC sub-rate channels of 2, 7, or 8 bits
- Lost/misordered packet compensation
- Glueless SDRAM interface up to 16 MB
- CBR, VBR, and HDLC payload types
- Compliant with latest IETF and MPLS Forum drafts
- RTOS-independent, abstracted host API source code
- Test Access Port for IEEE 1149.1 boundary scan
- 2.5v core, 3.3v I/O
- 256-pin PBGA, 27 x 27 mm (1.27 mm pitch)

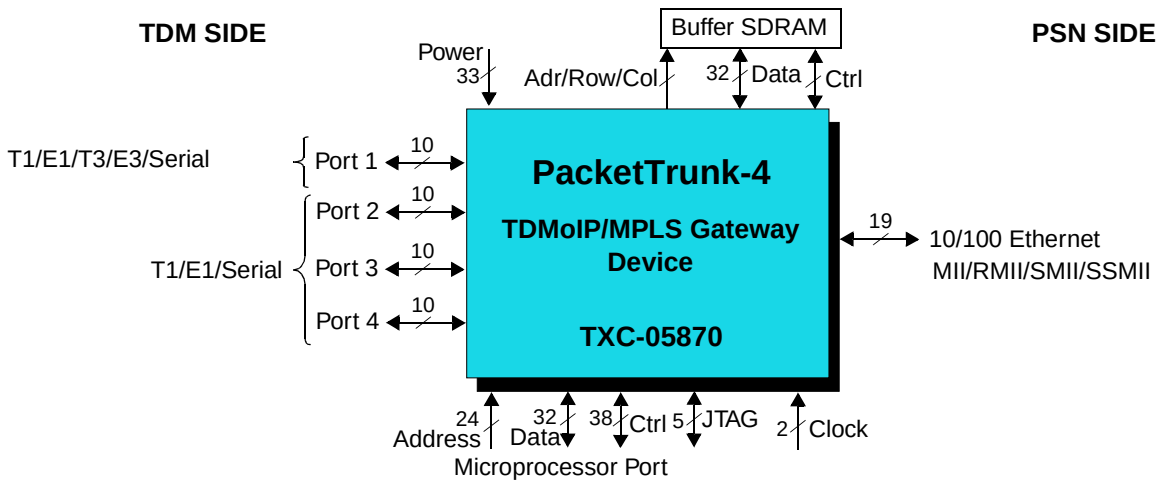
DESCRIPTION

The PacketTrunk-4™ device is a single-chip solution for implementing cost-effective, standards-compliant TDMoIP/MPLS interfaces and systems. PacketTrunk-4 provides all of the necessary interface, encapsulation, clock recovery, and QoS functionality for enabling transport of unstructured or structured TDM signals over a packet-switched network (PSN). An RTOS-independent, abstracted host API enables rapid application development.

PacketTrunk-4 is designed to interface with other TranSwitch devices such as the TEPro (TXC-06830), Envoy-8FE (TXC-06840), and EtherMap®-3 *Plus* (TXC-04236).

APPLICATIONS

- Carrier
  - TDM services over Ethernet MAN, broadband wireless, CATV
  - 2G / 2.5G cellular backhaul over IP/MPLS
  - HDLC-based traffic (ex. Frame Relay) trunking over IP/MPLS
  - T/E carrier grooming (via Ethernet backplane)
  - PSTN-IP network bridging
  - SS7 transport over IP
- Enterprise
  - Private line/toll bypass via Ethernet MAN
  - TDM PBX migration to Ethernet MAN
  - MTU/MDU



Document Number:
PRELIMINARY TXC-05870-MB, Ed. 3
December 2004

TranSwitch Corporation • 3 Enterprise Drive • Shelton, Connecticut 06484 • USA
Tel: 203-929-8810 • Fax: 203-926-9453 • www.transwitch.com

TABLE OF CONTENTS

Section **Page**

TABLE OF CONTENTS (cont.)

Section **Page**

TABLE OF CONTENTS (cont.)

TABLE OF CONTENTS (cont.)

Section **Page**

TABLE OF CONTENTS (cont.)

Please note that TranSwitch provides documentation for all of its products. Current editions of many documents are available from the Products page of the TranSwitch Web site at www.transwitch.com. Customers who are using a TranSwitch Product, or planning to do so, should register with the TranSwitch Marketing Department to receive relevant updated and supplemental documentation as it is issued. They should also contact the Applications Engineering Department to ensure that they are provided with the latest available information about the product, especially before undertaking development of new designs incorporating the product.

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES (cont.)

Figure **Page**

LIST OF TABLES

Table **Page**

LIST OF TABLES (cont.)

Table **Page**

LIST OF TABLES (cont.)

Table **Page**

LIST OF TABLES (cont.)

Table **Page**

LIST OF TABLES (cont.)

Table **Page**

LIST OF DATA SHEET CHANGES

This change list identifies those areas within this updated PacketTrunk-4 TXC-05870 Data Sheet that have significant differences relative to the previous and now superseded PacketTrunk-4 TXC-05870 Data Sheet.

Updated PacketTrunk-4 TXC-05870 Data Sheet: *PRELIMINARY* Edition 3, December 2004

Previous PacketTrunk-4 TXC-05870 Data Sheet: *PRODUCT PREVIEW* Edition 2, August 2004

The page numbers indicated below of this updated Data Sheet include significant changes relative to the previous data sheet.

Page Number of Updated Data Sheet	Summary of the Change
All	Changed edition number and date. Changed *PRODUCT PREVIEW* to *PRELIMINARY*
Where Occurred	Removed v.35 and STS-1.
2 - 13	Updated Table of Contents, List of Figures and List of Tables.
14	Changed List of Data Sheet Changes section.
17	Changed second paragraph above Figure 1-1. Basic PacketTrunk-4-Based System with 4 User Ports.
25	Changed Figure 2-2. PacketTrunk-4 Bottom View by removing pin numbers. Removed redundant Figure 2-3. PacketTrunk-4 Pinning – Bottom View, which followed Figure 2-2. PacketTrunk-4 Bottom View.
26 - 37	Removed Pin No. column from Table 2-2. SDRAM Interface Pinout through Table 2-12. Power Supply Pins.
38	Added Maximum Power parameter in Table 2-13. Absolute Maximum Ratings.
39	Changed “Core supply current (3.3V)" to "I/O supply current (3.3V)" in Table 2-15. DC Characteristics (VDDC = 2.5V ± 0.2V and VDDE = 3.3V ± 0.3V, TA = –40 to +85×C) (1/2).
96	Changed Descriptions for Bits 26 and 25:16 in Table 3-70. General_cfg_reg0 Map.
100	Changed Descriptions for Bits 3:2 in Table 3-70. General_cfg_reg0 Map (Cont.).
103	Changed Descriptions for Bits 3:2 in Table 3-73. Port2_cfg_reg Map (Cont.).
106	Changed Descriptions for Bits 3:2 in Table 3-74. Port3_cfg_reg Map (Cont.).
109	Changed Descriptions for Bits 3:2 in Table 3-75. Port4_cfg_reg Map (Cont.).
117	Changed text above Table 3-86. General_stat_reg Map. Changed Descriptions for Bits 27 through 16, 11, 10 in Table 3-86. General_stat_reg Map.
118	Changed Description for Bits 9 and 8 in Table 3-86. General_stat_reg Map (Cont.).
119	Added text above Table 3-87. Ports_stat_reg Map.Changed Descriptions for Bits 19 through 8 in Table 3-87. Ports_stat_reg Map.
120	Changed Description for in Table 3-88. Version Register Map.
122	Changed Descriptions for Bits 31:22 in Table 3-90. AAL1 Per-bundle Configuration [63:32] Map.
126	Changed Descriptions for Bits 29:20 in Table 3-92. AAL1 Per-bundle Configuration [127:96] Map.
132	Changed Descriptions for Bits 29:20 in Table 3-96. AAL2 Per-bundle Configuration [127:96] Map.
144	Added text below the Status Tables heading.
147	Changed Descriptions for Bits 12:7 and 6:0 in Table 3-135. Bank1 Per Timeslot Assignment Map.

Page Number of Updated Data Sheet	Summary of the Change
148	Changed Descriptions for Bits 12:7 and 6:0 in Table 3-136. Bank2 Per Timeslot Assignment Map.
158	Changed Table 3-167. Per Bundle Head Map.
165	Added Note below Table 3-182. Control Word Map.
165	Added text below the Clock Recovery heading.
166	Changed Descriptions for Bits 1:0 in Table 3-183. Receive Conditioning Data Map.
170	Changed Description for Bit 3 in Table 3-191. MAC Network Control Register Map.
190	Added Note and last sentence of paragraph at the end of the Clock Recovery section.
209	Changed paragraph above Figure 4-30. Jitter Buffer Parameters.

1.0 INTRODUCTION

1.1 TDMOIP CONCEPT

TDM over IP (TDMoIP) is a transport technology that provides a simple conversion strategy of TDM to IP or MPLS or pure Ethernet layer 2 networks. TDMoIP carries E1, T1, E3 or T3 services and serial data across a packet-switched network, transparent to all protocols and signaling. TDMoIP enables service providers to migrate to next generation networks while continuing to provide all their revenue-generating legacy voice and data services. TDMoIP also benefits data carriers by enabling them to offer lucrative leased-line services. This lets them maximize revenues from their packet-switched infrastructure by selling voice services as well as data. It enables enterprises to run voice and video over the same IP/Ethernet-based network that is currently used to run only LAN traffic, thereby minimizing network maintenance and operating costs.

Packet-switched networks, such as IP networks, were not designed to transport TDM data, and so have no inherent clock distribution mechanism. Hence, when transporting TDM over packet switched networks, the receiver needs to reconstruct the transmitter's TDM clock. TDMoIP ensures that recovered clock jitter and wander levels conform to ITU-T G.823/824, even for networks that introduce high packet delay variation and packet loss.

TDMoIP complements VoIP in those cases where VoIP is not applicable, and in those cases where VoIP price/performance is not sufficient. Most importantly, TDMoIP provides higher voice quality with much lower latency than VoIP. Unlike VoIP, TDMoIP can support all applications that run over E1/T1 circuits, not just voice. TDMoIP can provide traditional leased-line services over IP, and is transparent to protocols and signaling. Because TDMoIP provides an evolutionary (as opposed to revolutionary approach), investment protection is maximized.

Now that the communications market economy has become business oriented, customers are seeking solutions that can be smoothly integrated into their existing infrastructure and end-user equipment. TDMoIP offers transparent interworking between the most common network technologies – SDH/SONET, Frame-Relay, Ethernet, and ATM.

1.2 PACKETTRUNK-4 ARCHITECTURE

Overview

The PacketTrunk-4 is a highly integrated ASIC designed for use in a wide variety of network access applications. It serves as a building block for TDM over IP or TDM over MPLS gateway implementation. The PacketTrunk-4 provides a very high degree of system integration and cost effectiveness together with enhanced performance. The chip provides a scalable solution, from single chip-based systems to multiple chip-based systems.

The PacketTrunk-4 enables transparent transport of legacy TDM traffic over Packet-Switched-Networks (PSN). The chip implements the following payload methods of TDM transfer over IP/MPLS:

- AAL1-like method for circuit emulation (Constant Bit Rate or static allocation of TDM timeslots)
- AAL2-like method for loop emulation (dynamic allocation of TDM timeslots)
- HDLC method for efficient transfer or termination of frame based traffic.

A dedicated payload type machine implements each method.

The AAL1 payload type machine converts E1/T1/E3/T3/serial data flow into IP/MPLS packets, and vice versa. It supports E1/T1 structured mode with/without CAS, using a timeslot size of 8 bits, or unstructured mode (carrying serial interfaces, unframed E1/T1 or E3/T3 traffic).

The AAL2 payload type machine converts E1/T1 data flow into IP/MPLS packets, and vice versa, implementing dynamic timeslot allocation. It supports E1/T1 structured mode with/without CAS with 8-bit timeslot resolution.

The HDLC payload type machine converts and terminates HDLC-based E1/T1/serial flow into IP/MPLS packets and vice versa. It supports 2-, 7- and 8-bit timeslot resolution (i.e. 16, 56, and 64 kbps respectively (not supported for T1 SF interface)), as well as N × 64 kbps bundles (n=1 to 32). Supported applications of this machine include:

- Trunking of HDLC-based traffic (such as Frame Relay) implementing Dynamic Bandwidth Allocation over IP/MPLS networks
- HDLC-based signaling interpretation (such as ISDN D-channel signaling termination – BRI or PRI, V5.1/2, or GR-303).

The PacketTrunk-4 external connections are illustrated in the figures below.

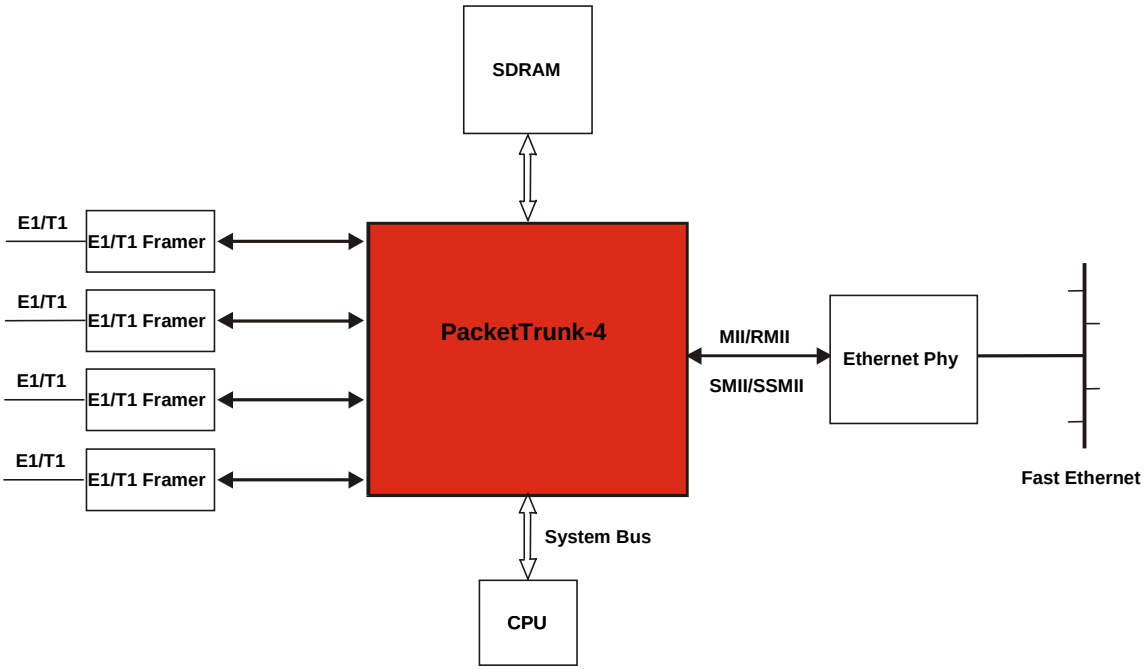


Figure 1-1. Basic PacketTrunk-4-Based System with 4 User Ports

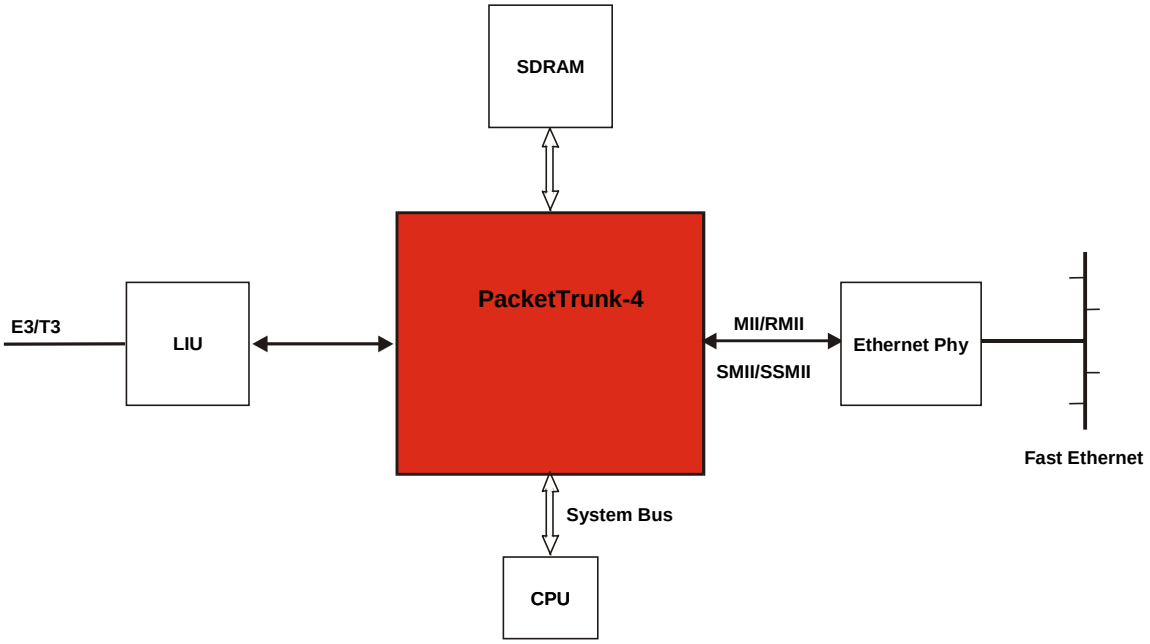


Figure 1-2. Basic PacketTrunk-4-Based System with One User Port

User Ports

The user side consists of either a single high-speed E3 or T3, or four ports, each independently supporting E1, T1 or serial data transfer.

- For the E3/T3 option, the AAL1 method is used.

For the E1/T1 option, the chip supports the following operation modes:

- Unframed – E1/T1 pass-through mode (AAL1 or HDLC payload type method)
- Structured – fractional E1/T1 support (all payload type methods)
- Structured with CAS – fractional E1/T1 with CAS support (AAL1 or AAL2 payload type method).
- The serial port option supports synchronous data transfer over the IP/MPLS network for legacy data services (such as Frame Relay or arbitrary continuous bit stream).

The serial port option supports the following synchronous serial data formats:

- Arbitrary continuous bit stream (using AAL1 payload type method)
  - In single-port high-speed mode, the first port operates at up to DS3 rate (44.736 Mbps) and the other ports are disabled. In this mode, the whole traffic is transferred using a single bundle/connection.
  - In four-port low-speed mode each port can operate at up to 4.64 Mbps with an aggregate rate of 9.3 Mbps.
- HDLC-based traffic (such as Frame Relay transferred using HDLC payload type method). Only the four-port low-speed mode is relevant (up to 4.64 Mbps with an aggregate rate of 9.3 Mbps).

All serial interface modes are capable of working with a gapped clock.

Network Port

The chip features one 10/100 ETH port with the option of MII/RMII/SMII/S-SMII. The port can work in half/full duplex mode and supports VLAN tagging and priority labeling according to 802.1 p&Q, including VLAN stacking.

Bundles

A bundle is defined as a stream of bits that have originated from the same physical interface and which are transmitted from a TDMoIP source device to a TDMoIP destination device. For example, bundles may comprise any number of 64 kbps timeslots originating from a single E1/T1, or an entire T3 or E3. Bundles are single direction streams, frequently coupled with bundles in the opposite direction to enable full duplex communications. More than one bundle can be transmitted between two TDMoIP edge devices.

Each bundle in the PacketTrunk-4 is transmitted using one of the payload type methods implemented the following:

- TDMoIP using either AAL1 or AAL2 payload type method
- TDMoMPLS using either AAL1 or AAL2 payload type method
- HDLCoIP
- HDLCoMPLS

The chip supports up to 64 bundles. The maximum number of bundles that can be allocated per port is configurable as follows:

Table 1-1. Maximum Bundle Allocation

Allocation No	Port 1	Port 2	Port 3	Port 4
1.	32	Disabled	32	Disabled
2.	32	Disabled	16	16
3.	16	16	32	Disabled
4.	16	16	16	16

Each TDMoIP bundle/connection may be assigned to one of the payload type machines or to the CPU. For E1/T1, the chip provides internal bundle cross-connect functionality, with DS0 resolution.

Clock Recovery

Sophisticated TDM clock recovery mechanisms, one for each E1/T1 interface, allow end-to-end TDM clock synchronization despite packet delay variation of the IP/MPLS network. These mechanisms provide both fast frequency acquisition and highly accurate phase tracking. Jitter and wander of the recovered clock are maintained at levels that conform to TDM standards (G.823/824).

Delay Variation Compensation

Large configurable jitter buffers, on a per bundle basis, that compensate for the delay variation introduced by the IP/MPLS network, with the following maximal depths:

- E1: up to 256 ms
- T1: unframed – up 340 ms
  - framed – up to 256 ms
  - framed with CAS – up to 192 ms
- E3: up to 60 ms
- T3: up to 46 ms
- Serial:depending on the interface rate (e.g., for 512 kbps up to 1 sec)

CAS Support

For structured with CAS bundles (with AAL1/AAL2), the chip provides termination of E1/T1 CAS signaling, eliminating the need for CPU intervention.

CPU Interface

The CPU interface provides a connection to a host with a 16/32-bit data bus. This allows configuration of chip control registers and statistics collection using the chip counters and status registers. It also provides access to the CPU transmit and receive buffers in the SDRAM, used for packets directed to/originating from the CPU (such as ARP, SNMP, etc.).

1.3 FEATURES

The PacketTrunk-4 offers:

- TDM Interfaces
  - 4 E1/T1/serial or
  - Single E3/T3
- Ethernet Interface
  - A single 10/100 Mbps, MII/RMII/SMII/SSMII
  - Half/Full duplex
  - VLAN support according to 802.1 p&Q
  - Fully compatible with IEEE 802.3 standard
- End-to-end TDM synchronization through the IP/MPLS domain by four independent on-chip TDM clock recovery mechanisms, on a per-port basis
- 64 independent bundles/connections, each with its own:
  - Transmit and receive queues
  - Configurable jitter-buffer depth
  - Connection level redundancy, with traffic duplication option

- Internal bundle cross-connect capability, with DS0 resolution
- Packet loss compensation and handling of misordered packets
- Glueless SDRAM interface
- Complies with MPLS-Frame Relay Alliance Implementation Agreements 4.0 and 5.0
- Conforms to drafts submitted to IETF
- 256-pin PBGA package

1.4 APPLICATIONS

The following features make the PacketTrunk-4 an attractive building block for implementing TDMoIP gateways:

- Scalability:
  - A single chip is suitable for a small, customer-located gateway
  - Multi-chip architecture enables building larger capacity devices for a large customer site or for central sites, where traffic is handed over to the PSTN
- Cost-effectiveness:
  - Per-port clock recovery mechanism eliminates the need for any additional timing-related hardware
  - Simplified designs due to minimized peripheral components required
  - Low-cost SDRAM
- Flexible protocol stack implementation

General TDMoIP Applications

Carrier

- Leased-line services over IP
- Private line over metro Ethernet/TLS (Transparent LAN Services)
- Cellular backhaul over IP/Ethernet
- T1/E1 or T3/E3 over Gigabit Ethernet
- SS7 transport over IP

Enterprise

- E1/T1, voice, video and TDM data over Ethernet
- Centralized voice services over Ethernet
- E1/T1 or E3/T3 over Gigabit Ethernet

PacketTrunk-4 Applications

- TDM circuit extension over IP/MPLS/ Ethernet
- Multiservice over unified IP/MPLS/Ethernet
- Point-to-Multipoint TDM connectivity over IP/Ethernet
- HDLC transport over IP/MPLS
- Legacy (Frame Relay, N × 64 kbps) transport over IP/MPLS/Ethernet
- Using a packet back-plane for multiservice concentrators

TDM Circuit Extension over IP/MPLS

The PacketTrunk-4 can serve as a building block for devices which transport TDM traffic over packet networks. AAL1 payload type usage enables providing a constant bit-rate end-to-end transparent TDM circuit extension, while the AAL2 payload type enables, on the fly, signaling/control dependent, dynamic timeslot allocation, for increased bandwidth efficiency.

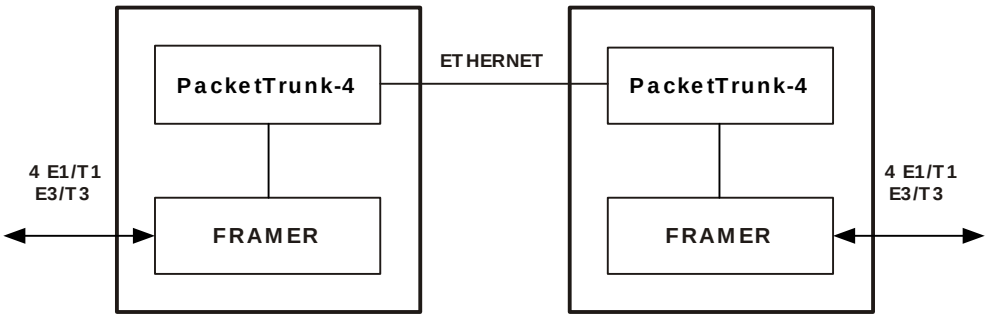


Figure 1-3. TDM Extension over an Ethernet Network

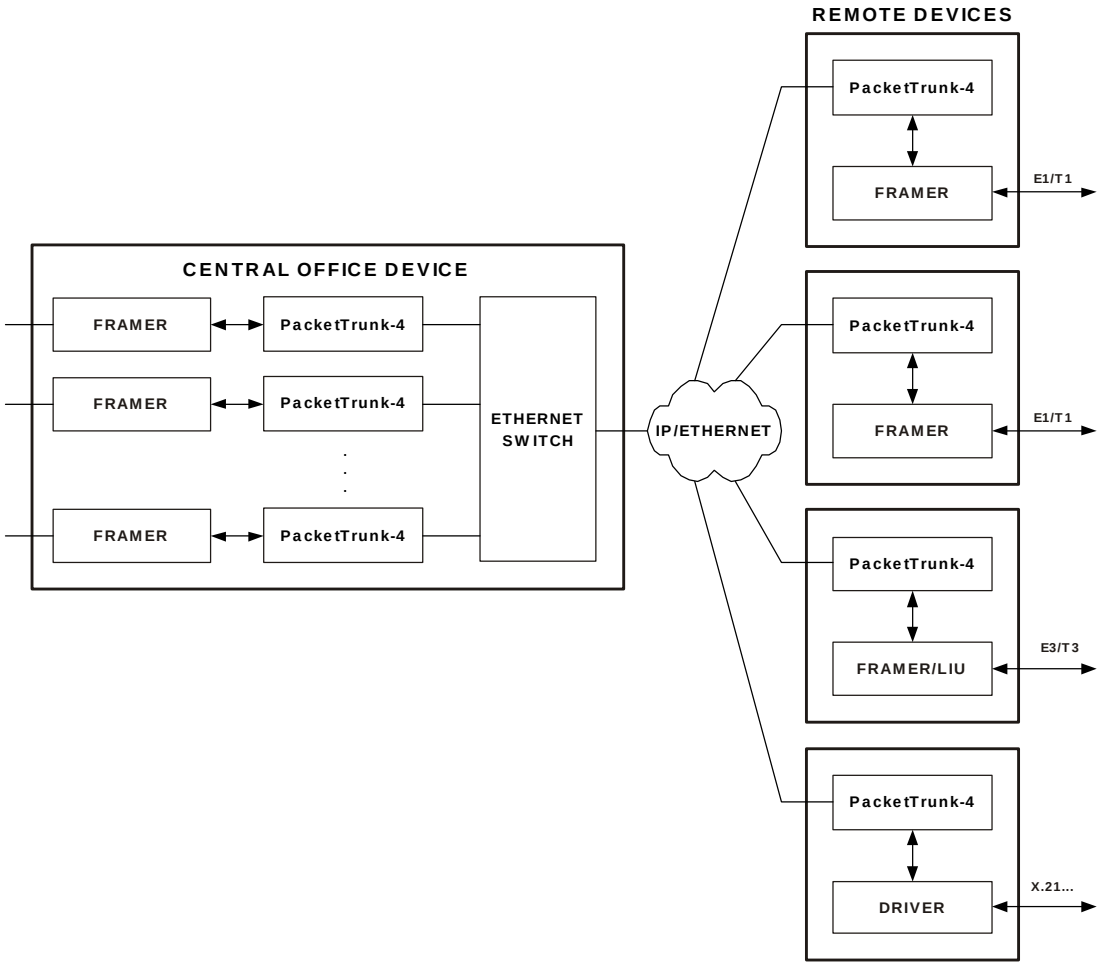


Figure 1-4. Multi-service over Unified IP/Ethernet

Point-to-Multipoint TDM Connectivity over IP/Ethernet

The PacketTrunk-4 supports DS0 level multiple bundles/connections with and without CAS (Channel Associated Signaling). There is no need for an external TDM cross-connect, as the packet domain can be used as a virtual cross-connect; any bundle of timeslots can be directed to another remote location on the packet domain.

HDLC Transport over IP/MPLS

TDM traffic streams often contain HDLC-based control and data traffic. Ideally, these data streams, when transported over a packet domain, should have been treated differently than the TDM time-sensitive payload. The PacketTrunk-4 includes HDLC controller capability, which enables termination of the HDLC frames. HDLC-based control protocols, such as ISDN BRI and PRI, SS7 etc. and other frame-based traffic, can be managed and transported.

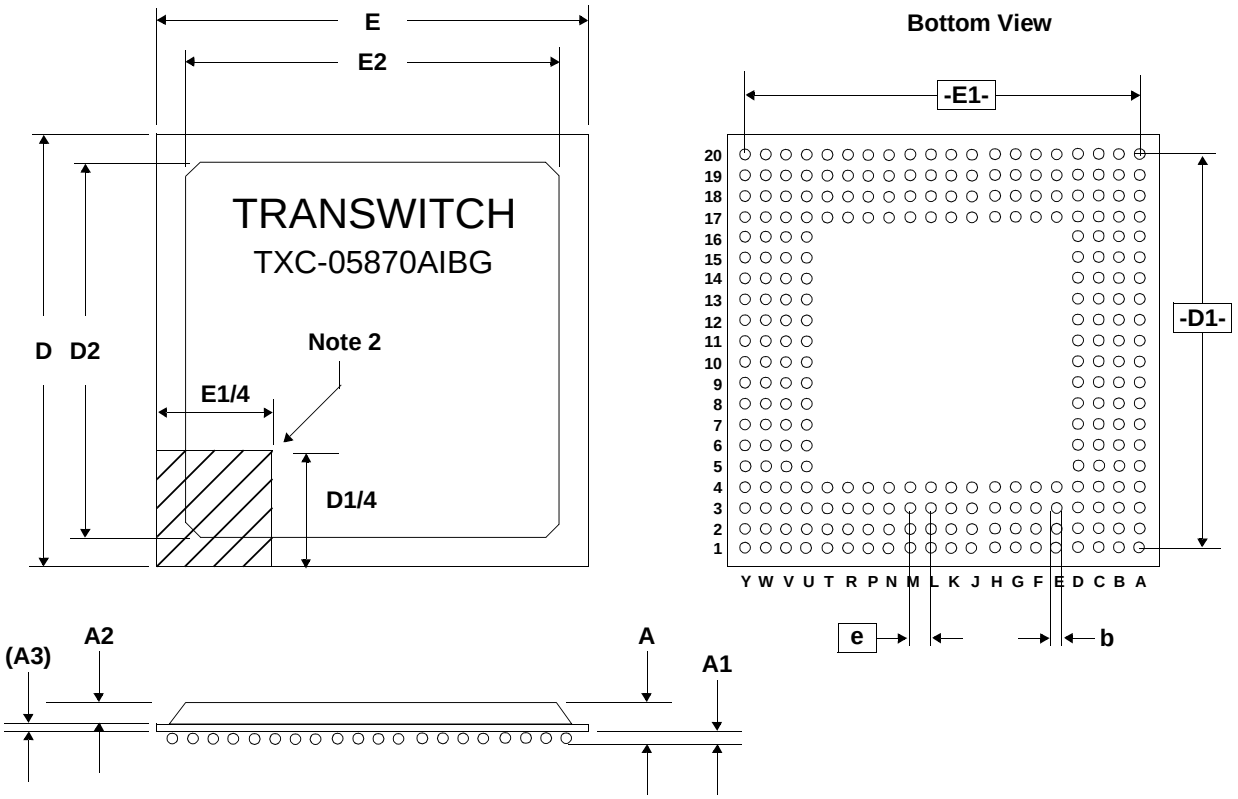
Using a Packet Backplane for Multiservice Concentrators

Using all the above-mentioned capabilities internally inside a communication device enables using a packet-based backplane instead of the more expensive TDM bus option. This enables a cost/effective and future-proof design of communication platforms with full support for both legacy and next-generation services.

2.0 PHYSICAL DESCRIPTION

2.1 PACKAGE AND DIMENSIONS

The PacketTrunk-4 is packaged in a 256-pin plastic BGA, shown in Figure 2-1. PacketTrunk-4 Packaging. Dimensions are provided in Figure 2-1. PacketTrunk-4 Packaging.



Dimension (Note 1)	Min	Max
A (Nom.)	2.13	
A1 A2	0.6 1.53	
A3 (Nom.)	0.36	
b (Ref.)	0.75	
D	27.00	
D1 (Nom.)	24.13	
D2	24.0	
E	27.00	
E1 (Nom.)	24.13	
E2	24.0	
e (Ref.)	1.27	

Notes:
1. All dimensions are in millimeters. Values shown are for reference only.
2. Identification of the solder ball A1 corner is contained within this shaded zone. This package corner may be a 90° angle, or chamfered for A1 identification.
3. Size of array: 22 x 22, JEDEC Code MS-034B

Figure 2-1. PacketTrunk-4 Packaging

2.2 PINOUT DESCRIPTION

The PacketTrunk-4 pinout is illustrated in Figure 2-2. PacketTrunk-4 Bottom View. The pins are described in Table 2-2. SDRAM Interface Pinout through Table 2-12. Power Supply Pins.

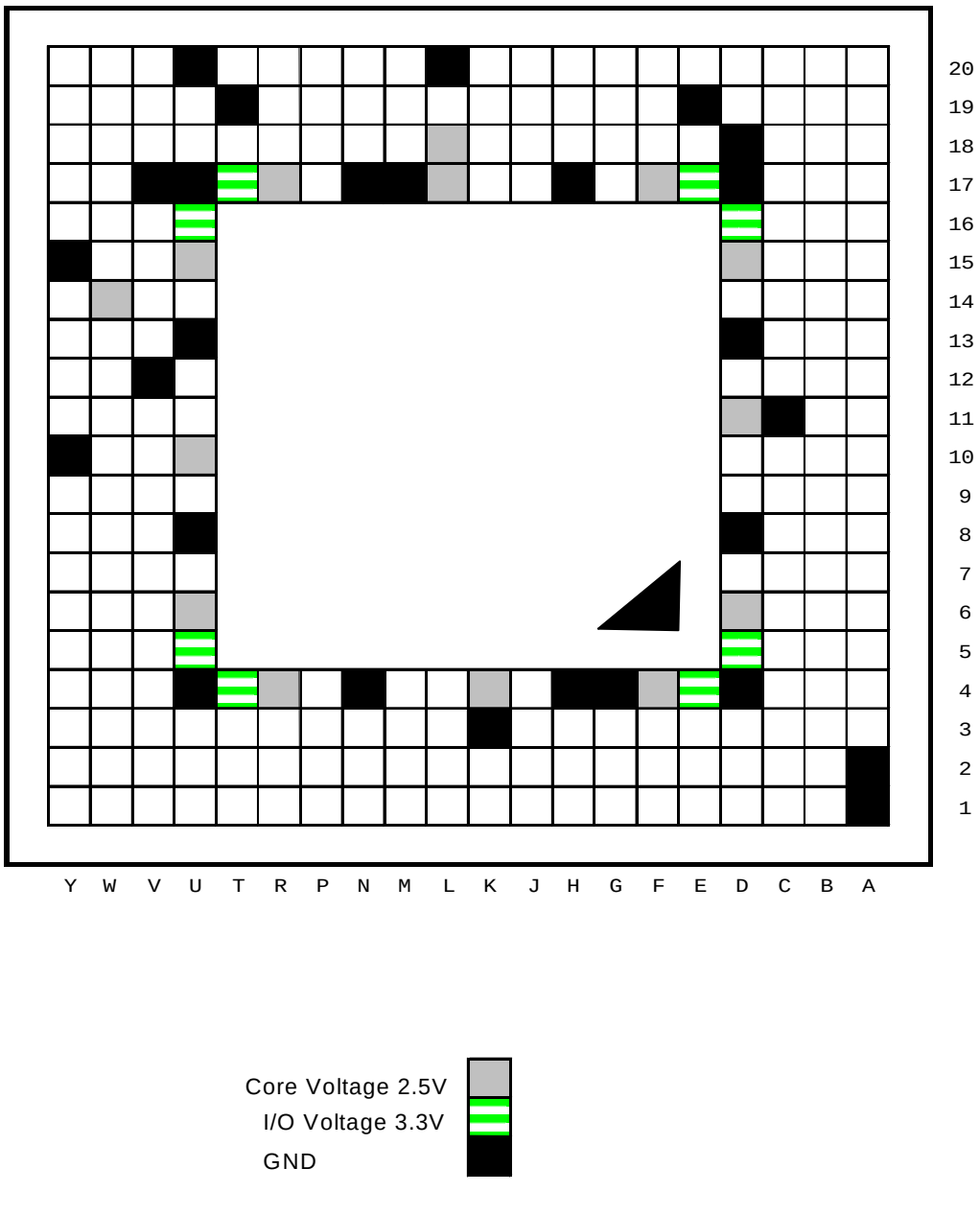


Figure 2-2. PacketTrunk-4 Bottom View

Table 2-2. SDRAM Interface Pinout

Pin Name	Location	Type	Description
SD_D (31:0)		In/Out 6 mA	Data bus towards SDRAM. MSB is SD_D[31]
SD_D (31)	U12		
SD_D (30)	Y12		
SD_D (29)	U11		
SD_D (28)	V11		
SD_D (27)	W11		
SD_D (26)	Y11		
SD_D (25)	V10		
SD_D (24)	W10		
SD_D (23)	Y9		
SD_D (22)	W9		
SD_D (21)	V9		
SD_D (20)	U9		
SD_D (19)	Y8		
SD_D (18)	W8		
SD_D (17)	V8		
SD_D (16)	Y7		
SD_D (15)	W7		
SD_D (14)	V7		
SD_D (13)	Y6		
SD_D (12)	W6		
SD_D (11)	U7		
SD_D (10)	V6		
SD_D (9)	Y5		
SD_D (8)	W5		
SD_D (7)	V5		
SD_D (6)	Y4		
SD_D (5)	Y3		
SD_D (4)	V4		
SD_D (3)	W4		

Table 2-2. SDRAM Interface Pinout

Pin Name	Location	Type	Description
SD_D (2)	Y2		
SD_D (1)	W3		
SD_D (0)	Y1		
SD_DQM(3:0)		Out 6 mA	Byte Enable towards SDRAM. Serve as a mask. SD_DQM[0] is connected to the least significant byte at SDRAM, while SD_DQM[3] is connected to the most significant byte at SDRAM
SD_DQM(3)	W20		
SD_DQM(2)	Y20		
SD_DQM(1)	W19		
SD_DQM(0)	V18		
SD_A(11:0)		Out 6 mA	Address bus towards SDRAM. MSB is SD_A[11]
SD_A(11)	V16		
SD_A(10)	Y17		
SD_A(9)	W16		
SD_A(8)	V15		
SD_A(7)	U14		
SD_A(6)	Y16		
SD_A(5)	W15		
SD_A(4)	V14		
SD_A(3)	Y14		
SD_A(2)	V13		
SD_A(1)	W13		
SD_A(0)	Y13		
SD_BA(1:0)		Out 6 mA	SDRAM bank select. Selects one bank out of four banks at SDRAM
SD_BA(1)	U19		
SD_BA(0)	V19		
SD_CLK	W12	Out 9 mA	SDRAM clock. Drives the clock towards SDRAM
SD_CS_N	W17	Out 6 mA	SDRAM Chip select towards SDRAM. Active low
SD_WE_N	Y18	Out 6 mA	Write enable towards SDRAM. Active low

Table 2-2. SDRAM Interface Pinout

Pin Name	Location	Type	Description
SD_RAS_N	W18	Out 6 mA	Row Address Strobe towards SDRAM. Active low
SD_CAS_N	Y19	Out 6 mA	Column Address Strobe towards SDRAM. Active low

Table 2-3. First TDM Interface Pinout

Pin Name	Location	Type	Description
TDM1_TX	C2	Out 6 mA	First interface serial transmit line
TDM1_RX	E2	In PU	First interface serial receive line
TDM1_TCLK	B1	In PU	Used for clocking TDM1_TX and TDM1_RX lines for framed interface (PCM), or TDM1_TX in case of unframed interface
TDM1_RCLK	E1	In PU	Used for clocking TDM1_RX line for unframed interface
TDM1_ACLK	C1	Out 6 mA	First interface recovered clock
TDM1_SYNC	D1	In PD	First interface transmit/receive frame sync pulse
TDM1_TX_MF_CD	D3	In/Out PD 3 mA	First interface transmit multiframe sync pulse input for framed interface (PCM), or Carrier Detect output in case of serial interface
TDM1_RX_MF	E3	In PD	First interface receive multiframe sync pulse
TDM1_TSIG_CTS_D2A CS_N	D2	Out 6 mA	First interface transmit signaling , or Clear to Send in case of serial interface or external D2A active low chip select
TDM1_RSIG_RTS	F3	In PU	First interface serial RX signaling input or Request To Send input in case of serial interface

Table 2-4. Second TDM Interface Pinout

Pin Name	Location	Type	Description
TDM2_TX_D2AWR_N	G2	Out 6 mA	Second interface serial transmit line or external DAC active low write enable
TDM2_RX	F2	In PU	Second Interface serial receive line
TDM2_TCLK	G1	In PU	Used for clocking TDM2_TX and TDM2_RX lines for framed interface (PCM), or TDM2_TX in case of unframed interface
TDM2_RCLK	F1	In PU	Used for clocking TDM2_RX line for unframed interface
TDM2_ACLK_ D2A0	H1	Out 6 mA	Second interface recovered clock or external DAC data [0]
TDM2_SYNC	H2	In PD	Second interface transmit/receive frame sync pulse
TDM2_TX_MF_CD	H3	In/Out PD 3 mA	Second interface transmit multiframe sync pulse input for framed interface (PCM), or Carrier Detect output in case of serial interface
TDM2_RX_MF	G3	In PD	Second interface receive multiframe sync pulse
TDM2_TSIG_CTS_D2A1	J3	Out 6 mA	Second interface serial TX signaling or Clear to Send for serial interface or external DAC data [1]
TDM2_RSIG_RTS	J4	In PU	Second interface serial RX signaling input or Request To Send input in case of serial interface

Table 2-5. Third TDM Interface Pinout

Pin Name	Location	Type	Description
TDM3_TX_D2A2	K2	Out 6 mA	Third Interface serial transmit line or external DAC data [2]
TDM3_RX	L2	In PU	Third interface serial receive line
TDM3_TCLK	K1	In PU	Used for clocking TDM3_TXand TDM3_RX lines for framed interface (PCM), or TDM3_TX for unframed interface
TDM3_RCLK	L1	In PU	Used for clocking TDM3_RX line for unframed interface
TDM3_ACLK_D2A3	J1	Out 6 mA	Third interface recovered clock, or external DAC data [3]
TDM3_SYNC	M1	In PD	Third interface transmit/receive frame sync pulse
TDM3_TX_MF_CD	L3	In/Out PD 3 mA	Third interface transmit multiframe sync pulse input for framed interface (PCM), or Carrier Detect output for serial interface
TDM3_RX_MF	M2	In PD	Third interface receive multiframe sync pulse
TDM3_TSIG_CTS_D2A4	J2	Out 6 mA	Serial TX signaling, or Clear to Send for serial interface, or external DAC data [4]
TDM3_RSIG_RTS	L4	In PU	Third interface serial RX signaling input, or Request To Send input in case of serial interface

Table 2-6. Fourth TDM Interface Pinout

Pin Name	Location	Type	Description
TDM4_TX_D2A5	P2	Out 6 mA	Fourth Interface serial transmit line or external DAC data [5]
TDM4_RX	N2	In PU	Fourth interface serial receive line
TDM4_TCLK	P1	In PU	Used for clocking TDM4_TX and TDM4_RX lines for framed interface (PCM), or TDM4_TX for unframed interface
TDM4_RCLK	N1	In PU	Used for clocking TDM4_RX line for unframed interface
TDM4_ACLK_D2A6	R1	Out 6 mA	Fourth interface recovered clock or external DAC data [6]
TDM4_SYNC	M3	In PD	Fourth interface transmit/receive frame sync pulse
TDM4_TX_MF_CD	R2	In/Out PD 3 mA	Fourth interface transmit multiframe sync pulse input for framed interface (PCM), or Carrier Detect output for serial interface
TDM4_RX_MF	M4	In PD	Fourth interface receive multiframe sync pulse
TDM4_TSIG_CTS_D2A7	P3	Out 6 mA	Serial TX signaling, or Clear to Send for serial interface, or external DAC data [7]
TDM4_RSIG_RTS	N3	In PU	Fourth interface serial RX signaling input, or Request To Send input in case of serial interface

Table 2-7. MII/RMII/SMII/SSMII Interface Pinout

Pin Name	Location	Type	Description
CLK_MII_RX	N20	In	MII receive clock or SSMII receive clock
MII_RXD(0)	N19	In	MII receive data (0), or SMII/SSMII receive data
MII_RXD(1)	N18	In	MII receive data (1), or receive SSMII sync
MII_RXD(3)	M20	In	MII receive data (3), or RMII receive data (1)
MII_RXD(2)	M19	In	MII receive data (2), or RMII receive data (0)
MII_RX_DV	P20	In	MII receive data valid, or RMII carrier sense/data valid
MII_RX_ERR	M18	In	MII receive error, or RMII receive error
MII_COL	P18	In	MII collision detection
MII_CRS	U18	In	MII carrier sense
CLK_MII_TX	T18	In	MII transmit clock or RMII ref clock or SMII/SSMII ref clock
CLK_SSMII_TX	V20	Out 12 mA	SSMII transmit clock (125MHz)
MII_TXD(0)	T20	Out 9 mA	MII transmit data (0), or SMII/SSMII transmit data
MII_TXD(1)	R18	Out 9 mA	MII transmit data (1), or SMII sync or SSMII transmit sync
MII_TXD(3)	R20	Out 6 mA	MII transmit data (3), or RMII transmit data (1)
MII_TXD(2)	R19	Out 6 mA	MII transmit data (2), or RMII transmit data (0)
MII_TX_EN	P19	Out 6 mA	MII transmit enable or RMII transmit enable
MII_TX_ERR	P17	Out 6 mA	MII transmit error
MDIO	L19	In/Out PU,TS 6 mA	Management data, synchronized to MDC
MDC	K20	Out 6 mA	Management data clock

Table 2-8. Host Interface Pinout

Pin Name	Location	Type	Description
H_D (31:0)		In/Out 6 mA	Host data bus MSB is: • H_D[31] when Host data bus width is 32 bits • H_D[15] when Host data bus width is 16 bits When using 16-bit data bus, H_D[31:16] should be connected to pull-up resistors.
H_D (31)	C10		
H_D (30)	B10		
H_D (29)	A10		
H_D (28)	A11		
H_D (27)	B11		
H_D (26)	A12		
H_D (25)	B12		
H_D (24)	C12		
H_D (23)	D12		
H_D (22)	A13		
H_D (21)	B13		
H_D (20)	C13		
H_D (19)	A14		
H_D (18)	B14		
H_D (17)	C14		
H_D (16)	A15		
H_D (15)	B15		
H_D (14)	D14		
H_D (13)	C15		
H_D (12)	A16		
H_D (11)	B16		
H_D (10)	C16		
H_D (9)	A17		
H_D (8)	A18		
H_D (7)	C17		
H_D (6)	B17		
H_D (5)	B18		
H_D (4)	A19		

Table 2-8. Host Interface Pinout (Cont.)

Pin Name	Location	Type	Description
H_D (3)	A20		
H_D (2)	B19		
H_D (1)	C18		
H_D (0)	B20		
H_AD (24:1)		In	Host address bus, MSB is H_AD[24]
H_AD (24)	B2		
H_AD (23)	B3		
H_AD (22)	C4		
H_AD (21)	A3		
H_AD (20)	B4		
H_AD (19)	C5		
H_AD (18)	A4		
H_AD (17)	B5		
H_AD (16)	C6		
H_AD (15)	D7		
H_AD (14)	A5		
H_AD (13)	B6		
H_AD (12)	C7		
H_AD (11)	A6		
H_AD (10)	B7		
H_AD (9)	A7		
H_AD (8)	C8		
H_AD (7)	B8		
H_AD (6)	A8		
H_AD (5)	D9		
H_AD (4)	C9		
H_AD (3)	B9		
H_AD (2)	A9		
H_AD (1)	D10		
			Should be connected to GND when the Host data bus is 32 bits
H_CS_N	G20	In	Host chip select
H_R_W_N	G19	In	Host read/write
H_WR_BE0_N	F19	In	H_D[7:0] write enable, active low
H_WR_BE1_N	G18	In	H_D[15:8] write enable, active low

Table 2-8. Host Interface Pinout (Cont.)

Pin Name	Location	Type	Description
H_WR_BE2_N	E20	In	H_D[23:16] write enable, active low
H_WR_BE3_N	G17	In	H_D[31:24] write enable, active low
H_READY_N	F20	Out PU,TS 6 mA	Host ready, active low. This pin requires the use of an external pull-up resistors. The signal is actively driven high (‘1’) before it becomes tri-state.
H_INT	D20	Out 6 mA	Host Interrupt, active low
DAT_32_16_N	E18	In PU	Selects the host data bus width: ‘0’ – 16 bits ‘1’– 32 bits
H_CPU_SYNC	C19	In PD	‘0’ – for asynchronus CPU operation (different clock source is provided to the chip Clk Sys and the CPU) ’1’ – for synchronus CPU operation (the same clock source is provided to the chip Clk Sys and the CPU)
RST_SYS_N	J17	In PU	System reset, active low

Table 2-9. Boundary Scan Pinout

Pin Name	Location	Type	Description
JTAG_TMS	K19	In PU	Boundary scan JTAG interface, Test Mode Select
JTAG_TCK	K18	In PD	Boundary scan JTAG interface, test clock
JTAG_TDI	K17	In PU	Boundary scan JTAG interface, test data in
JTAG_TDO	J20	Out TS, PU 6 mA	Boundary scan JTAG interface, test data out
JTAG_TRST	J19	In PD	Boundary scan JTAG interface, test reset, active low

Table 2-10. Factory Tests Pinout

Pin Name	Location	Type	Description
TMC1	C20	In	Used for factory tests. **Must be** tied to GND for normal operation.
TMC2	D19	In	Used for factory tests. **Must be** tied to GND for normal operation.
BIST_TDO	J18	Out TS, PU 6 mA	Used for factory tests. **Must be** left unconnected for normal operation.
BIST_TRST	H20	In PD	Used for factory tests. **Must be** tied to GND for normal operation. Active low.
SCAN_EN	H19	In PD	Used for factory tests. **Must be** tied to GND for normal operation.
TEST_MODE	H18	In PD	Factory tests logic enable, active high. **Must be** tied to GND for normal operation.

Table 2-11. Clocks

Pin Name	Location	Type	Description
CLK_HIGH	C3	In	38.88 MHz clock input used for E1/T1 clock recovery machines. Must be tied to GND when none of the recovered clock outputs (TDM1_ACLK-TDM4_ACLK) are used or when the chip is in single-port high-speed mode. For more details about clock accuracy, refer to the Clock Recovery section.
CLK_SYS	F18	In	Clock input used to drive all the chip internal circurity, 50 or 75 MHz (±50 ppm or better)

Notes: For all tables:

- PU – Input/Output with internal Pull-Up resistor
- PD – Input/Output with internal Pull-Down resistor
- TS – Tri-State output

Table 2-12. Power Supply Pins

Pin Name	Location	Description
GND	A1, Y10, Y15, U20, L20, A2, T19, E19, K3, V12, V17, D18, C11, D4, G4, H4, N4, U4, U8, U13, U17, N17, M17, H17, D17, D13, D8	Ground pin
VDDC (2.5V)	W14, L18, F4, K4, R4, U6, U10, U15, R17, L17, F17, D15, D11, D6	Core supply voltage
VDDE (3.3V)	E4, T4, U5, U16, T17, E17, D16, D5	I/O supply voltage
NC	T1, U1, V1, W1, T2, U2, V2, W2, R3, T3, U3, V3, P4	NC pins should be left unconnected

2.3 DC/AC CHARACTERISTICS

DC Characteristics

Absolute Maximum Ratings

Table 2-13. Absolute Maximum Ratings

Parameter	Symbol	Conditions	Ratings	Unit
Core supply voltage	V_{DDC}		–0.5 to +3.6	V
I/O supply voltage	V_{DDE}		–0.5 to +4.6	V
I/O voltage	V_I/V_O	$V_I/V_O < V_{DDE} + 0.5V$	–0.5 to +4.6	V
Output current	I_O	I_{OL} = 3 mA	10	mA
		I_{OL} = 6 mA	20	mA
		I_{OL} = 9 mA	30	mA
		I_{OL} = 12 mA	40	mA
Maximum Power	P_{MAX}	+ 25 °C	500	mW
Operating temperature	T_A		–40 to +85	°C
Junction temperature	T_j		–40 to +125	°C
Storage temperature	T_{stg}		–65 to +150	°C

Caution: Product quality may suffer if the absolute maximum rating is exceeded for even a single parameter, or even momentarily. In other words, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

Note: Always apply voltage to I/O pins after determining the power supply voltage.

Recommended Operating Conditions

Table 2-14. Recommended Operating Conditions

Parameter	Symbol	Min.	Typ	Max.	Unit
Core supply voltage	V_{DDC}	2.3	2.5	2.7	V
I/O supply voltage	V_{DDE}	3	3.3	3.6	V
Low-level input voltage	V_{IL}	0		0.8	V
High-level input voltage	V_{IH}	2.0		V_{DDE}	V

Note: No power-up sequence is necessary for the supply voltages V_{DDC} (2.5V) and V_{DDE} (3.3V).

Table 2-15. DC Characteristics (V_{DDC} = 2.5V ± 0.2V and V_{DDE} = 3.3V ± 0.3V, T_A = –40 to +85°C) (1/2)

Parameter	Symbol	Conditions	Min.	Typ	Max.	Unit
Static current consumption[Note1]	I_{DDS}	$V_I = V_{DDE}$ or GND		24	480	µA
Off-state current 3.3V output	I_{OZ}	$V_O = V_{DDE}$ or GND			±10	µA
Output short circuit current[Note2]	I_{OS}	V_O = GND			-250	mA
Input leakage current	I_{LI}					
Normal input		$V_I = V_{DDE}$ or GND		$\pm 10^{-4}$	±10	µA
With pull-up resistor (50 kΩ)		V_I = GND	-24	-72	-180	µA
With pull-down resistor (50 kΩ)		$V_I = V_{DDE}$	24	72	180	µA
Core supply current (2.5V)	$I_{DD2.5V}$	CLK_SYS=50 MHz Clock_recovery_en =1		180		mA
		CLK_SYS =50 MHz Clock_recovery_en =0		149		
		CLK_SYS =75 MHz Clock_recovery_en =1		250		
		CLK_SYS =75 MHz Clock_recovery_en =0		208		
I/O supply current (3.3V)	$I_{DD3.3V}$	CLK_SYS =50 MHz Clock_recovery_en =1		22		mA
		CLK_SYS =50 MHz Clock_recovery_en =0		22		
		CLK_SYS =75 MHz Clock_recovery_en =1		24		
		CLK_SYS =75 MHz Clock_recovery_en =0		24		

Notes:

1. If an I/O with internal pull-up or pull-down resistors is used, the static current consumption will increase.
2. The output short circuit time is one second or less and is only for one pin on the LSI.

Table 2-16. DC Characteristics (V_{DDC} = 2.5V ± 0.2V and V_{DDE} = 3.3V ± 0.3V, T_A = –40 to +85°C) (2/2)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Low-level output current	I_{OL}	V_{OL} = 0.4V				
3 mA output type			3.0			mA
6 mA output type			6.0			mA
9 mA output type			9.0			mA
12 mA output type			12.0			mA
High-level output current	I_{OH}	V_{OH} = 2.4V				
3 mA output type			-3.0			mA
6 mA output type			-6.0			mA
9 mA output type			-9.0			mA
12 mA output type			-12.0			mA
Low-level output voltage	V_{OL}	I_{OL} = 0 mA			0.1	V
High-level output voltage	V_{OH}	V_{OL} = 0 mA	V_{DDE} – 0.1			V

Pull-up/Pull-down Resistance Values

Table 2-17. Pull-up/Pull-down Resistance Values (V_{DDC} = 2.5V ± 0.2V and V_{DDE}= 3.3V ± 0.3V, T_A = –40 to +85°C)

Parameter	Library Expression	Min.	Typ	Max.	Unit
Pull-up resistor	50 kΩ	20	45	125	kΩ
Pull-down resistor	50 kΩ	20	45	125	kΩ

Pin Capacitance

Table 2-18. Pin Capacitance

Parameter	Min.	Max.	Unit
Input	5	7	pF
Output	5	7	pF

Note: V_{DD} = 0V, T_J = 25°C, f = 1 MHz.

Thermal Resistance

Table 2-19. Thermal Resistance

Parameter	Symbol	Value	Unit
Junction to ambient	ja	30	°C/W

AC Characteristics

Table 2-20. CLK_SYS - AC Characteristics

Name	Description	50 MHz			75 MHz			Unit
		Min	Typ	Max	Min	Typ	Max	
	CLK_SYS frequency		50			75		MHz
T2	CLK_SYS duty cycle	40		60	40		60	%

Note: CLK_SYS should be 25 ppm or better.

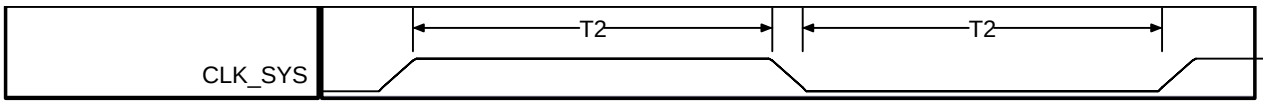


Figure 2-4. CLK_SYS Timing

Table 2-21. RST_SYS_N - AC Characteristics

Name	Description	Min	Max	Unit
T5	RST_SYS_N width	200		µs

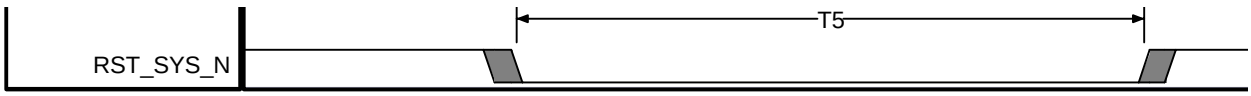


Figure 2-5. RST_SYS_N Timing

Table 2-22. H_INT - AC Characteristics

Name	Description	Min	Max	Unit
T6	CLK_SYS to H_INT output valid		12	ns
T7	CLK_SYS to H_INT output hold	3.5		ns

Notes:

1. Timing reference voltage is 1.5V.
2. The output timing specified assumes 50 pf load.

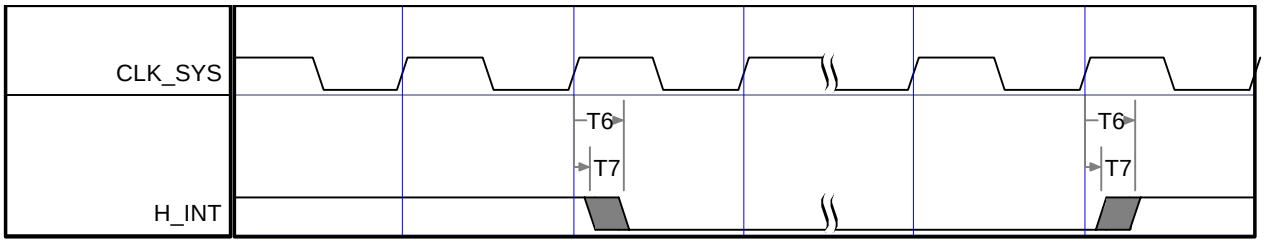


Figure 2-6. CPU Interface – H_INT Timing

Table 2-23. CPU Interface – Synchronous Mode – AC Characteristics

Name	Description	Min	Max	Unit
T11	H_CS_N input setup prior to CLK_SYS	6		ns
T12	H_CS_N input hold after CLK_SYS	1		ns
T13	H_D[31:0] input setup prior to CLK_SYS	5		ns
T14	H_D[31:0] input hold after CLK_SYS	1		ns
T15	H_AD[24:1] input setup prior to CLK_SYS	5		ns
T16	H_AD[24:1] input hold after CLK_SYS	4		ns
T17	H_R_W_N input setup prior to CLK_SYS	4		ns
T18	H_R_W_N input hold after CLK_SYS	3.5		ns
T19	H_WR_BEx_N input setup prior to CLK_SYS	7		ns
T20	H_WR_BEx_N input hold after CLK_SYS	1		ns

Table 2-23. CPU Interface – Synchronous Mode – AC Characteristics (Cont.)

Name	Description	Min	Max	Unit
T21	CLK_SYS to H_D[31:0] output valid		12	ns
T22	H_CS_N deasserted or H_R_W_N low to H_D[31:0] high-Z		7	ns
T23	CLK_SYS to H_D[31:0] output hold	3.5		ns
T24	CLK_SYS to H_READY_N output valid		8	ns
T25	H_CS_N deasserted to H_READY_N output hold	1		ns
T26	H_READY_N active pull up	1.8	3.8	ns

Notes:

1. Synchronous mode is applicable at CLK_SYS = 50 MHz only.
2. Timing reference voltage is 1.5V.
3. The output timing specified assumes 50 pf load.

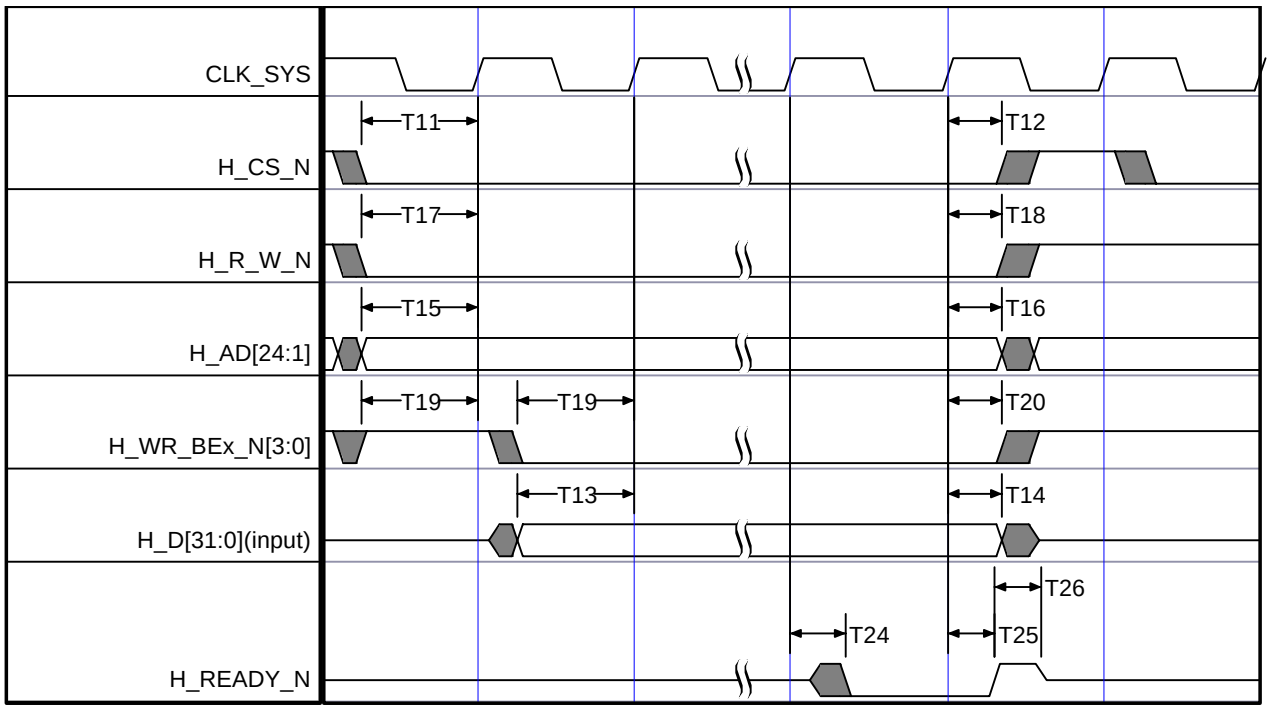


Figure 2-7. CPU Interface – Synchronous Mode – Write Cycle

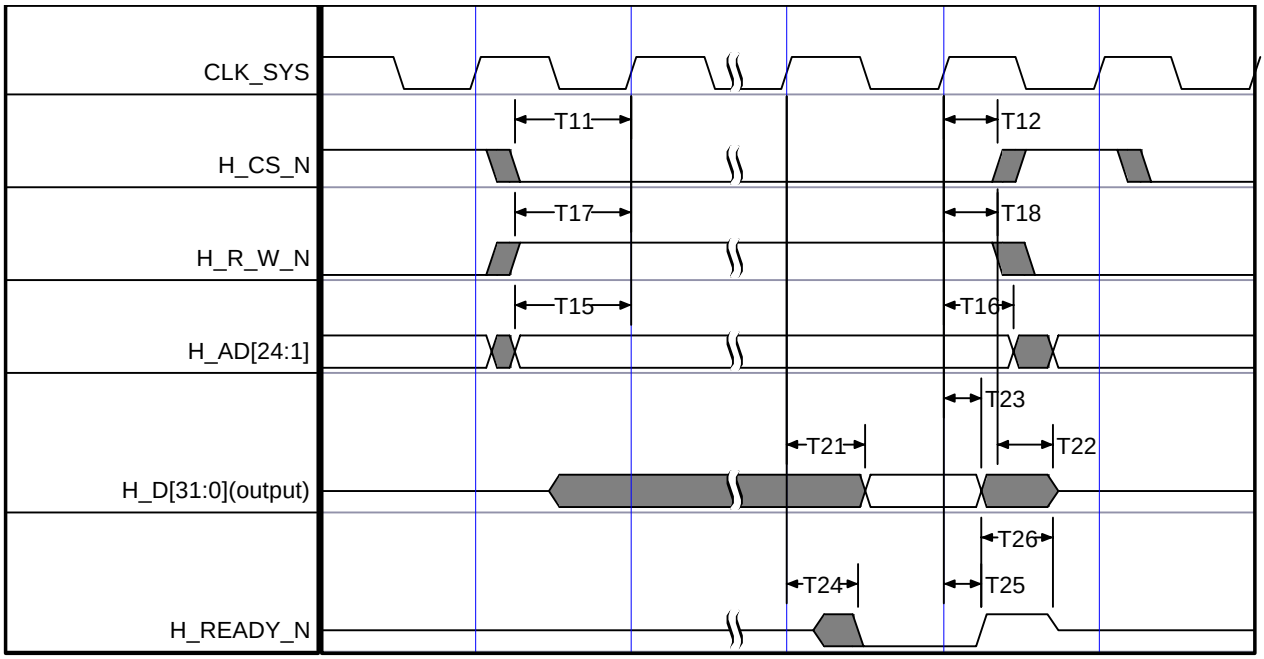


Figure 2-8. CPU Interface – Synchronous Mode – Read Cycle

Table 2-24. CPU Interface – Asynchronous mode – AC characteristics

Name	Description	Min	Max	Unit
T31	Latest between H_WR_BEx_N or H_CS_N - asserted to H_D[31:0] valid		0	ns
T32	H_CS_N deasserted to H_D[31:0] not valid		0	ns
T33	H_CS_N asserted to H_AD[24:1] valid		0	ns
T34	H_CS_N deasserted to H_AD[24:1] not valid		0	ns
T35	H_CS_N asserted to H_R_W_N valid		0	ns
T36	H_CS_N deasserted to H_R_W_N not valid		0	ns
T37	H_CS_N deasserted to H_READY_N high		7	ns
T40	H_CS_N deasserted to H_WR_BEx_N not valid		0	ns
T43	Delay between two successive accesses	1.5		clk_sys cycles
T44	H_D[31:0] valid before H_READY_N active low	1		ns

Notes:

1. Timing reference voltage is 1.5V.
2. The output timing specified assumes 50 pf load.

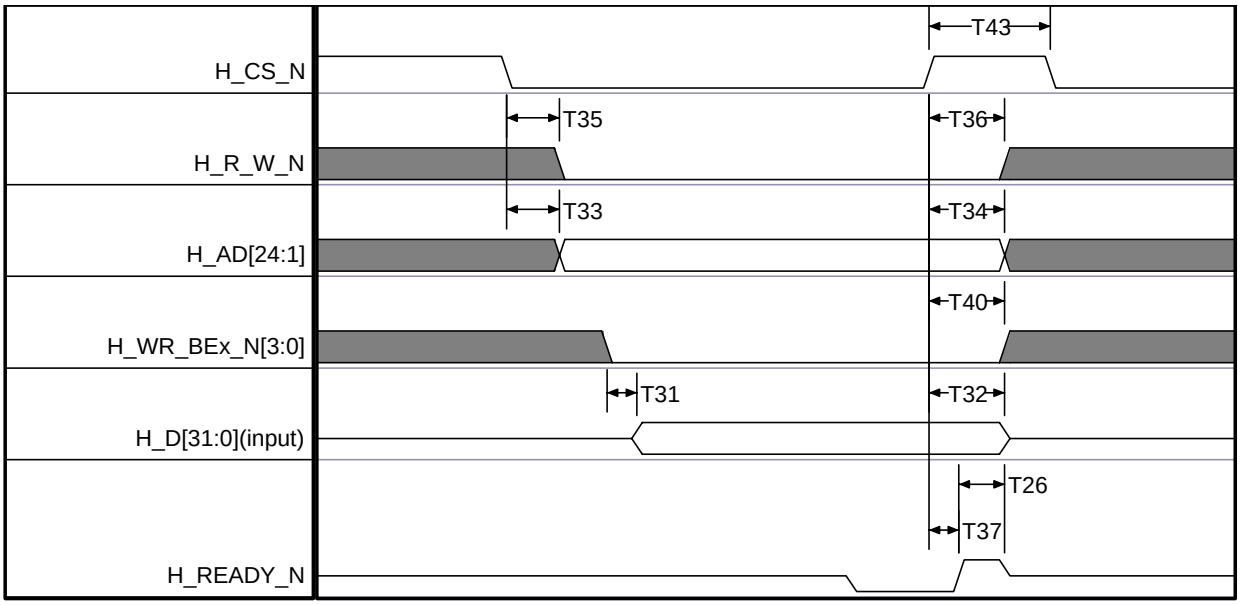


Figure 2-9. CPU Interface – Asynchronous Mode – Write Cycle

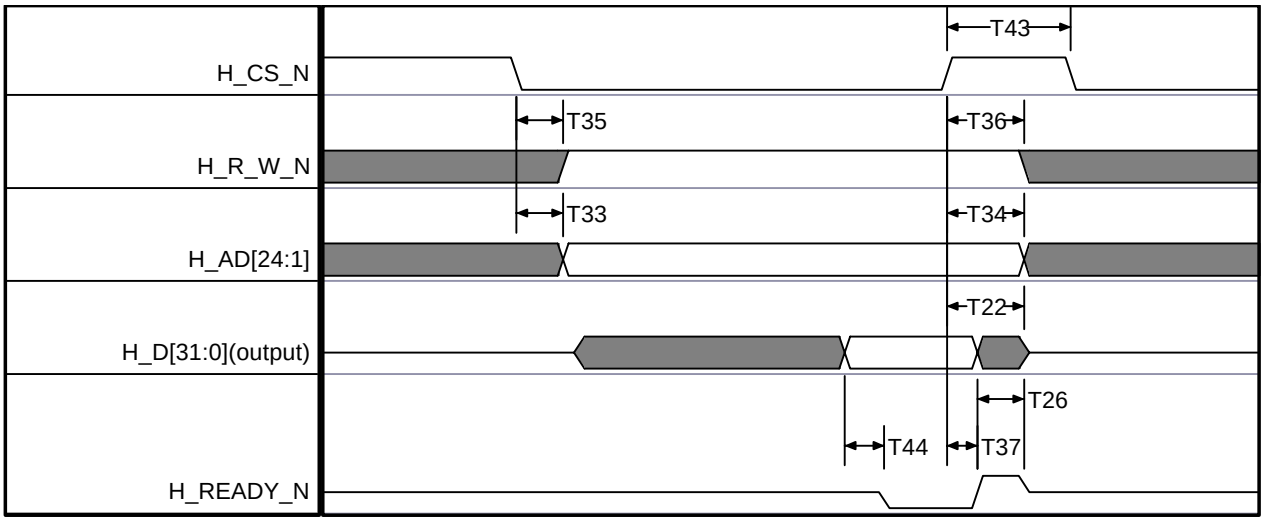


Figure 2-10. CPU Interface – Asynchronous Mode – Read Cycle

Table 2-25. SDRAM Interface AC Characteristics

Name	Description	Min	Max	Unit
T51	SD_CLK to SD_CS_N, SD_RAS_N, SD_CAS_N, SD_WE_N, SD_DQM[3:0], SD_A[11:0], SD_BA[1:0] output hold	2		ns
T52	SD_CLK to SD_CS_N, SD_RAS_N, SD_CAS_N, SD_WE_N, SD_DQM[3:0], SD_A[11:0], SD_BA[1:0] output valid		8	ns
T59	SD_CLK to SD_D[31:0] output hold	1.5		ns
T60	SD_CLK to SD_D[31:0] output valid		8.5	ns
T69	SD_D[31:0] input setup prior to SD_CLK	3		ns
T70	SD_D[31:0] input hold after SD_CLK	1.5		ns

Notes:

1. Timing reference voltage is 1.5V.
2. The output timing specified assumes 30 pf load.

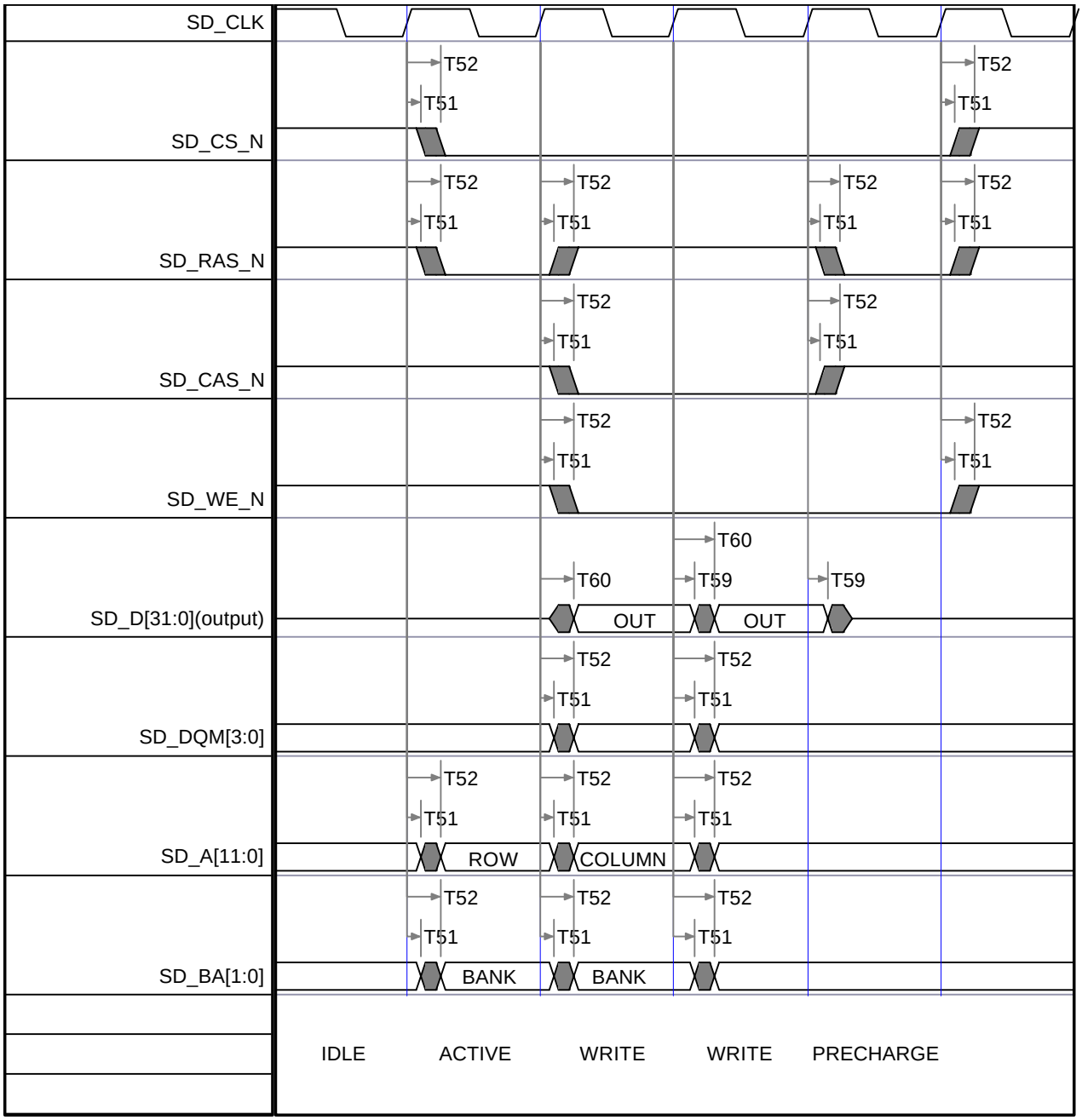


Figure 2-11. SDRAM Interface – Write Cycle

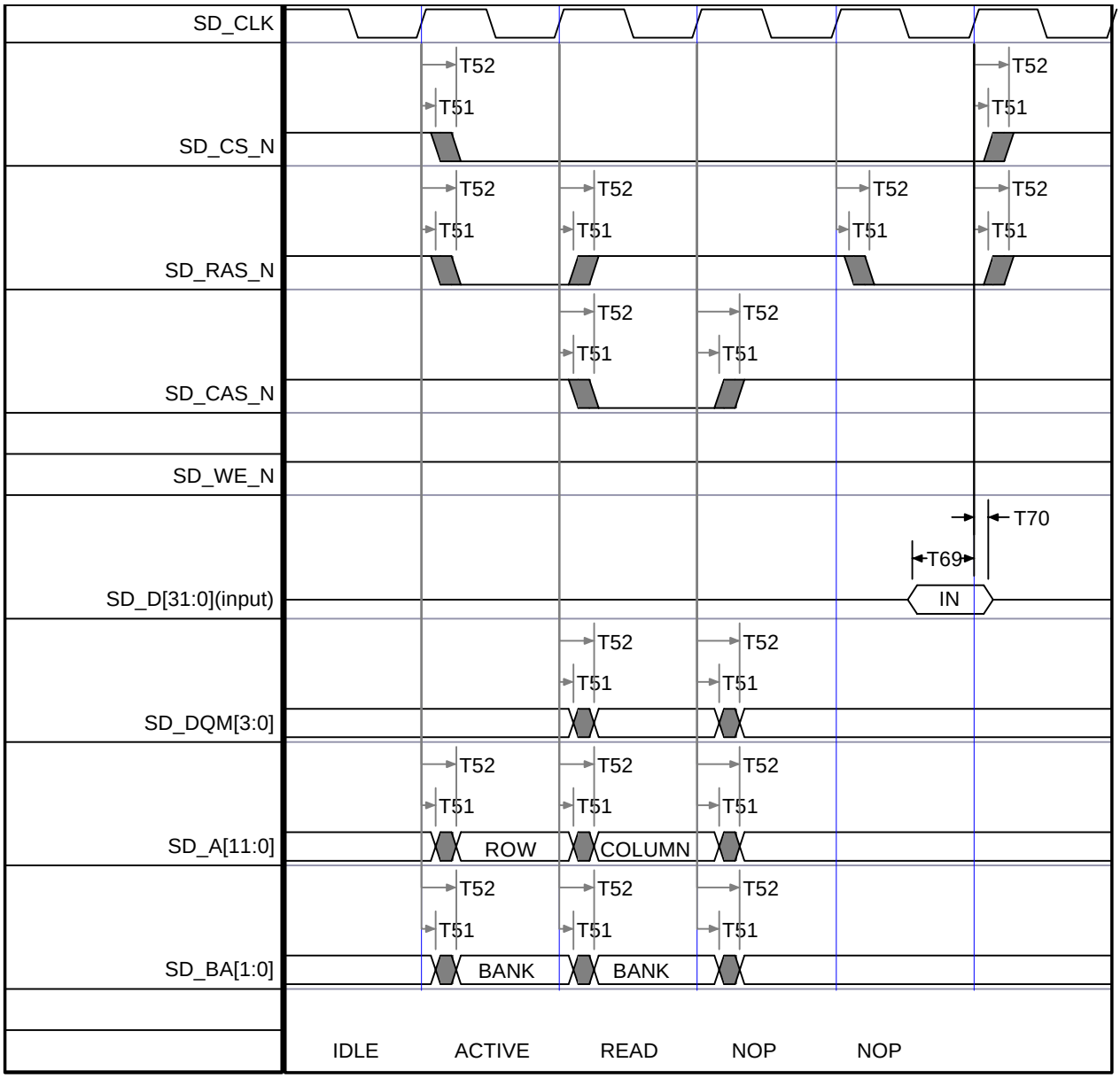


Figure 2-12. SDRAM Interface – Read Cycle

Table 2-26. TDM Interfaces AC Characteristics

Name	Description	Min	Typ	Max	Unit
T101	TDMn_RX, TDMn_SYNC, TDMn_RX_MF, TDMn_TX_MF, TDMn_RSIG input setup prior to TDMn_TCLK for E1/T1/serial interface	3.9			ns
T102	TDMn_RX, TDMn_SYNC, TDMn_RX_MF, TDMn_TX_MF, TDMn_RSIG input hold after TDMn_TCLK for E1/T1/serial interface	2.9			ns
T103	TDMn_TCLK to TDMn_TX, TDMn_TSIG output hold for E1/T1/serial interface	2			ns
T104	TDMn_TCLK to TDMn_TX, TDMn_TSIG output valid for E1/T1/serial interface			9.4	ns
T103	TDMn_TCLK to TDMn_TX, TDMn_TSIG output hold for high speed interface	4.4			ns
T104	TDMn_TCLK to TDMn_TX, TDMn_TSIG output valid for high speed interface			12.8	ns
T109	TDMn_RX input setup prior to TDMn_RCLK for serial interface	3.9			ns
T110	TDMn_RX input hold after TDMn_RCLK for serial interface	2.9			ns
T109	TDM1_RX input setup prior to TDMn_RCLK for high speed interface	3.9			ns
T110	TDM1_RX input hold after TDMn_RCLK for high speed interface	0.1			ns

Table 2-27. TDM Clocks Timing

Name	Description	Min	Typ	Max	Unit
T100	TDMn_TCLK frequency for E1 interface		2.048M		Hz
T100	TDMn_TCLK frequency for T1 interface		1.544M		Hz
T105	TDMn_TCLK duty cycle for E1/T1 interface	30		70	%
T106	TDMn_RCLK, TDMn_TCLK frequency for serial interface	16K		4.65M	Hz
T106	TDM1_RCLK, TDM1_TCLK frequency for high speed interface	16K		51.84M	Hz
T107	TDMn_RCLK, TDMn_TCLK duty cycle for serial interface	30		70	%
T107	TDM1_RCLK, TDM1_TCLK frequency for high speed interface	40		60	%

Notes:

1. The output timing specified for TDM interfaces assumes 30 pF load.
2. Timing reference voltage is 1.5V.

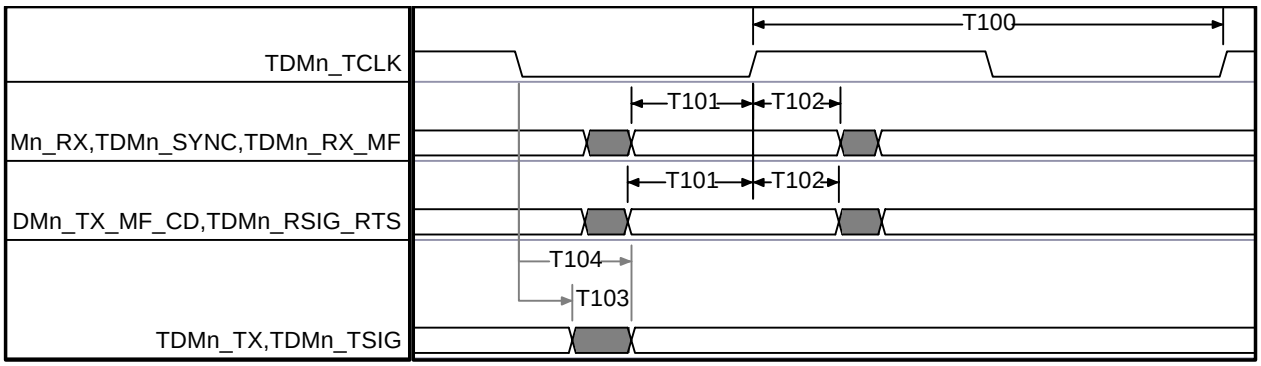


Figure 2-13. E1/T1 Interface (tx_sample=1)

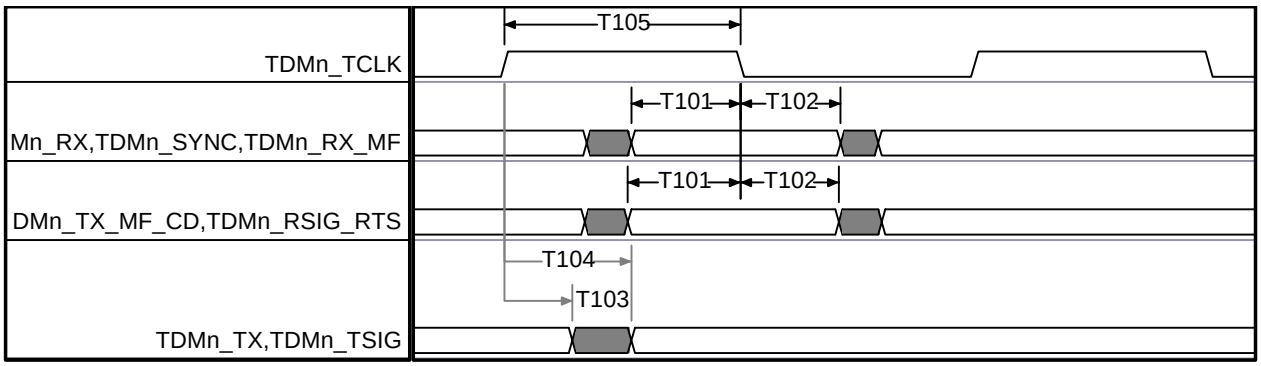


Figure 2-14. E1/T1 Interface (tx_sample=0)

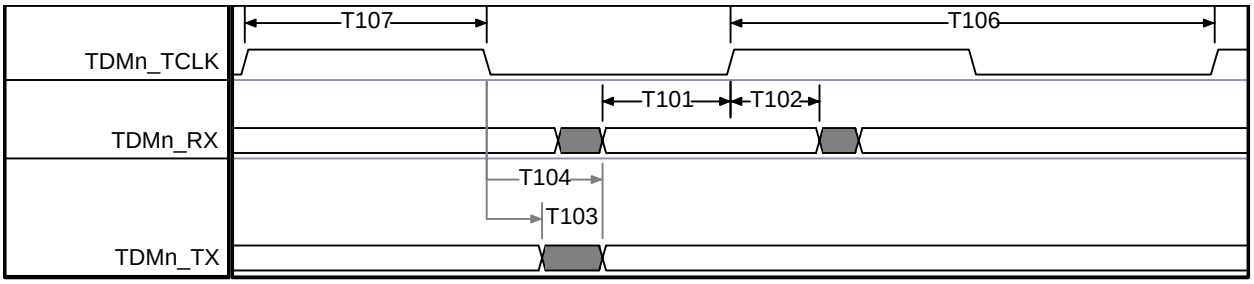


Figure 2-15. Serial Interface Working At Single Clock Operation (two_clocks=0, tx_sample=1)

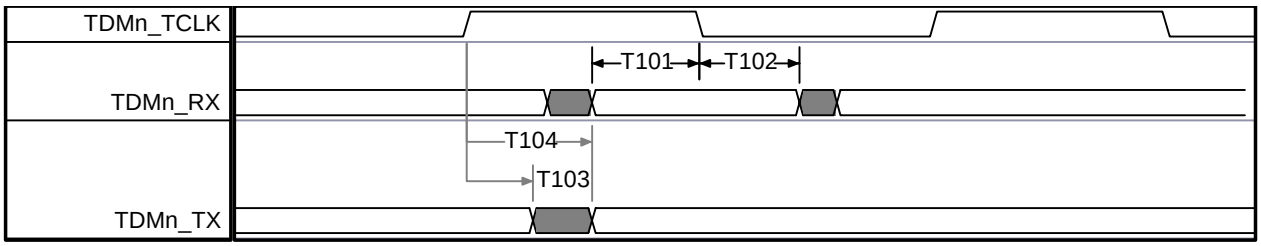


Figure 2-16. Serial Interface Working At Single Clock Operation (two_clocks=0, tx_sample=0)

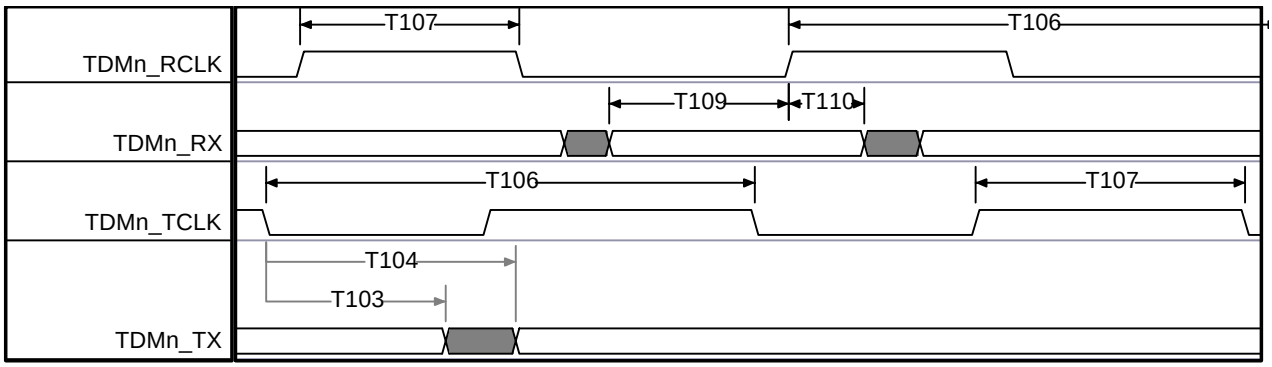


Figure 2-17. High speed or Serial Interface Working At Two Clocks Operation (two_clocks=1, tx_sample=1, rx_sample=1)

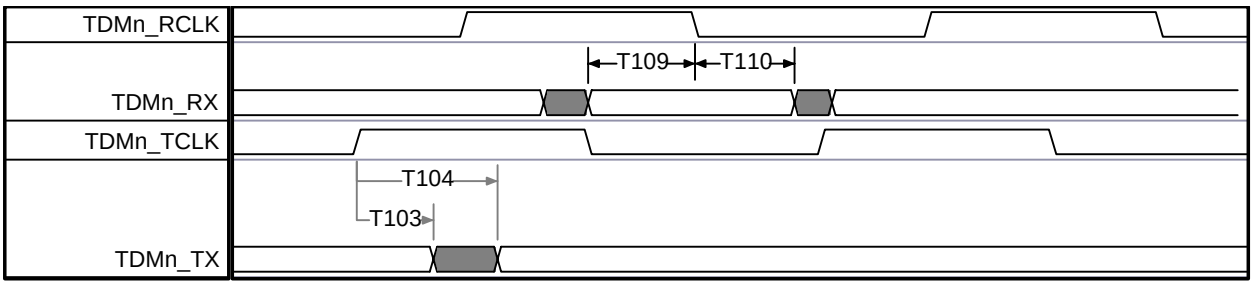


Figure 2-18. High Speed or Serial Interface Working At Two Clocks Operation (two_clocks=1, tx_sample=0, rx_sample=0)

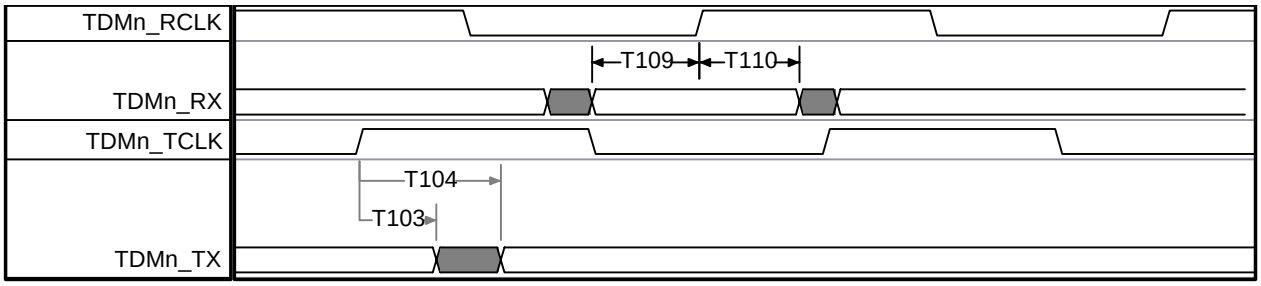


Figure 2-19. High Speed Or Serial Interface Working At Two Clocks Operation (two_clocks=1, tx_sample=0, rx_sample=1)

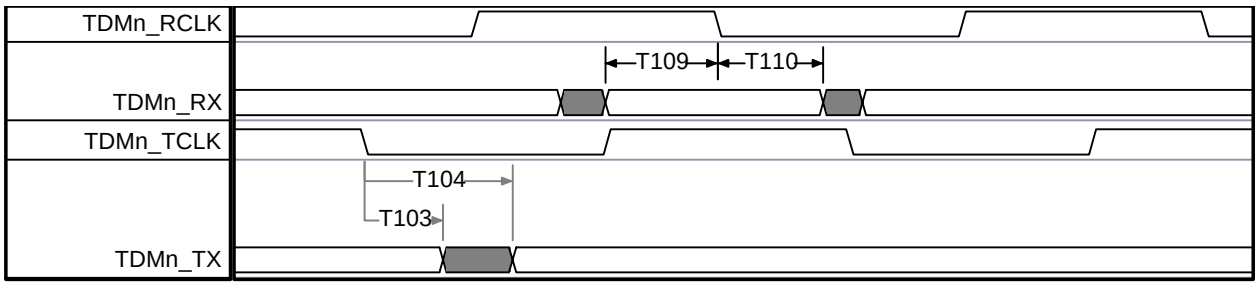


Figure 2-20. High Speed Or Serial Interface Working At Two Clocks Operation (two_clocks=1, tx_sample=1, rx_sample=0)

MII/RMII/SMII/SSMII

Table 2-28. MII Management Interface AC Characteristics

Name	Description	Min	Typ	Max	Unit
T150	MDC period Note		320		ns
T151	MDC to MDIO output hold Note	140			ns
T152	MDC to MDIO output valid Note			180	ns
T153	MDIO input setup prior to MDC rising	10			ns
T154	MDIO input hold after MDC rising	10			ns

Note: Valid for CLK_SYS = 50 MHz and MDC_frequency = ”01”.

Table 2-29. MII Interface AC Characteristics

Name	Description	Min	Typ	Max	Unit
T156	CLK_MII_TX rising to MII_TXD, MII_TX_ERR, MII_TX_EN output hold	10.3			ns
T157	CLK_MII_TX rising to MII_TXD, MII_TX_ERR, MII_TX_EN output valid			27.5	ns
T159	MII_RXD, MII_RX_DV, MII_RX_ERR input setup prior to CLK_MII_RX rising	5.8			ns
T160	MII_RXD, MII_RX_DV, MII_RX_ERR input hold after to CLK_MII_RX rising	0			ns

Table 2-30. MII Clocks Timing

Name	Description	Min	Typ	Max	Unit
T158	CLK_MII_TX frequency		25		MHz
T158	CLK_MII_RX frequency		25		MHz
T180	CLK_MII_TX duty cycle	40		60	%
T180	CLK_MII_RX duty cycle	40		60	%

Table 2-31. RMII Interface AC Characteristics

Name	Description	Min	Typ	Max	Unit
T162	CLK_MII_TX rising to MII_TXD(3:2), MII_TX_EN output hold	2			ns
T163	CLK_MII_TX rising to MII_TXD(3:2), MII_TX_EN output valid			7.5	ns
T164	MII_RXD(3:2), MII_RX_DV, MII_RX_ERR input setup prior to CLK_MII_TX rising	5.5			ns
T165	MII_RXD(3:2), MII_RX_DV, MII_RX_ERR input hold after CLK_MII_TX rising	2			ns

Table 2-32. RMII Clock Timing

Name	Description	Min	Typ	Max	Unit
T161	CLK_MII_TX frequency		50		MHz
T183	CLK_MII_TX duty cycle	40		60	%

Table 2-33. SMII Interface AC Characteristics

Name	Description	Min	Typ	Max	Unit
T167	CLK_MII_TX rising to MII_TXD(1:0) output hold	1.5			ns
T168	CLK_MII_TX rising to MII_TXD(1:0) output valid			5.5	ns
T169	MII_RXD(0) input setup prior to CLK_MII_TX rising	1.5			ns
T170	MII_RXD(0) input hold after CLK_MII_TX rising	1			ns

Table 2-34. SMII Clock Timing

Name	Description	Min	Typ	Max	Unit
T166	CLK_MII_TX frequency		125		MHz
T186	CLK_MII_TX duty cycle	40		60	%

Table 2-35. SSMII Interface AC Characteristics

Name	Description	Min	Typ	Max	Unit
T172	CLK_SSMII_TX rising to MII_TXD(1:0) output hold	2.5			ns
T173	CLK_SSMII_TX rising to MII_TXD(1:0) output valid			5	ns
T175	MII_RXD(1:0) input setup prior to CLK_MII_RX rising	1.5			ns
T176	MII_RXD(1:0) input hold after CLK_MII_RX rising	1			ns

Table 2-36. SSMII Clock Timing

Name	Description	Min	Typ	Max	Unit
T171	CLK_SSMII_TX frequency		125		MHz
T189	CLK_SSMII_TX duty cycle	40		60	%
T171	CLK_MII_RX frequency		125		MHz
T189	CLK_MII_RX duty cycle	40		60	%

Notes:

1. The output timing specified for MII/RMII/SMII/SSMII interfaces assumes 15 pF load.
2. Timing reference voltage is 1.5V.

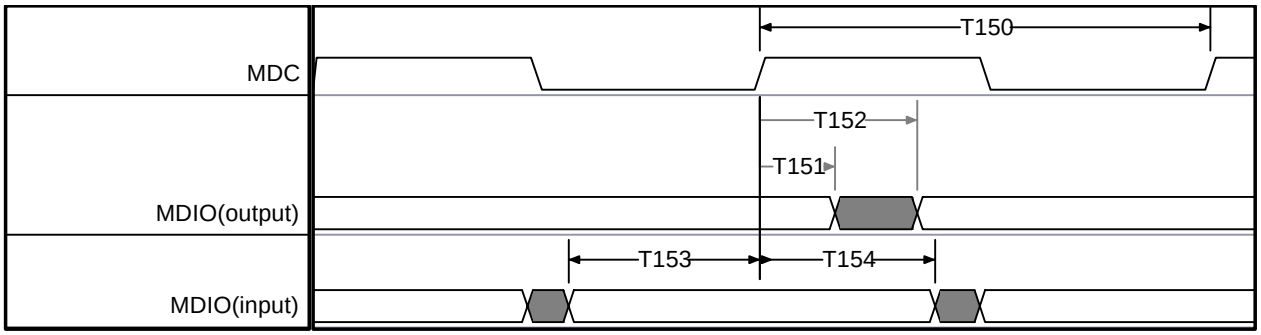


Figure 2-21. MII Management Interface

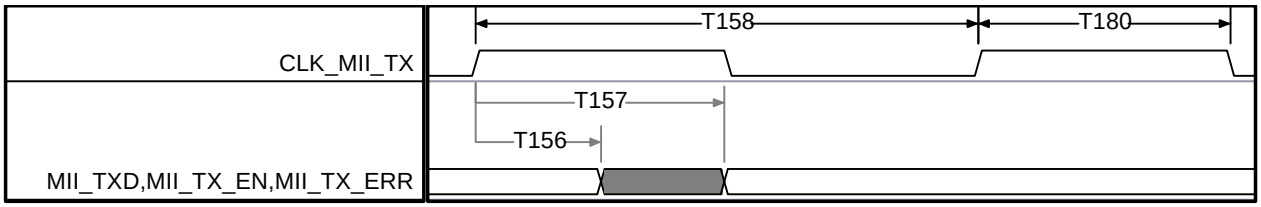


Figure 2-22. MII Interface Output Signals

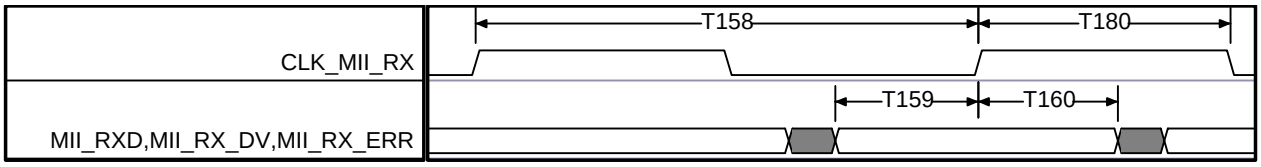


Figure 2-23. MII Interface Input Signals

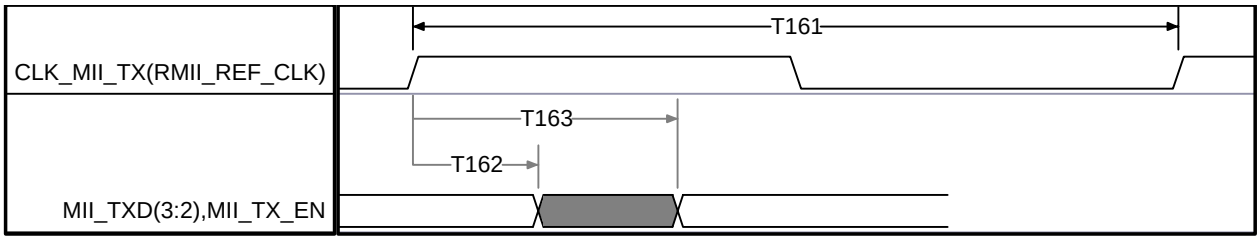


Figure 2-24. RMII Interface Output Signals

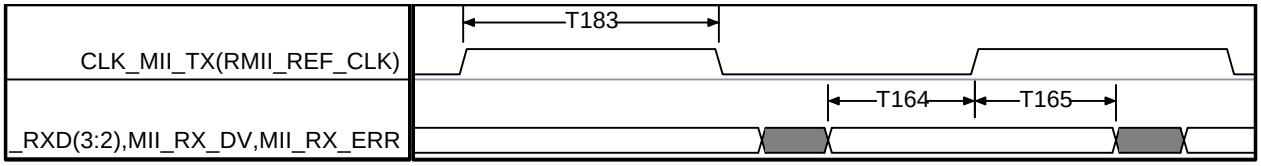


Figure 2-25. RMII Interface Input Signals

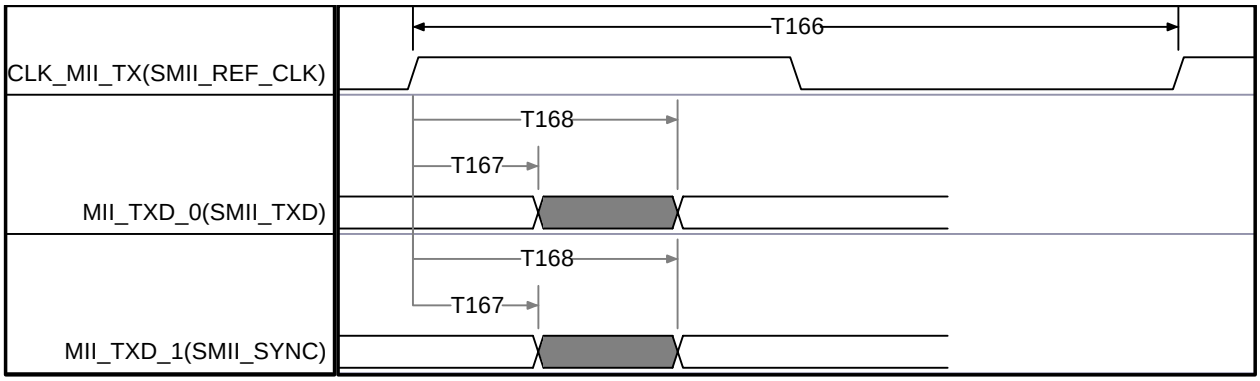


Figure 2-26. SMII Interface Output Signals

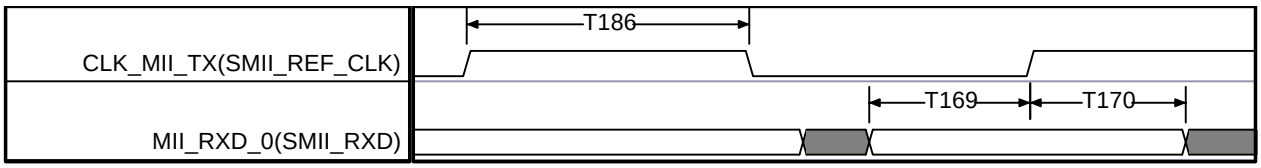


Figure 2-27. SMII Interface Input Signals

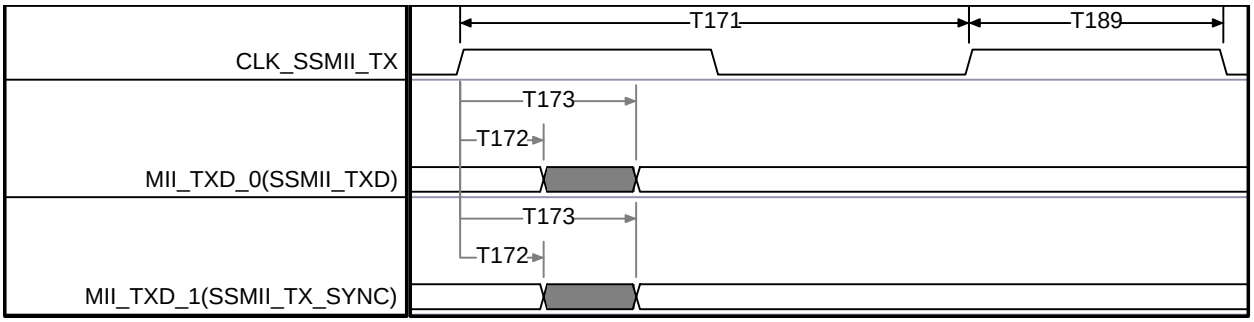


Figure 2-28. SSMII Interface Output Signals

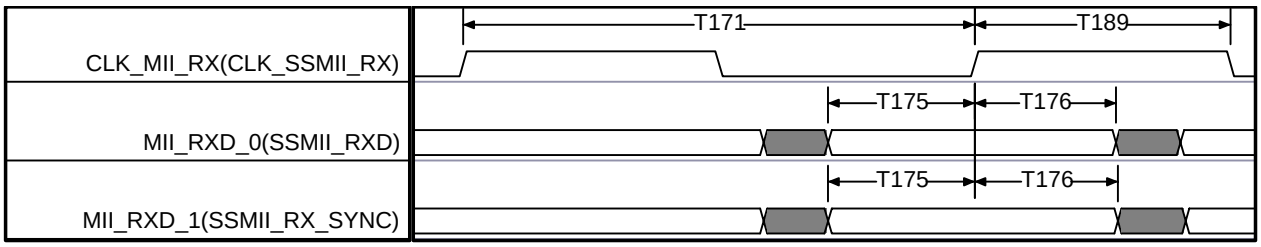


Figure 2-29. SSMII Interface Input Signals

JTAG Interface AC Characteristics

Table 2-37. JTAG Interface AC Characteristics

Name	Description	Min	Max	Unit
	JTAG_TCK frequency		20	MHz
T201	JTAG_TCK falling to JTAG_TDO output hold	4.5		ns
T202	JTAG_TCK falling to JTAG_TDO output valid		15	ns
T203	JTAG_TMS input setup prior to JTAG_TCK	8		ns
T204	JTAG_TMS input hold after JTAG_TCK	9.5		ns
T205	JTAG_TDI input setup prior to JTAG_TCK	0		ns
T206	JTAG_TDI input hold after JTAG_TCK	16		ns

Notes:

1. Timing reference voltage is 1.5V.
2. The output timing specified assumes 30pf load.

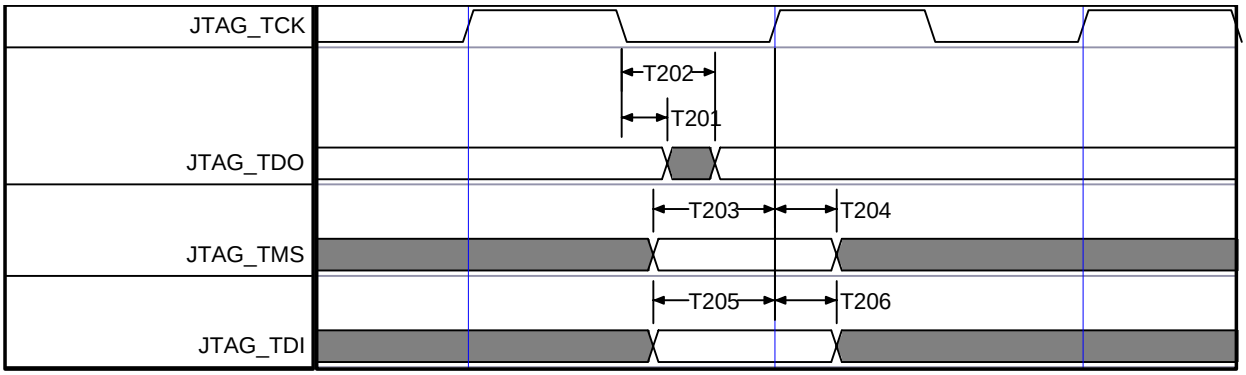


Figure 2-30. JTAG Interface

3.0 MEMORY MAP

3.1 GENERAL MAP AND ADDRESSING

Address Range

The address range of PacketTrunk-4 is mapped to the internal memory and to the external SDRAM, as follows:

Table 3-1. PacketTrunk-4 Memory Map

Address Range	Memory Type	Comments
0,000,000 – 0,07F,FFF		Subdivided into areas, accessible by means of a base address and offset
1,000,000 – 1,FFF,FFF	External SDRAM	Can be accessed through the chip, 1, 2 or 4 bytes at a time

Memory addresses comprise 24 bits, ADD[24:1]. ADD[0] is not connected.

Note: The address range of the PacketTrunk-4 can be decreased in the following manner:
ADD24 pin of the PacketTrunk-4 is used as a selector between the Internal Chip Memory and the External SDRAM. If the required SDRAM size is less than 16 MB, then the most significant address pin of the CPU that is connected to the chip can be connected to the chip's ADD24 pin, and any unused ADD pins of the chip should be connected to GND or VDD. For example, if the SDRAM memory size is 8 MB, then A23 pin of the CPU can be connected to ADD24 pin of the PacketTrunk-4, and ADD23 pin of the PacketTrunk-4 should be connected to GND or VDD.

Internal Chip

The internal memory subdivisions are listed in the table below.

Table 3-2. Internal Memory Map

Base Address	Contents	Page
0,000,000	Registers	63
0,008,000	Bundle Configuration Tables	64
0,010,000	Counters	66
0,012,000	Status Tables	70
0,018,000	Timeslot Assignment Tables	71
0,020,000	CPU Queues	73
0,028,000	Transmit Buffers Pool	75
0,030,000	Jitter Buffer Control	76
0,038,000	Transmit Software CAS	77
0,040,000	Receive Line CAS	78
0,048,000	Clock Recovery	80
0,050,000	Receive SW Defined Data	90
0,058,000	Receive SW Defined CAS	91
0,060,000	Timeslot to CID Table	92
0,068,000	Interrupt Controller	94
0,070,000	ETH MAC	95

Addressing

The internal memory can be accessed either 2 or 4 bytes at a time, controlled by an external configuration pin. In the 16-bit addressing mode, addresses are multiples of 2, while in 32-bit addressing, they are multiples of 4.

Note: In this document addresses are shown for both modes in the format:
0x32-bit address (0x16-bit address). Example: 0x134 (0x136).

The byte order for both addressing modes is “big-endian” (the most significant byte has the lowest address – see byte order numbers in grey in Figure 3-1. 16-bit Addressing and Figure 3-2. 32-bit Addressing).

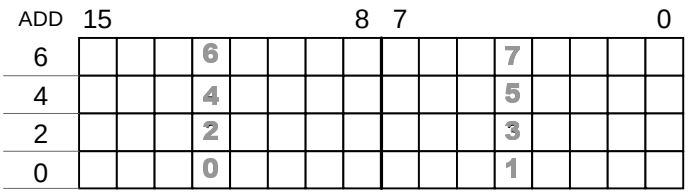


Figure 3-1. 16-bit Addressing

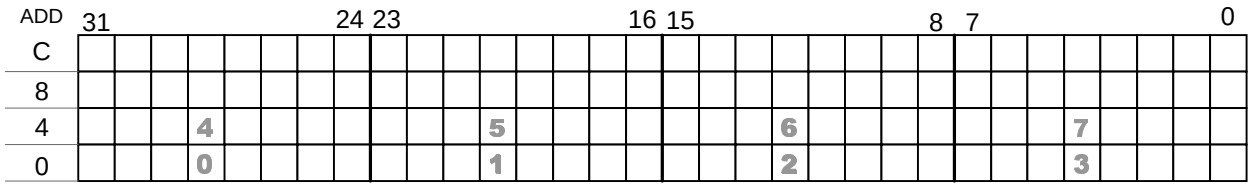


Figure 3-2. 32-bit Addressing

Partial data elements (shorter than 16 or 32 bits) are always positioned from LSB to MSB with the rest of the bits left unused. Thus, the bit numbers of data elements shorter than 16 bits are identical for both addressing modes (see bits [12:0] in Figure 3-3) and the CPU can access all bits by a single read/write.

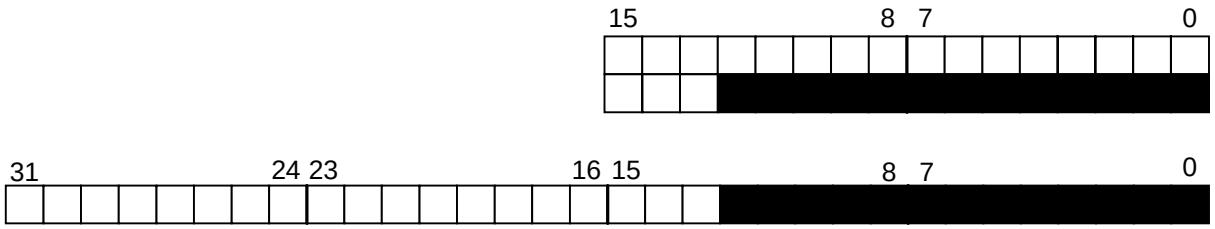


Figure 3-3. Partial Data Element (shorter than 16 bits)

Data elements 16 to 32 bits long need one read/write access in 32-bit addressing and two in 16-bit addressing. In the figure below, the 20-bit data element needs one 32-bit CPU access (bits [19:0]) and two 16-bit accesses (bits [15:0] and then [3:0]).

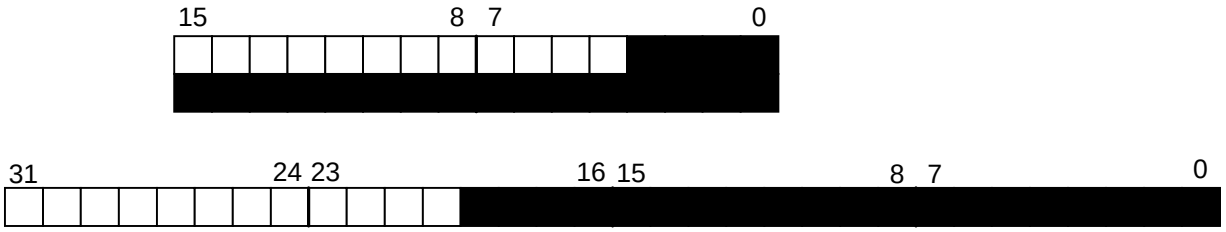


Figure 3-4. Partial Data Elements (16 to 32 bits long)

Registers 0,000,000

Table 3-3. Registers

Name	Offset	Description	Page
General_cfg_reg0	0x00	Chip configuration register0	96
General_cfg_reg1	0x04	Chip configuration register1	97
Port1_cfg_reg	0x08	Port1 configuration register	98
Port2_cfg_reg	0x0C	Port2 configuration register	101
Port3_cfg_reg	0x10	Port3 configuration register	103
Port4_cfg_reg	0x14	Port4 configuration register	107
Rst_reg	0x18	Reset register	110
TDM_cond _data _reg	0x1C	TDM AAL1/AAL2 conditioning data register	111
ETH_cond_data_reg	0x20	Ethernet AAL1 conditioning data register	111
AAL2_HDLC_recovery_reg	0x24	AAL2/HDLC/Clock Recovery configuration register	112
Packet_classifier_cfg_reg0	0x28	Packet classifier configuration register0	113
Packet_classifier_cfg_reg1	0x2C	Packet classifier configuration register1	114
Packet_classifier_cfg_reg2	0x30	Packet classifier configuration register2	114
Packet_classifier_cfg_reg3	0x34	Packet classifier configuration register3	115
Packet_classifier_cfg_reg4	0x38	Packet classifier configuration register4	116
Packet_classifier_cfg_reg5	0x3C	Packet classifier configuration register5	116

Table 3-4. Status Registers

Name	Offset	Description	Page
General_stat_reg	80	Per chip sticky bits register	117
Ports_stat_reg	84	Per port status register	119

Table 3-5. Version Register

Name	Offset	Description	Page
Version Register	C0	PacketTrunk-4 version	120

Bundle Configuration Tables – 0,008,000

Table 3-6. Bundle Configuration Tables [31:0]

Table Name	Offset	Description	Page
AAL1_Bundle0_cfg[31:0]	0x000	AAL1 bundle0 Configuration [31:0]	121
AAL2_Bundle0_cfg[31:0]		AAL2 bundle0 Configuration [31:0]	129
HDLC_Bundle0_cfg[31:0]		HDLC bundle0 Configuration [31:0]	132
AAL1_Bundle1_cfg[31:0]	0x004	AAL1 bundle1 Configuration [31:0]	121
AAL2_Bundle1_cfg[31:0]		AAL2 bundle1 Configuration [31:0]	129
HDLC_Bundle1_cfg[31:0]		HDLC bundle1 Configuration [31:0]	132
…	…		
AAL1_Bundle63_cfg[31:0]	0x0FC	AAL1 bundle63 Configuration [31:0]	121
AAL2_Bundle63_cfg[31:0]		AAL2 bundle63 Configuration [31:0]	129
HDLC_Bundle63_cfg[31:0]		HDLC bundle63 Configuration [31:0]	132

Table 3-7. Bundle Configuration Table [63:32]

Table Name	Offset	Description	Page
AAL1_Bundle0_cfg[63:32]	0x100	AAL1 bundle0 Configuration [63:32]	122
AAL2_Bundle0_cfg[63:32]		AAL2 bundle0 Configuration [63:32]	130
HDLC_Bundle0_cfg[63:32]		HDLC bundle0 Configuration [63:32]	133
AAL1_Bundle1_cfg[63:32]	0x104	AAL1 bundle1 Configuration [63:32]	122
AAL2_Bundle1_cfg[63:32]		AAL2 bundle1 Configuration [63:32]	130
HDLC_Bundle1_cfg[63:32]		HDLC bundle1 Configuration [63:32]	133
…	…		
AAL1_Bundle63_cfg[63:32]	0x1FC	AAL1 bundle63 Configuration [63:32]	122
AAL2_Bundle63_cfg[63:32]		AAL2 bundle63 Configuration [63:32]	130
HDLC_Bundle63_cfg[63:32]		HDLC bundle63 Configuration [63:32]	133

Table 3-8. Bundle Configuration Table [95:64]

Table Name	Offset	Description	Page
AAL1_Bundle0_cfg[95:64]	0x200	AAL1 bundle0 Configuration [95:64]	125
AAL2_Bundle0_cfg[95:64]		AAL2 bundle0 Configuration [95:64]	131
HDLC_Bundle0_cfg[95:64]		HDLC bundle0 Configuration [95:64]	133
AAL1_Bundle1_cfg[95:64]	0x204	AAL1 bundle1 Configuration [95:64]	125
AAL2_Bundle1_cfg[95:64]		AAL2 bundle1 Configuration [95:64]	131
HDLC_Bundle1_cfg[95:64]		HDLC bundle1 Configuration [95:64]	133
…	…		
AAL1_Bundle63_cfg[95:64]	0x2FC	AAL1 bundle63 Configuration [95:64]	125
AAL2_Bundle63_cfg[95:64]		AAL2 bundle63 Configuration [95:64]	131
HDLC_Bundle63_cfg[95:64]		HDLC bundle63 Configuration [95:64]	133

Table 3-9. Bundle Configuration Table [127:96]

Table Name	Offset	Description	Page
AAL1_Bundle0_cfg[127:96]	0x300	AAL1 bundle0 Configuration [127:96]	126
AAL2_Bundle0_cfg[127:96]		AAL2 bundle0 Configuration [127:96]	132
HDLC_Bundle0_cfg[127:96]		HDLC bundle0 Configuration [127:96]	134
AAL1_Bundle1_cfg[127:96]	0x304	AAL1 bundle1 Configuration [127:96]	126
AAL2_Bundle1_cfg[127:96]		AAL2 bundle1 Configuration [127:96]	132
HDLC_Bundle1_cfg[127:96]		HDLC bundle1 Configuration [127:96]	134
…	…		
AAL1_Bundle63_cfg[127:96]	0x3FC	AAL1 bundle63 Configuration [127:96]	126
AAL2_Bundle63_cfg[127:96]		AAL2 bundle63 Configuration [127:96]	132
HDLC_Bundle63_cfg[127:96]		HDLC bundle63 Configuration [127:96]	134

Counters – 0,010,000

Each counter can be read from two different addresses. The first address is 0,010,000+ offset and the second address is 0,011,000+ offset. Reading from the first address has no impact on the counter value. When reading from the second address the counter is cleared.

Table 3-10. Counters Types

Name	Base Address	R/W	Reset Values
Counters – no clear on read	0,010,000	R/O	No
Counters – clear on read	0,011,000	R/Clear	No

Note: When accessing counters, wider than 16 bits, at 16-bit mode, perform the following:
Read from address 2, i.e. A1='1' (This will give the current counter data of D[15:0] and also lock internally D[31:16] which are read from the counter).
Read from address 0, i.e. A1='0' (This will give the counter data D[31:16] that was locked in previous access to address 2).

Table 3-11. ETH Rx Good Packets Counters

Name	Offset	Size	Description	Page
Bundle0_ctr	0x000	32	Good packets received from Ethernet. Wrap around on maximum value.	135
Bundle1_ctr	0x004	32	"	135
…	…			
Bundle63_ctr	0x0FC	32	"	135

Table 3-12. ETH Tx Good Packets Counters

Name	Offset	Size	Description	
Bundle0_ctr	0x200	32	Good packets transmitted to Ethernet. Wrap around on maximum value.	135
Bundle1_ctr	0x204	32	"	135
…	…			
Bundle63_ctr	0x2FC	32	"	135

Table 3-13. ETH Rx Lost Packets Counters

Name	Offset	Size	Description	Page
Bundle0_ctr	0x400	16	Ethernet Lost Packets. Stuck on maximum value. Receives counts from Rx AAL1 or Rx HDLC.	135
Bundle1_ctr	0x404	16	"	135
…	…			
Bundle63_ctr	0x4FC	16	"	135

Table 3-14. Rx/AAL1 Lost Cells/AAL2-CPS Packets with Error Counters

Name	Offset	Size	Description	Page
Bundle0_ctr	0x600	16	Rx AAL1 Lost Cells /AAL2-CPS Packets received with Incorrect HEC. Stuck on maximum value.	136
Bundle1_ctr	0x604	16	"	136
….	….			
Bundle63_ctr	0x6FC	16	"	136

Table 3-15. TDM Tx HDLC Frames with Error Counters

Name	Offset	Size	Description	Page
Bundle0_ctr	0x800	16	HDLC frames received from TDM with any error. Stuck on maximum value.	136
Bundle1_ctr	0x804	16	"	136
…	…			
Bundle63_ctr	0x8FC	16	"	136

Table 3-16. TDM Tx HDLC Good Frames Counters

Name	Offset	Size	Description	Page
Bundle0_ctr	0xA00	32	HDLC good frames received from TDM. Wrap around on maximum value.	136
Bundle1_ctr	0xA04	32	"	136
…	…			
Bundle63_ctr	0xAFC	32	"	136

Table 3-17. Jitter Buffer Underrun/Overrun Events Counters

Name	Offset	Size	Description	Page
Timeslot0_ctr	0xC00	8	For AAL1 bundles/AAL2 timeslots – count of underrun events. AAL1 counter does not include underruns caused by pointer mismatches. For HDLC bundles – count of overrun events. Stuck on maximum value.	136
Timeslot1_ctr	0xC04	8	"	136
…	…			
Timeslot127_ctr	0xDFC	8	"	136

Table 3-18. Ethernet Counters

Name	Offset	Size	Description	Page
ETH_bytes_received	0xE00	32	Total bytes received from Ethernet (good packets only). CRC bytes are not counted. Wrap around on maximum value	137
ETH bytes_transmitted	0xE04	32	Total bytes transmitted to Ethernet (good packets only). CRC bytes are not counted. Wrap around on maximum value	137
Classified_packets	0xE08	32	Counts all packets that pass the packet classifier. Wrap around on maximum value	137

Ethernet MAC Counters – 0,070,000

Table 3-19. ETH MAC Counters

Name	Offset	Description	Page
Pause_Packets_Received_OK	0x3C		138
Packets_Transmitted_OK	0x40		138
Single_Collision_Packets	0x44		138
Multiple_Collision_Packets	0x48		139
Packets_Received_OK	0x4C		139
Packet_Check_Sequence_ Errors	0x50		139
Alignment_Errors	0x54		140
Deferred_Transmission_ Packets	0x58		140
Late_Collisions	0x5C		140
Excessive_Collisions	0x60		141
Transmit_Underrun_Errors	0x64		141
Carrier_Sense_Errors	0x68		141
Receive_Symbol_Errors	0x74		142
Excessive_Length_Errors	0x78		142
Receive_Jabbers	0x7C		142
Undersize_Packets	0x80		143
SQE_Test_Errors	0x84		143
Transmitted_Pause_Packets	0x8C		143

Note: When accessing elements, wider than 16 bits of the ETH MAC, at 16 bit mode, use the following procedure:

Read:

Read from address 2, i.e. A1='1' (This will give the current MAC data of D[15:0] and also lock internally D[31:16] which are read from the MAC).

Read from address 0, i.e. A1='0' (This will give the MAC data D[31:16] that was locked in previous access to address 2).

Write:

Write to address 2, i.e. A1='1' (This will lock internally the current 16 bit as D[15:0] but will not be written to the MAC yet).

Write to address 0, i.e. A1='0' (This will write to the MAC the current 16 bit to the MAC D[31:16] plus the previous write access 16 bit to the MAC D[15:0]).

Status Tables – 0,012,000

Table 3-20. Rx Payload Type Machine Status Bits

Name	Offset	Description	Page
Bundle0_status	0x000	Status bits of receive Payload Type machine	144
Bundle1_status	0x004	“	144
…	…		
Bundle63_status	0x0FC	“	144

Table 3-21. Tx Payload Type Machine Status Bits

Name	Offset	Description	Page
Bundle0_status	0x200	Status bits of transmit Payload Type machine	145
Bundle1_status	0x204	“	145
…	…		
Bundle63_status	0x2FC	“	145

Table 3-22. Tx Buffers Status Bits

Name	Offset	Description	Page
Bundle0_status	0x400	Transmit buffers status bits	145
Bundle1_status	0x404	“	145
…	…		
Bundle63_status	0x4FC	“	145

Table 3-23. Packet Classifier Status Bits

Name	Offset	Description	Page
Bundle0_status	0x600	Packet classifier status bits	145
Bundle1_status	0x604	“	145
…	…		
Bundle63_status	0x6FC	“	145

Table 3-24. Rx AAL2 Status Bits

Name	Offset	Description	Page
Timeslot0_status	0x800	Receive AAL2 status bits	146
Timeslot1_status	0x804	“	146
….	….		
Timeslot127_status	0x9FC	“	146

Table 3-25. Jitter Buffer Control Status Bits

Name	Offset	Description	Page
Timeslot0_status	0xA00	Jitter Buffer Control status bits	146
Timeslot1_status	0xA04	"	146
…	…		
Timeslot127_status	0xBFC	"	146

Timeslot Assignment (TSA) Tables – 0,018,000

Table 3-26. Port1 TSA Bank1

Name	Offset	Description	Page
Timeslot0_entry	0x000	Entry of first Timeslot	147
Timeslot1_entry	0x004	Entry of second Timeslot	147
...	...		
Timeslot31_entry	0x07C	"	147

Table 3-27. Port2 TSA Bank1

Name	Offset	Description	Page
Timeslot0_entry	0x080	Entry of first Timeslot	147
Timeslot1_entry	0x084	Entry of second Timeslot	147
...	...		
Timeslot31_entry	0x0FC	"	147

Table 3-28. Port3 TSA Bank1

Name	Offset	Description	Page
Timeslot0_entry	0x100	Entry of first timeslot	147
Timeslot1_entry	0x104	Entry of second timeslot	147
...	...		
Timeslot31_entry	0x17C	"	147

Table 3-29. Port4 TSA Bank1

Name	Offset	Description	Page
Timeslot0_entry	0x180	Entry of first timeslot	147
Timeslot1_entry	0x184	Entry of second timeslot	147
...	...		
Timeslot31_entry	0x1FC	"	147

Table 3-30. Port1 TSA Bank2

Name	Offset	Description	Page
Timeslot0_entry	0x200	Entry of first timeslot	148
Timeslot1_entry	0x204	Entry of second timeslot	148
...	...		
Timeslot31_entry	0x27C	"	148

Table 3-31. Port2 TSA Bank2

Name	Offset	Description	Page
Timeslot0_entry	0x280	Entry of first timeslot	148
Timeslot1_entry	0x284	Entry of second timeslot	148
...	...		
Timeslot31_entry	0x2FC	"	148

Table 3-32. Port3 TSA Bank2

Name	Offset	Description	Page
Timeslot0_entry	0x300	Entry of first timeslot	148
Timeslot1_entry	0x304	Entry of second timeslot	148
...	...		
Timeslot31_entry	0x37C	"	148

Table 3-33. Port4 TSA Bank2

Name	Offset	Description	Page
Timeslot0_entry	0x380	Entry of first timeslot	148
Timeslot1_entry	0x384	Entry of second timeslot	148
...	...		
Timeslot31_entry	0x3FC	"	148

CPU Queues 0,020,000

Table 3-34. CPU Queues

Name	Offset	Description	Page
TDM_CPU_pool_insert	0x00 (0x02)	Writing to this address causes a single buffer to be inserted to the TDM-CPU Pool. Only data bits [12:0] are written; these will be the 13 MSBs of the buffer's address.	149
TDM_CPU_pool_level	0x04 (0x06)	Holds the number of buffers stored in the TDM-CPU Pool.	149
TDM_CPU_pool_thresh	0x08 (0x0A)	Threshold of the TDM-CPU Pool. If the TDM_CPU_pool_level goes below this level an interrupt will be generated.	149
TDM_to_CPU_q_read	0x0C (0x0E)	Holds data bits [12:0]; these will be the 13 MSBs of the first buffer in the TDM_to_CPU queue.	150
TDM_to_CPU_q_level	0x10 (0x12)	Number of buffers in the TDM to CPU Queue	150
TDM_to_CPU_q_thresh	0x14 (0x16)	Threshold of the TDM to CPU Queue. If the TDM_to_CPU_q_level exceeds this level an interrupt will be generated.	150
CPU_to_ETH_q_insert	0x18 (0x1A)	Writing to this address causes a single buffer to be inserted to the CPU_to_eth queue. Only data bits [12:0] are written; these will be the 13 MSBs of the buffer's address.	151
CPU_to_ETH_q_level	0x1C (0x1E)	Number of buffers in the CPU to ETH Queue	151
CPU_to_ETH_q_thresh	0x20 (0x22)	Threshold of the CPU_to_eth Queue. If the CPU_to_eth_q_level goes below this level an interrupt will be generated.	151
ETH_CPU_pool_insert	0x24 (0x26)	Writing to this address causes a single buffer to be inserted to the ETH-CPU Pool. Only data bits [12:0] are written; these will be the 13 MSBs of the buffer's address.	152

Table 3-34. CPU Queues (Cont.)

Name	Offset	Description	Page
ETH_CPU_pool_level	0x28 (0x2A)	Number of buffers in the ETH_CPU Pool.	152
ETH_CPU_pool_thresh	0x2C (0x2E)	Threshold of the ETH_to_CPU Pool. If the eth_CPU_pool_level goes below this level an interrupt will be generated.	152
ETH_to_CPU_q_read	0x30 (0x32)	Holds data bits [12:0]; these will be the 13 MSBs of the first buffer in the ETH_to_CPU queue.	153
ETH_to_CPU_q_level	0x34 (0x36)	Number of buffers in the ETH_to_CPU Queue.	153
ETH_to_CPU_q_thresh	0x38 (0x3A)	Threshold of the ETH_to_CPU Queue. If the eth_to_CPU_q_level goes below this level an interrupt will be generated.	153
AAL2_CPU_pool_insert	0x3C (0x3E)	Writing to this address causes a single buffer to be inserted to the AAL2 CPU Pool. Only data bits [12:0] are written; these will be the 13 MSBs of the buffer's address.	154
AAL2_CPU_pool_level	0x40 (0x42)	Number of buffers in the AAL2_CPU Pool	154
AAL2_CPU_pool_thresh	0x44 (0x46)	Threshold of the AAL2-CPU Pool. If the AAL2_CPU_pool_level goes below this level an interrupt will be generated.	154
AAL2_to_CPU_q_read	0x48 (0x4A)	Holds data bits [12:0]; these will be the 13 MSBs of the first buffer in the AAL2_to_CPU queue	155
AAL2_to_CPU_q_level	0x4C (0x4E)	Number of buffers in the AAL2_to_CPU Queue	155

Table 3-34. CPU Queues (Cont.)

Name	Offset	Description	Page
AAL2_to_CPU_q_thresh	0x50 (0x52)	Threshold of the AAL2_to_CPU Queue. If the AAL2_to_CPU_q_level goes below this level an interrupt will be generated.	155
CPU_to_TDM_q_insert	0x54 (0x56)	Writing to this address causes a single buffer to be inserted to the CPU_to_TDM queue. Only data bits [12:0] are written; these will be the 13 MSBs of the buffer's address.	156
CPU_to_TDM_q_level	0x58 (0x5A)	Number of buffers in the TDM_to_CPU Queue	156
CPU_to_TDM_q_thresh	0x5C (0x5E)	Threshold of the CPU_to_TDM Queue. If the CPU_to_TDM_q_level goes below this level an interrupt will be generated	156
Tx_return_q_read	0x60 (0x62)	Holds data bits [12:0]; these will be the 13 MSBs of the first buffer in the CPU_tx_return queue.	157
Tx_return_q_level	0x64 (0x66)	Number of buffers in the CPU_tx_return Queue	157
Tx_return_q_thresh	0x68 (0x6A)	Threshold of the CPU_tx_return Queue. If the Tx_return_level goes below this level an interrupt will be generated.	157
Rx_return_q_read	0x6C (0x6E)	Holds data bits [12:0]; these will be the 13 MSBs of the first buffer in the CPU_rx_return queue.	157
Rx_return_q_level	0x70 (0x72)	Number of buffers in the CPU_rx_return Queue.	158
Rx_return_q_thresh	0x74 (0x76)	Threshold of the CPU_rx_return Queue. If the Rx_return_level goes below this level an interrupt will be generated	158

Transmit Buffers Pool – 0,028,000

Table 3-35. Transmit Buffers Pool – Head Table

Name	Offset	Description	Page
Bundle0_head	0x800 (0x802)	Initialized by CPU to hold the lists and heads for all open bundles	158
Bundle1_head	0x804 (0x806)	"	158
…	…		
Bundle63_head	0x8FC (0x8FE)	"	158

Table 3-36. Transmit Buffers Pool – Next Buffer Table

Name	Offset	Description	Page
Next_buffer0	0x000 (0x002)	A pointer to the next buffer in the list. Initialized by CPU to hold the lists for all open bundles.	159
Next_buffer1	0x004 (0x006)	"	159
…	…		
Next_buffer511	0x7FC (0x7FE)	"	159

Jitter Buffer Control - 0,030,000

Table 3-37. Per Jitter Buffer Index Status Table

Name	Offset	Description	Page
Status_and_level0	0x000 (0x000, 0x002)	JBC status for all bundle types and the level for the bundle types that are specified in the detailed description.	159
Min_and_max_level0	0x004 (0x004, 0x006)	Minimal and maximal levels for AAL1 and AAL2 bundle types. For other bundle types this register is unused and must not be read. See Note 2 below.	162
.	.		
.	.		
Status_and_level127	0x3F8 (0x3F8, 0x3FA)	JBC status for all bundle types and the level for the bundle types that are specified in the detailed description.	159
Min_and_max_level127	0x3FC (0x3FC, 0x3FE)	Minimal and maximal levels for Structured AAL1 and AAL2 bundle types. For other bundle types this register is unused and must not be read. See Note 2 below.	162

Notes:

1. At high speed mode, Hs_status_and_level and Hs_min_and_max_level reside in Status_and-level0 and Min_and_max_level0 registers, respectively.
2. The CPU should never try to read Min/Max Level from an HDLC bundle. When the CPU performs an access to these registers, it causes some bits to be changed- bits that are used for other purposes in HDLC bundles and thus may cause sever problems.

Table 3-38. Bundle Timeslot Table

Name	Offset	Description	Page
Bundle_ts0	0x700 (0x700, 0x702)	Assigned timeslots of the bundle: 1 – Bit is assigned 0 – Bit is not assigned	164
Bundle_ts1	0x704 (0x704, 0x706)	Assigned timeslots of the bundle: 1 – Bit is assigned 0 – Bit is not assigned	164
…			
Bundle_ts63	0x7FC (0x7FC, 0x7FE)	Assigned timeslots of the bundle: 1 – Bit is assigned 0 – Bit is not assigned	164

Transmit Software CAS – 0,038,000

Table 3-39. Port1 Timeslots Transmit Software CAS

Name	Offset	Description	Page
Tx_SW_CAS_TS1_TS0	0x00 (0x02)	CAS signaling for TS1 and TS0 of Port1. Used instead of CAS bits coming from RSIG1 per bundle configuration.	164
Tx_SW_CAS_TS3_TS2	0x04 (0x06)	CAS signaling for TS3 and TS2 of Port1. Used instead of CAS bits coming from RSIG1 per bundle configuration.	164
…	…		
Tx_SW_CAS_TS31_TS30	0x3C (0x3E)	CAS signaling for TS31 and TS30 of Port1. Used instead of CAS bits coming from RSIG1 per bundle configuration.	164

Table 3-40. Port2 Timeslots Transmit Software CAS

Name	Offset	Description	Page
Tx_SW_CAS_TS1_TS0	0x40 (0x42)	CAS signaling for TS1 and TS0 of Port2. Used instead of CAS bits coming from RSIG2 per bundle configuration.	164
Tx_SW_CAS_TS3_TS2	0x44 (0x46)	CAS signaling for TS3 and TS2 of Port2. Used instead of CAS bits coming from RSIG2 per bundle configuration.	164
…	…		
Tx_SW_CAS_TS31_TS30	0x7C (0x7E)	CAS signaling for TS31 and TS30 of Port2. Used instead of CAS bits coming from RSIG2 per bundle configuration.	164

Table 3-41. Port3 Timeslots Transmit Software CAS

Name	Offset	Description	Page
Tx_SW_CAS_TS1_TS0	0x80 (0x82)	CAS signaling for TS1 and TS0 of Port3. Used instead of CAS bits coming from RSIG3 per bundle configuration.	164
Tx_SW_CAS_TS3_TS2	0x84 (0x86)	CAS signaling for TS3 and TS2 of Port3. Used instead of CAS bits coming from RSIG3 per bundle configuration.	164
…	…		
Tx_SW_CAS_TS31_TS30	0xBC (0xBE)	CAS signaling for TS31 and TS30 of Port3. Used instead of CAS bits coming from RSIG3 per bundle configuration.	164

Table 3-42. Port4 Timeslots Transmit Software CAS

Name	Offset	Description	Page
Tx_SW_CAS_TS1_TS0	0xC0 (0xC2)	CAS signaling for TS1 and TS0 of Port4. Used instead of CAS bits coming from RSIG4 per bundle configuration.	164
Tx_SW_CAS_TS3_TS2	0xC4 (0xC6)	CAS signaling for TS3 and TS2 of Port4. Used instead of CAS bits coming from RSIG4 per bundle configuration.	164
…	…		
Tx_SW_CAS_TS31_TS30	0xFC (0xFE)	CAS signaling for TS31 and TS30 of Port4. Used instead of CAS bits coming from RSIG4 per bundle configuration.	164

Receive Line CAS – 0,040,000

Table 3-43. Port1 Receive Line CAS

Name	Offset	Description	Page
Timeslot0_rx_CAS	0x000 (0x002)	Timeslot0 CAS bits to be transmitted on TSIG	164
Timeslot1_rx_CAS	0x004 (0x006)	Timeslot1 CAS bits to be transmitted on TSIG	164
…	…		
Timeslot31_rx_CAS	0x07C (0x1FE)	Timeslot31 CAS bits to be transmitted on TSIG	164

Table 3-44. Port2 Receive Line CAS

Name	Offset	Description	Page
Timeslot0_rx_CAS	0x080 (0x082)	Timeslot0 CAS bits to be transmitted on TSIG	164
Timeslot1_rx_CAS	0x084 (0x086)	Timeslot1 CAS bits to be transmitted on TSIG	164
…	…		
Timeslot31_rx_CAS	0x0FC (0x0FE)	Timeslot31 CAS bits to be transmitted on TSIG	164

Table 3-45. Port3 Receive Line CAS

Name	Offset	Description	Page
Timeslot0_rx_CAS	0x100 (0x102)	Timeslot0 CAS bits to be transmitted on TSIG	164
Timeslot1_rx_CAS	0x104 (0x106)	Timeslot1 CAS bits to be transmitted on TSIG	164
…	…		
Timeslot31_rx_CAS	0x17C (0x17E)	Timeslot31 CAS bits to be transmitted on TSIG	164

Table 3-46. Port4 Receive Line CAS

Name	Offset	Description	Page
Timeslot0_rx_CAS	0x180 (0x182)	Timeslot0 CAS bits to be transmitted on TSIG	164
Timeslot1_rx_CAS	0x184 (0x186)	Timeslot1 CAS bits to be transmitted on TSIG	164
…	…		
Timeslot31_rx_CAS	0x17C (0x1FE)	Timeslot31 CAS bits to be transmitted on TSIG	164

Clock Recovery – 0,048,000

Table 3-47. Clock Recovery Control Words

Name	Offset	Description	Page
Control_word_port1	0x0000 (0x0002)	Port1 clock recovery control bits	165
Control_word_port2	0x0400 (0x0402)	Port2 clock recovery control bits	165
Control_word_port3	0x0800 (0x0802)	Port3 clock recovery control bits	165
Control_word_port4	0x0C00 (0x0C02)	Port4 clock recovery control bits	165

Table 3-48. Port1 Clock Recovery Acquisition and Tracking Characteristics

Name	Offset	Description	Page
Ack_lshft0	0x0004 (0x0006)	5-bit register (unsigned)	–
Ack_lshft1	0x0008 (0x000A)	5-bit register (unsigned)	–
Ack_lshft2	0x000C (0x000E)	5-bit register (unsigned)	–
Trk_lshft0	0x0010 (0x0012)	5-bit register (unsigned)	–
Trk_lshft1	0x0014 (0x0016)	5-bit register (unsigned)	–
Trk_lshft2	0x0018 (0x001A)	5-bit register (unsigned)	–
Ack_den_coef0	0x001C (0x001E)	26-bit register (signed)	–
Ack_den_coef1	0x0020 (0x0022)	26-bit register (signed)	–
Trk_den_coef0	0x0024 (0x0026)	26-bit register (signed)	–
Trk_den_coef1	0x0028 (0x002A)	26-bit register (signed)	–
I_ack_gain	0x002C (0x002E)	3-bit register (unsigned)	–
I_trk_gain	0x0030 (0x0032)	3-bit register (unsigned)	–
P_ack_gain	0x0034 (0x0036)	3-bit register (unsigned)	–
P_trk_gain	0x0038 (0x003A)	3-bit register (unsigned)	–
I_ack_rshft	0x003C (0x003E)	5-bit register (unsigned)	–
I_trk_rshft	0x0040 (0x0042)	5-bit register (unsigned)	–
JBC_lvl_gain	0x0044 (0x0046)	24-bit register (signed)	–
Loop_gain_rshft	0x0048 (0x004A)	5-bit register (unsigned)	–
Ticks_gen_m1	0x004C (0x004E)	10-bit register (unsigned)	–

Table 3-48. Port1 Clock Recovery Acquisition and Tracking Characteristics (Cont.)

Name	Offset	Description	Page
Ticks_gen_n1	0x0050 (0x0052)	19-bit register (unsigned)	–
Ticks_gen_n2	0x0054 (0x0056)	19-bit register (unsigned)	–
DPLL_c	0x0058 (0x005A)	10-bit register (unsigned)	–
P_ack_rshft	0x005C (0x005E)	5 bit reg (Unsigned)	–
P_trk_rshft	0x0060 (0x0062)	5 bit reg (Unsigned)	–
Acq_tick_decimation_factor Trk2_tick_decimation_factor	0x0064 (0x0066)	[6:0] - (Unsigned) [13:7] - (Unsigned)	–
Acq_sett_ok_time	0x0068	18 bit reg (Unsigned)	–
Acq_wo_int_time	0x006C	14 bit reg (Unsigned)	–
Pll_acq_jaf_tresh	0x0070	30 bit reg (Unsigned)	–
Trk2_additional_lshft Trk2_jb_lvl_tresh	0x0074	[4:0] - (Unsigned) [25:16] - (Unsigned)	–
Acq_stbl_alrm_time	0x0078	19 bit reg (Unsigned)	–
Trk_stbl_alrm_time	0x007C	19 bit reg (Unsigned)	–

Table 3-49. Port2 Clock Recovery Acquisition and Tracking Characteristics

Name	Offset	Description	Page
Ack_lshft0	0x0404 (0x0406)	5-bit register (unsigned)	–
Ack_lshft1	0x0408 (0x040A)	5-bit register (unsigned)	–
Ack_lshft2	0x040C (0x000E)	5-bit register (unsigned)	–
Trk_lshft0	0x0410 (0x0412)	5-bit register (unsigned)	–
Trk_lshft1	0x0414 (0x0416)	5-bit register (unsigned)	–
Trk_lshft2	0x0418 (0x041A)	5-bit register (unsigned)	–
Ack_den_coef0	0x041C (0x041E)	26-bit register (signed)	–
Ack_den_coef1	0x0420 (0x0422)	26-bit register (signed)	–
Trk_den_coef0	0x0424 (0x0426)	26-bit register (signed)	–
Trk_den_coef1	0x0428 (0x042A)	26-bit register (signed)	–
I_ack_gain	0x042C (0x042E)	3-bit register (unsigned)	–
I_trk_gain	0x0430 (0x0432)	3-bit register (unsigned)	–
P_ack_gain	0x0434 (0x0436)	3-bit register (unsigned)	–
P_trk_gain	0x0438 (0x043A)	3-bit register (unsigned)	–
I_ack_rshft	0x043C (0x043E)	5-bit register (unsigned)	–
I_trk_rshft	0x0440 (0x0442)	5-bit register (unsigned)	–
JBC_lvl_gain	0x0444 (0x0446)	24-bit register (signed)	–
Loop_gain_rshft	0x0448 (0x044A)	5-bit register (unsigned)	–
Ticks_gen_m1	0x044C (0x044E)	10-bit register (unsigned)	–

Table 3-49. Port2 Clock Recovery Acquisition and Tracking Characteristics (Cont.)

Name	Offset	Description	Page
Ticks_gen_n1	0x0450 (0x0452)	19-bit register (unsigned)	–
Ticks_gen_n2	0x0454 (0x0456)	19-bit register (unsigned)	–
DPLL_c	0x0458 (0x045A)	10-bit register (unsigned)	–
P_ack_rshft	0x045C (0x045E)	5 bit reg (Unsigned)	–
P_trk_rshft	0x0460 (0x0462)	5 bit reg (Unsigned)	–
Acq_tick_decimation_factor Trk2_tick_decimation_factor	0x0064 (0x0066)	[6:0] - (Unsigned) [13:7] - (Unsigned)	–
Acq_sett_ok_time	0x0468	18 bit reg (Unsigned)	–
Acq_wo_int_time	0x046C	14 bit reg (Unsigned)	–
Pll_acq_jaf_tresh	0x0470	30 bit reg (Unsigned)	–
Trk2_additional_lshft Trk2_jb_lvl_tresh	0x0074	[4:0] - (Unsigned) [25:16] - (Unsigned)	–
Acq_stbl_alrm_time	0x0478	19 bit reg (Unsigned)	–
Trk_stbl_alrm_time	0x047C	19 bit reg (Unsigned)	–

Table 3-50. Port3 Clock Recovery Acquisition and Tracking Characteristics

Name	Offset	Description	Page
Ack_lshft0	0x0804 (0x0806)	5-bit register (unsigned)	–
Ack_lshft1	0x0808 (0x080A)	5-bit register (unsigned)	–
Ack_lshft2	0x080C (0x080E)	5-bit register (unsigned)	–
Trk_lshft0	0x0810 (0x0812)	5-bit register (unsigned)	–
Trk_lshft1	0x0814 (0x0816)	5-bit register (unsigned)	–
Trk_lshft2	0x0818 (0x081A)	5-bit register (unsigned)	–
Ack_den_coef0	0x081C (0x081E)	26-bit register (signed)	–
Ack_den_coef1	0x0820 (0x0822)	26-bit register (signed)	–
Trk_den_coef0	0x0824 (0x0826)	26-bit register (signed)	–
Trk_den_coef1	0x0828 (0x082A)	26-bit register (signed)	–
I_ack_gain	0x082C (0x082E)	3-bit register (unsigned)	–
I_trk_gain	0x0830 (0x0832)	3-bit register (unsigned)	–
P_ack_gain	0x0834 (0x0836)	3-bit register (unsigned)	–
P_trk_gain	0x0838 (0x083A)	3-bit register (unsigned)	–
I_ack_rshft	0x083C (0x083E)	5-bit register (unsigned)	–
I_trk_rshft	0x0840 (0x0842)	5-bit register (unsigned)	–
JBC_lvl_gain	0x0844 (0x0846)	24-bit register (signed)	–
Loop_gain_rshft	0x0848 (0x084A)	5-bit register (unsigned)	–
Ticks_gen_m1	0x084C (0x084E)	10-bit register (unsigned)	–

Table 3-50. Port3 Clock Recovery Acquisition and Tracking Characteristics (Cont.)

Name	Offset	Description	Page
Ticks_gen_n1	0x0850 (0x0852)	19-bit register (unsigned)	–
Ticks_gen_n2	0x0854 (0x0856)	19-bit register (unsigned)	–
DPLL_c	0x0858 (0x085A)	10-bit register (unsigned)	–
P_ack_rshft	0x085C (0x085E)	5 bit reg (Unsigned)	–
P_trk_rshft	0x0860 (0x0862)	5 bit reg (Unsigned)	–
Acq_tick_decimation_factor Trk2_tick_decimation_factor	0x0064 (0x0066)	[6:0] - (Unsigned) [13:7] - (Unsigned)	–
Acq_sett_ok_time	0x0868	18 bit reg (Unsigned)	–
Acq_wo_int_time	0x086C	14 bit reg (Unsigned)	–
Pll_acq_jaf_tresh	0x0870	30 bit reg (Unsigned)	–
Trk2_additional_lshft Trk2_jb_lvl_tresh	0x0074	[4:0] - (Unsigned) [25:16] - (Unsigned)	–
Acq_stbl_alrm_time	0x0878	19 bit reg (Unsigned)	–
Trk_stbl_alrm_time	0x087C	19 bit reg (Unsigned)	–

Table 3-51. Port4 Clock Recovery Acquisition and Tracking Characteristics

Name	Offset	Description	Page
Ack_lshft0	0x0C04 (0x0C06)	5-bit register (unsigned)	–
Ack_lshft1	0x0C08 (0x000A)	5-bit register (unsigned)	–
Ack_lshft2	0x0C0C (0x0C0E)	5-bit register (unsigned)	–
Trk_lshft0	0x0C10 (0x0C12)	5-bit register (unsigned)	–
Trk_lshft1	0x0C14 (0x0C16)	5-bit register (unsigned)	–
Trk_lshft2	0x0C18 (0x0C1A)	5-bit register (unsigned)	–
Ack_den_coef0	0x0C1C (0x0C1E)	26-bit register (signed)	–
Ack_den_coef1	0x0C20 (0x0C22)	26-bit register (signed)	–
Trk_den_coef0	0x0C24 (0x0C26)	26-bit register (signed)	–
Trk_den_coef1	0x0C28 (0x0C2A)	26-bit register (signed)	–
I_ack_gain	0x0C2C (0x0C2E)	3-bit register (unsigned)	–
I_trk_gain	0x0C30 (0x0C32)	3-bit register (unsigned)	–
P_ack_gain	0x0C34 (0x0C36)	3-bit register (unsigned)	–
P_trk_gain	0x0C38 (0x0C3A)	3-bit register (unsigned)	–
I_ack_rshft	0x0C3C (0x0C3E)	5-bit register (unsigned)	–
I_trk_rshft	0x0C40 (0x0C42)	5-bit register (unsigned)	–
JBC_lvl_gain	0x0C44 (0x0C46)	24-bit register (signed)	–
Loop_gain_rshft	0x0C48 (0x0C4A)	5-bit register (unsigned)	–
Ticks_gen_m1	0x0C4C (0x0C4E)	10-bit register (unsigned)	–

Table 3-51. Port4 Clock Recovery Acquisition and Tracking Characteristics (Cont.)

Name	Offset	Description	Page
Ticks_gen_n1	0x0C50 (0x0C52)	19-bit register (unsigned)	–
Ticks_gen_n2	0x0C54 (0x0C56)	19-bit register (unsigned)	–
DPLL_c	0x0C58 (0x0C5A)	10-bit register (unsigned)	–
P_ack_rshft	0x0C5C (0x0C5E)	5 bit reg (Unsigned)	–
P_trk_rshft	0x0C60 (0x0C62)	5 bit reg (Unsigned)	–
Acq_tick_decimation_factor Trk2_tick_decimation_factor	0x0064 (0x0066)	[6:0] - (Unsigned) [13:7] - (Unsigned)	–
Acq_sett_ok_time	0x0C68	18 bit reg (Unsigned)	–
Acq_wo_int_time	0x0C6C	14 bit reg (Unsigned)	–
Pll_acq_jaf_tresh	0x0C70	30 bit reg (Unsigned)	–
Trk2_additional_lshft Trk2_jb_lvl_tresh	0x0074	[4:0] - (Unsigned) [25:16] - (Unsigned)	–
Acq_stbl_alrm_time	0x0C78	19 bit reg (Unsigned)	–
Trk_stbl_alrm_time	0x007C	19 bit reg (Unsigned)	–

Table 3-52. Port1 DPLL Conversion Tables

Name	Offset	Description	Page
DPLL_config_Entry0	0x1000 (0x1002)	a – [31:16] b – [15:0]	–
DPLL_config_Entry1	0x1004 (0x1006)	a – [31:16] b – [15:0]	–
...	...		–
DPLL_config_Entry255	0x13FC (0x13FE)	a – [31:16] b – [15:0]	–

Table 3-53. Port2 DPLL Conversion Tables

Name	Offset	Description	Page
DPLL_config_Entry0	0x1400 (0x1402)	a – [31:16] b – [15:0]	–
DPLL_config_Entry1	0x1404 (0x1406)	a – [31:16] b – [15:0]	–
...	...		–
DPLL_config_Entry255	0x17FC (0x17FE)	a – [31:16] b – [15:0]	–

Table 3-54. Port3 DPLL Conversion Tables

Name	Offset	Description	Page
DPLL_config_Entry0	0x1800 (0x1802)	a – [31:16] b – [15:0]	–
DPLL_config_Entry1	0x1804 (0x1806)	a – [31:16] b – [15:0]	–
...	...		–
DPLL_config_Entry255	0x1BFC (0x1BFE)	a – [31:16] b – [15:0]	–

Table 3-55. Port4 DPLL Conversion Tables

Name	Offset	Description	Page
DPLL_config_Entry0	0x1C00 (0x1C02)	a – [31:16] b – [15:0]	–
DPLL_config_Entry1	0x1C04 (0x1C06)	a – [31:16] b – [15:0]	–
...	...		–
DPLL_config_Entry255	0x1FFC (0x1FFE)	a – [31:16] b – [15:0]	–

Receive SW Defined Data – 0,050,000

Table 3-56. Port1 Receive SW Defined Data

Name	Offset	Description	Page
Timeslot0_cond_data_sel	0x000 (0x002)	2 bits, which select the value of the conditioning octet to be transmitted on TDM1_Tx, when timeslot is not assigned. This value is also the conditioning octet that is inserted into the jitter buffer for lost packet compenation.	166
Timeslot1_cond_data_sel	0x004 (0x006)	"	166
…	…		
Timeslot31_cond_data_sel	0x07C (0x07E)	"	166

Table 3-57. Port2 Receive SW Defined Data

Name	Offset	Description	Page
Timeslot0_cond_data_sel	0x080 (0x082)	2 bits, which select the value of the conditioning octet to be transmitted on TDM2_Tx, when timeslot is not assigned. This value is also the conditioning octet that is inserted into the jitter buffer for lost packet compenation.	166
Timeslot1_cond_data_sel	0x084 (0x086)	"	166
…	…		
Timeslot31_cond_data_sel	0x0FC (0x0FE)	"	166

Table 3-58. Port3 Receive SW Defined Data

Name	Offset	Description	Page
Timeslot0_cond_data_sel	0x100 (0x102)	2 bits, which select the value of the conditioning octet to be transmitted on TDM3_Tx, when timeslot is not assigned. This value is also the conditioning octet that is inserted into the jitter buffer for lost packet compenation.	166
Timeslot1_cond_data_sel	0x104 (0x106)	"	166
…	…		
Timeslot31_cond_data_sel	0x17C (0x17E)	"	166

Table 3-59. Port4 Receive SW Defined Data

Name	Offset	Description	Page
Timeslot0_cond_data_sel	0x180 (0x182)	2 bits, which select the value of the conditioning octet to be transmitted on TDM4_Tx, when timeslot is not assigned. This value is also the conditioning octet that is inserted into the jitter buffer for lost packet compenation.	166
Timeslot1_cond_data_sel	0x184 (0x186)	"	166
…	…		
Timeslot31_cond_data_sel	0x1FC (0x1FE)	"	166

Receive SW Defined CAS – 0,058,000

Table 3-60. Port1 Receive SW Defined CAS

Name	Offset	Description	Page
Timeslot0_CAS	0x000 (0x002)	Timeslot0 CAS bits to be transmitted on TSIG when Timeslot0 is not assigned or CAS is not transferred for this timeslot. Must be different from '0000'	166
Timeslot1_CAS	0x004 (0x006)	"	166
...	...		
Timeslot31_CAS_	0x07C (0x07E)	"	166

Table 3-61. Port2 Receive SW Defined CAS

Name	Offset	Description	Page
Timeslot0_CAS	0x080 (0x082)	Timeslot0 CAS bits to be transmitted on TSIG when Timeslot0 is not assigned or CAS is not transferred for this timeslot. Must be different from '0000'	166
Timeslot1_CAS	0x084 (0x086)	"	166
...	...		
Timeslot31_CAS	0x0FC (0x0FE)	"	166

Table 3-62. Port3 Receive SW Defined CAS

Name	Offset	Description	Page
Timeslot0_CAS	0x100 (0x102)	Timeslot0 CAS bits to be transmitted on TSIG when Timeslot0 is not assigned or CAS is not transferred for this timeslot. Must be different from '0000'	166
Timeslot1_CAS	0x104 (0x106)	"	166
...	...		
Timeslot31_CAS	0x17C (0x17E)	"	166

Table 3-63. Port4 Receive SW Defined CAS

Name	Offset	Description	Page
Timeslot0_CAS	0x180 (0x002)	Timeslot0 CAS bits to be transmitted on TSIG when Timeslot0 is not assigned or CAS is not transferred for this timeslot. Must be different from '0000'	166
Timeslot1_CAS	0x184 (0x186)	"	166
...	...		
Timeslot31_CAS	0x1FC (0x1FE)	"	166

Timeslot to CID Table – 0,060,000

Table 3-64. Port1 AAL2 Timeslot to CID Table

Name	Offset	Description	Page
TS0_CID_value	0x00 (0x02)	CID value associated with TS0	166
TS1_CID_value	0x04 (0x06)	CID value associated with TS1	166
…	...		
TS31_CID_value	0x7C (0x7E)	CID value associated with TS31	166

Table 3-65. Port2 AAL2 Timeslot to CID Table

Name	Offset	Description	Page
TS0_CID_value	0x80 (0x82)	CID value associated with TS0	166
TS1_CID_value	0x84 (0x86)	CID value associated with TS1	166
…	...		
TS31_CID_value	0xFC (0xFE)	CID value associated with TS31	166

Table 3-66. Port3 AAL2 Timeslot to CID Table

Name	Offset	Description	Page
TS0_CID_value	0x100 (0x102)	CID value associated with TS0	166
TS1_CID_value	0x104 (0x106)	CID value associated with TS1	166
…	...		
TS31_CID_value	0x17C (0x17E)	CID value associated with TS31	166

Table 3-67. Port4 AAL2 Timeslot to CID Table

Name	Offset	Description	Page
TS0_CID_value	0x180 (0x182)	CID value associated with TS0	166
TS1_CID_value	0x184 (0x186)	CID value associated with TS1	166
…	...		
TS31_CID_value	0x1FC (0x1FE)	CID value associated with TS31	166

Interrupt Controller – 0,068,000

Table 3-68. Interrupt Controller Registers

Name	Offset	Description	Page
Intpend	0x000	Interrupt controller pending interrupts register. Writing '1' to one of its bits clears it ('0') if it was set. Writing '0' to one of its bits has no effect.	167
Intmask	0x004	Interrupt controller Mask register. Each bit corresponds to a bit in the INTPEND register. 0 – don't mask interrupt to the CPU 1 – mask the interrupt to the CPU	168
Rx_CAS_change_reg_P1	0x040	Each bit represents a change in Port1 receive line CAS table at one of the timeslots. Writing '1' to one of its bits clears it ('0') if it was set. Writing '0' to one of its bits has no effect.	168
Rx_CAS_change_reg_P2	0x044	Each bit represents a change in Port2 receive line CAS table at one of the timeslots. Writing '1' to one of its bits clears it ('0') if it was set. Writing '0' to one of its bits has no effect.	168
Rx_CAS_change_reg_P3	0x048	Each bit represents a change in Port3 receive line CAS table at one of the timeslots. Writing '1' to one of its bits clears it ('0') if it was set. Writing '0' to one of its bits has no effect.	168
Rx_CAS_change_reg_P4	0x04C	Each bit represents a change in Port4 receive line CAS table at one of the timeslots. Writing '1' to one of its bits clears it ('0') if it was set. Writing '0' to one of its bits has no effect.	168
JBC_underrun_reg_P1	0x080	JBC underrun pending register of Port1. Writing '1' to one of its bits clears it ('0') if it was set. Writing '0' to one of its bits has no effect.	169
JBC_underrun_reg_P2	0x088	JBC underrun pending register of Port2. Writing '1' to one of its bits clears it ('0') if it was set. Writing '0' to one of its bits has no effect.	169
JBC_underrun_reg_P3	0x090	JBC underrun pending register of Port3. Writing '1' to one of its bits clears it ('0') if it was set. Writing '0' to one of its bits has no effect.	169
JBC_underrun_reg_P4	0x098	JBC underrun pending register of Port4. Writing '1' to one of its bits clears it ('0') if it was set. Writing '0' to one of its bits has no effect.	169

Table 3-68. Interrupt Controller Registers (Cont.)

Name	Offset	Description	Page
JBC_underrun_mask_ reg_P1	0x084	JBC underrun Mask register. Each bit corresponds to a bit in the JBC_underrun_reg_P1 register. 0 - pass interrupt to the CPU 1 - don't pass interrupt to the CPU	169
JBC_underrun_mask_ reg_P2	0x08C	JBC underrun Mask register. Each bit corresponds to a bit in the JBC_underrun_reg_P2 register. 0 - pass interrupt to the CPU 1 - don't pass interrupt to the CPU	169
JBC_underrun_mask_ reg_P3	0x094	JBC underrun Mask register. Each bit corresponds to a bit in the JBC_underrun_reg_P3 register. 0 - pass interrupt to the CPU 1 - don't pass interrupt to the CPU	169
JBC_underrun_mask_ reg_P4	0x09C	JBC underrun Mask register. Each bit corresponds to a bit in the JBC_underrun_reg_P4 register. 0 - pass interrupt to the CPU 1 - don't pass interrupt to the CPU	169

ETH MAC – 0,070,000

Table 3-69. ETH MAC Registers

Name	Offset	Description	Page
MAC_network_control	0x00		170
MAC_network_configuration	0x04		171
MAC_network_status	0x08		173
MAC_transmit_status	0x14		173
MAC_interrupt_status	0x24		174
MAC_interrupt_enable	0x28		175
MAC_interrupt_disable	0x2C		175
MAC_interrupt_mask	0x30		176
MAC_PHY_maintenance	0x34		177
Pause_Time_Register	0x38		177
MAC_Specific_Bottom_Register	0x98		178
MAC_Specific_Top_Register	0x9C		178
MAC_transmit_pause_ quantum	0xBC		178
PHY_SMII_status	0xC0		178

3.2 REGISTERS

General_cfg_reg0 0x0,000,000

This register holds chip-related parameters.

Table 3-70. General_cfg_reg0 Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:27]	Reserved			
[26]	Clock_recovery_en	R/W	'0'	1 – Normal operation 0 – Clock recovery block is disabled (power saving mode) Should be cleared to reduce the chip power consumption when clock recovery is not used. When cleared, the clock recovery block (offset 48000h) must not be accessed by the CPU.
[25:16]	Rx_fifo_priority_lvl	R/W	'0100000000'	Rx_fifo threshold level in dwords. If Rx_fifo level is higher than this threshold, then the Rx_fifo receives the higher priority instead of the cross-connect queue and the CPU-TDM queue (which has the lowest priority). This parameter is relevant only when there are bundles configured as cross-connect. The recommended value is 0x3FF (maximal value).
[15:14]	MII_mode_select	R/W	'00'	00 – MII 01 – RMII 10 – Non source sync SMII (SMII) 11 – Source sync SMII (SSMII)
[13]	Reserved	R/W	'0'	Must be set to zero
[12]	Reserved	R/W	'1'	Must be set to one
[11]	High_speed	R/W	'0'	1 – Port1 is active only (E3/T3) 0 – Up to 4 ports of E1/T1 may be active
[10:8]	Bundles_ports_allocation	R/W	'111'	Maximum number of bundles and active ports
[7]	Mem_size	R/W	'0'	SDRAM size: 0 – 64 Mb 1 – 128 Mb

	Port1	Port2	Port3	Port4
010	32	disabled	32	disabled
110	32	disabled	16	16
011	16	16	32	disabled
111	16	16	16	16

Table 3-70. General_cfg_reg0 Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[6:5]	Fq	R/W	'00'	SDRAM clock: 00 – 50 MHz 01 – 75 MHz 10 – Reserved 11 – Reserved
[4:3]	Col_width	R/W	'00'	SDRAM columns and rows 00 – 8 bit (256 columns) 01 – 9 bit (512 columns) 10 – 10 bit (1K columns) 11 – 11 bit (2K columns)
[2:1]	CAS_latency	R/W	'10'	SDRAM CAS latency: 00 – Reserved 01 – CAS latency = 1 10 – CAS latency = 2 11 – CAS latency = 3
[0]	Rst_SDRAM_n	R/W	'0'	Resets SDRAM controller. Active low. Needs to be reset ('0') and then released ('1') only after all the configuration bits of the SDRAM controller are valid.

General_cfg_reg1 0x0,000,004

This register holds chip-related parameters.

Table 3-71. General_cfg_reg1 Map

Bits	Data Element Name	R/W	Reset Values	Description
[31]	Reserved			
[30:24]	Sw_packet_offset	R/W	'0000100'	4 – 127 bytes of offset between start of CPU buffer to first byte of first packet/AAL2-CPS Packet
[23:19]	Tx_payload_offset	R/W	'00000'	Number of 32-bit dwords between the start of transmit buffer to start of the payload (or the control word, if exists)
[18:10]	JBC_sig_base_add	R/W	'001100000'	Base address (9 MSBits) of receive jitter buffer signaling section in SDRAM
[9:6]	Tx_buf_base_add	R/W	'0010'	Base address (4 MSBits) of transmit buffers in SDRAM
[5:3]	Reserved			
[2:0]	JBC_data_base_add	R/W	'000'	Base address (3 MSBits) of receive jitter buffer data section in SDRAM

Port1_cfg_reg 0x0,000,008

This register holds port1-related configuration parameters.

Table 3-72. Port1_cfg_reg Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:21]	Clock recovery_PDVT_1 [10:0]	R/W	'0..0'	PDVT value for the bundle used for port1 clock recovery. Holds the configured PDVT of this bundle plus half packet creation time. The resolution is determined by the interface and bundle types as follows: • For AAL1 structured/structured with CAS and for AAL2 bundles: 0.5 ms (only [9:0] bits are relevant). • For mode- different then High Speed: 32 * interface bit period. • For high speed interface: 128 * interface bit period. Used in conjunction with Clock recovery_PDVT_1[14:11] for unstructured bundle.
[20]	Port_Rx_enable_1	R/W	'0'	1 – Outgoing TDM traffic towards Port1 is enabled 0 – Outgoing TDM traffic towards Port1 is discarded (TDM1_Tx and TDM1_TSIG are set) *Note: This bit also applies to high speed.*
[19]	CTS_1	R/W	'1'	Value of TDM1_TSIG_CTS_D2ACS_N, which acts as CTS (Clear To Send) output signal in case of serial interface.
[18]	CD_en_1	R/W	'0'	Output enable of CD_1. Active high. This bit should be activated only for serial interface.
[17]	CD_1	R/W	'1'	Value of TDM1_TX_MF_CD, which acts as CD (Carrier Detect) output signal in case of serial interface.
[16]	Loss_1	R/W	'0'	Loss of sync on TDM port1. Causes setting of the corresponding bit in the control word of all bundles of Port1.
[15:11]	Adapt_JBC_indx_1	R/W	'00000'	Index of the jitter buffer used by the clock recovery block to generate the clock of Port1

Table 3-72. Port1_cfg_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[10:9]	SF_to_ESF_low_CAS_bits_1	R/W	'00'	Bits that are added as C, D CAS bits to the A, B CAS bits for cases where SF interface is connected by a structured with CAS bundle to an ESF interface.
[8]	TSA_act_blk_1	R/W	'0'	0 – TSA bank1 is the active bank for Port1. 1 – TSA bank2 is the active bank for Port1. Swapping banks takes effect at the next sync input assertion.
[7]	Port_Tx_enable_1	R/W	'0'	1 – Incoming TDM traffic from Port1 is enabled 0 – Incoming TDM traffic from Port1 is discarded *Note: This bit also applies to High-Speed.*
[6]	Rx_sample_1	R/W	'1'	When interface is framed or unframed with two_clocks_1='0' this signal has no meaning. When interface is unframed with Two_clocks_1 = '1': • If rx_sample_1 ='0': TDM1_Rx is sampled at the falling edge of TDM1_RCLK. • If rx_sample_1 ='1': TDM1_Rx is sampled at the rising edge of TDM1_RCLK.

Table 3-72. Port1_cfg_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[5]	Tx_sample_1	R/W	'0'	When interface is unframed with two_clocks_1='0' or framed: • If tx_sample_1 ='0': TDM1_SYNC, TDM1_TX_MF, TDM1_RX_MF and TDM1_RX are sampled on the falling edge of TDM1_TCLK whereas TDM1_TX and TDM1_TSIG_CTS_D2A_CS_N are updated at the rising edge of TDM1_TCLK. • If tx_sample_1 ='1': TDM1_SYNC, TDM1_TX_MF, TDM1_RX_MF and TDM1_RX are sampled at the rising edge of TDM1_TCLK whereas TDM1_TX and TDM1_TSIG_CTS_D2A_CS_N are updated at the falling edge of TDM1_TCLK. When interface is unframed with Two_clocks_1 = '1': • If tx_sample_1 ='0': TDM1_Tx and TDM1_TSIG_CTS_D2A_CS_N are updated at the rising edge of TDM1_TCLK. • If tx_sample_1 ='1': TDM1_Tx and TDM1_TSIG_CTS_D2A_CS_N are updated at the falling edge of TDM1_TCLK.
[4]	Two_clocks_1	R/W	'1'	When interface is configured to high speed mode, this bit must be set. When interface is framed this bit has no meaning. When interface is unframed: • If two_clocks ='0' TDM1_TCLK is used for both receive and transmit paths. • If two_clocks ='1' TDM_RCLK is used for the receive path and TDM1_TCLK is used for the transmit path.
[3:2]	Int_framed_type1	R/W	'00'	00 – unframed(no sync, no mf sync) 01 – framed(sync only, no mf sync) 10 – MF(E1)/SF(T1) (sync and mf sync) 11 – ESF(T1) (sync and mf sync) Changing value from 10 or 11 to 00 or 01 must performed only after asserting RST_SYS_N.
[1:0]	Int_type_1	R/W	'01'	00 – Serial 01 – E1 10 – T1 11 – Reserved

Port2_cfg_reg 0x0,000,00C

This register holds Port2-related configuration parameters.

Table 3-73. Port2_cfg_reg Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:21]	Clock recovery_PDVT_2 [10:0]	R/W	'0..0'	PDVT value for the bundle used for port2 clock recovery. Holds the configured PDVT of this bundle plus half packet creation time. The resolution is determined by the interface and bundle types as follows: • For AAL1 structured/structured with CAS and for AAL2 bundles: 0.5 ms (only [9:0] bits are relevant). • For unstructured bundles: 32 * interface bit period. Used in conjunction with Clock recovery_PDVT_2 [14:11] for unstructured bundle.
[20]	Port_Rx_enable_2	R/W	'0'	1 – Outgoing TDM traffic towards Port2 is enabled 0 – Outgoing TDM traffic towards Port2 is discarded (TDM2_Tx and TDM2_TSIG are set)
[19]	CTS_2	R/W	'1'	Value of TDM2_TSIG_CTS_D2ACS_N, which acts as CTS (Clear To Send) output signal in case of serial interface.
[18]	CD_en_2	R/W	'0'	Output enable of CD_2. Active high. This bit should be activated only for serial interface.
[17]	CD_2	R/W	'1'	Value of TDM2_TX_MF_CD, which acts as CD (Carrier Detect) output signal in case of serial interface.
[16]	Loss_2	R/W	'0'	Loss of sync on TDM Port2. Causes setting of the corresponding bit in the control word of all bundles of Port2.
[15:11]	Adapt_JBC_indx_2	R/W	'00000'	Index of the jitter buffer used by the clock recovery block to generate the clock of Port2
[10:9]	SF_to_ESF_low_CAS_bits_2	R/W	'00'	Bits that are added as C, D CAS bits to the A, B CAS bits for cases where SF interface is connected by a structured with CAS bundle to an ESF interface.
[8]	TSA_act_blk_2	R/W	'0'	When cleared, TSA bank1 is the active bank for Port2. When set, TSA bank2 is the active bank for Port2. Swapping banks takes effect at the next sync input assertion.

Table 3-73. Port2_cfg_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[7]	Port_Tx_enable_2	R/W	'0'	1 – Incoming TDM traffic from Port2 is enabled 0 – Incoming TDM traffic from Port2 is discarded
[6]	Rx_sample_2	R/W	'1'	When interface is framed or unframed with two_clocks_2='0' this signal has no meaning. When interface is unframed with Two_clocks_2 = '1': • If rx_sample_2 ='0': TDM2_Rx is sampled at the falling edge of TDM2_RCLK. • If rx_sample_2 ='1': TDM2_Rx is sampled at the rising edge of TDM2_RCLK.
[5]	Tx_sample_2	R/W	'0'	When interface is unframed with two_clocks_2='0' or framed: • If tx_sample_2 ='0': TDM2_SYNC, TDM2_TX_MF, TDM2_RX_MF and TDM2_RX are sampled on the falling edge of TDM2_TCLK whereas TDM2_TX and TDM2_TSIG_CTS_D2A1 are updated at the rising edge of TDM2_TCLK. • If tx_sample_2 ='1': TDM2_SYNC, TDM2_TX_MF, TDM2_RX_MF and TDM2_RX are sampled at the rising edge of TDM2_TCLK whereas TDM2_TX and TDM2_TSIG_CTS_D2A1 are updated at the falling edge of TDM2_TCLK. When interface is unframed with Two_clocks_2 = '1': • If tx_sample_2 ='0': TDM2_Tx and TDM2_TSIG_CTS_D2A1 are updated at the rising edge of TDM2_TCLK. • If tx_sample_2 ='1': TDM2_Tx and TDM2_TSIG_CTS_D2A1 are updated at the falling edge of TDM2_TCLK.

Table 3-73. Port2_cfg_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[4]	Two_clocks_2	R/W	'1'	When interface is framed this bit has no meaning. When interface is unframed: • If two_clocks ='0' TDM2_TCLK is used for both receive and transmit paths. • If two_clocks ='1' TDM_RCLK is used for the receive path and TDM2_TCLK is used for the transmit path.
[3:2]	Int_framed_type2	R/W	'00'	00 – unframed (no sync, no mf sync) 01 – framed (sync only, no mf sync) 10 – MF(E1)/SF(T1)(sync and mf sync) 11 – ESF(T1) (sync and mf sync) Changing value from 10 or 11 to 00 or 01 must performed only after asserting RST_SYS_N.
[1:0]	Int_type_2	R/W	'01'	00 – Serial 01 – E1 10 – T1 11 – Reserved

Port3_cfg_reg 0x0,000,010

This register holds port3-related configuration parameters.

Table 3-74. Port3_cfg_reg Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:21]	Clock recovery_PDVT_3 [10:0]	R/W	'0..0'	PDVT value for the bundle used for port3 clock recevery. Holds the configured PDVT of this bundle plus half packet creation time. The resolution is determined by the interface and bundle types as follows: • For AAL1 structured/structured with CAS and for AAL2 bundles: 0.5 ms (only [9:0] bits are relevant). • For unstructured bundles: 32 * interface bit period. Used in conjunction with Clock recovery_PDVT_3[14:11] for unstructured bundle.

[20]	Port_Rx_enable_3	R/W	'0'	1 – Outgoing TDM traffic towards Port3 is enabled 0 – Outgoing TDM traffic towards Port3 is discarded (TDM3_Tx and TDM3_TSIG are set)
[19]	CTS_3	R/W	'1'	Value of TDM3_TSIG_CTS_D2ACS_N, which acts as CTS (Clear To Send) output signal in case of serial interface.
[18]	CD_en_3	R/W	'0'	Output enable of CD_3. Active high. This bit should be activated only for serial interface.
[17]	CD_3	R/W	'1'	Value of TDM3_TX_MF_CD, which acts as CD (Carrier Detect) output signal in case of serial interface.
[16]	Loss_3	R/W	'0'	Loss of sync on TDM Port3. Causes setting of the corresponding bit in the control header of all bundles of Port3.
[15:11]	Adapt_JBC_indx_3	R/W	'00000'	Index of the jitter buffer used by the clock recovery block to generate the clock of Port3.
[10:9]	SF_to_ESF_low_CAS_bits_3	R/W	'00'	Bits that are added as C, D CAS bits to the A, B CAS bits for cases where SF interface is connected by a structured with CAS bundle to an ESF interface.
[8]	TSA_act_blk_3	R/W	'0'	When cleared, TSA bank1 is the active bank for Port3. When set, TSA bank2 is the active bank for Port3. Swapping banks takes effect at the next sync input assertion.

Table 3-74. Port3_cfg_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[7]	Port_Tx_enable_3	R/W	'0'	1 – Incoming TDM traffic from Port3 is enabled 0 – Incoming TDM traffic from Port3 is discarded
[6]	Rx_sample_3	R/W	'1'	When interface is framed or unframed with two_clocks_3='0' this signal has no meaning. When interface is unframed with Two_clocks_3 = '1': • If rx_sample_3 ='0': TDM3_Rx is sampled at the falling edge of TDM3_RCLK. • If rx_sample_3 ='1': TDM3_Rx is sampled at the rising edge of TDM3_RCLK.
[5]	Tx_sample_3	R/W	'0'	When interface is unframed with two_clocks_3='0' or framed: • If tx_sample_3 ='0': TDM3_SYNC, TDM3_TX_MF, TDM3_RX_MF and TDM3_RX are sampled on the falling edge of TDM3_TCLK whereas TDM3_TX and TDM3_TSIG_CTS_D2A4 are updated at the rising edge of TDM3_TCLK. • If tx_sample_3 ='1': TDM3_SYNC, TDM3_TX_MF, TDM3_RX_MF and TDM3_RX are sampled at the rising edge of TDM3_TCLK whereas TDM3_TX and TDM3_TSIG_CTS_D2A4 are updated at the falling edge of TDM3_TCLK. When interface is unframed with Two_clocks_3 = '1': • If tx_sample_3 ='0': TDM3_Tx and TDM3_TSIG_CTS_D2A4 are updated at the rising edge of TDM3_TCLK. • If tx_sample_3 ='1': TDM3_Tx and TDM3_TSIG_CTS_D2A4 are updated at the falling edge of TDM3_TCLK.

Table 3-74. Port3_cfg_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[4]	Two_clocks_3	R/W	'1'	When interface is framed this bit has no meaning. When interface is unframed: • If two_clocks ='0' TDM3_TCLK is used for both receive and transmit paths. • If two_clocks ='1' TDM_RCLK is used for the receive path and TDM3_TCLK is used for the transmit path.
[3:2]	Int_framed_type3	R/W	'00'	00 – unframed (no sync, no mf sync) 01 – framed (sync only, no mf sync) 10 – MF(E1)/SF(T1)(sync and mf sync) 11 – ESF(T1) (sync and mf sync) Changing value from 10 or 11 to 00 or 01 must performed only after asserting RST_SYS_N.
[1:0]	Int_type_3	R/W	'01'	00 – Serial 01 – E1 10 – T1 11 – Reserved

Port4_cfg_reg 0x0,000,014

This register holds port4-related configuration parameters.

Table 3-75. Port4_cfg_reg Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:21]	Clock recovery_PDVT_4 [10:0]	R/W	’0..0’	PDVT value for the bundle used for port4 clock recevery. Holds the configured PDVT of this bundle plus half packet creation time. The resolution is determined by the interface and bundle types as follows: • For AAL1 structured/structured with CAS and for AAL2 bundles: 0.5 ms (only [9:0] bits are relevant). • For unstructured bundles: 32 * interface bit period. Used in conjunction with Clock recovery_PDVT_4[14:11] for unstructured bundle.
[20]	Port_Rx_enable_4	R/W	‘0’	1 – Outgoing TDM traffic towards Port4 is enabled 0 – Outgoing TDM traffic towards Port4 is discarded (TDM4_Tx and TDM4_TSIG are set)
[19]	CTS_4	R/W	‘1’	Value of TDM4_TSIG_CTS_D2ACS_N, which acts as CTS (Clear To Send) output signal in case of serial interface.
[18]	CD_en_4	R/W	‘0’	Output enable of CD_4. Active high. This bit should be activated only for serial interface.
[17]	CD_4	R/W	‘1’	Value of TDM4_TX_MF_CD, which acts as CD (Carrier Detect) output signal in case of serial interface.
[16]	Loss_4	R/W	‘0’	Loss of sync on TDM Port4. Causes setting of the corresponding bit in TDMoIP header of all bundles of Port4.
[15:11]	Adapt_JBC_indx_4	R/W	‘00000’	Index of the jitter buffer used by the clock recovery block to generate the clock of Port4
[10:9]	SF_to_ESF_low_CAS_bits_4	R/W	‘00’	Bits that are added as C, D CAS bits to the A, B CAS bits for cases where SF interface is connected by a structured with CAS bundle to an ESF interface.
[8]	TSA_act_blk_4	R/W	‘0’	When cleared, TSA bank1 is the active bank for Port4. When set, TSA bank2 is the active bank for Port4. Swapping banks takes effect at the next sync input assertion.

Table 3-75. Port4_cfg_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[7]	Port_Tx_enable_4	R/W	'0'	1 – Incoming TDM traffic from Port4 is enabled 0 – Incoming TDM traffic from Port4 is discarded
[6]	Rx_sample_4	R/W	'1'	When interface is framed or unframed with two_clocks_4='0' this signal has no meaning. When interface is unframed with Two_clocks_4 = '1': • If rx_sample_1 ='0': TDM4_Rx is sampled at the falling edge of TDM4_RCLK. • If rx_sample_4 ='1': TDM4_Rx is sampled at the rising edge of TDM4_RCLK.
[5]	Tx_sample_4	R/W	'0'	When interface is unframed with two_clocks_4='0' or framed: • If tx_sample_4 ='0': TDM4_SYNC, TDM4_TX_MF, TDM4_RX_MF and TDM4_RX are sampled on the falling edge of TDM4_TCLK whereas TDM4_TX and TDM4_TSIG_CTS_D2A7 are updated at the rising edge of TDM4_TCLK. • If tx_sample_4 ='1': TDM4_SYNC, TDM4_TX_MF, TDM4_RX_MF and TDM4_RX are sampled at the rising edge of TDM4_TCLK whereas TDM4_TX and TDM4_TSIG_CTS_D2A7 are updated at the falling edge of TDM4_TCLK. When interface is unframed with Two_clocks_4 = '1': • If tx_sample_4 ='0': TDM4_Tx and TDM4_TSIG_CTS_D2A7 are updated at the rising edge of TDM4_TCLK. • If tx_sample_4 ='1': TDM4_Tx and TDM4_TSIG_CTS_D2A7 are updated at the falling edge of TDM4_TCLK.

Table 3-75. Port4_cfg_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[4]	Two_clocks_4	R/W	'1'	When interface is framed this bit has no meaning. When interface is unframed: • If two_clocks ='0' TDM4_TCLK is used for both receive and transmit paths. • If two_clocks ='1' TDM_RCLK is used for the receive path and TDM4_TCLK is used for the transmit path.
[3:2]	Int_framed_type4	R/W	'00'	00 – unframed (no sync, no mf sync) 01 – framed (sync only, no mf sync) 10 – MF(E1)/SF(T1)(sync and mf sync) 11 – ESF(T1) (sync and mf sync) Changing value from 10 or 11 to 00 or 01 must performed only after asserting RST_SYS_N.
[1:0]	Int_type_4	R/W	'01'	00 – Serial 01 – E1 10 – T1 11 – Reserved

Rst_reg 0x0,000,018

This register holds reset bits for receive and transmit paths.

Table 3-76. Rst_reg Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:24]	Reserved			
[23:18]	Rst_tx_bundle_num	R/W	'000000'	Bundle number associated with rst_tx.
[17]	Rst_tx_init	R/W	'0'	1 – Indicates first rst_tx after chip reset. 0 – Regular transmit reset (rst_tx). This bit is also used at high speed mode.
[16]	Rst_tx	R/set	'0'	If set, the relevant transmit Payload Type machine will reset its variables (should be given with bundle number and the appropriate Rst_tx_init). The CPU should poll this bit until it is '0' meaning, "reset acknowledged". This bit is also used at high speed mode
[15:14]	Reserved			
[13;12]	Rst_rx_interface_num	R/W	'00'	Interface number associated with rst_rx (AAL2 only).
[11:7]	Rst_rx_time_slot_num	R/W	'00000'	Timeslot associated with rst_rx (AAL2 only).
[6:1]	Rst_rx_bundle_num	R/W	'000000'	Bundle number associated with rst_rx
[0]	Rst_rx	R/set	'0'	1 – Packet classifier generates a reset frame (Rst_rx_time_slot_num and Rst_rx_bundle_num are valid). The CPU should poll this bit until it finds '0'; this means "reset acknowledged".

TDM_cond_data_reg 0x0,000,01C

This register holds four octets to be transmitted as conditioning data towards TDM. This applies to all bundle types.

Table 3-77. TDM_cond_data_reg Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:24]	TDM_cond_octet_a	R/W	'00000000'	TDM Conditioning octet A (Port 1) Must be set to 0x7E for HDLC bundles Also used at high-speed mode
[23:16]	TDM_cond_octet_b	R/W	'00000000'	TDM Conditioning octet B (Port 2) Must be set to 0x7E for HDLC bundles
[15:8]	TDM_cond_octet_c	R/W	'00000000'	TDM Conditioning octet C (Port 3) Must be set to 0x7E for HDLC bundles
[7:0]	TDM_cond_octet_d	R/W	'00000000'	TDM Conditioning octet D (Port 4) Must be set to 0x7E for HDLC bundles

ETH_cond_data_reg 0x0,000,020

This register holds four octets to be transmitted as conditioning data towards network. This applies only to AAL1 bundles.

Table 3-78. Eth_cond_data_reg Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:24]	ETH_cond_octet_d	R/W	'00000000'	ETH Conditioning octet D
[23:16]	ETH_cond_octet_c	R/W	'00000000'	ETH Conditioning octet C
[15:8]	ETH_cond_octet_b	R/W	'00000000'	ETH Conditioning octet B
[7:0]	ETH_cond_octet_a	R/W	'00000000'	ETH Conditioning octet A

AAL2_HDLC_recovery_reg 0x0,000,024

This register holds the configuration of HDLC and AAL2 payload type machines.

Table 3-79. AAL2_HDLC_ recovery_reg Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:29]	Rx_HDLC_min_flags	R/W	'0'	Minimum flags between 2 adjacent HDLC frames trasnmitted towards TDM. The real value is equal to Rx_hdlc_min_flags plus 1 (range: 1 – 8).
[28:25]	AAL2_uui_cnst_val	R/W	'0000'	Constant value for transmitted UUI if AAL2_uui_is_cnst is set. Range: 0 – 15
[24]	AAL2_uui_is_cnst	R/W	'0'	Indicates a constant UUI value as in AAL2_uui_cnst_val
[23:16]	Reserved			
[15:12]	Clock recovery_PDVT_4 [14:11]	R/W	'0000'	PDVT value for the bundle used for port4 clock recovery. Holds the configured PDVT of this bundle plus half packet creation time. The Resolution is 32 * interface bit period. Relevant and Used in conjunction with Clock recovery_PDVT_4[10:0] for unstructured bundle.
[11:8]	Clock recovery_PDVT_3 [14:11]	R/W	'0000'	PDVT value for the bundle used for port3 clock recovery. Holds the configured PDVT of this bundle plus half packet creation time. The Resolution is 32 * interface bit period. Relevant and Used in conjunction with Clock recovery_PDVT_3[10:0] for unstructured bundle.

Table 3-79. AAL2_HDLC_ recovery_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[7:4]	Clock recovery_PDVT_2 [14:11]	R/W	'0000'	PDVT value for the bundle used for port2 clock recovery. Holds the configured PDVT of this bundle plus half packet creation time. The Resolution is 32 * interface bit period. Relevant and Used in conjunction with Clock recovery_PDVT_2[10:0] for unstructured bundle.
[3:0]	Clock recovery_PDVT_1 [14:11]	R/W	'0000'	PDVT value for the bundle used for port1 clock recovery. Holds the configured PDVT of this bundle plus half packet creation time. The Resolution is determined by the interface and bundle types as follows: • For mode- different then High Speed: 32 * interface bit period. • For high speed interface: 128 * interface bit period. Relevant and Used in conjunction with Clock recovery_PDVT_1[10:0] for unstructured bundle.

Packet_classifier_cfg_reg0 0x0,000,028

This register holds the chip's IP Address1.

Table 3-80. Packet_classifier_cfg_reg0 Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	IP_add1	R/W	'0'	First chip's IP Address. Relevant only for Ethernet to TDM path

Packet_classifier_cfg_reg1 0x0,000,02C

This register holds the chip's IP Address2.

Table 3-81. Packet_classifier_cfg_reg1 Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	IP_add2	R/W	'0'	Second chip's IP Address. Relevant only for Ethernet to TDM path

Packet_classifier_cfg_reg2 0x0,000,030

This register holds the chip's MAC address.

Table 3-82. Packet_classifier_cfg_reg2 Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	MAC_add	R/W	'0'	MAC address bits [31:0]. Relevant only for Ethernet to TDM path

Packet_classifier_cfg_reg3 0x0,000,034

Table 3-83. Packet_classifier_cfg_reg3 Map

Bits	Data Element Name	R/W	Reset Values	Description
[31]	Reserved			
[30:24]	Lst_label_msbits	R/W	'1'	Each MPLS packet inner label MSbits should be compared to these bits
[23]	Discard_switch_7	R/W	'1'	0 – Forward to CPU TDMoIP control packets 1 – Discard those packets
[22]	Discard_switch_6	R/W	'0'	0 – Forward to CPU TDMoIP packets whose Ext_Bundle_Num doesn't match any of the chip's assigned bundles 1 – Discard those packets
[21]	Discard_switch_5	R/W	'0'	0 – Forward to CPU IP/UDP packets whose UDP_Dst_Port_Num is different from TDMoIP_Port_Num 1 – Discard those packets
[20]	Discard_switch_4	R/W	'0'	0 – Forward to CPU IP frames whose Protocol is different from UDP 1 – Discard those packets
[19]	Discard_switch_3	R/W	'0'	0 – Forward to CPU ARP frames whose DST_IP address is the same as chip's IP addresses 1 – Discard those packets
[18]	Discard_switch_2	R/W	'0'	0 – Forward to CPU packets whose Ether Type is different from IP, MPLS or ARP 1 – Discard those packets
[17]	Discard_switch_1	R/W	'1'	0 – Forward to CPU IP packets whose DST_IP address is different from chip's IP addresses 1 – Discard those packets
[16]	Discard_switch_0	R/W	'0'	0 – Forward to CPU ARP packets whose DST_IP address is different from chip's IP addresses 1 – Discard those packets
[15:0]	MAC_add	R/W	0000h	MAC address bits [47:32]. Relevant only for Ethernet to TDM path

Packet_classifier_cfg_reg4 0x0,000,038

Table 3-84. Packet_classifier_cfg_reg4 Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	TDMoIP_port_num2	R/W	085Eh	Packets with UDP destination port equal to "TDMoIP_port_num2" are recognized as "TDMoIP packets"
[15:0]	TDMoIP_port_num1	R/W	085Eh	Packets with UDP destination port equal to "TDMoIP_port_num1" are recognized as "TDMoIP packets"

Packet_classifier_cfg_reg5 0x0,000,03C

Table 3-85. Packet_classifier_cfg_reg5 Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:26]	Reserved			
[25]	En_timestamp	R/W	'0'	If set, Standard format control packets are timestamped by the packet classifier.
[24:13]	Reserved			
[12:0]	OAM_bundle_num	R/W	'0000000000000'	Compared to the received packet external bundle number to identify OAM packets.

General_stat_reg 0x0,000,080

This register holds indications of hardware events. Value '1' indicates that the event occurred. Writing '1' to a bit changes its value to '0'. Writing a '0' to a bit does not change its value.

Table 3-86. General_stat_reg Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:28]	Reserved			
[27]	sticky_filter_ovrflw_4	R/set	'0'	0 – Normal Operation 1 – Port4 clock recovery loop filter is overflowed
[26]	sticky_filter_ovrflw_3	R/set	'0'	0 – Normal Operation 1 – Port3 clock recovery loop filter is overflowed
[25]	sticky_filter_ovrflw_2	R/set	'0'	0 – Normal Operation 1 – Port2 clock recovery loop filter is overflowed
[24]	sticky_filter_ovrflw_1	R/set	'0'	0 – Normal Operation 1 – Port1 clock recovery loop filter is overflowed
[23]	Adapt_delta_rx_q_err_4	R/set	'0'	Provided for debug purposes
[22]	Adapt_delta_rx_q_err_3	R/set	'0'	Provided for debug purposes
[21]	Adapt_delta_rx_q_err_2	R/set	'0'	Provided for debug purposes
[20]	Adapt_delta_rx_q_err_1	R/set	'0'	Provided for debug purposes
[19]	Stability_Clock_Alarm4	R/set	'0'	0 – Normal operation 1 – Port4 recovered clock is not stable (the clock recovery machine does not proceed from acquisition to tracking phase)
[18]	Stability_Clock_Alarm3	R/set	'0'	0 – Normal Operation 1 – Port3 recovered clock is not stable (the clock recovery machine does not proceed from acquisition to tracking phase)
[17]	Stability_Clock_Alarm2	R/set	'0'	0 – Normal Operation 1 – Port2 recovered clock is not stable (the clock recovery machine does not proceed from acquisition to tracking phase)
[16]	Stability_Clock_Alarm1	R/set	'0'	0 – Normal Operation 1 – Port1 recovered clock is not stable (the clock recovery machine does not proceed from acquisition to tracking phase)
[15:12]	Reserved			
[11]	Adapt_freeze_state4	R/set	'0'	Provided for debug purposes
[10]	Adapt_freeze_state3	R/set	'0'	Provided for debug purposes

Table 3-86. General_stat_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[9]	Adapt_freeze_state2	R/set	'0'	Provided for debug purposes
[8]	Adapt_freeze_state1	R/set	'0'	Provided for debug purposes
[7]	Rx_fifo_sof_err	R/set	'0'	'1' – Rx_fifo was flushed due to bundle configuration error '0' – Rx_fifo content is not discarded
[6]	TDM_CPU_buff_err	R/set	'0'	Frames received from TDM discarded due to lack of buffers at TDM_CPU pool.
[5]	Rx_fifo_full	R/set	'0'	Packet received from Ethernet discarded because rx_fifo is full
[4]	MPLS_err	R/set	'0'	Received MPLS packet with more than three labels has been detected
[3]	OAM_ETH_to_CPU_q_full	R/set	'0'	OAM packet received from Ethernet and destined to CPU discarded because ETH_to_CPU queue is full.
[2]	OAM_SW_buff_err	R/set	'0'	OAM packet received from Ethernet and destined to CPU discarded due to lack of SW buffers
[1]	Non_OAM_ETH_to_CPU_q_full	R/set	'0'	Non-OAM packet received from Ethernet and destined to CPU discarded because ETH_to_CPU queue is full.
[0]	Non_OAM_SW_buff_err	R/set	'0'	Non-OAM packet received from Ethernet and destined to CPU discarded due to lack of SW buffers.

Ports_stat_reg 0x0,000,084

This register holds current values of the variables. These values are temporary (not latched) and may change. This register is read-only.

Table 3-87. Ports_stat_reg Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:20]	Reserved			
[19:17]	Adapt_current_state4	R/O	'000'	Port4 clock recovery current state: 0 – Idle 2 – Acquisition 5 – Jitter Buffer Centering 6 – Tracking
[16:14]	Adapt_current_state3	R/O	'000'	Port3 clock recovery current state: 0 – Idle 2 – Acquisition 5 – Jitter Buffer Centering 6 – Tracking
[13:11]	Adapt_current_state2	R/O	'000'	Port2 clock recovery current state: 0 – Idle 2 – Acquisition 5 – Jitter Buffer Centering 6 – Tracking
[10:8]	Adapt_current_state1	R/O	'000'	Port1 clock recovery current state: 0 – Idle 2 – Acquisition 5 – Jitter Buffer Centering 6 – Tracking
[7]	RTS_P4	R/O	'0'	Value of TDM4_rsig_RTS, which acts as RTS (Request To Send) input signal in case of serial interface.
[6]	RTS_P3	R/O	'0'	Value of TDM3_rsig_RTS, which acts as RTS (Request To Send) input signal in case of serial interface.

Table 3-87. Ports_stat_reg Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[5]	RTS_P2	R/O	'0'	Value of TDM2_rsig_RTS, which acts as RTS (Request To Send) input signal in case of serial interface.
[4]	RTS_P1	R/O	'0'	Value of TDM1_rsig_RTS, which acts as RTS (Request To Send) input signal in case of serial interface.
[3]	TSA_int_act_blk_P4	R/O	'0'	Indicates which bank is active: 0 – Port4 TSA bank1 is active 1 – Port4 TSA bank2 is active
[2]	TSA_int_act_blk_P3	R/O	'0'	Indicates which bank is active: 0 – Port3 TSA bank1 is active 1 – Port3 TSA bank2 is active
[1]	TSA_int_act_blk_P2	R/O	'0'	Indicates which bank is active: 0 – Port2 TSA bank1 is active 1 – Port2 TSA bank2 is active
[0]	TSA_int_act_blk_P1	R/O	'0'	Indicates which bank is active: 0 – Port1 TSA bank1 is active 1 – Port1 TSA bank2 is active

Version Register 0x0,000,0C0

Table 3-88. Version Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	Chip_version_reg	R/O		Contains PacketTrunk-4 version, where [7:0] refers to the version (ABCDEF00)

3.3 BUNDLE CONFIGURATION TABLES

AAL1 Per-bundle Configuration [31:0] 0x0,008,000 – 0x0,008,0FC

This table holds 32 less significant bits of AAL1 bundle configuration.

Table 3-89. AAL1 Per-bundle Configuration [31:0] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31]	Rx_ discard_sanity_fail	R/W	None	1 – Discard AAL1 packets which fail the sanity check 0 – Don't discard the above packets
[30:26]	Packet_size	R/W	None	AAL1 cells per Ethernet packet Range: 1–30
[25:21]	Tx_ext_bundle_num	R/W	None	5 upper bits of Tx_ext_bundle_num
[20:19]	Tx_AAL1_bundle_type	R/W	None	Bundle type of transmitted payload: 00 – Unstructured 01 – Structured 10 – Structured with CAS 11 – Reserved
[18]	Tx_CAS_source	R/W	None	Source of transmit CAS bits: 0 – RSIG input 1 – Tx software CAS table
[17]	Tx_dest_framing	R/W	None	Applies to T1 framed traffic 0 – Destination framer operates in SF framing 1 – Destination framer operates in ESF framing
[16]	Tx_cond_data	R/W	None	0 – Regular operation 1 – Use conditioning octet according to ETH_cond_data_reg for transmitted packets.
[15:3]	Rx_ext_bundle_num	R/W	None	Receive external bundle number
[2]	Rx_discard	R/W	None	0 – Pass packets received from Ethernet 1 – Discard packets received from Ethernet
[1]	Rx_dest	R/W	None	Destination of packets received from Ethernet: 0 – HW (TDM) 1 – SW (CPU)
[0]	Rx_ext_bundle_assigned	R/W	None	0 – Bundle is not assigned 1 – Bundle is assigned

AAL1 Per-bundle Configuration [63:32] 0x0,008,100 – 0x0,008,1FC

This table holds 32 bits of AAL1 bundle configuration.

Table 3-90. AAL1 Per-bundle Configuration [63:32] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:22]	Rx_pdvt [9:0]	R/W	None	PDVT value for AAL1 bundles. For unstructured AAL1 bundles (unframed E1/T1, serial or high speed interfaces) used in conjunction with Rx_pdvt (14:10) parameter. The resolution is determined by the interface type as follows: • For framed/multiframed E1 and framed T1: 0.5 ms • For T1 SF: 0.5 ms. It is necessary to adjust the required value as follows: Rx_pdvt = value + [value/3], where [value/3] is the integral part without remainder. For example: if the required value is 5.5 ms (11) the configured Rx_pdvt should be: Rx_pdvt = 11 + [11/3] = 14 • For T1 ESF: 0.5 ms. It is necessary to adjust the required value as follows: Rx_pdvt = value + 2*[value/6] Where [value/6] is the integral part without remainder. For example: If the required value is 5.5 ms (11) the configured Rx_pdvt should be: Rx_pdvt = 11 + 2*[11/6] = 13 • For unframed E1/T1 or serial bundles: 32 * interface bit period. • For high speed interface: (128 * interface bit period). Allowed values: • Minimum allowed value: (For all interfaces types) 3. • For T1 SF: Two LSBits of the configured Rx_pdvt binary value ≠ ‘11’. Following procedure explained above ensures that the '11' value does not occur. • For T1 ESF: Three LSBits of the configured Rx_pdvt binary value ≠ ‘110’ or ‘111’. Following procedure explained above ensures that the ’110’ and ‘111’ values do not occur.

[21:20]	Payload_type_machine	R/W	None	00 – HDLC 01 – AAL1 10 – AAL2 11 – Reserved
[19]	IP_RTP_MPLS	R/W	None	0 – Tx: For IP packet RTP header is not used. For MPLS packet only one label is used. Rx: For IP packet RTP header is not used. 1 – Tx: For IP packet RTP header is used. For MPLS packet two labels are used. Rx: For IP packet RTP header is used.
[18]	Control_Word	R/W	None	0 – Backwards compatible (experimental) format (without control word) 1 – Standard format (with control word)

Table 3-90. AAL1 Per-bundle Configuration [63:32] Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[17:16]	Tx_dest	R/W	None	Destination of packets: 00 – Reserved 01 – Ethernet 10 – CPU 11 – TDM-Rx (Cross Connect)
[15:9]	Rx_max_lost_packets	R/W	None	The maximum number of receive packets inserted upon detection of lost packets
[8:4]	Number_of_ts	R/W	None	One less than number of assigned timeslots per bundle. When Rx_AAL1_bundle_type='00' (unstructured) then Number_of_ts=31; this applies also to high speed mode.
[3:2]	Reserved			
[1]	Tx_IP_MPLS	R/W	None	0 – MPLS framing 1 – IP framing
[0]	Tx_R_bit	R/W	None	0 – Don't set R bit in packet header 1 – Set R bit in packet header

AAL1 Per-bundle Configuration [95:64] 0x0,008,200 – 0x0,008,2FC

This table holds 32 bits of AAL1 bundle configuration.

Table 3-91. AAL1 Per-bundle Configuration [95:64] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:11]	Reserved			
[10:6]	Rx_pdvt [14:10]	R/W	None	Rx_pdvt Msbits. Relevant for unstructured AAL1 only (unframed E1/T1, serial or high speed interfaces). See Rx_pdvt [9:0] description.
[5:4]	Tx_cond_octet_type	R/W	None	Selects ETH_cond_octet to be transmitted towards network: 00 – ETH_cond_octet_a 01 – ETH_cond_octet_b 10 – ETH_cond_octet_c 11 – ETH_cond_octet_d
[3:2]	Rx_AAL1_bundle_type	R/W	None	Bundle type of received packets: 00 – Unstructured 01 – Structured 10 – Structured with CAS 11 – Reserved
[1:0]	Protection_mode	R/W	None	00 – Stop sending packets 01 – Send each packet once with the first header 10 – Send each packet once with the second header 11 – Send each packet twice: one with the first header and one with the second header

AAL1 Per-bundle Configuration [127:96] 0x0,008,300 – 0x0,008,3FC

This table holds 32 MS bits of AAL1 bundle configuration.

Table 3-92. AAL1 Per-bundle Configuration [127:96] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31]	Rx_sig_cond_src	R/W	None	Source of signaling towards TDM: 0 – SDRAM after underrun (it will be frozen after the last signaling is read) 1 – Receive SW defined CAS tables
[30]	Rx_cell_chk_ignore	R/W	None	0 – Discard cells with header parity/CRC errors 1 – Ignore cell header (CRC /parity) checks including AAL1 pointer parity error

[29:20]	Rx_max_buff_size	R/W	None	This is the maximal time interval for which data will be stored before overrun is reached. The Resolution is determined by the interface type as follows: • For framed/multiframed E1 and framed T1: 0.5 ms. • For T1 SF: 0.5 ms. It is necessary to adjust the required value as follows: Rx_max_buff_size = value + [value/3], where [value/3] is the integral part without remainder. For example: If the required value is 5.5 ms (11) the configured Rx_max_buff_size should be: Rx_max_buff_size = 11 + [11/3] = 14 • For T1 ESF: 0.5 ms. It is necessary to adjust the required value as follows: Rx_max_buff_size = value + 2*[value/6], where [value/6] is the integral part without the reminder. For example: If the required value is 5.5 ms (11) the configured Rx_max_buff_size should be: Rx_max_buff_size = 11 + 2*[11/6] = 13 • For unframed E1/T1 or serial bundles: 1024 * interface bit period. • For high speed interface: (4096 * interface bit period). Allowed values: • For T1 SF: Two LSBits of the configured Rx_max_buff_size binary value ≠ '11'. Following procedure explained above ensures that the '11' value does not occur. • For T1 ESF: Three LSBits of the configured Rx_max_buff_size binary value ≠ '110' or '111'. Following procedure explained above ensures that the '110' and '111' values do not occur. • For all interface types the RX_max_buff_size must be greater than Rx_pdvt + PCT (Packet Creation Time) Note: For unframed interfaces, the RX_max_buff_size resolution is different from the Rx_pdvt resolution.
[19:18]	Tx_VLAN_Tag	R/W	None	0 – No VLAN tag in header 1 – VLAN tag exists in header 2 – 2 VLAN tags exist in header 3 – Reserved

[17:0]	Reserved	R/W	None	Must be set to '0'

AAL2 Per-bundle Configuration [31:0] 0x0,008,000 – 0x0,008,0FC

This register holds the configuration of HDLC and AAL2 payload type machines and part of the clock recovering configuration.

Table 3-93. AAL2 Per-bundle Configuration [31:0] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Tx_IP_checksum	R/W	None	IP header checksum for IP total length equals zero
[15:3]	Rx_Ext_Bundle_Num	R/W	None	Receive external bundle number
[2]	Rx_discard	R/W	None	0 – Pass packets received from Ethernet 1 – Discard packets received from Ethernet
[1]	Rx_dest	R/W	None	Destination of packets received from Ethernet: 0 – HW (TDM) 1 – SW (CPU)
[0]	Rx_ext_bundle_assigned	R/W	None	0 – Bundle is not assigned 1 – Bundle is assigned

AAL2 Per-bundle Configuration [63:32] 0x0,008,100 – 0x0,008,1FC

Table 3-94. AAL2 Per-bundle Configuration [63:32] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:22]	Rx_pdvt [9:0]	R/W	None	PDVT value for AAL2 bundles. The Resolution is 0.5msec. Allowed values: Minimum allowed value (for all interfaces types): 3
[21:20]	Payload_type_machine	R/W	None	00 – HDLC 01 – AAL1 10 – AAL2 11 – Reserved
[19]	IP_RTP_MPLS	R/W	None	0 – Tx: No RTP header or MPLS header has one label Rx: No RTP header 1 – Tx: RTP header exists, when MPLS packet header has two labels Rx: RTP header exists in packets
[18]	TDMoIP_header	R/W	None	0 – MPLS framing (without TDMoIP header) 1 – MPLS framing (with TDMoIP header) Ignored when TDMoIP framing
[17:16]	Tx_dest	R/W	None	Destination of packets: 00 – Reserved 01 – Ethernet 10 – CPU 11 – TDM-Rx (Cross Connect)

Table 3-94. AAL2 Per-bundle Configuration [63:32] Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[15]	Reserved			
[14:9]	CPS_packet_size	R/W	None	CPS packet payload size minus one, in bytes
[8]	Rx_CAS_drop	R/W	None	Drop CAS AAL2-CPS packets
[7]	Rx_type3_drop	R/W	None	Drop type 3 AAL2-CPS packets that are not CAS or OAM
[6]	Rx_framed_drop	R/W	None	Drop framed mode signaling AAL2-CPS packets
[5]	Rx_OAM_drop	R/W	None	Drop OAM AAL2-CPS packets
[4]	Rx_unknown_drop	R/W	None	Drop AAL2-CPS packets with unknown UUI value (any value that differs from 0-15,24,26, 27,31)
[3]	Rx_SID_drop	R/W	None	Drop SID AAL2-CPS packets
[2]	Reserved			
[1]	Tx_IP/MPLS	R/W	None	0 – MPLS framing 1 – TDMoIP framing
[0]	Tx_R_bit	R/W	None	1 – Set R bit in packet header 0 – Don’t set R bit in packet header

AAL2 Per-bundle Configuration [95:64] 0x0,008,200 – 0x0,008,2FC

Table 3-95. AAL2 Per-bundle Configuration [95:64] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	Dynamic_ts_assignment	R/W	None	Dynamic assignment of timeslots of the bundle. If bit x is set, timeslot x is assigned. Note that assigned bits must also be assigned at the TSA.

AAL2 Per-bundle Configuration [127:96] 0x0,008,300 – 0x0,008,3FC

Table 3-96. AAL2 Per-bundle Configuration [127:96] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31]	Reserved			
[30]	Rx_cell_chk_ignore	R/W	None	0 – Discard CPS packets with header HEC errors and CAS CPS-packets with CRC-10 error 1 – Ignore AAL2 CPS-packet header HEC checks and CAS CPS-packets with CRC-10 error
[29:20]	Rx_max_buff_size	R/W	None	Maximal time interval for which data will be stored before overrun is reached. The Resolution is 0.5 msec. Allowed values: RX_max_buff_size must be greater than Rx_pdvt + PCT (Packet Creation Time).
[19:18]	Tx_VLAN_tag	R/W	None	0 – No VLAN tag in header 1 – VLAN tag exists in header 2 – 2 VLAN tags exist in header 3 – Reserved
[17:0]	Reserved	R/W	None	Must be set to zero

HDLC Per-bundle Configuration [31:0] 0x0,008,000 – 0x0,008,0FC

Table 3-97. HDLC Per-bundle Configuration [31:0] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Tx_IP_checksum	R/W	None	IP header checksum for IP total length equals zero
[15:3]	Rx_ext_bundle_num	R/W	None	Receive external bundle number
[2]	Rx_discard	R/W	None	0 – Pass packets received from Ethernet 1 – Discard packets received from Ethernet
[1]	Rx_dest	R/W	None	Destination of packets received from Ethernet: 0 – HW (TDM) 1 – SW (CPU)
[0]	Rx_ext_bundle_assigned	R/W	None	0 – Bundle is not assigned 1 – Bundle is assigned

HDLC Per-bundle Configuration [63:32] 0x0,008,100 – 0x0,008,1FC

Table 3-98. HDLC Per-bundle Configuration [63:32] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:22]	Reserved			
[21:20]	Payload_type_machine	R/W	None	00 – HDLC 01 – AAL1 10 – AAL2 11 – Reserved
[19]	RTP_MPLS	R/W	None	0 – Tx: No RTP header Or MPLS header has one label Rx: No RTP header 1 – Tx: RTP header exists or MPLS header has two labels Rx: RTP header exists in packet
[18]	Control_Word	R/W	None	0 – Backwards compatible (experimental) format (without control word) 1 – Standard format (with control word)
[17:16]	Tx_dest	R/W	None	Destination of packets: 00 – Reserved 01 – Ethernet 10 – CPU 11 – TDM-Rx (Cross Connect)
[15:2]	Reserved			
[1]	Tx_IP_MPLS	R/W	None	0 – MPLS framing 1 – TDMoIP framing
[0]	Tx_R_bit	R/W	None	Set R bit in packet header

HDLC Per-bundle Configuration [95:64] 0x0,008,200 – 0x0,008,2FC

Table 3-99. HDLC Per-bundle Configuration [95:64] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:2]	Tx_max_frame_size	R/W	None	Tx HDLC maximum transmitted packet size in bytes This does not include FCS.
[1:0]	Tx_stop	R/W	None	When ‘00’ stop sending packets, otherwise normal operation

HDLC Per-bundle Configuration [127:96] 0x0,008,300 – 0x0,008,3FC

Table 3-100. HDLC Per-bundle Configuration [127:96] Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:20]	Reserved			
[19:18]	Tx_VLAN_tag	R/W	None	00 – No VLAN tag in header 01 – VLAN tag exists in header 10 – 2 VLAN tags exist in header 11 – Reserved
[17:0]	Reserved			Must be set to '0'

3.4 COUNTERS

Per Bundle Counters

Ethernet Rx Good Packets 0x0,010,000 – 0x0,010,0FC

Table 3-101. Ethernet Rx Good Packets Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	Good_packets_received	R/Clear	None	Good packets received from Ethernet

Ethernet Tx Good Packets 0x0,010,200 – 0x0,010,2FC

Table 3-102. Ethernet Tx Good Packets Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	Good_packets_transmitted	R/Clear	None	Good packets transmitted to Ethernet

Ethernet Rx Lost Packets Counter 0x0,010,400 – 0x0,010,4FC

Table 3-103. Ethernet Rx Lost Packets Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved			
[15:0]	Lost_AAL1_packets_received/ Lost_HDLC_packets_received	R/Clear	None	Number of lost packets encountered by rx_aal1/rx_hdlc hardware machine: For AAL1 – The counter is increased by the difference between the received frame sequence number and the expected frame sequence number only when this difference is smaller or equal to the configured max_lost_frames value, For HDLC – The counter is increased by the difference between the received frame sequence number and the expected frame sequence number only when this difference is smaller than 32768.

Rx AAL1 Lost Cells/AAL2 CPS Packets with Error 0x0,010,600 – 0x0,010,6FC

Table 3-104. AAL1 Cells Received with SN errors/AAL2 CPS Packets with Error Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved			
[15:0]	Lost_AAL1_received_cells/ errored_AAL2_received_ CPS_packets	R/Clear	None	Lost AAL1 Cells / AAL2 CPS Packets received with incorrect HEC

TDM Tx HDLC Frames with Error 0x0,010,800 – 0x0,010,8FC

Table 3-105. TDM Tx HDLC Frames with Error Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved			
[15:0]	TDM_HDLC_err_frames	R/Clear	None	HDLC frames received with any error (including CRC/alignment/abort/short/long)

TDM Tx HDLC Good Frames 0x0,010,A00 – 0x0,010,AFC

Table 3-106. TDM Tx HDLC Good Frames Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	TDM_HDLC_good_frames	R/Clear	None	HDLC good frames received from TDM

Per Jitter Buffer Index Counters

Jitter Buffer Underrun/Overrun Events 0x0,010,C00 – 0x0,010,DFC

Table 3-107. Jitter Buffer Underrun/Overrun Events Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	JBC_events	R/Clear	None	Number of jitter buffer underrun/overrun events

General Counters

Ethernet Bytes Received (of Good Packets) 0x0,010,E00

Table 3-108. Ethernet Bytes Received (of good packets) Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	ETH_bytes_received	R/Clear	0	Ethernet bytes received (of good packets)

Ethernet Bytes Transmitted (of good packets) 0x0,010,E04

Table 3-109. Ethernet Bytes Transmitted (of good packets) Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	ETH_bytes_transmitted	R/Clear	0	Ethernet bytes transmitted (of good packets)

Classified Packets Counter 0x0,010,E08

Table 3-110. Classified Packets Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	Classified_packets	R/Clear	0	Packets that pass packet classifier

MAC Counters

These registers reset to zero on a read and stick at all ones when they count to their maximum value. They should be read frequently enough to prevent loss of data. The receive statistics registers are only incremented when the receive enable bit is set in the network control register.

To write to these registers, bit 7 must be set in the network control register. The statistics register block contains the following registers:

Pause Packets Received OK Counter 0x0,070,03C

Table 3-111. Pause Packets Received OK Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved			
[15:0]	Pause_packets_received_OK	R/W	0	A 16-bit register counting the number of good pause packets received. A good packet has a length of 64 to 1518 (1536 if bit 8 set in network configuration register) and has no FCS, alignment or receive symbol errors

Packets Transmitted OK Counter 0x0,070,040

Table 3-112. MAC Packets Transmitted OK Map Counter

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	Packets_transmitted_OK	R/W	0	A 32-bit register counting the number of packets successfully transmitted, i.e., no underrun and not too many retries

Single Collision Packets Counter 0x0,070,044

Table 3-113. Single Collision Packets Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved			
[15:0]	Single_collision_packets	R/W	0	A 16-bit register counting the number of packets experiencing a single collision before being successfully transmitted, i.e. no underrun

Multiple Collision Packets Counter 0x0,070,048

Table 3-114. Multiple Collision Packets Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved			
[15:0]	Multiple_collision_packets	R/W	0	A 16-bit register counting the number of packets experiencing between two and fifteen collisions prior to being successfully transmitted, i.e., no underrun and not too many retries

Packets Received OK Counter 0x0,070,04C

Table 3-115. Packets Received OK Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:24]	Reserved			
[23:0]	Packets_received_OK	R/W	0	A 24-bit register counting the number of good packets received, i.e. packet length is 64 to 1518 bytes (1536 if bit 8 set in network configuration register) and has no FCS, alignment or receive symbol errors.

Packet Check Sequence Errors Counter 0x0,070,050

Table 3-116. Packet Check Sequence Errors Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Packet_check_sequence_errors	R/W	0	An 8-bit register counting packets that are an integral number of bytes, have bad CRC and are between 64 and 1518 bytes in length (1536 if bit 8 set in network configuration register).

Alignment Errors Counter 0x0,070,054

Table 3-117. Alignment Errors Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Alignment_errors	R/W	0	An 8-bit register counting packets that are not an integral number of bytes long and have bad CRC when their length is truncated to an integral number of bytes and are between 64 and 1518 bytes in length (1536 if bit 8 set in network configuration register).

Deferred Transmission Packets Counter 0x0,070,058

Table 3-118. Deferred Transmission Packets Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved			
[15:0]	Deferred_transmission_packets	R/W	0	A 16-bit register counting the number of packets experiencing deferral due to carrier sense being active on their first attempt at transmission. Packets involved in any collision are not counted nor are packets that experienced a transmit underrun.

Late Collisions Counter 0x0,070,05C

Table 3-119. Late Collisions Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Late_collisions	R/W	0	An 8-bit register counting the number of packets that experience a collision after the slot time (512 bits) has expired. A late collision is counted twice i.e. both as a collision and a late collision.

Excessive Collisions Counter 0x0,070,060

Table 3-120. Excessive Collisions Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Excessive_collisions	R/W	0	An 8-bit register counting the number of packets that failed to be transmitted because they experienced 16 collisions.

Transmit Underrun Errors Counter 0x0,070,064

Table 3-121. Transmit Underrun Errors Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Transmit_underruns	R/W	0	An 8-bit register counting the number of packets not transmitted due to a transmit FIFO underrun. If this register is incremented then no other statistics register is incremented.

Carrier Sense Errors Counter 0x0,070,068

Table 3-122. Carrier Sense Errors Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Carrier_sense_errors	R/W	0	An 8-bit register counting the number of packets transmitted where carrier sense was not seen during transmission or where carrier sense was deasserted after being asserted in a transmit packet without collision (no underrun). Only incremented in half duplex mode. The only effect of a carrier sense error is to increment this register. The behaviour of the other statistics registers is unaffected by the detection of a carrier sense error.

Receive Symbol Errors Counter 0x0,070,074

Table 3-123. Receive Symbol Errors Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Receive_symbol_errors	R/W	0	An 8-bit register counting the number of packets that had rx_er asserted during reception.

Excessive Length Errors Counter 0x0,070,078

Table 3-124. Excessive Length Errors Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Excessive_length_packets	R/W	0	An 8-bit register counting the number of packets received exceeding 1518 bytes (1536 if bit 8 set in network configuration register) in length but do not have either a CRC error, an alignment error nor a receive symbol error.

Receive Jabbers Counter 0x0,070,07C

Table 3-125. Receive Jabbers Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Receive_jabbers	R/W	0	An 8-bit register counting the number of packets received exceeding 1518 bytes (1536 if bit 8 set in network configuration register) in length and have either a CRC error, an alignment error or a receive symbol error.

Undersize Packets Counter 0x0,070,080

Table 3-126. Undersize Packets Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Undersize_packets	R/W	0	An 8-bit register counting the number of packets received less than 64 bytes in length, that do not have either a CRC error or an alignment error.

SQE Test Errors Counter 0x0,070,084

Table 3-127. SQE Test Errors Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	SQE_test_errors	R/W	0	An 8-bit register counting the number of packets where col was not asserted within 96 bit times (an interpacket gap) of tx_en being deasserted in half duplex mode.

Transmitted Pause Packets Counter 0x0,070,08C

Table 3-128. Transmitted Pause Packets Counter Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved			
[15:0]	Transmitted_pause_packets	R/W	0	A 16-bit register counting the number of pause packets transmitted.

3.5 STATUS TABLES

The status tables hold indications of hardware events (value of '1' indicates that the event occurred). Once a bit has a value of '1' it maintains the value unless the host CPU changes it. The host CPU can only change a bit value from '1' to '0' by writing '1' to the required bit location. Writing a '0' to a bit does not change its value.

Per Bundle Tables

Rx Payload Type Machine Status 0x0,012,000 – 0x0,012,0FC

Table 3-129. Rx Payload Type Machine Status Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:4]	Reserved			
[3]	Rx_AAL1_ cell_hdr_err	R/set	None	For Rx AAL1 – Cells received with incorrect SN (Sequence Number) protection fields (CRC/parity), corrected and not corrected cell header
[2]	Rx_AAL1_SN_drop_err/ Rx_AAL2_uui_val_err/ Rx_HDLC_frame_sn_oo_window	R/set	None	For Rx AAL1– Packet SN (Sequence Number) error outside the window (window configured by Max_lost_packets). Packets are dropped. For Rx AAL2 – OAM/voice/CAS AAL2 CPS packets received from Ethernet with incorrect (unknown) UUI (relevant only if AAL2 not configured for constant UUI value) For Rx HDLC – Packet SN (Sequence Number) error outside window (window of 32768)
[1]	Rx_AAL1_SN_loss_err/ Rx_AAL2_HEC_err/ Rx_HDLC_frame_sn_in_window	R/set	None	For Rx AAL1– Packet SN (Sequence Number) error within window (window configured by Max_lost_packets). Packets are passed to Rx_FIFO. For Rx AAL2 – Cells received with incorrect HEC. Ignored when Rx_cell_chk_ignore is set. For Rx HDLC – Packet SN (Sequence Number) error within window (window of 32768)
[0]	Rx_AAL1_ptr_mismatch/ Rx_AAL2_CID_err	R/set	None	For Rx AAL1– Cells received with pointer mismatch For Rx AAL2 – AAL2-CPS Packets received with incorrect CID

Tx Payload Type Machine Status 0x0,012,200 – 0x0,012,2FC

Table 3-130. Tx Payload Type Machine Status Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:5]	Reserved			
[4]	Tx_HDLC_abort	R/set	None	For Tx HDLC – TDM frame received with abort indication
[3]	Tx_HDLC_short	R/set	None	For Tx HDLC – TDM received frame shorter than 4 bytes (including CRC bytes)
[2]	Tx_HDLC_long	R/set	None	For Tx HDLC – TDM received frame longer than maximum allowed length
[1]	Tx_HDLC_align_err	R/set	None	For Tx HDLC – TDM received frame with alignment error
[0]	Tx_AAL1_start_mf_mismatch/ Tx_HDLC_CRC_err	R/set	None	For Tx AAL1 – Start of TDM frame or start of TDM multiframe mismatch For Tx HDLC – TDM received frame with CRC error

Tx Buffers Status 0x0,012,400 – 0x0,012,4FC

Table 3-131. Tx Buffers Status Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:1]	Reserved			
[0]	TDM_to_ETH_buff_err	R/set	None	Frames received from TDM are discarded due to lack of Tx buffers

Packet Classifier Status 0x0,012,600 – 0x0,012,6FC

Table 3-132. Packet Classifier Status Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:4]	Reserved			
[3]	Rx_sync_loss	R/set	None	Received packet with “L” indication
[2]	Rx_remote_fail	R/set	None	Received packet with “R” indication
[1]	Rx_length_mismatch_discard	R/set	None	Packet discarded due to mismatch between the packet length and the configuration (for AAL1 bundles only)
[0]	Rx_version_mismatch_discard	R/set	None	Packet discarded due to mismatch between the packet version and the configuration

Per Timeslot Tables

Rx AAL2 Status 0x0,012,800 – 0x0,012,9FC

Table 3-133. Rx AAL2 Status Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:3]	Reserved			
[2]	Rx_AAL2_LI_err	R/set	None	Voice AAL2-CPS Packet received with LI value different from the AAL2-CPS Packet length configured value.
[1]	Rx_AAL2_CRC10_err	R/set	None	AAL2 CAS CPS packet received with CRC-10 error. Not relevant when Rx_cell_chk_ignore is set.
[0]	Rx_AAL2_UUI_seq_err	R/set	None	Voice AAL2-CPS Packets received with non-sequential UUI.

Per JBC Index Tables

Jitter Buffer Control Status 0x0,012,A00 – 0x0,012,BFC

Table 3-134. Jitter Buffer Control Status Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:1]	Reserved			
[0]	JBC_overrun	R/set	None	For AAL1 – overrun has occurred For HDLC – overrun has occurred For AAL2 – overrun has occurred

3.6 TIMESLOT ASSIGNMENT TABLES

Bank1 Per Timeslot Assignment 0x0,018,000 – 0x0,018,1FC

Table 3-135. Bank1 Per Timeslot Assignment Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:19]	Reserved			
[18]	Structured_AAL1	R/W		Must be set for timeslots that are part of AAL1 structured or structured with CAS bundles
[17:16]	Timeslot_width	R/W	None	00 – Reserved 01 – 2 bits + 6 unassigned bits (only for HDLC bundles) 10 – 7 bits + 1 unassigned bit (only for HDLC bundles) 11 – 8 bits
[15]	First_in_bundle	R/W	None	Must be set for the first timeslot of an AAL1/AAL2 bundle. Must be cleared for HDLC bundles.
[14]	Receive_assigned	R/W	None	0 – timeslot is not assigned for the receive path 1 – timeslot is assigned for the receive path
[13]	Transmit_assigned	R/W	None	0 – timeslot is not assigned for the transmit path 1 – timeslot is assigned for the transmit path
[12:7]	Bundle_number	R/W	None	Number of the bundle, which the timeslot is assigned to.
[6:0]	Jitter_buffer_index	R/W	None	Jitter buffer index used for the timeslot. The 5 LSbits should be configured with the number of the first timeslot assigned to the bundle. The 2 MSbits should be configured with the port number: 00 for port1, 01 for port2 etc.

Bank2 Per Timeslot Assignment 0x0,018,200 – 0x0,018,3FC

Table 3-136. Bank2 Per Timeslot Assignment Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:19]	Reserved			
[18]	Structured_AAL1	R/W		Must be set for timeslots that are part of AAL1 structured or structured with CAS bundles
[17:16]	Timeslot_width	R/W	None	00 – Reserved 01 – 2 bits + 6 unassigned bits (only for HDLC bundles) 10 – 7 bits + 1 unassigned bit (only for HDLC bundles) 11 – 8 bits
[15]	First_in_bundle	R/W	None	Must be set for the first timeslot of an AAL1/AAL2 bundle. Must be cleared for HDLC bundles.
[14]	Receive_assigned	R/W	None	0 – timeslot is not assigned for the receive path 1 – timeslot is assigned for the receive path
[13]	Transmit_assigned	R/W	None	0 – timeslot is not assigned for the transmit path 1 – timeslot is assigned for the transmit path
[12:7]	Bundle_number	R/W	None	Number of the bundle, which the timeslot is assigned to
[6:0]	Jitter_buffer_index	R/W	None	Jitter buffer index used for the timeslot. The 5 LSbits hould be configured with the number of the first timeslot assigned to the bundle. The 2 MSbits should be configured with the port number: 00 for port1, 01 for port2 etc.

3.7 CPU QUEUES

TDM CPU Pool

TDM_CPU_pool_insert 0x0,020,000

Table 3-137. TDM_CPU_pool_insert Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:0]	SDRAM_offset	WO	None	13 MSBs of the SDRAM address of the buffer. The lower 9 bits are always zero since all payload type buffers start at 2 kbps

TDM_CPU_pool_level 0x0,020,004

Table 3-138. TDM_CPU_pool_level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Level	RO	'00000000'	Number of buffers that are currently in the pool. These are the buffers that are still available to the Tx payload type machines. Range: 0 to 128.

TDM_CPU_pool_thresh 0x0,020,008

Table 3-139. TDM_CPU_pool_thresh Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Threshold	RW	'00000000'	If the number of buffers in the pool ≤ this level, an interrupt will be generated. Range: 0 to 128.

TDM to CPU Queue

TDM_to_CPU_q_read 0x0,020,00C

Table 3-140. TDM_to_CPU_q_read Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:0]	SDRAM_offset	RO	None	13 MSBs of the SDRAM address of the buffer. The lower 9 bits are always zero since all payload type buffers start at 2 kbytes.

TDM_to_CPU_q_level 0x0,020,010

Table 3-141. TDM_to_CPU_q_level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Level	RO	'00000000'	This field holds the number of buffers that are currently in the queue. These are the buffers still waiting to be handled by the CPU. Range: 0 to 128.

TDM_to_CPU_q_thresh 0x0,020,014

Table 3-142. TDM_to_CPU_q_thresh Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Threshold	RW	'00000000'	The threshold of the TDM to CPU queue. If the number of buffers in the queue ≥ this level, an interrupt will be generated. Range: 0 to 128.

CPU to Ethernet Queue

CPU_to_ETH_q_insert 0x0,020,018

Table 3-143. CPU_to_ETH_q_insert Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:0]	SDRAM_offset	WO	None	13 MSBs of the SDRAM address of the buffer. The lower 9 bits are always zero since each payload type buffer size is 2 kbytes.

CPU_to_ETH_q_level 0x0,020,01

Table 3-144. CPU_to_ETH_q_level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Level	RO	'000000'	Number of buffers that are currently in the queue. Range: 0 to 32.

CPU_to_ETH_q_thresh 0x0,020,020

Table 3-145. CPU_to_ETH_q_thresh Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Threshold	RW	'000000'	If the number of buffers in the queue ≥ this level an interrupt will be generated. Range: 0 to 32.

Ethernet CPU Pool

ETH_CPU_pool_insert 0x0,020,024

Table 3-146. ETH_CPU_pool_insert Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:0]	SDRAM_offset	WO	None	13 MSBs of the SDRAM address of the buffer. The lower 9 bits are always zero since each payload type buffer size is 2 kbytes.

ETH_CPU_pool_level 0x0,020,028

Table 3-147. ETH_CPU_pool_level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:7]	Reserved			
[6:0]	Level	RO	'0000000'	Number of buffers that are currently in the pool. These are the buffers that are still available to the Rx Arbiter. Range: 0 to 64.

ETH_CPU_pool_thresh 0x0,020,02C

Table 3-148. ETH_CPU_pool_thresh Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:7]	Reserved			
[6:0]	Threshold	RW	'0000000'	The threshold of the pool. If the number of buffers in the pool ≤ this level an interrupt will be generated. Range: 0 to 64.

Ethernet to CPU Queue

ETH_to_CPU_q_read 0x0,020,030

Table 3-149. ETH_to_CPU_q_read Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:0]	SDRAM_offset	RO	None	13 MSBs of the SDRAM address of the buffer. The lower 9 bits are always zero since each payload type buffer size is 2 kbytes.

ETH_to_CPU_q_level 0x0,020,034

Table 3-150. ETH_to_CPU_q_level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:7]	Reserved			
[6:0]	Level	RO	'0000000'	Number of buffers that are currently in the queue. These are the buffers still waiting to be handled by the CPU. Range: 0 to 64.

ETH_to_CPU_q_thresh 0x0,020,038

Table 3-151. ETH_to_CPU_q_thresh Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:7]	Reserved			
[6:0]	Threshold	RW	'0000000'	Threshold of the ETH to CPU queue. If the number of buffers in the queue ≥ this level an interrupt will be generated. Range: 0 to 64.

AAL2 CPU Pool

AAL2_CPU_pool_insert 0x0,020,03C

Table 3-152. AAL2_CPU_pool_insert Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:0]	SDRAM_offset	WO	None	13 MSBs of the SDRAM address of the buffer. The lower 9 bits are always zero since each payload type buffer size is 2 kbytes.

AAL2_CPU_pool_level 0x0,020,040

Table 3-153. AAL2_CPU_pool_level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Level	RO	'000000'	Number of buffers that are currently in the pool. These are the buffers that are still available to the AAL2. Range: 0 to 32.

AAL2_CPU_pool_thresh 0x0,020,044

Table 3-154. AAL2_CPU_pool_thresh Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Threshold	RW	'000000'	The threshold of the AAL2 CPU Pool. If the number of buffers in the pool ≤ this level an interrupt will be generated. Range: 0 to 32.

AAL2 to CPU Queue

AAL2_to_CPU_q_read 0x0,020,048

Table 3-155. AAL2_to_CPU_ q_read Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:0]	SDRAM_offset	RO	None	13 MSBs of the SDRAM address of the buffer. The lower 9 bits are always zero since each payload type buffer size is 2 kbytes.

AAL2_to_CPU_q_level 0x0,020,04C

Table 3-156. AAL2_to_CPU_ q_level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Level	RO	'000000'	Number of buffers that are currently in the queue. Range: 0 to 32.

AAL2_to_CPU_q_thresh 0x0,020,050

Table 3-157. AAL2_to_CPU_ q_thresh Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Threshold	RW	'000000'	If the number of buffers in the queue ≥ this level an interrupt will be generated. Range: 0 to 32.

CPU to TDM Queue

CPU_to_TDM_q_insert 0x0,020,054

Table 3-158. CPU_to_TDM_q_insert Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:0]	SDRAM_offset	WO	None	13 MSBs of the SDRAM address of the buffer. The lower 9 bits are always zero since each payload type buffer size is 2 kbytes.

CPU_to_TDM_q_level 0x0,020,058

Table 3-159. CPU_to_TDM_q_level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Level	RO	'000000'	Number of buffers that are currently in the queue. Range: 0 to 32.

CPU_to_TDM_q_thresh 0x0,020,05C

Table 3-160. CPU_to_TDM_q_thresh Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Threshold	RW	'000000'	If the number of buffers in the queue ≥ this level an interrupt will be generated. Range: 0 to 32.

Tx Return Queue

Tx_return_q_read 0x0,020,060

Table 3-161. Tx_return_q_read Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:0]	SDRAM_offset	RO	None	13 MSBs of the SDRAM address of the buffer. The lower 9 bits are always zero since each payload type buffer size is 2 kbytes.

Tx_return_q_level 0x0,020,064

Table 3-162. Tx_return_q_level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Level	RO	'000000'	Number of buffers that are currently in the queue. Range: 0 to 32.

Tx_return_q_thresh 0x0,020,068

Table 3-163. Tx_return_q_thresh Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Threshold	RW	'000000'	If the number of buffers in the queue ≥ this level an interrupt will be generated. Range: 0 to 32.

Rx Return Queue

Rx_return_q_read 0x0,020,06C

Table 3-164. Rx_return_q_read Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:13]	Reserved			
[12:0]	SDRAM_offset	RO	None	13 MSBs of the SDRAM address of the buffer. The lower 9 bits are always zero since each payload type buffer size is 2 kbytes.

Rx_return_q_level 0x0,020,070

Table 3-165. Rx_return_q_level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Level	RO	'000000'	Number of buffers that are currently in the queue. Range: 0 to 32.

Rx_return_q_thresh 0x0,020,074

Table 3-166. Rx_return_q_thresh Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:6]	Reserved			
[5:0]	Threshold	RW	'000000'	If the number of buffers in the queue > this level an interrupt will be generated. Range: 0 to 32.

3.8 TRANSMIT BUFFER POOL

Per Bundle Head 0x0,028,800 – 0x0,028,8FC

The RAM should be initialized by the CPU in order to hold the heads of the lists for all open bundles.

Table 3-167. Per Bundle Head Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:10]	Reserved			
[9]	Buffer_valid	R/W	None	0 – The head contains non-valid information. That is, the pool is empty. 1 – The head points to a valid free buffer.
[8:0]	Buffer_id	R/W	None	The full address of the buffer consists of the Tx buffer base address (defined in General_cfg_reg1 [9:6]), concatenated with the buffer offset and eleven '0's.

Per Bundle Next Buffer 0x0,028,000 – 0x0,028,7FC

A pointer to the next buffer in the list.

The RAM should be initialized by the CPU, in order to hold the lists for all the bundles.

Table 3-168. Per Bundle Next Buffer Map

Bits	Data Element Name	R/W	Reset values	Description
[31:9]	Reserved			
[8:0]	Buffer_offset	R/W	None	The offset of the next buffer in the linked list at the SDRAM area dedicated to the Tx payload type machines. The full address of the buffer consists of the Tx buffer base address (defined in General_cfg_reg1 [9:6]) concatenated with the buffer offset and eleven '0's.

3.9 JITTER BUFFER CONTROL

Status_and_level Registers 0x0,030,000 – 0x0,030,3F8

HDLC

Table 3-169. HDLC Status and Level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:2]	Reserved			
[1:0]	Status	RO	None	The status of the bundle's jitter buffer: 00 – Jitter Buffer is empty 01 – Jitter Buffer is OK 10 – Jitter Buffer is full 11 – Reserved

High Speed HDLC

Table 3-170. HDLC High Speed Status and Level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Current_level	RO	'00000000'	The 16 MSBs of the current level (the level is 19 bits wide)
[15:2]	Reserved			
[1:0]	Status	RO	'00000000'	The status of the bundle's Jitter Buffer: 00 – Jitter Buffer is empty 01 – Jitter Buffer is OK 10 – Jitter Buffer is full 11 – Reserved

Structured AAL1

Table 3-171. Structured AAL1 Status and Level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:26]	Reserved			
[25:16]	Current_level	RO	None	The current level for the bundle. The resolution is 0.5msec.
[15:2]	Reserved			
[1:0]	Status	RO	None	The status of the bundle's Jitter Buffer: 00 – Jitter Buffer is empty 01 – Jitter Buffer is OK 10 – Jitter Buffer is full 11 – Reserved

Unstructured AAL1

Table 3-172. Unstructured AAL1 Status and Level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:2]	Reserved			
[1:0]	Status	RO	None	The status of the bundle's Jitter Buffer: 00 – Jitter Buffer is empty 01 – Jitter Buffer is OK 10 – Jitter Buffer is full 11 – Reserved

High Speed AAL1

Table 3-173. AAL1 High Speed Status and Level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Current_level	RO	'00000000'	The 16 MSBs of the current level (the level is 17 bits wide). The resolution is 64*interface bit period.
[15:2]	Reserved			
[1:0]	Status	RO	'00000000'	The status of the bundle's Jitter Buffer: 00 – Jitter Buffer is empty 01 – Jitter Buffer is OK 10 – Jitter Buffer is full 11 – Reserved

AAL2

Table 3-174. AAL2 Status and Level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:26]	Reserved			
[25:16]	Current-level	RO	None	Current level for the bundle. The resolution is 0.5 ms.
[15:2]	Reserved			
[1:0]	Status	RO	None	The status of the bundle's Jitter Buffer: 00 – Jitter Buffer is empty 01 – Jitter Buffer is OK 10 – Jitter Buffer is full 11 – Reserved

Min_and_max_levels 0x0,030,004 – 0x0,030,3FC

Structured AAL1

Table 3-175. Structured AAL1 Minimum and Maximum Levels Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:26]	Reserved			
[25:16]	Minimal_level	RO	None	The minimal level that the JBC has reached since the last read (read operation causes the minimal level register to be all ones). When underrun is reached, the value remains zero until it is read by CPU. The resolution is 0.5 ms.
[15:10	Reserved			
[9:0]	Maximal_level	RO	None	The maximal level that the JBC has reached since the last read (read operation causes the level to be zero). When overrun is reached, the value remains equal to max_buf_size until it is read by CPU. The resolution is 0.5 ms.

Unstructured AAL1

Table 3-176. Unstructured AAL1 Minimum and Maximum Levels Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:26]	Reserved			
[25:16]	Minimal_level	RO	None	The 10 MSBs of the minimal level that the JBC has reached since the last read (read operation causes the minimal level register to be all ones). When underrun is reached, the value remains zero until it is read by CPU. The resolution is 1024* interface bit period..
[15:10]	Reserved			
[9:0]	Maximal_level	RO	None	The 10 MSBs of the maximal level that the JBC has reached since the last read (read operation causes the level to be zero). When overrun is reached, the value remains equal to max_buf_size until it is read by CPU. The resolution is 1024* interface bit period.

High Speed AAL1

Table 3-177. AAL1 High Speed Minimum and Maximum Level Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Minimal_level	RO	0xFFFF	The 16 MSBs of the minimal level that the JBC has reached since the last read (read operation causes the minimal level register to be all ones). When underrun is reached, the value remains zero until it is read by CPU. The level is 17 bits wide. The resolution is 64*interface bit period
[15:0]	Maximal_level	RO	'00000000'	The 16 MSBs of maximal level that the JBC has reached since the last read (read operation causes the level to be zero). When overrun is reached, the value remains equal to max_buf_size until it is read by CPU. The level is 17 bits wide. The resolution is 64*interface bit period

AAL2

Table 3-178. AAL2 Minimum and Maximum Levels Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:26]	Reserved			
[25:16]	Minimal_level	RO	None	The minimal level that the JBC has reached since the last read (read operation causes the minimal level register to be all ones). When underrun is reached, the value remains zero until it is read by CPU. The resolution is 0.5 ms.
[15:10	Reserved			
[9:0]	Maximal_level	RO	None	The maximal level that the JBC has reached since the last read (read operation causes the level to be zero). When overrun is reached, the value remains equal to max_buf_size until it is read by CPU. The resolution is 0.5 ms.

Bundle Timeslot Registers 0x0,030,700 – 0x0,030,7FC

Table 3-179. Bundle Timeslot Registers Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	Ts_assigned			Assigned timeslot of the bundle: 1 – Bit is assigned 0 – Bit is unassigned *Note: When the interface type is Nx64 the Bundle TS RAM entry for the bundle should be configured to all 1's.*

3.10 TRANSMIT SOFTWARE CAS

Transmit Software CAS 0x0,038,000 – 0x0,038,0FC

Table 3-180. Transmit Software CAS Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Two_CAS_nibbles	R/W	None	CAS signaling of two adjacent timeslots

3.11 RECEIVE LINE CAS

Timeslot Receive Line CAS 0x0,040,000 – 0x0,040,17C

Table 3-181. Timeslot Receive Line CAS Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:4]	Reserved			
[3:0]	Rx_CAS	R/O	None	Timeslot CAS towards TSIG

3.12 CLOCK RECOVERY

When using the clock recovery mechanism of a certain port, its Clock recovery_PDVT parameter in the Portn_cfg_reg must also be configured.

Control Word 0x0,048,000 – 0x0,048,C00

Table 3-182. Control Word Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:17]	Reserved			
[16]	Cpu_trk2_jb_lvl_tresh_en	R/W	0	Set according to the HAL function
[15]	Trk1_after_frz	R/W	0	Set according to the HAL function
[14]	Smart_fll_en	R/W	0	Set according to the HAL function
[13]	Acq_after_frz	R/W	0	Set according to the HAL function
[12]	Jb_centering_phase_en	R/W	0	Set according to the HAL function
[11]	Watch_dog_en	R/W	0	Set according to the HAL function
[10]	Pll_Acq	R/W	0	Set according to the HAL function
[9]	Acq_tick_src	R/W	0	Set according to the HAL function
[8]	Trk_tick_src	R/W	0	Set according to the HAL function
[7]	Jb_lv_resolution	R/W	0	Set according to the HAL function
[6]	Fine_trk_en	R/W	0	Set according to the HAL function
[5]	Jb_lvl_avg_en	R/W	0	Set according to the HAL function
[4]	Tracking_disable	R/W	0	Set according to the HAL function
[3]	Port–en	R/W	0	Set according to the HAL function
[2]	Acq_enable	R/W	0	Set according to the HAL function
[1]	Freeze	R/W	0	Set according to the HAL function
[0]	Rst	W/O	0	Set according to the HAL function

Note: Reading the clock register 0x0,048,000 when the Clock block is disabled can hang up the µP interface.

3.13 RECEIVE CONDITIONING DATA

Receive Conditioning Data 0x0,050,000 – 0x0,050,1FC

Table 3-183. Receive Conditioning Data Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:2]	Reserved			
[1:0]	Cond_octet_sel	R/W	None	2 bits, which select the value of the conditioning octet to be transmitted on TDM_Tx, when timeslot is not assigned. This value is also the conditioning octet that is inserted into the jitter buffer for lost packet compenation. 00 – TDM_cond_octet_a 01 – TDM_cond_octet_b 10 – TDM_cond_octet_c 11 – TDM_cond_octet_d

3.14 RECEIVE SW DEFINED CAS

Receive SW Defined CAS 0x0,058,000 – 0x0,058,1FC

Table 3-184. Receive SW Defined CAS Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:4]	Reserved			
[3:0]	Rx_CAS	R/W	None	Receive CAS bits of the timeslot

3.15 TIMESLOT TO CID TABLE

Timeslot to CID Table 0x0,060,000 – 0x0,060,1FC

Table 3-185. Transmit AAL2 Timeslot CID Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:8]	Reserved			
[7:0]	Ts_CID	R/W	'00000000'	CID of the timeslot

3.16 INTERRUPT CONTROLLER

Intpend Register 0x0,068,000

Table 3-186. Intpend Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:23]	Reserved			
[22]	TDM_CPU_pool_thresh	R/W	'0'	1 – TDM CPU pool level < threshold.
[21]	TDM_to_CPU_q_thresh	R/W	'0'	1 – TDM to CPU queue level > threshold.
[20]	CPU_to_ETH_q_thresh	R/W	'0'	1 – CPU to Ethernet queue level < threshold.
[19]	ETH_CPU_pool_thresh	R/W	'0'	1 – Ethernet to CPU pool level < threshold.
[18]	ETH_to_CPU_q_thresh	R/W	'0'	1 – Ethernet to CPU queue level >threshold
[17]	AAL2_CPU_ pool_thresh	R/W	'0'	1 – AAL2 to CPU pool level < threshold
[16]	AAL_to_CPU_q_thresh	R/W	'0'	1 – AAL2 to CPU queue level >threshold.
[15]	CPU_to_TDM_q_thresh	R/W	'0'	1 – CPU to TDM queue level < threshold.
[14]	Tx_return_q_thresh	R/W	'0'	1 – CPU TX return queue level >threshold.
[13]	Rx_ return_q_thresh	R/W	'0'	1 – CPU RX return queue level went above threshold.
[12]	JBC_underrun_P4	R/W	'0'	One of the Port4 Jitter Buffers is in underrun state.
[11]	JBC_underrun_P3	R/W	'0'	One of the Port3 Jitter Buffers is in underrun state.
[10]	JBC_underrun_P2	R/W	'0'	One of the Port2 Jitter Buffers is in underrun state.
[9]	JBC_underrun_P1	R/W	'0'	One of the Port1 Jitter Buffers is in underrun state.
[8]	RTS4_ changed	R/W	'0'	RTS4 input level was changed.
[7]	RTS3_ changed	R/W	'0'	RTS3 input level was changed.
[6]	RTS2_ changed	R/W	'0'	RTS2 input level was changed.
[5]	RTS1_ changed	R/W	'0'	RTS1 input level was changed.
[4]	ETH_MAC	R/W	'0'	Ethernet MAC interrupt
[3]	Rx_CAS_change_P4	R/W	'0'	1 – A change has occurred at Port4 receive line CAS table, at one of the timeslots.
[2]	Rx_CAS_change_P3	R/W	'0'	1 – A change has occurred at Port3 receive line CAS table, at one of the timeslots.
[1]	Rx_CAS_change_P2	R/W	'0'	1 – A change has occurred at Port2 receive line CAS table, at one of the timeslots.
[0]	Rx_CAS_change_P1	R/W	'0'	1 – A change has occurred at Port1 receive line CAS table, at one of the timeslots.

Intmask Register 0x0,068,004

Table 3-187. Intmask Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:23]	Reserved			
[22]	TDM_CPU_pool_thresh	R/W	'1'	Mask of TDM_CPU_pool_thresh
[21]	TDM_to_CPU_q_thresh	R/W	'1'	Mask of TDM_to_CPU_q_thresh
[20]	CPU_to_ETH_q_thresh	R/W	'1'	Mask of CPU_to_ETH_q_ thresh
[19]	ETH_CPU_pool_thresh	R/W	'1'	Mask of ETH_CPU_pool_ thresh
[18]	ETH_to_CPU_q_thresh	R/W	'1'	Mask of ETH_to_CPU_q_ thresh
[17]	AAL2_CPU_ pool_thresh	R/W	'1'	Mask of AAL2_CPU_pool_thresh
[16]	AAL2_to_CPU_q_thresh	R/W	'1'	Mask of AAL2_to_CPU_q_ thresh
[15]	CPU_to_TDM_q_thresh	R/W	'1'	Mask of CPU_to_TDM_q_ thresh
[14]	Tx_return_q_thresh	R/W	'1'	Mask of tx_return_ thresh
[13]	Rx_ return_q_thresh	R/W	'1'	Mask of rx_ return _ thresh
[12]	JBC_underrun_P4	R/W	'1'	Mask of JBC_underrun_P4
[11]	JBC_underrun_P4	R/W	'1'	Mask of JBC_underrun_P3
[10]	JBC_underrun_P2	R/W	'1'	Mask of JBC_underrun_P2
[9]	JBC_underrun_P1	R/W	'1'	Mask of JBC_underrun_P1
[8]	RTS4_ changed	R/W	'1'	Mask of RTS4_ changed
[7]	RTS3_ changed	R/W	'1'	Mask of RTS3_ changed
[6]	RTS2_ changed	R/W	'1'	Mask of RTS2_ changed
[5]	RTS1_ changed	R/W	'1'	Mask of RTS1_ changed
[4]	ETH_MAC	R/W	'1'	Mask of ETH_MAC
[3]	Rx_CAS_change_P4	R/W	'1'	Mask of Rx_CAS_change_P4
[2]	Rx_CAS_change_P3	R/W	'1'	Mask of Rx_CAS_change_P3
[1]	Rx_CAS_change_P2	R/W	'1'	Mask of Rx_CAS_change_P2
[0]	Rx_CAS_change_P1	R/W	'1'	Mask of Rx_CAS_change_P1

Rx_CAS_change Registers 0x0,068,040/0x0,068,044/0x0,068,048/0x0,068,04C

Table 3-188. Rx_CAS_change Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	Rx_CAS_change	R/W	'00000000 00000000 00000000 00000000'	Each bit set represents a change in the Rx CAS, at one of the port's timeslots.

JBC_Underrun_Reg Registers 0x0,068,080/0x0,068,088/0x0,068,090/0x0,068,098

Table 3-189. JBC Underrun Pending Register

Bits	Name	R/W	Reset Values	Description
[31:0]	JBC_underrun	R/W	0x00000000	Each bit set represents an underrun, at one of the port's timeslots.

JBC_Underrun_Mask_Reg Registers 0x0,068,084/0x0,068,08C/0x0,068,094/0x0,068,09C

Table 3-190. JBC Underrun Mask Register

Bits	Name	R/W	Reset Values	Description
[31:0]	JBC_underrun	R/W	0xFFFFFFFF	Each bit set represents mask for the corresponding bit in JBC underrun pending register

3.17 ETH MAC BLOCK

MAC_Network_Control_Register 0x0,070,000

Table 3-191. MAC Network Control Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31: 13]	Reserved.	RO	'0'	Read as zero, ignored on write
[12]	Transmit_zero_quantum_ pause_ packet.	WO		Writing a '1' to this bit will transmit a pause packet with zero pause quantum, at the next available transmitter idle time
[11]	Transmit_pause_packet	WO		Writing '1' to this bit transmits a pause packet with the pause quantum from the Transmit Pause Quantum register — at the next available transmitter idle time.
[10:9]	Reserved			
[8]	Back_pressure	R/W	'0'	If set in half duplex mode will force collisions on all received packets.
[7]	Write_enable_for_statistics_ registers	R/W	'0'	Setting this bit to '1' makes the statistics registers writable for functional test purposes.
[6]	Increment_statistics_registers	WO	'0'	Writing a one increments all the statistics registers by one for test purposes.
[5]	Clear_statistics_registers	WO	'0'	Writing a one clears the statistics registers.
[4]	Management_port_enable	R/W	'0'	Set to '1' to enable the management port. 0 – Forces MDIO to high impedance state and MDC low.
[3]	Transmit_enable	R/W	'0'	1 – Enables the Ethernet transmitter to send data. 0 – Transmission will stop immediately, the transmit FIFO and control registers will be cleared and the transmit queue pointer register will reset to point to the start of the transmit descriptor list. This bit must be set during normal operation.
[2]	Receive_enable	R/W	'0'	1 – Enables the MAC to receive data 0 – Packet reception will stop immediately
[1:0]	Reserved			

MAC Network Configuration Register 0x0,070,004

Table 3-192. MAC Network Configuration Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:20]	Reserved		'0'	Read aas zero, ignored on write
[19]	Ignore_Rx_FCS	R/W	'0'	When set packets with FCS/CRC errors will not be rejected and no FCS error statistics will be counted. for normal operation, this bit must be set to 0
[18]	Enable_half_duplex_Rx	R/W	'0'	Enable packets to be received in half-duplex mode while transmitting.
[17]	Reserved			
[16]	Receive_length_field_checking _ enable	R/W	'0'	When set, packets with measured lengths shorter than their length fields will be discarded. Packets containing a type ID in bytes 13 and 14 — length/type ID ≥0600 — will not be counted as length errors
[15:14]	Reserved			
[13]	Pause_enable	R/W	'0'	When set transmission will pause when a valid pause packet is received

Table 3-192. MAC Network Configuration Register Map (Cont.)

Bits	Data Element Name	R/W	Reset Values	Description
[12]	Retry_test	R/W	'0'	Must be set to zero for normal operation. If set to one the back off between collisions will always been one slot time. Setting this bit to one helps testing the too many retries condition. Also used in the pause packet tests to reduce the pause counters decrement time from 512 bit times, to every rx_clk cycle
[11:10]	MDC_frequency	R/W	'10'	Set according to clk_sys speed. This determines by what number clk_sys will be divided to generate MDC. For conformance with 802.3 MDC must not exceed 2.5MHz (MDC is only active during MDIO read and write operations). 00 – divide clk_sys by 8 (clk_sys up to 20 MHz) 01 – divide clk_sys by 16 (clk_sys up to 40 MHz) 10 – divide clk_sys by 32 (clk_sys up to 80 MHz) 11 – divide clk_sys by 64 (clk_sys up to 160 MHz)
[9]	Reserved			
[8]	Receive_1536_byte_packets	R/W	'0'	Setting this bit means the MAC will receive packets up to 1536 bytes in length. Normally the MAC would reject any packet above 1518 bytes
[7:5]	Reserved			Must be set to zero
[4]	Reserved	R/W	'0'	Must be set to 1
[3:2]	Reserved			Must be set to zero
[1]	Full_duplex	R/W	'0'	If set to 1 the transmit block ignores the state of collision and carrier sense and allows receive while transmitting.
[0]	Speed	R/W	'0'	1 – Indicates 100 Mbit/s operation 0 – Indictaes 10 Mbit/s operation. Used only for RMII and SMII interfaces.

MAC Network Status Register 0x0,070,008

Table 3-193. MAC Network Status Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:3]	Reserved			
[2]	PHY_access_has_completed	RO	'1'	The PHY management logic is idle.
[1:0]	Reserved			

MAC Transmit Status Register 0x0,070,014

Description

This register, when read, provides details of the status of a transmit. Once read, individual bits may be cleared by writing 1 to them. It is not possible to set a bit to 1 by writing to the register.

Table 3-194. MAC Transmit Status Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:7]	Reserved			
[6]	Transmit_underrun	R/W	'0'	Set when MAC transmit FIFO was read while was empty. If this happens the transmitter will force bad CRC and tx_er high. Cleared by writing a '1' to this bit
[5:3]	Reserved			
[2]	Retry_limit_exceeded	R/W	'0'	Cleared by writing a '1' to this bit
[1]	Collision_occurred	R/W	'0'	Set by the assertion of collision. Cleared by writing a '1' to this bit
[0]	Reserved			

MAC Interrupt Status Register 0x0,070,024

Description

The MAC generates a single interrupt. This register indicates the source of this interrupt. For test purposes each bit can be set or reset by directly writing to the interrupt status register regardless of the state of the mask register. Otherwise the corresponding bit in the mask register must be cleared for a bit to be set. All bits are reset to zero on read. If any bit is set in this register the Ethernet_int signal will be asserted

Table 3-195. MAC Interrupt Status Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:14]	Reserved	RO	'0'	Read 0, ignored on write
[13]	Pause_time_zero	R/W	'0'	Set when the pause time register, 0x38 decrements to zero. Cleared on a read
[12]	Pause_packet_ received	R/W	'0'	Indicates a valid pause has been received. Cleared on a read.
[11:6]	Reserved			
[5]	Retry_limit_exceeded	R/W	'0'	Transmit error. Cleared on read.
[4]	Ethernet_transmit_underrun	R/W	'0'	Set when MAC transmit FIFO was read while was empty. If this happens the transmitter will force bad CRC and tx_er high. Cleared on read
[3:1]	Reserved			
[0]	Management_packet_sent	R/W	'0'	The PHY maintenance register has completed its operation. Cleared on read.

MAC Interrupt Enable Register 0x0,070,028

Description

At reset all interrupts are disabled. Writing a one to the relevant bit location enables the required interrupt. This register is write-only.

Table 3-196. MAC Interrupt Enable Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:14]	Reserved			Set to zero
[13]	Enable_pause_time_zero_ interrupt	WO		
[12]	Enable_pause_packet_ received_interrupt	WO		
[11:6]	Reserved			Set to zero
[5]	Enable_retry_limit_exceeded_ interrupt	WO		
[4]	Enable_transmit_buffer_ underrun_interrupt	WO		
[3:1]	Reserved	WO		Set to zero
[0]	Enable_management_done_ interrupt	WO		

MAC Interrupt Disable Register 0x0,070,02C

Description

Writing a 1 to the relevant bit location disables that particular interrupt. This register is write-only.

Table 3-197. MAC Interrupt Disable Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:14]	Reserved			Each bit set to one
[13]	Disable_pause_time_zero_ interrupt	WO		
[12]	Disable_pause_packet_received_interrupt	WO		
[11:6]	Reserved			Each bit set to one
[5]	Disable_retry_limit_exceeded_ interrupt	WO		
[4]	Disable_transmit_buffer_ underrun_interrupt	WO		
[3:1]	Reserved			Each bit set to one
[0]	Disable_management_done_ interrupt	WO		

MAC Interrupt Mask Register 0x0,070,030

Description

The mask register determines whether the interrupt event should be passed to the interrupt controller or not. Note that the interrupt enable/disable registers are pseudo-registers and are therefore write-only. The mask register is read-only.

Table 3-198. MAC Interrupt Mask Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:14]	Reserved	RO	'0'	
[13]	Pause_time_zero_interrupt_ masked	RO	'1'	
[12]	Pause_packet_received_ interrupt_masked	RO	'1'	
[11:6]	Reserved			
[5]	Retry_limit_exceeded_ interrupt_masked	RO	'1'	
[4]	Transmit_buffer_underrun_ interrupt_masked	RO	'1'	
[3:1]	Reserved			
[0]	Management_done_interrupt_ masked	RO	'1'	

MAC PHY Maintenance Register 0x0,070,034

Description

This register enables the MAC to communicate with a PHY by means of the MDIO interface. It is used during auto negotiation to ensure that the MAC and the PHY are configured for the same speed and duplex configuration.

The PHY maintenance register is implemented as a shift register. Writing to the register starts a shift operation which is signaled as complete when bit two is set in the network status register, about 2000 clk_sys cycles later when bit ten is set to zero, and bit eleven is set to one in the network configuration register. An interrupt is generated as this bit is set. During this time, the MSB of the register is output on the MDIO pin and the LSB updated from the MDIO pin with each MDC cycle. This causes transmission of a PHY management packet on MDIO. See Section 22.2.4.5 of the IEEE 802.3 standard. Reading during the shift operation will return the current contents of the shift register.

At the end of management operation, the bits will have shifted back to their original locations. For a read operation, the data bits will be updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management packet is produced.

For a description of MDC generation, see the MAC Network Configuration Register in Table 3-193. MAC Network Status Register Map.

Table 3-199. PHY Maintenance Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:30]	Start_of_packet	R/W	'0'	Must be written 01 for a valid packet
[29:28]	Operation	R/W	'0'	10 – read 01 – write
[27:23]	PHY_address	R/W	'0'	Specifies the PHY address to access
[22:18]	Register_address	R/W	'0'	Specifies the register in the PHY to access
[17:16]	Must_be_written_to_10	R/W	'0'	Will read as written
[15:0]		R/W	'0'	For a write operation this is written with the data to be written to the PHY. After a read operation this contains the data read from the PHY

Pause Time Register 0x0,070,038

Table 3-200. Pause Time Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved	RO	0	Read 0, ignored on write
[15:0]	Pause time	RO	0	Stores the current value of the pause time register which is decremented every 512 bit times.

MAC Specific Bottom Register 0x0,070,098

Table 3-201. MAC Specific Address Bottom Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:0]	MAC Specific Address Bottom bits	R/W	0	Least significant bits of the MAC Specific Address

MAC Specific Top Register 0x0,070,09C

Table 3-202. MAC Specific Address Top Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved	RO	0	Read 0, ignored on write
[15:0]	MAC Specific Address Top bits	R/W	0	Most significant bits of the MAC Specific Address, that is bits 47 to 32

MAC Transmit Pause Quantum Register 0x0,070,0BC

Table 3-203. MAC Transmit Pause Quantum Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:16]	Reserved			
[15:0]	Pause_time	R/W	0xFFFF	Transmit pause quantum. Used in hardware generation of transmitted pause packets as value for pause quantum

PHY SMII Status Register 0x0,070,0C0

Table 3-204. SMII Status Register Map

Bits	Data Element Name	R/W	Reset Values	Description
[31:21]	Reserved	RO		
[20]	SMII_speed	RO		Speed recovered from rxd_SMII
[19]	SMII_Duplex	RO		Duplex recovered from rxd_SMII
[18]	SMII_link	RO		Link recovered from rxd_SMII
[17]	SMII_Jabber	RO		Jabber recovered from rxd_SMII
[16]	SMII_False_Carrier	RO		False carrier recovered from rxd_SMII
[15:0]	Reserved	RO		

4.0 FUNCTIONAL DESCRIPTION

4.1 INTRODUCTION

TDMoIP Packet Format

To transport TDM data through packet switched networks, the PacketTrunk-4 encapsulates it into Ethernet packets, as depicted in Figure 4-1. TDMoIP Encapsulation in an Ethernet Packet.

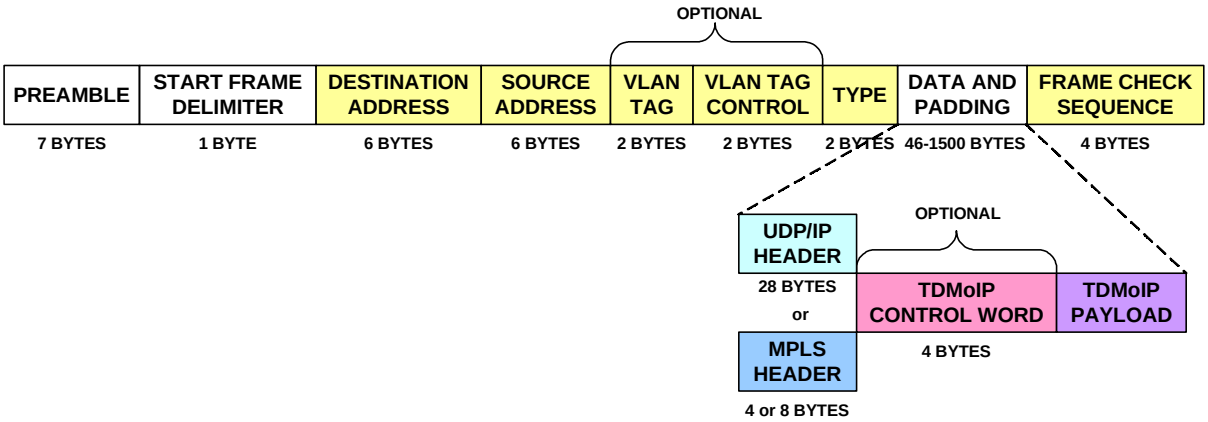


Figure 4-1. TDMoIP Encapsulation in an Ethernet Packet

Table 4-1. Ethernet Packet Structure

Field	Description
Preamble	A sequences of 56 bits (alternating 1 and 0 values) used for synchronization. Gives components in the network time to detect the presence of a signal.
Start frame delimiter	A sequence of 8 bits (10101011) that indicates the start of the packet.
Destination and source addresses	The Destination Address field identifies the station or stations that are to receive the packet. The Source Address identifies the station that originated the packet. A Destination Address may specify either an "individual address" destined for a single station, or a "multicast address" destined for a group of stations. A Destination Address of all 1 bits refers to all stations on the LAN and is called a "broadcast address".

Table 4-2. Ethernet Packet Structure (Cont.)

Field	Description
Type	Protocol type
Data and padding	This field contains the data transferred from the source station to the destination station or stations. The maximum size of this field is 1500 bytes. If the size of this field is less than 46 bytes, then padding is used to bring the packet size up to the minimum length. A minimum Ethernet packet size is 64 bytes from the Destination Address field through the Frame Check Sequence.
Frame check sequence	This field contains a 4-byte cyclical redundancy check (CRC) value used for error checking. When a source station assembles a packet, it performs a CRC calculation on all the bits in the packet from the Destination Address through the Pad fields (that is, all fields except the preamble, start frame delimiter, and frame check sequence). The source station stores the value in this field and transmits it as part of the packet. When the packet is received by the destination station, it performs an identical check. If the calculated value does not match the value in this field, the destination station assumes an error has occurred during transmission and discards the packet.

VLAN Option

The original Ethernet standards defined the minimum packet size as 64 bytes and the maximum as 1518 bytes. These numbers include all bytes from the Destination Address field through the Frame Check Sequence field. The Preamble and Start Frame Delimiter fields are not included when quoting the size of a packet. The IEEE 802.3ac standard released in 1998 extended the maximum allowable packet size to 1522 bytes to allow a VLAN tag to be inserted into the Ethernet packet format. If present, the 2-byte VLAN tag is set to 0x8100, a reserved Type field assignment indicating the presence of the VLAN tag and a 'traditional' Type field 4 bytes further into the packet.

The 2-byte VLAN Tag Control field consists of:

First 3 bits User Priority field, used to assign a priority level to the Ethernet packet
Next 1 bit Canonical Format Indicator indicating the presence of a Router Information Field
Last 12 bits VLAN Identifier, uniquely identifying the VLAN to which the Ethernet packet belongs.

The packet may contain two VLAN tags to support VLAN stacking.

UDP/IP Header

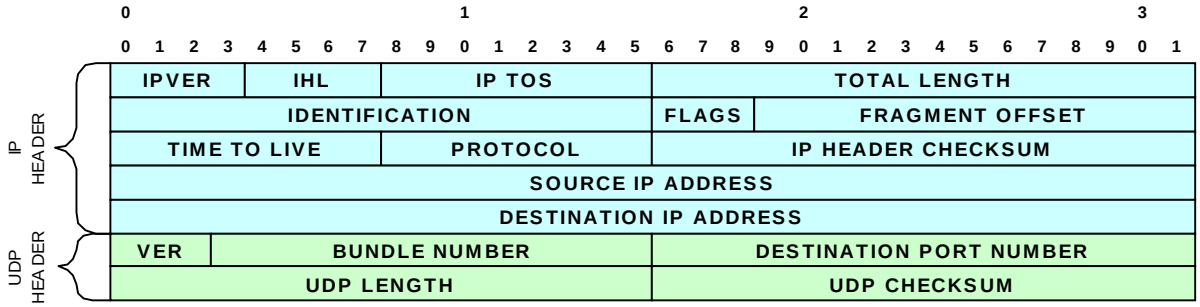


Figure 4-2. UDP/IP Header

Table 4-3. IP Header Structure

Field	Description
IPVER	IP version number, e.g. for IPv4 IPVER=4
IHL	Length in 32-bit words of the IP header, IHL=5
IP TOS	IP type of service
Total length	Length in octets of header and data
Identification	IP fragmentation identification
Flags	IP control flags; must be set to 010 to avoid fragmentation
Fragment offset	Indicates where in the datagram the fragment belongs; not used for TDMoIP
Time to live	IP time to live field; datagrams with zero in this field are to be discarded
Protocol	Must be set to 0x11 to signify UDP
IP Header checksum	Checksum for the IP header
Source IP address	IP address of the source
Destination IP address	IP address of the destination

Table 4-4. UDP Header Structure

Field	Description
VER	TDMoIP version number. Presently VER=001
Bundle number	This field is usually dedicated to the Source Port Number, but here identifies the unique data stream emanating from a given trunk and sharing a common destination. This nonstandard use of a UDP port number is similar to RTP/RTCP's use of port numbers to uniquely identify sessions, and the common practice (sanctioned in H.225) of randomly allocating port numbers for VoIP sessions. Here placing the bundle identifier in the UDP header rather than the application area enables fast switching. Possible values are: 1-8063: valid bundle numbers 0: reserved 8191 (1FFF): used for OAM control messages the 127 ports 8064-8190: reserved.
Destination port number	Set to 0x085E (2142), the user port number which has been assigned to TDMoIP by the Internet Assigned Numbers Authority (IANA)
UDP length	Length in octets of UDP header and data
UDP checksum	Checksum of UDP/IP header and data. If not computed it must be set to zero.

MPLS Header

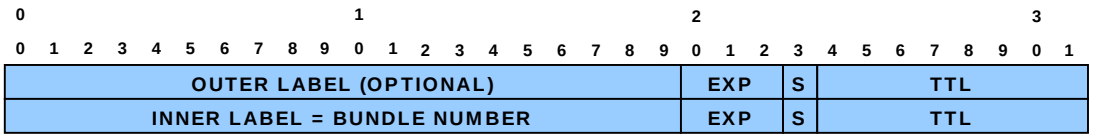


Figure 4-3. MPLS Header

Table 4-5. MPLS Header Structure

Field	Description
Outer label	MPLS label which identifies the MPLS LSP used to tunnel the TDM frames through the MPLS network. It is also known as the tunnel label or the transport label. The label number can be assigned either by manual provisioning or via the MPLS control protocol. There can be zero or one outer label row.
EXP	Experimental field
S	Stacking bit: 1 indicates stack bottom. S=0 for all outer labels
TTL	MPLS time to live
Inner label	MPLS inner label (also known as the PW label or the interworking label), contains the bundle identifier used to multiplex multiple bundles within the same tunnel. Always be at the bottom of the MPLS label stack, and hence its stacking bit is set. For OAM packets (see "TDMoIP Payload on page 185"), it must be set to 1FFF.

TDMoIP Control Word

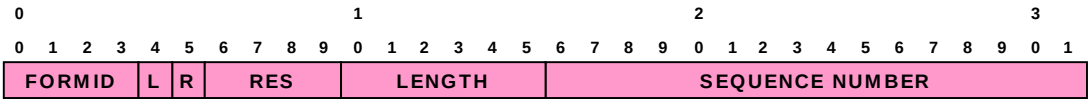


Figure 4-4. TDMoIP Control Word

Table 4-6. TDMoIP Control Word Structure

Field	Description
FORMID	Format Identifier: 1001 AAL2 1100 AAL1 unstructured 1101 AAL1 structured 1110 AAL1 structured with CAS 1111 HDLC
L	Local loss of sync failure. This bit is set by the CPU. A set L bit indicates that the source has detected or has been informed of a TDM physical layer fault impacting the data to be transmitted. This bit can be used to indicate physical layer LOS that should trigger AIS generation at the far end. Once set, if the TDM fault is rectified, the L bit must be cleared.
R	Remote receive failure. This bit is set by the CPU. A set R bit indicates that the source is not receiving packets at the Ethernet port, i.e., there is a failure of that direction of the bi-directional connection. This indication can be used to signal congestion or other network related faults. Receiving remote failure indication may trigger fall-back mechanisms for congestion avoidance. The R bit must be set after a preconfigured number of consecutive packets are not received, and must be cleared once packets are once again received.
RES	Reserved bits. Must be set to zero.
Length	The length of the TDMoIP packet (control word and payload), in case padding is employed to meet minimum transmission unit requirements of the PSN. It is used if the total packet length (including PSN, control word, and payload) is less than 64 bytes. Otherwise, it is set to zero.
Sequence number	TDMoIP sequence number, defined separately for each bundle and incremented by one for each TDMoIP packet sent for that bundle. The initial value of the sequence number is random (unpredictable) for security purposes, and the value is incremented modulo 2^{16} separately for each bundle. Used by the receiver to detect packet loss and restore packet sequence. The AAL2 Payload Type machine does not support this field. Its value is set to a constant zero. For OAM packets (see "TDMoIP Payload on page 185"), it uniquely identifies the message. Its value is unrelated to the sequence number of the TDMoIP data packets for the bundle in question. It is incremented in query messages, and replicated without change in replies.

TDMoIP Payload

This field can contain the following payload types:

- AAL1
- AAL2
- HDLC
- OAM

The AAL1, AAL2 and HDLC payload type details are provided in the "AAL1 Payload Type Machine on page 197", "AAL2 Payload Type Machine on page 200" and "HDLC Payload Type Machine on page 205" sections respectively. The OAM payload type format is described below.

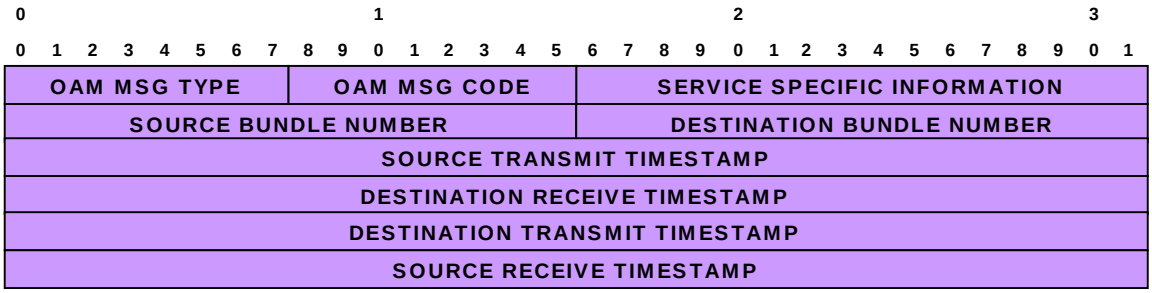


Figure 4-5. OAM Payload

Table 4-7. OAM Payload Structure

Field	Description
OAM Msg Type	Indicates the function of the message. At present, the following are defined: 0: one way connectivity query message 8: one way connectivity reply message
OAM Msg Code	Information related to the message; its interpretation depends on the message type. For type 0 (connectivity query) messages, the following codes are defined: 0: validate connection 1: do not validate connection For type 8 (connectivity reply) messages, the available codes are: 0: acknowledge valid query 1: invalid query (configuration mismatch)
Service specific information	Can be used to exchange configuration information between gateways. If not used, it must contain zero. Its interpretation depends on the FORMID (see "TDMoIP Control Word on page 183") field. At present, the following is defined for AAL1 payloads: First byte (bits 16-23): Number of timeslots being transported, e.g. 24 for full T1 Seconds byte (bits 24-31): Number of 48-octet AAL1 subframes per packet, e.g. 8 when packing 8 subframes per packet
Source bundle number	The bundle number used for TDMoIP traffic from the source to destination gateway
Destination bundle number	The bundle number used for TDMoIP traffic from the destination to source gateway

Table 4-8. OAM Payload Structure (Cont.)

Field	Description
Source Transmit Timestamp	The time the source gateway transmitted the query message in units of 100 microseconds. This field and the following ones only appear if delay is being measured.
Destination Receive Timestamp	The time the destination gateway received the query message in units of 100 microseconds
Destination Transmit Timestamp	The time the destination gateway transmitted the reply message in units of 100 microseconds
Source Receive Timestamp	The time the source gateway received the reply message in units of 100 microseconds

For more details about OAM Signaling, see "OAM Signaling on page 234".

Typical Application

In the application below, the PacketTrunk-4 is embedded in a TDMoIP Gateway to achieve TDM connectivity over a PSN. The TDMoIP packet formats for both IP and MPLS are shown in Figure 4-7. TDMoIP Packet Format in a Typical Application and Figure 4-8. TDMoMPLS Packet Format in a Typical Application, respectively.

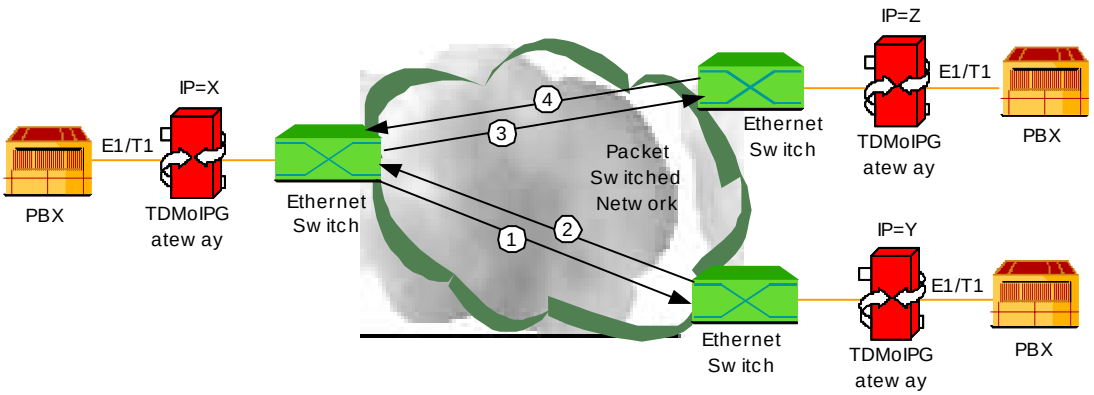


Figure 4-6. TDM Connectivity over a PSN

	DA	SA	VLAN Tag	Eth Type	IP Header	UDP Header	Control Word	Payload Type	CRC32
1			Optional	IP	Src. IP=X Dst. IP=Y	Bundle no.=A	Optional	AAL1/AAL2/ HDLC/OAM	
2			Optional	IP	Src. IP=Y Dst. IP=X	Bundle no.=B	Optional	AAL1/AAL2/ HDLC/OAM	
3			Optional	IP	Src. IP=X Dst. IP=Z	Bundle no.=C	Optional	AAL1/AAL2/ HDLC/OAM	
4			Optional	IP	Src. IP=Z Dst. IP=X	Bundle no.=D	Optional	AAL1/AAL2/ HDLC/OAM	

Figure 4-7. TDMoIP Packet Format in a Typical Application

Note: The UDP source port number is used as the bundle number designator, While UDP destination port number is set to 0x085E (2142), the user port number assigned by IANA to TDMoIP.

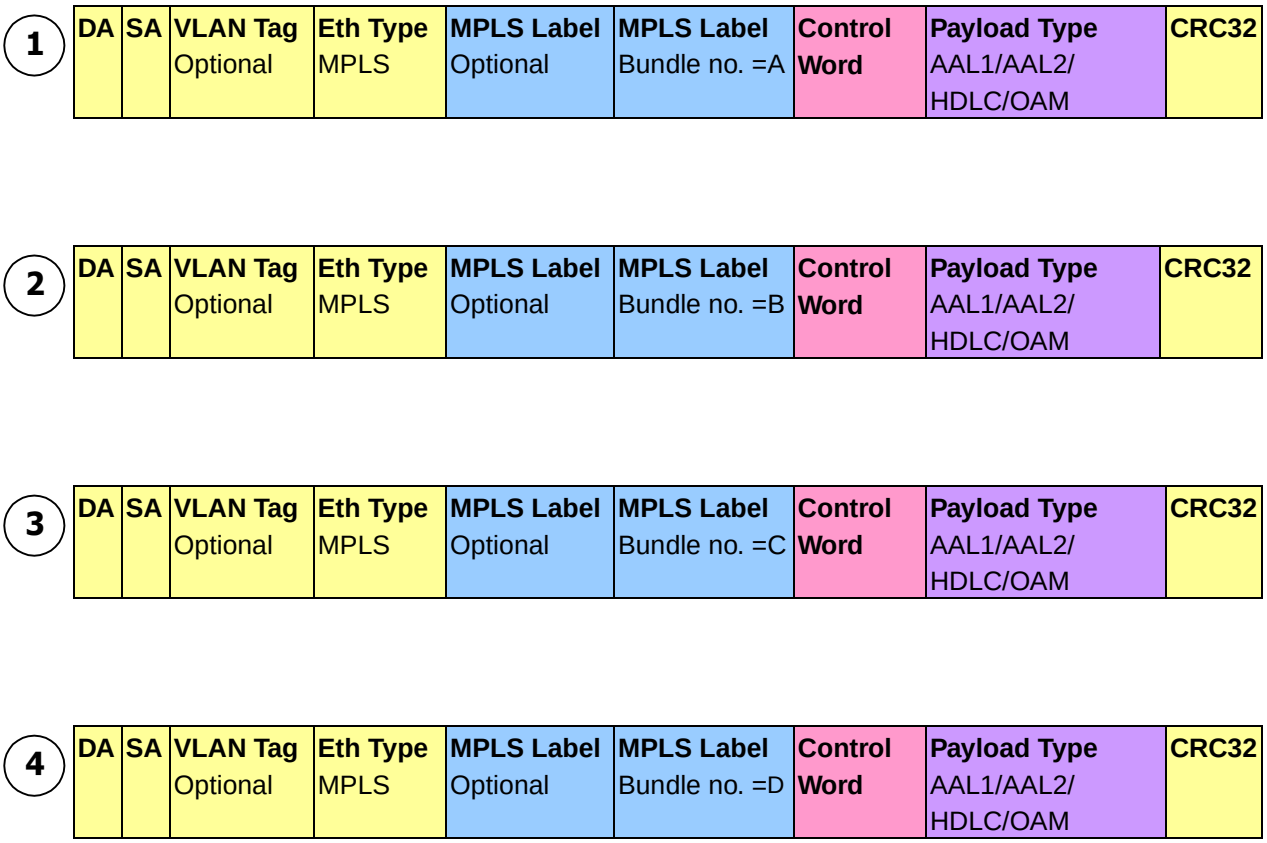


Figure 4-8. TDMoMPLS Packet Format in a Typical Application

Note: A stack of up to two MPLS labels is supported, while the inner label is used as the bundle number designator.

4.2 BLOCK DIAGRAM

In order to transform the TDM frames into IP or MPLS packets and vice versa, the PacketTrunk-4 is divided into internal blocks, each performing a defined function. In addition, it uses an external SDRAM to store the receive and transmit data buffers. A CPU interface enables an external CPU to configure and control the chip.

The PacketTrunk-4 block diagram is shown below.

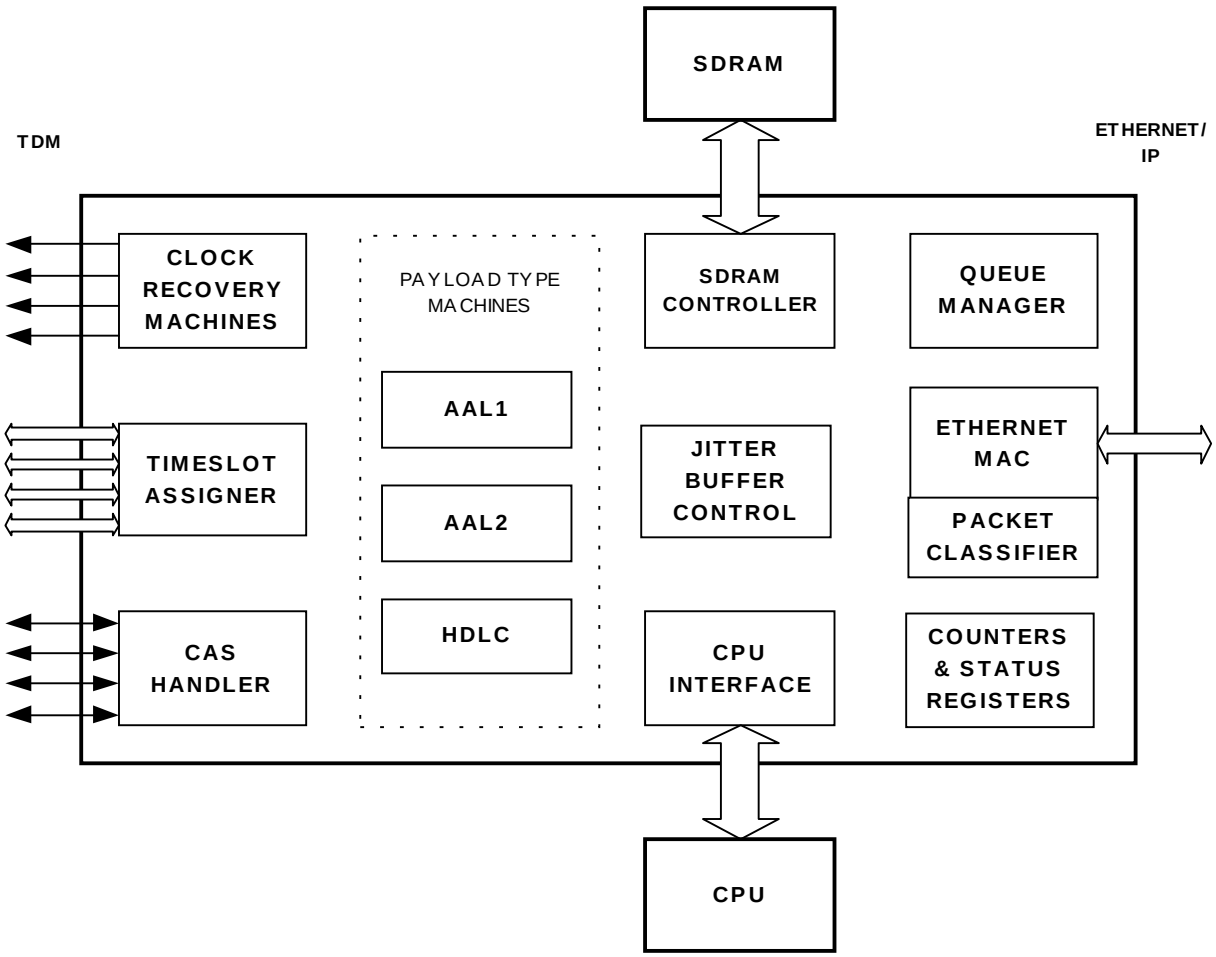


Figure 4-9. PacketTrunk-4 Block Diagram

4.3 CLOCK RECOVERY

Packet switched networks, such as IP networks, were not designed to transport TDM data and thus have no inherent clock distribution mechanism. Hence when transporting TDM over packet switched networks, the receiver needs to reconstruct the transmitter's TDM clock. The conventional means to accomplish this is by slaving a local clock (e.g., a VCXO) to the fill level of the receiver's jitter buffer. Initial frequency discrepancy is eventually compensated for, and the receiver's jitter buffer will settle on the level corresponding to precise frequency alignment between the two clocks. Although such an adaptive clock is highly robust, it has several major faults. First, the entire network jitter is transferred to the local clock, causing it to be unstable and vulnerable to packet loss. Second, the jitter buffer level may settle down far from its desired position at buffer center. Lastly, the scheme exhibits relatively long convergence time.

The PacketTrunk-4's innovative clock recovery scheme retains the robustness of the above-mentioned scheme, while offering improved capabilities. The process is divided into two successive phases. In the *acquisition* phase, rapid *frequency lock* is attained. In the *tracking* phase, the achieved frequency lock is sustained while gradually bringing the jitter buffer level back to its center (*jitter buffer centering*). During the tracking phase, jitter is effectively attenuated to comply with the relevant standards even for packet-switched networks with relatively large packet delay variation, and packet loss immunity is also significantly improved.

During the acquisition phase, a direct estimation for the frequency discrepancy between the far-end and near-end service clocks continuously drives either an internal or an external frequency synthesis device through a band-limited control loop. As a result, frequency acquisition is achieved rapidly (less than 10 seconds for a full E1/T1 bundle). The clock recovery capture range is ±128 ppm around the nominal service clock for both E1 and T1 rates.

Once the frequency-monitoring unit has detected a steady frequency lock, the system switches to its tracking phase. During the tracking phase the fill level of the incoming network packet's jitter buffer drives either the internal or the external frequency synthesizer through a similar band-limited control loop. Once in tracking phase, two tasks are performed. First, the far-end service clock frequency is slowly and accurately tracked, while compelling the regenerated near-end service clock jitter and wander levels to conform to ITU-T G.823/824, even for networks that introduce high packet delay variation and packet loss. This important characteristic can be attained due to a very efficient jitter attenuation mechanism, combined with a high resolution internal digital PLL (f=1 ppm). Second, the incoming jitter buffer is maintained at its optimal fill level, regardless of the initial frequency discrepancy between the clocks. As a result, the latency added by the mechanism is minimized, while immunity against overflow/underflow events (caused by extreme packet delay variation events) is substantially enhanced. The time duration from the moment the system switches to its tracking phase until a stable phase lock is achieved (i.e. until the jitter buffer is centered at its optimal fill level) is typically only a few dozens of seconds for a full E1/T1 bundle.

For low-speed interfaces, an on-chip digital PLL synthesizes the recovered clock frequency using CLK_HIGH. CLK_HIGH is a 38.88 MHz clock input used for the E1/T1 clock recovery engine. The frequency stability characteristics of this clock depend on the wander requirements of the recovered TDM clock. For applications where the recovered TDM clock should comply with G.823/4 wander requirements (traffic interfaces), the frequency stability within the operating temperature range should be less than 1 ppm. This can be achieved by using a TCXO component to drive this input. For applications where the wander requirements are not stringent, a standard oscillator (± 25 ppm or better) would suffice.

When none of the recovered TDM clock outputs (TDM1_ACLK-TDM4_ACLK) is being used, it is recommended to tie this input to GND in order to save power. Also, when none of the recovered TDM clock outputs are being used, the chip power consumption may be reduced by clearing the Clock_recovery_en bit.

Note: When this bit is cleared the clock recovery block (offset 48000h) must not be accessed by the CPU (as no ready signal is to be returned by the block).

For the high-speed interface (up to 44.736 MHz) the frequency synthesis is performed by an external VCXO, using the chip's external DAC interface. For more details, refer to Applications section.

4.4 TIMESLOT ASSIGNER (TSA)

The PacketTrunk-4 contains four Timeslot Assigners, one for each E1/T1 port (framed or multi-framed) using a PCM interface. (The TSA is bypassed for E1/T1, serial or high speed unframed interfaces.)

The TSA assigns 2-, 7- and 8-bit wide timeslots to a specific bundle and specific receive queue. For 2-bit wide timeslots, the next 6 bits of the interface are unassigned as a default and cannot be assigned. For 7-bit wide timeslots, the next bit is unassigned as a default and cannot be assigned. The 2- and 7-bit timeslots may be assigned only to the HDLC Payload Type machine (Both the AAL1 and AAL2 Payload Type machines support only 8-bit timeslots).

Each port has two TSA tables: one active and the other one shadow. The CPU can only write to the shadow table. After TSA entries are changed in the shadow table, the TSA tables should be swapped, so that the active table becomes the shadow table and the shadow table becomes the active table. Changes take effect at the next frame sync signal.

Each table consists of 32 entries, one entry per timeslot. The first entry refers to the first 8 bits of the frame (where the frame sync signal indicates the first bit of the frame). The second entry refers to the 8 bits coming after the first 8 bits and so on. An entry indicating assignment of less than 8 bits (i.e., 2 or 7 bits), the rest of the bits of this octet (8 bits) cannot be assigned by default. Each entry contains 19 bits, as described in the table below.

Table 4-9. TSA Entry Format

Bits	Data Element Name	Description
[6:0]	Jitter buffer index	The Jitter buffer index used for this timeslot
[12:7]	Bundle number	The bundle number that the timeslot belongs to
[13]	TX assigned	When set, the timeslot is assigned to a bundle that its number is written in Bundle Number field. Otherwise the timeslot data is discarded.
[14]	RX Assigned	When set, the data transferred to PCM for this timeslot is from the jitter buffer that its index is written in Jitter buffer index field. Otherwise, conditioning data is transferred to PCM for this timeslot.
[15]	First in bundle	Must be set for the first timeslot of an AAL1/AAL2 bundle. Must be cleared for HDLC bundles.
[17:16]	Timeslot width	00 – Reserved 01 – 2 bits + 6 unassigned bits (used for HDLC bundles only) 10 – 7 bits + 1 unassigned bit(used for HDLC bundles only) 11 – 8 bits
[18]	Structured AAL1	Must be set for timeslots that are part of AAL1 structured or structured with CAS bundles

4.5 CAS HANDLER

In the TDM to Ethernet direction, the CAS Handler receives the CAS bits (for structured with CAS AAL1 bundles) from the E1/T1 Framer through the TDMn_RSIG_RTS pins. According to the per bundle Tx_cas_source configuration bit, the CAS Handler inserts either the CAS bits of the corresponding RSIG input or the values from the Transmit SW CAS tables (configured by the CPU) into the AAL1 packets, in order to deliver the signaling as part of the AAL1 payload type packets.

The Transmit SW CAS tables may contain conditioning bits set by the CPU during configuration (per timeslot). The CPU can read the CAS bits directly from the Framer's Received CAS Bits internal registers, alter them and write them into the Transmit SW CAS tables (upon change).

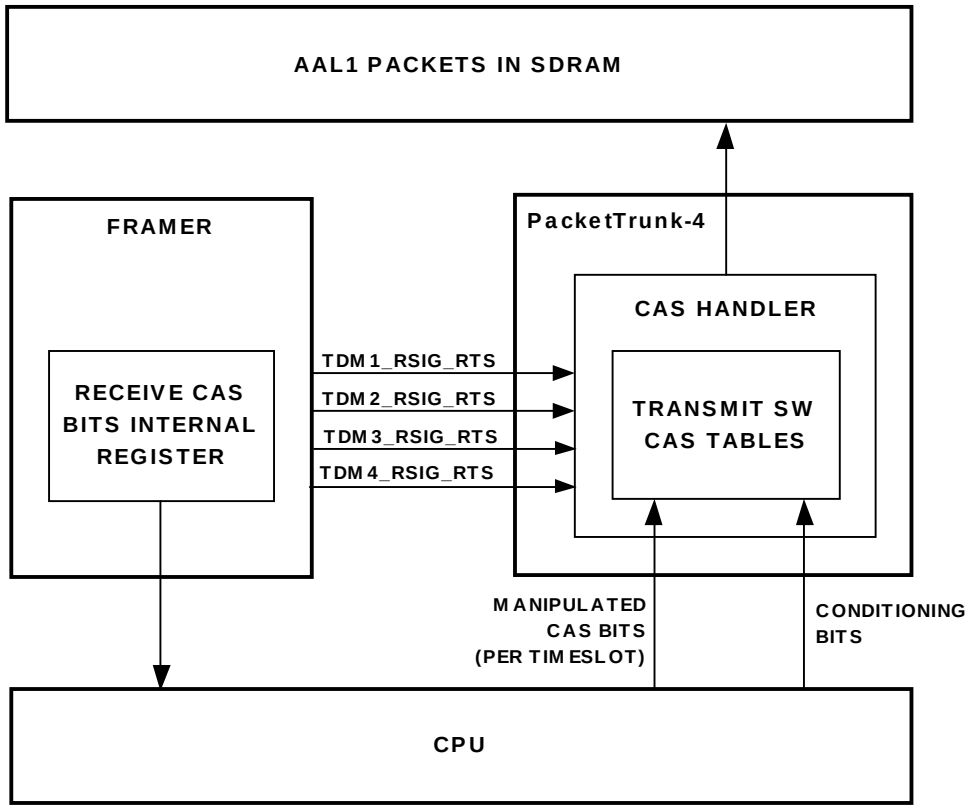


Figure 4-10. CAS Transmitted in the TDM to Ethernet Direction

There are four Transmit SW CAS tables, one per each port. Each table consists of 16 rows and each row contains the CAS bits of two timeslots (e.g., the first row contains the CAS bits of the first and second timeslots and the second row contains the CAS bits of the third and fourth timeslots).

7							0
A (TS0)	B (TS0)	C (TS0)	D (TS0)	A (TS1)	B (TS1)	C (TS1)	D (TS1)
A (TS2)	B (TS2)	C (TS2)	D (TS2)	A (TS3)	B (TS3)	C (TS3)	D (TS3)

Figure 4-11. Transmit SW CAS Table Format for E1 MF/T1 ESF Interfaces

7							0
A (TS0)	B (TS0)	B (TS0)	B (TS0)	B (TS0)	B (TS0)	B (TS0)	B (TS0)
A (TS2)	B (TS2)	B (TS2)	B (TS2)	B (TS2)	B (TS2)	B (TS2)	B (TS2)

Figure 4-12. Transmit SW CAS Table Format forT1 SF Interfaces

The supported interface connections are shown below.

Table 4-10. CAS – Supported Interface Connections

Interface Connection	Transmitted Bits
E1 MF to E1 MF T1 SF to T1 SF	CAS bits are transferred as is
T1 ESF to T1 ESF	
T1 ESF to T1 SF	Only A and B bits
T1 SF to T1 ESF	A, B bits of the SF interface and the configured SF_to_ESF_low_CAS_bits are used as C and D bits

For structured with CAS bundles connecting two T1 SF/ESF interfaces, the per-bundle Tx_dest_framing configuration bit indicates the destination interface framing type (SF or ESF).

The diagrams below show the location of the CAS bits over the RSIG input of each port.

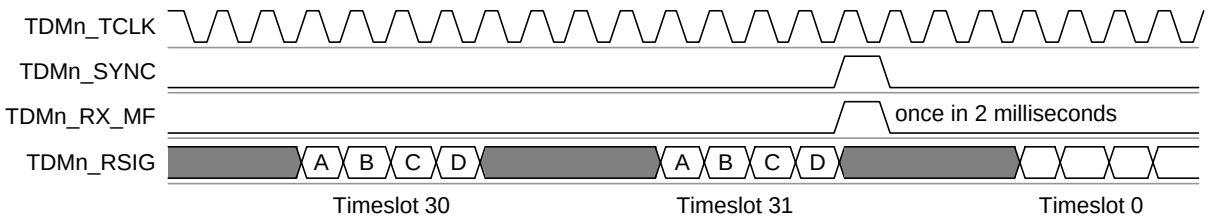


Figure 4-13. E1 MF Interface RSIG – Timing Diagram

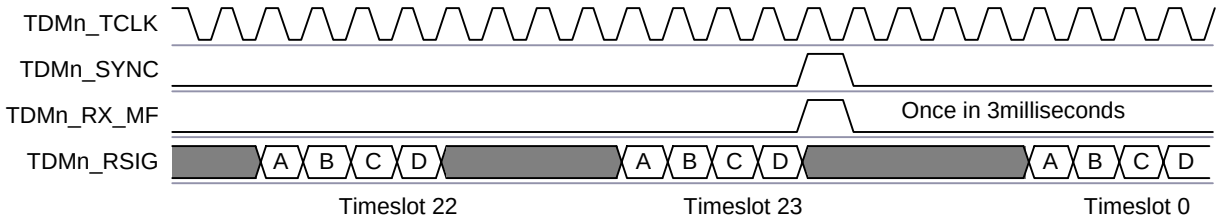


Figure 4-14. T1 ESF Interface RSIG – Timing Diagram

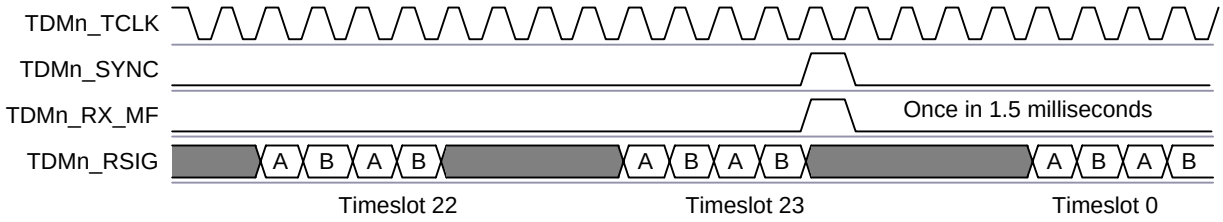


Figure 4-15. T1 SF Interface RSIG – Timing Diagram

TDMn_RX_MF can be left unconnected or connected to ground if the framer cannot drive it. The chip has internal free running counter which generates this signal internally when not driven by external source. This internally generated multiframe sync signal is synchronized to the SYNC input pulse.

In the Ethernet to TDM direction, the CAS is received from the TDMoIP packets payload.

The AAL1 Payload Type machine extracts the CAS bits from the TDMoIP payload and writes them into the CAS Jitter Buffers in the SDRAM (for structured with CAS AAL1 bundles only). The CAS Jitter Buffers store the CAS information of up to 128 timeslots of the four user ports.

The AAL2 Payload Type machine extracts the CAS bits from the valid CAS type CPS-AAL2 packets and writes them into the Receive AAL2/SW Defined CAS tables.

Selectors in the CAS Handler send the CAS bits either from the CAS Jitter buffers or from the Receive AAL2/SW Defined CAS tables to the Next MF CAS tables. The Selectors' decision is shown below.

Table 4-11. Selector Decision

Condition	Source of CAS bits Driven on TSIG for this Timeslot
Timeslot not assigned or assigned to a bundle which is not an AAL1 structured bundle (rx_assigned or AAL1_struct are cleared for its TSA entry)	Receive AAL2/SW Defined CAS tables
AAL1 bundle jitter buffer is in underrun state and rx_sig_cond_src is set	
Timeslot assigned to an AAL1 structured bundle (rx_assigned and AAL1_struct are set for its TSA entry)	Corresponding CAS Jitter buffer (CAS value is the latest received)
AAL1 bundle jitter buffer is in underrun state and rx_sig_cond_src is zero	

The Receive AAL2/SW Defined CAS tables can contain CAS bits received from a valid CAS type CPS-AAL2 packet or bits written by the CPU.

Each Receive Line CAS table is updated with the CAS bits stored in the Port_n Next MF CAS table upon TDMn_TX_MF input assertion. For E1 interface, CAS bits are updated once in 2 milliseconds, for T1 SF interface, once in 1.5 milliseconds and for T1 ESF once in 3 milliseconds.

If the data in one of the Receive Line CAS table changes upon write, the CPU receives a maskable interrupt for the port the data change occurred. The interrupt sources are recorded in four 32-bit wide registers, Rx_CAS_Change_n (n = 1 through 4). Each bit in the register represents a change in the corresponding timeslot CAS bits of the port.

There are four Receive Line CAS tables, one for each port. These tables hold the CAS information being transmitted on TSIG outputs. Each table contains 32 rows and each row holds the CAS bits of one timeslot. (Only the first 24 rows are used for T1 interfaces.) For E1 and T1 ESF interfaces, each row holds the A, B, C, and D bits. For T1 SF interface where only two CAS bits exists, each row holds the A and B bits duplicated i.e. A, B, A, B. The CPU can read the contents of these tables.

The bits in the Receive Line CAS table are sent to the Framer over the TSIG output, as shown in the figures below.

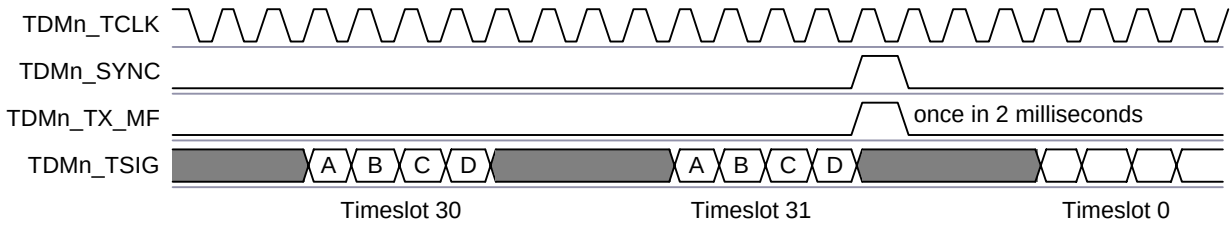


Figure 4-16. E1 MF Interface TSIG – Timing Diagram

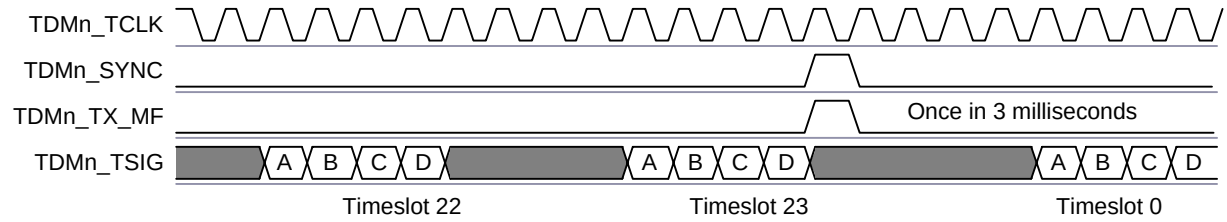


Figure 4-17. T1 ESF Interface TSIG – Timing Diagram

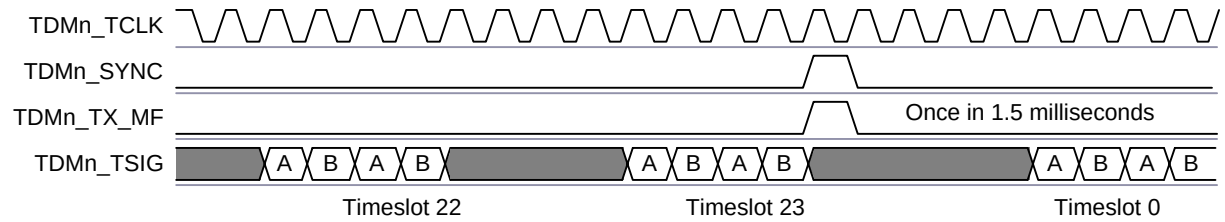


Figure 4-18. T1 SF Interface TSIG – Timing Diagram

TDMn_TX_MF can be left unconnected or connected to ground if the framer cannot drive it. The chip has internal free running counter which generates this signal internally when not driven by external source. This internally generated multiframe sync signal is synchronized to the SYNC input pulse.

The CPU can also read the CAS bits from the Next MF CAS tables, manipulate them and then write them directly into the Framer's internal register. In this case, the Framer should be configured to use the CAS information from its CAS registers and not from its TSIG inputs.

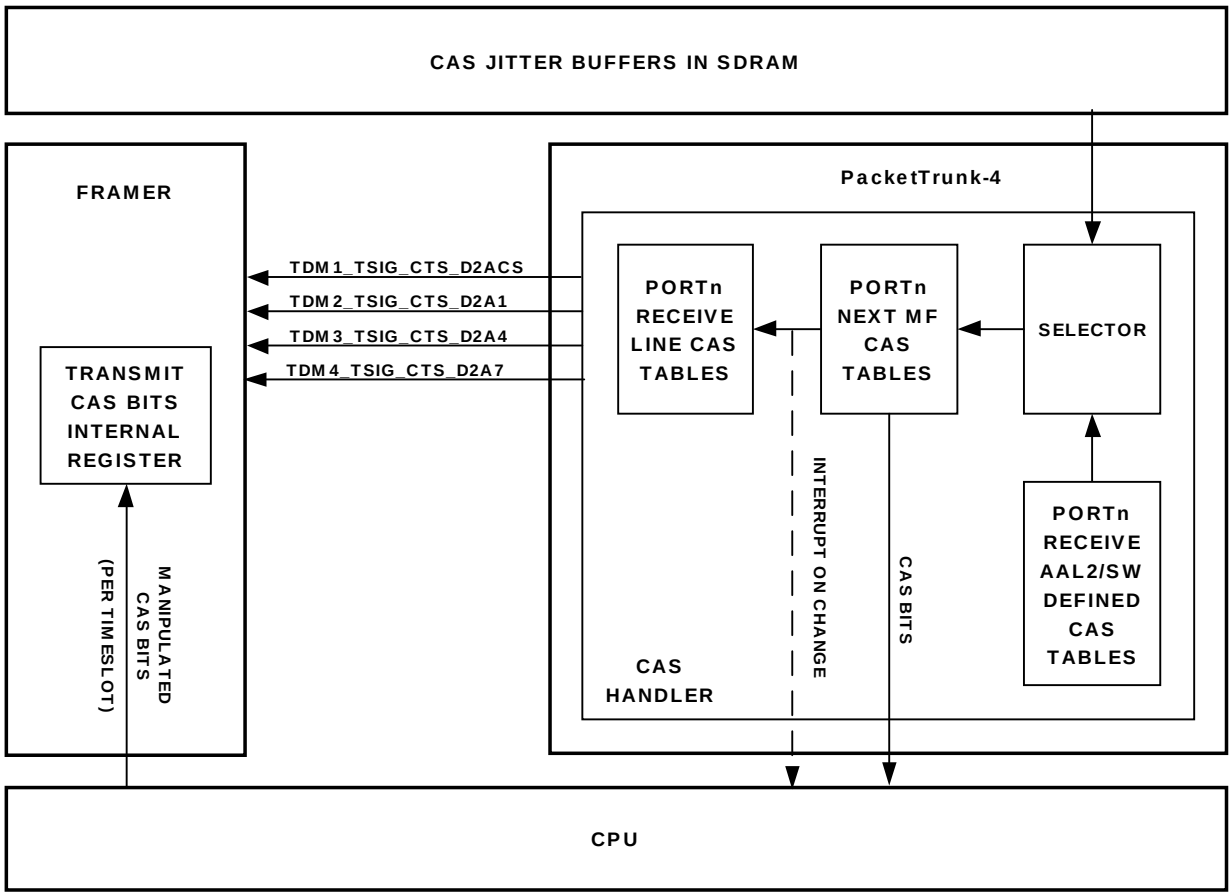


Figure 4-19. CAS Transmitted in the Ethernet to TDM Direction

4.6 AAL1 PAYLOAD TYPE MACHINE

For the prevalent case for which the timeslot allocation is static and no activity detection is performed, the payload can be efficiently encoded using constant bit rate AAL1 adaptation.

In the TDM to Ethernet direction, the AAL1 Payload Type machine concatenates the bundle's timeslots payload into structures and then slices it into 48-octet AAL1 cells. After adding the AAL1 header, the cells are concatenated into the Ethernet packet payload.

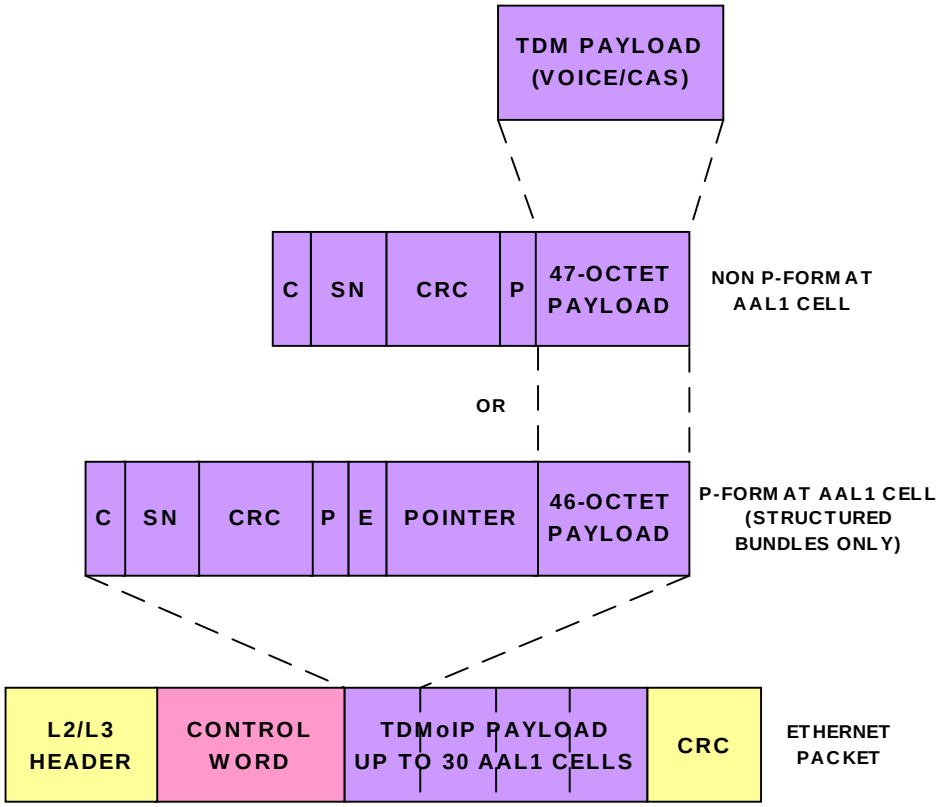


Figure 4-20. AAL1 Processing

The structure of the AAL1 header is as provided below.

Table 4-12. AAL1 Header Structure

Field	Length [bits]	Description
C	1	Indicates if there is a pointer in the 2nd octet of the cell. When set, a pointer exists.
SN	3	Cell sequence number
CRC	3	Error cyclic redundancy code on C and SN
P	1	Even parity bit on C, SN and CRC or the even byte parity LSB for the sequence number octet (P format cells only)
E	1	(P format cells only) Even byte parity MSB for pointer octet
Pointer	7	(P format cells only) Indicates the next structure boundary. It is always located at the first possible position in the sequence number cycle, in which a structure boundary occurs. The pointer indicates one of 93 octets (46 octets of the current cell + 47 octets of the next cell). P=0 indicates that the first octet of the current cell's payload is the first octet of the structure. P=93 indicates that the last octet of the next cell is the final octet of the structure.

The AAL1 block supports the following bundle types:

- Unstructured
- Structured without CAS
- Structured with CAS.

Unstructured bundles, as part of the E1/T1 interface, support rates of N × 64 kbps, where N is the number of timeslots configured to be assigned to a bundle. Unstructured bundles may also carry traffic of the whole low-speed interface (up to 4.6 Mbps), T1 interface (1.544 Mbps) and high-speed interface (up to 44.736 Mbps).

The 47-octet AAL1 type cell's payload contains 376 bits of pure TDM bit stream, without synchronization. Each cell has a one-byte header containing a sequence number and protection fields.

Structured without CAS bundles, as part of the E1/T1 interface, support rates of
N × 64 kbps, where N is the number of timeslots configured to be assigned to a bundle. For this format, the timeslots belonging to the bundle are sequentially placed in a structure, one octet per timeslot, until all 47 octets are filled. The 47th octet may contain a timeslot other than the last one in the bundle. Therefore the 1st octet of the next cell will contain the next timeslot of the same bundle.

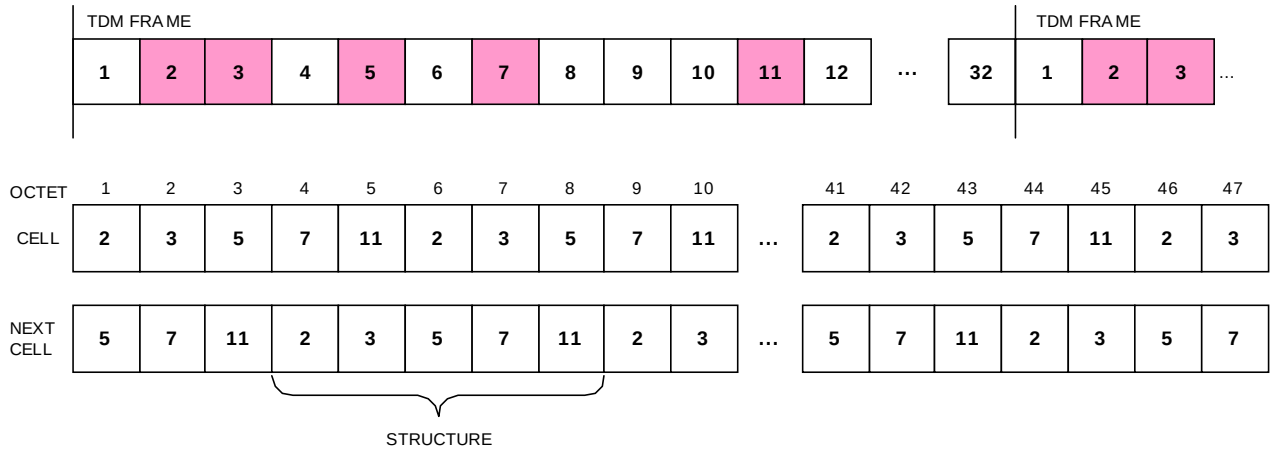


Figure 4-21. AAL1 Processing for Structured without CAS Bundles

This means that each cell can start with a different timeslot. In order for the far end to recognize the first timeslot in the bundle, a pointer to it is sent periodically in one of the 8 cells of every SN cycle. When this pointer is sent, a P-format cell is used.

Structured with CAS bundles as part of the E1/T1 interface, support rates of N × 64 kbps, where N is the number of timeslots configured to be assigned to a bundle. For this format, the AAL1 header is identical with the previously explained one. The addition of CAS only affects the structure contents. CAS data of one timeslot is 4 bits long, meaning one octet can contain CAS data of 2 timeslots. Bundles containing an odd number of timeslots need a padding of 4 zeroes in the last CAS octet. For example, a 3-timeslot bundle of an E1 frame with CAS will yield the following structure octet sequence: TS1, TS2, TS3 repeated 16 times and then CAS1+CAS2, CAS3+padding.

In the Ethernet to TDM direction, cells/packets of a bundle are being received only after the synchronization process. The synchronization process includes cell SN synchronization, packet SN synchronization and pointer synchronization. Cells with CRC or parity errors in their header are discarded. Pointer mismatch imposes jitter buffer under-run and bundle resynchronization. Cell header errors or pointer errors may be ignored depending on per-bundle configuration. Missing cells or packets are detected and restored in the jitter buffer.

4.7 AAL2 PAYLOAD TYPE MACHINE

The AAL2 Payload Type machine enables dynamic allocation of timeslots, as opposed to AAL1, which is limited to static allocation only. When timeslot allocation is dynamic or activity detection is performed, the payload can be more efficiently encoded using variable bit rate AAL2 adaptation. The variable bit rate AAL2 format is described in ITU-T Recommendation I.363.2 (11/00) B-ISDN ATM Adaptation Layer (AAL) specification: Type 2. Its use for loop emulation over ATM is explained in ATM forum specification atm-vmoa-0145 (LES) Voice and Multimedia over ATM - Loop Emulation Service Using AAL2.

AAL2 is subdivided into the Common Part Sublayer (CPS) and the Service Specific Convergence Sublayer (SSCS). The PacketTrunk-4 implements only CPS.

In TDM to Ethernet direction, AAL2 receives the TDM payload of a certain bundle, i.e. the CPS packet payload, transforms it into AAL2-CPS packets (by attaching AAL2-CPS packet header – 3 octets) and then concatenates these packets into the payload of IP or MPLS packets.

In Ethernet to TDM direction, the machine performs the opposite operation, i.e., retrieves the TDM payload from IP or MPLS packets and prepares it for sending over the TDM network.

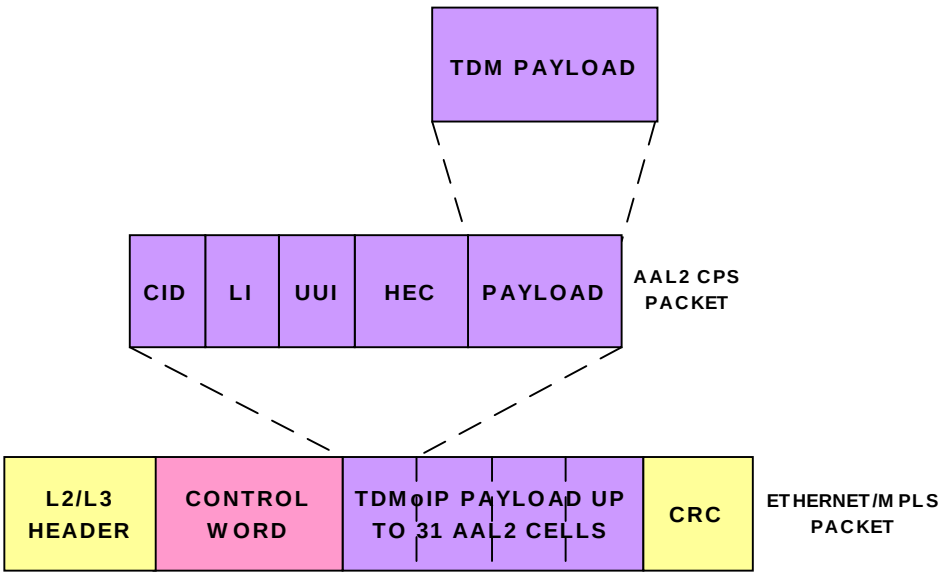


Figure 4-22. AAL2 Processing

The AAL2 block supports different AAL2 CPS packet types, identified by a combination of values of the header fields. The packet types supported are described in the "AAL2 CPS Packet Types on page 203" section.

In the TDM to Ethernet direction, the transmitted packets contain either AAL2 voice CPS packets or another type of AA2-CPS packets, but not a combination of the two. Before sending CAS CPS packets only, the SW must duplicate and place them in the Ethernet packet one after the other. For example, for transmitting CAS-CPS packets 1 and 2, the Ethernet packet is organized as follows:

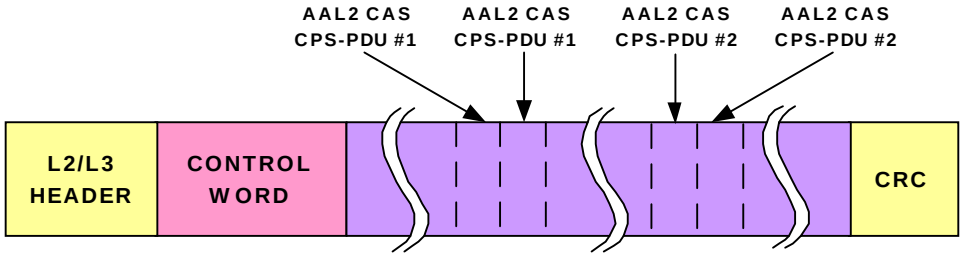


Figure 4-23. Sending AAL2 CAS CPS-PDU Packets from Ethernet to TDM

In the Ethernet to TDM direction, all possible combinations are allowed. Padding is supported as defined in ITU-T Recommendation I.363.2 (11/00) - B-ISDN ATM Adaptation Layer (AAL) specification: Type2. If an AAL2 CPS packet ends (according to its LI field) and the following bytes are set to zero, the AAL2 block ignores them until a byte other than zero appears. This byte is treated as a CID value of a new AAL2 CPS packet.

If the AAL2-CPS Packet Type is voice, and UUI is not configured with a constant value, the AAL2 Payload Type machine is programmed to receive consecutive UUI values for consecutively-received AAL2 voice CPS Packets, where each packet is associated with the same CID value. If the present AAL2 voice CPS Packet contains a UUI value that is not consecutive to the previous one, the following calculation is analyzed:

If $UUI_{received} - UUI_{expected} < 8$, then conditioning bytes, in the amount of the lost bytes, are inserted towards the TDM (by means of the Jitter Buffer).

If $UUI_{received} - UUI_{expected} \geq 8$, then the present AAL2 voice CPS Packet is dropped, and the expected UUI value is maintained for the next incoming
AAL2-CPS voice Packet.

AAL2 CPS Packet Format

The AAL2-CPS packet data elements are provided below:

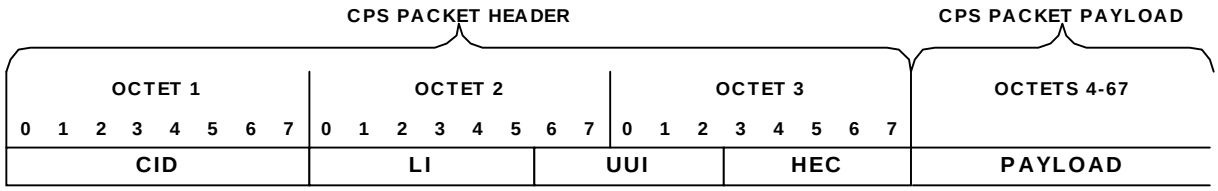


Figure 4-24. AAL2 CPS Packet Format

Table 4-13. AAL2-CPS Packet Structure

Field	Description
CID	Channel (timeslot) identifier unique for the bundle; values below 8 are reserved. Therefore there are 248 possible CID values.
LI	Length indicator, one less than the length of the payload in octets. (Note that the payload is limited to 64 octets.)
UUI	User-to-user indication, the higher layer (application) identifier and counter. Possible values are: 0-15: For non-constant UUI configuration: voice CPS packets incremented modulo each time an AAL2 voice CPS packet with the same CID is sent. For constant UUI configuration: the constant value is sent for voice CPS packets; The receiver accepts any value between 0 to 15. 24: TYPE3 packet including CAS 26 or 27: Framed mode packets, such as ELCP, CCS 31: OAM packet
HEC	Header error control
Payload (voice)	A block of length indicated by LI of voice samples are placed as-is into the AAL2-CPS packet. The AAL2 payload size is configurable, subject to the following restrictions: The maximum permitted size is 64 The minimum permitted size is 1 The configured payload size is per bundle The supported TS size is 8 bits
Payload (CAS)	Formatted as a Type-3 packet (in the notation of ITU-T Recommendation I.363.2 (11/00) - B-ISDN ATM Adaptation Layer (AAL) specification: Type2) in order to ensure error protection. The signaling is sent with the same CID as the corresponding voice channel. Signaling is sent whenever the state of the ABCD bits changes, and is sent with triple redundancy, i.e. sent three times spaced 5 milliseconds apart. In addition, the entire set of the signaling bits is sent periodically to ensure reliability

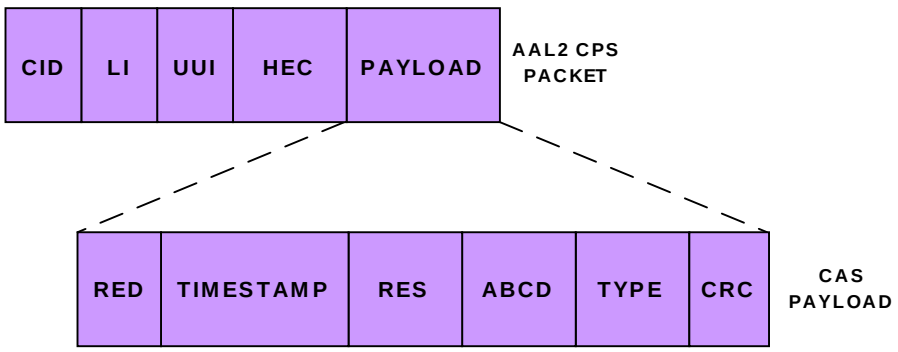


Figure 4-25. AAL2-CPS Packets – CAS Payload Format

Table 4-14. AAL2-CPS Packets – CAS Payload Structure

Field	Length [Bits]	Description
RED	2	Triple redundancy counter. For the first packet it takes the value 00, for the second 01 and for the third 10. RED=11 means non-redundant information and is used for periodic refresh of the CAS information.
Timestamp	14	The same for all three redundant transmissions
RES	4	Reserved and must be set to zero
ABCD	4	CAS bits
Type	6	For CAS, this is 000011
CRC-10	10	10-bit CRC error detection code. The AAL2 machine can be configured not to check this field by means of bundle configuration.

AAL2 CPS Packet Types

The AAL2 CPS packet types that supported by the chip are detailed below.

Table 4-15. AAL2 CPS Packet Types

Packet Type	LI Value	UUI Value	TYPE Value	Description
CAS	4	24	000011	The RED, RES and Timestamp fields are ignored by the receiver. HEC and CRC-10 are checked (can be disabled by per-bundle configuration for cases when the transmitter does not calculate it). The ABCD bits can either be written to the TDMx_TSIG_CTS_D2ACS_N pins via the Portx Received AAL2/SW defined CAS tables, or read by the CPU and rewritten to the E1/T1 framer via its registers. Also, the SW can configure the chip to drop all AAL2 CAS packets associated with a certain bundle.

Table 4-16. AAL2 CPS Packet Types (Cont.)

Packet Type	LI Value	UUI Value	TYPE Value	Description
Type-3 (except OAM and CAS)	0-3, 5-63	24	000011	After HEC is checked (can be disabled by per-bundle configuration), the packets are sent to the CPU for further processing. It can be configured (per bundle) to drop these packets.
Framed mode signaling, e.g., CCS and ELCP	Any	26 or 27		After HEC is checked (can be disabled by per-bundle configuration), the packets are sent to the CPU for further processing. It can be configured (per bundle) to ignore these packets.
OAM	Any	31		After HEC is checked (can be disabled by per-bundle configuration), the packets are sent to the CPU for further processing. It can be configured (per bundle) to ignore OAM packets.
Voice	As configured per bundle	0-15		HEC is checked (can be disabled by per-bundle configuration). The payloads are assigned to the corresponding timeslots in the corresponding T1/E1 interfaces. If LI differs from the configured value, the packet is dropped.
SID	0	0-15		Expected only when the cell payload size is not 1 (otherwise they can be confused with voice packets due to their UUI value). After HEC is checked (can be disabled by per-bundle configuration), the packets are sent to the CPU for further procesing. It can be configured (per bundle) to ignore SID packets.
Unknown	Any	0-15, 24, 26, 27, 31		After HEC is checked (can be disabled by per-bundle configuration), the packates are sent to the CPU for further processing. It can be configured (per bundle) to drop these packets.

4.8 HDLC PAYLOAD TYPE MACHINE

Handling HDLC in TDMoIP ensures efficient transport of CCS (common channel signaling, such as SS7), embedded in the TDM stream or other HDLC-based traffic, such as Frame Relay.

For the E1 interface, each bundle supports the rates of 16 kbps or N × 64 kbps, where N is the number of timeslots configured to be assigned to a bundle (between 1 to 32). For the T1 interface, each bundle supports the rates of 16 kbps, 56 kbps (not supported for T1 SF interface), full T1 (1.544 Mbps) or N × 64 kbps, where N varies from 1 to 24.

In the TDM to Ethernet direction, the HDLC block monitors flags until a frame is detected. It collects the contents of the frame and checks the correctness of the CRC, alignment and frame length.

Note: 2 bytes < Correct frame length < Tx_max_frame_size

Erroneous frames are discarded. Others are sent as-is in the payload (without the CRC, flags or transparency zero-insertions).

In Ethernet to TDM direction when a packet is received, its CRC is calculated, and the original HDLC frame reconstituted (flags are added, bit stuffing is performed and CRC added).

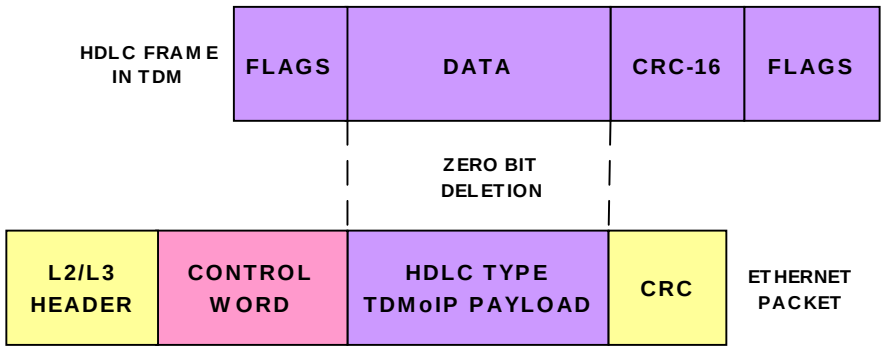


Figure 4-26. HDLC Processing

4.9 SDRAM CONTROLLER

The PacketTrunk-4 requires an external SDRAM for its operation. The SDRAM stores Ethernet outgoing packets (Tx buffers) and TDM incoming payload data after processed by the Payload Type machines (Jitter buffers). Another usage of the SDRAM is to store buffers sent to or by the CPU.

The built-in SDRAM Controller allows glueless connection to an external SDRAM (the chip supplies the SDRAM clock). Supported SDRAM devices are listed in the Connecting SDRAM Devices section.

The chip HW uses about either 1.5 or 3 MB of SDRAM space, depending on configuration. The CPU may use the rest.

The supported resolutions of the CPU access to the SDRAM are shown below.

Table 4-17. SDRAM Access Resolution

Data Bus Width	Access to SDRAM
32 bits	8, 16, 32 bit
16 bits	8, 16 bit

Prior to operation, the SDRAM Controller configuration bits (see the General_cfg_reg0 register detailed in Memory Map section) must be configured. First, the CPU must set the configuration bits while maintaining the SDRAM Controller Reset bit low (‘0’). Then, it should release the Reset bit (‘1’). The Reset bit must not be changed during operation.

The SDRAM Controller operates at either 50 or 75 MHz with the following CAS latency options:

Table 4-18. Frequency to CAS Latency Correlation

Frequency [MHz]	CAS Latency [clock cycles]
50	2
75	2 or 3

During operation, the controller’s arbiter receives access requests from various internal HW blocks and the CPU, and grants access permissions based on predefined priorities. The SDRAM Controller automatically refreshes the external SDRAM approximately once every 16 msec.

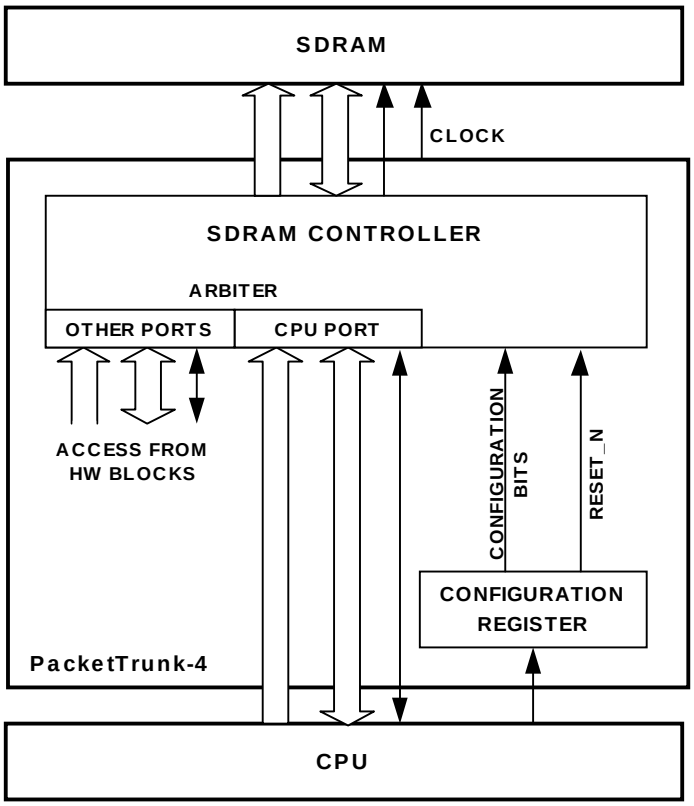


Figure 4-27. SDRAM Access via the SDRAM Controller

4.10 JITTER BUFFER CONTROL (JBC)

TDM networks have a single clock source, used by all devices in the network. Destination TDM devices derive the clock from incoming data and use it for transmitting data (loopback timing), as depicted below.

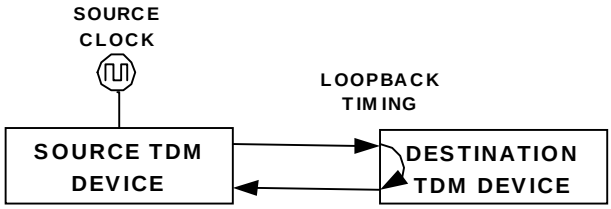


Figure 4-28. Loopback Timing in TDM Networks

When replacing the physical TDM connection with an IP/MPLS network and two TDMoIP devices (as depicted below), the receiving TDMoIP device (slave) receives TDMoIP packets with variable delays in arrival time. After processing, the device should send TDM data to the TDM side at the constant rate of the TDM network. To achieve this, the device works in Clock Recovery mode to reconstruct the source TDM clock, so the destination TDM device can still work in loopback timing mode.

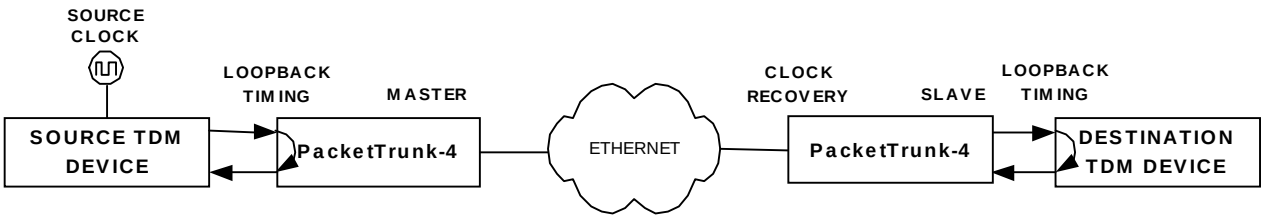


Figure 4-29. Timing in TDMoIP

The Jitter Buffer, located in the SDRAM, has two main roles:

- Compensates for delay variation
- Helps reconstructing the far end TDM clock on a TDMoIP slave device.

The data enters the buffer at a variable rate derived from the received Ethernet packet arrival time and leaves it at a constant TDM rate. In Clock Recovery mode, the level of the Jitter Buffer provides an indication for the clock recovery mechanism.

Two separate sections are allocated in the external SDRAM, one for data and another for signaling. Both data and signaling Jitter Buffers are divided into four identical sections, one for each E1/T1 interface. Each section contains the data of 32 timeslots for E1 or 24 timeslots for T1. A single E1/T1 timeslot (in structured E1/T1 mode) is allocated a maximum of 4 kB of space. The Jitter Buffer can contain the data of up to 128 timeslots. In serial or E1/T1 unstructured mode, the per-timeslot allocation is meaningless. For unstructured mode, the Jitter Buffer is divided into four 128-kB areas, one for each interface. For high-speed mode (E3/T3), the Jitter Buffer acts as one large buffer, without division into sections.

The signaling Jitter Buffer is divided into multi-frame sections, with each section containing the signaling nibbles of up to 32 timeslots.

The Jitter Buffer maximum depth in time units is calculated according to the following formula:

$$\frac{1}{2} \times \textit{Buffer area per interface} \times \frac{8}{Rate}$$

where:

½	Two halves of the buffer
Buffer area per interface	512 kB for a single high-speed interface or 128 kB for a low-speed interface
8	Number of bits per byte
Rate	Transmission rate (e.g., 2.048 Mbps)

Note: For T1 framed with CAS, multiply the above formula by 0.75.

The Jitter buffer depth is defined by the Rx_max_buff_size parameter found in the 'bundle configuration tables'. When the jitter buffer level reaches the value of Rx_max_buff_size, an overrun situation is declared.

The Rx_pdvt parameter (also found in the 'bundle configuration tables') defines the amount of data to be stored in the jitter buffer to compensate for network delay variation. This parameter has two implications:

- Rx_pdvt defines the chip immunity to the Ethernet network delay variation.
- The data arriving from the network is delayed by Rx_pdvt before it is sent to the TDM line.

Rx_pdvt should be smaller than Rx_max_buff_size. Also, the difference between Rx_max_buff_size and Rx_pdvt must be larger than the time that it takes to reconstruct a packet (otherwise an overrun may occur when the packet arrives). Typically, the recommended value for the Rx_max_buff_size is 2* Rx_pdvt + PCT (packet creation time). This provides equal immunity for both delayed and bursty packets. Configuring the Jitter Buffer parameters correctly avoids under-run and overrun situations. Under-run occurs when the Jitter Buffer is empty (the entering rate is lower than the exiting one). In case of an under-run event, the chip transmits conditioning data instead of actual data towards the TDM interface. Overrun occurs when the jitter buffer is full and there is no room for new data to enter (the entering rate exceeds the exiting one). Under-run and overrun require special treatment from the chip HW, depending on the bundle type.

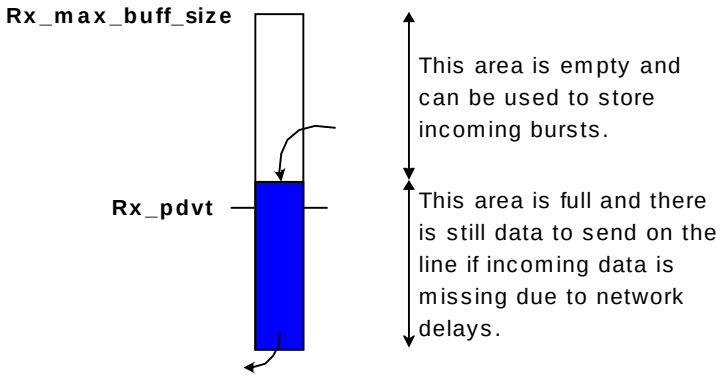


Figure 4-30. Jitter Buffer Parameters

The JBC statistics are stored in a 128-entry table. Each TDM port has 32 dedicated entries – one per time slot. The Jitter Buffer Status table stores the statistics of an active jitter buffer for each active bundle. A configurable parameter called “Jitter_buffer_index” (located in the TSA tables) defines the entry in the table the jitter buffer statistics are stored (and read from). The user should define the value of Jitter_buffer_index as follows:

- For AAL1/HDLC bundles: the Jitter_buffer_index is the number of the lowest time slot in the bundle (for example, if the bundle consists of time slots 2, 4, 17 the Jitter_buffer_index is 2).
- For AAL2 bundles: each time slot data is stored in its own jitter buffer and so the Jitter_buffer_index is identical with the time slot number (for example, if the bundle consists of time slots 2, 4, 17 there are three jitter buffers – one for each timeslot- and the Jitter_buffer_index values are 2, 4 and 17, respectively).

The software accesses the Jitter Buffer Status table according to the Jitter_buffer_index. The status table contains the current jitter buffer status (such as, the current jitter buffer level and its current state – e.g. ‘ok’, ‘underrun’ or ‘overrun’). It also contains 2 variables, which report the jitter buffer Min and Max levels (available for AAL1/AAL2 bundles only).

These variables provide information about network characteristics for the user. For example, using these values, the user can calculate the margins from the top (Rx_max_buff_size) and the bottom of the jitter buffer. If there is spare capacity, the user may want to reduce Rx_pdvt and so reduce the latency added by the jitter buffer to the incoming data.

The payload type machine detects that a packet was lost, either by sequence number error in AAL1, or by UUI error if AAL2. If packets are lost, conditioning data is inserted into the jitter buffer in order to compensate for the lost data and to maintain the bit integrity (i.e. the number of bits that are inserted into the jitter buffer must equal the number of bits that were transmitted by the far end).

4.11 CPU INTERFACE

The CPU Interface enables an external CPU to configure and control the chip, as well as to collect statistics from the chip. The CPU Interface identifies when the CPU initiates access (read or write) into the chip, forwards the access request within the chip or to the SDRAM and replies to the CPU with the requested data, if required.

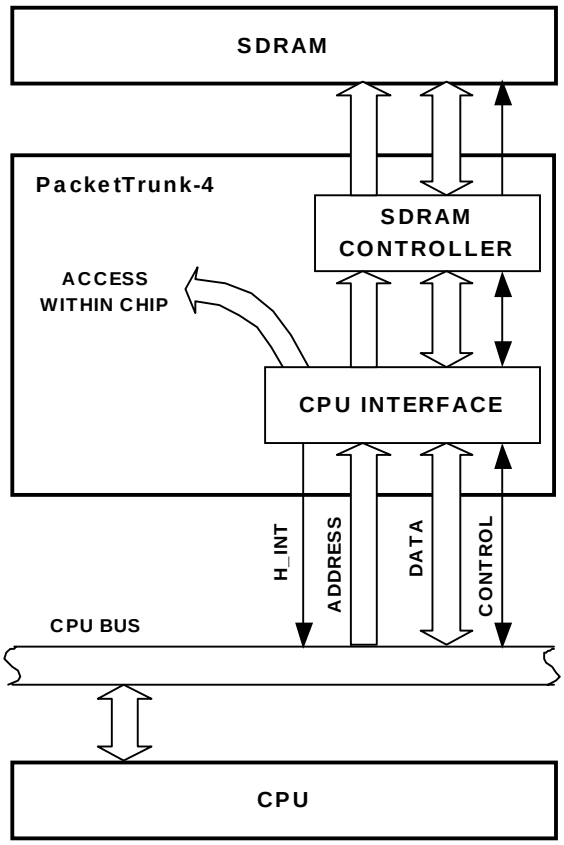


Figure 4-31. CPU Access via the CPU Interface

The operation of the CPU Interface depends on the following two external configuration pins:

- H_CPU_SYNC - determines the clock mode
- DAT_32_16_N – determines the data bus width.

The CPU and the chip can work synchronously, using the same clock, (for CPU such as Motorola MPC 860) or asynchronously (different clocks). In asynchronous mode (H_CPU_SYNC = 0), the CPU access takes longer since the chip needs more time to identify the access.

The chip can work with 16 or 32-bit CPU data bus width. The data bus width is conveyed to the chip by means of the DAT_32_16_N pin as follows:

Table 4-19. CPU Data Bus Width

DAT_32_16_N Value	Data Bus Width	Access to Chip Internal Resources	Access to SDRAM	Data Bus Bits	MSB	H_WR_BE Used
1	32 bits	32 bit only	8, 16, 32 bit	H_D[31-0]	HD[31]	3-0
0	16 bits	16 bit only	8, 16 bit	H_D[15-0]	HD[15]	1-0

Burst access is not supported.

The PacketTrunk-4 uses the Big-Endian byte order, as explained in the General Map and Addressing section.

The CPU starts accessing the chip by asserting the H_CS_N signal (active low), accompanied by H_R_W_N, address, H_WR_BEx_N and valid data (in case of write access). In response, the PacketTrunk-4 asserts H_RDY_N to indicate that the access has been carried out. The ready assertion implies that data from CPU has been written into the PacketTrunk-4/external SDRAM (for write access) or that valid data exists on data bus (for read access). In response to H_RDY_N assertion, the CPU de-asserts H_CS_N. This causes the PacketTrunk-4 to de-assert H_RDY_N, and by that finish the CPU access.

In order to make CPU operation more efficient, a write access receives an immediate “ready” towards CPU. Successive accesses (write or read) will receive “ready” only after the previous write has been completed.

Four signals, H_WR_BE0_N to H_WR_BE3_N are used as write byte enable and so define the low address space (replace the functionality of A[1:0] in the 32-bit data bus, A[0] in the 16-bit data bus) for write access.

This enables byte resolution access to the external SDRAM.

When performing a write access to internal chip resources, all H_WR_BEx_N inputs should be asserted since accessing internal chip resources can be performed in data bus width only.

Examples of CPU access on 32- and 16-bit buses are shown in the figures below.

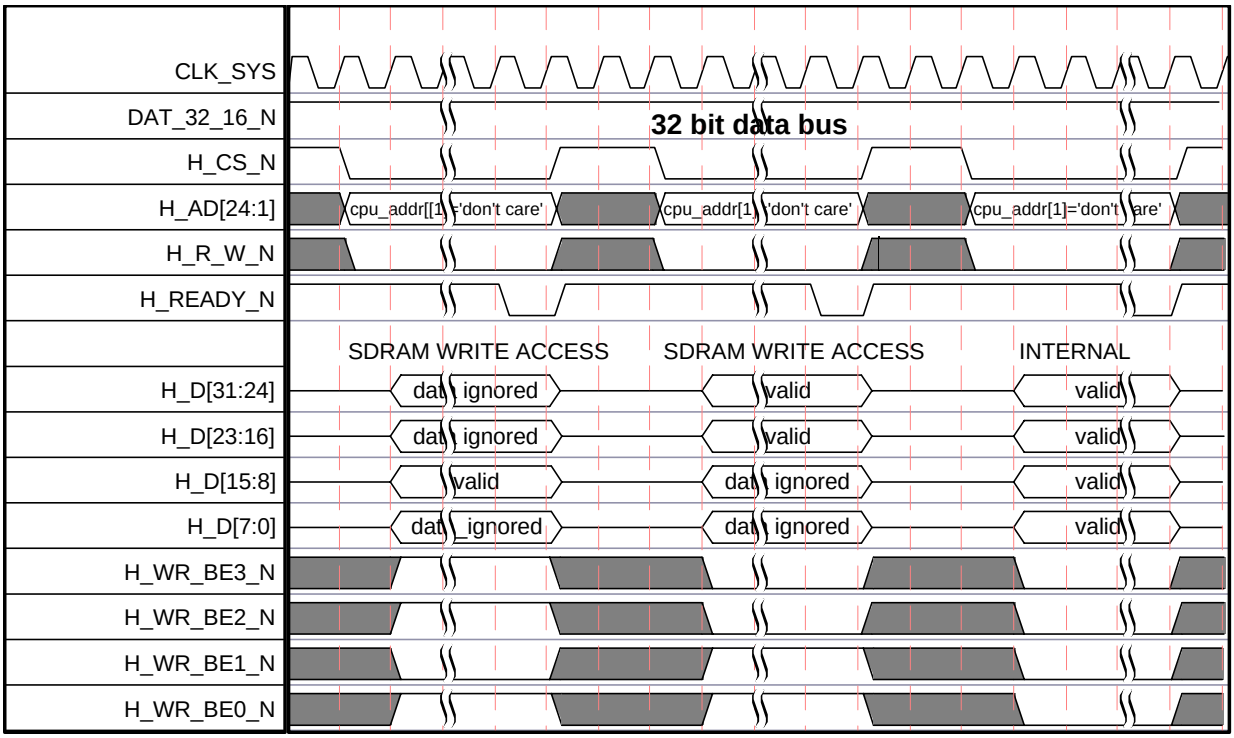


Figure 4-32. 32-bit Bus CPU Write Access

Figure 4-32. 32-bit Bus CPU Write Access section shows two write accesses to the SDRAM, one to a byte (at address 2) and the other to a word (at addresses 0 and 1), followed by an access to the internal chip resources.

The write access to the SDRAM is different than the write access to the chip. The SDRAM can be accessed at byte resolution achieved by four byte write enables. The SDRAM can be accessed by any combination of the above 4 bytes.

Internal chip resources are always accessed at CPU data bus width (32 bits in Figure 4-32. 32-bit Bus CPU Write Access section). For 32-bit CPU bus width cpu_addr[1] is irrelevant, since there is no 16-bit resolution. Byte enable signals should always be asserted when accessing internal chip resources for write.

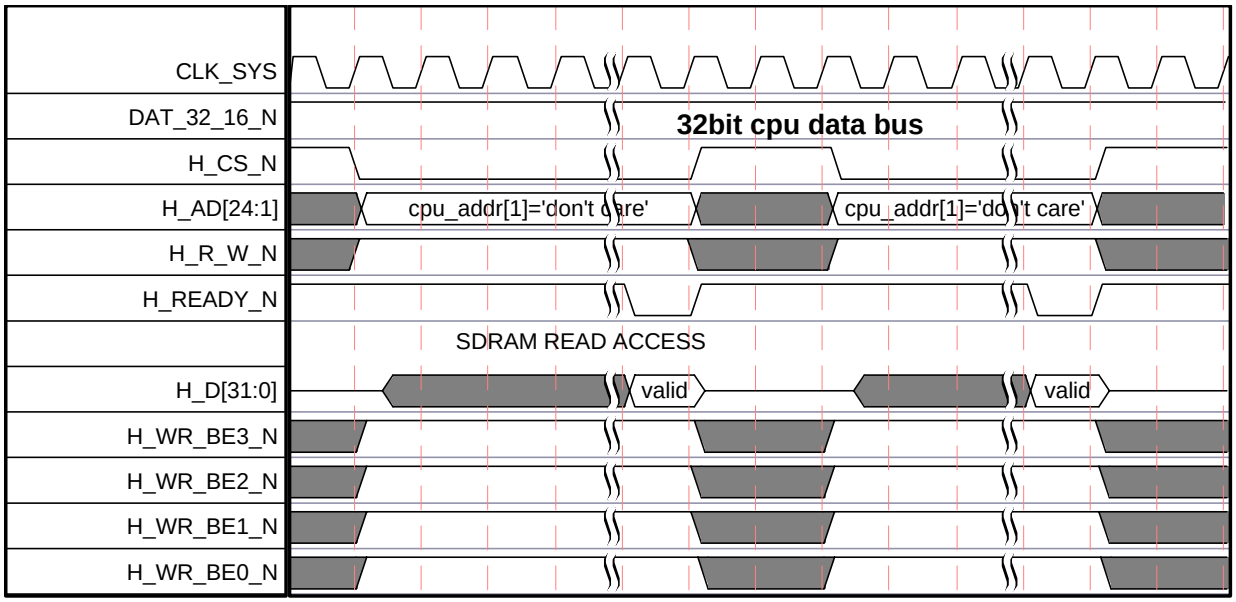


Figure 4-33. 32-bit Bus CPU Read Access

Figure 4-33. 32-bit Bus CPU Read Access section shows a read access to the SDRAM, followed by a read access to the internal chip resources. In read access, always 32 bits are read.

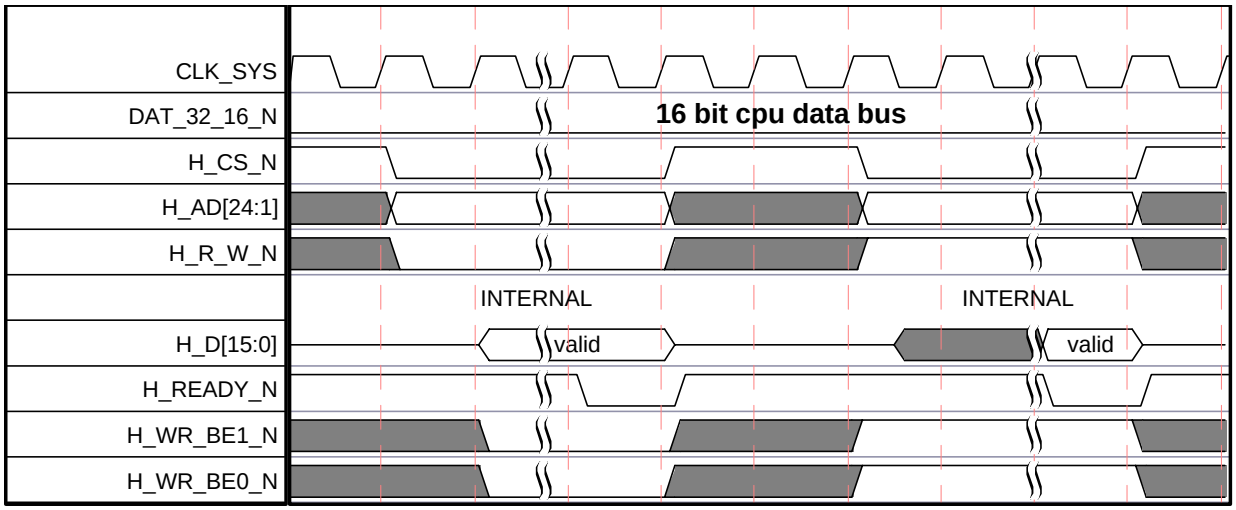


Figure 4-34. 16-bit Bus CPU Access to the PacketTrunk-4

Figure 4-34. 16-bit Bus CPU Access to the PacketTrunk-4 section shows a write access to the chip, followed by a read access. The H_AD[1] signal is relevant for 16-bit CPU data bus. Write access to the SDRAM can still be at byte resolution, as depicted in Figure 4-35. 16-bit Bus CPU Write Access to the SDRAM section.

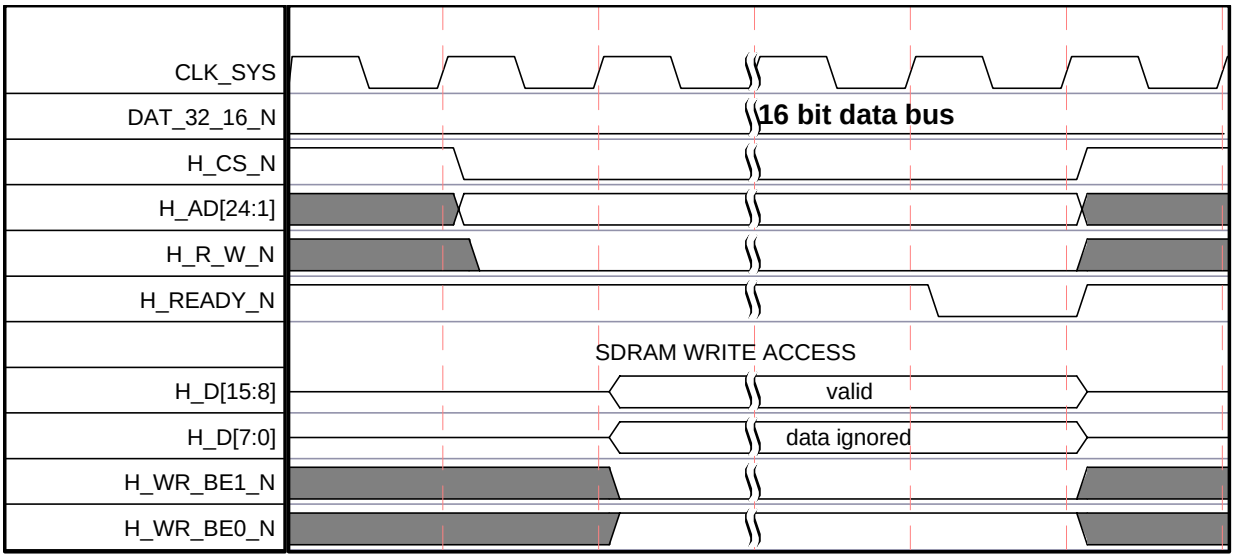


Figure 4-35. 16-bit Bus CPU Write Access to the SDRAM

For 16-bit data bus width, read access to SDRAM is always 16 bits, as in Figure 4-36. 16-bit Bus CPU Read Access to the SDRAM section.

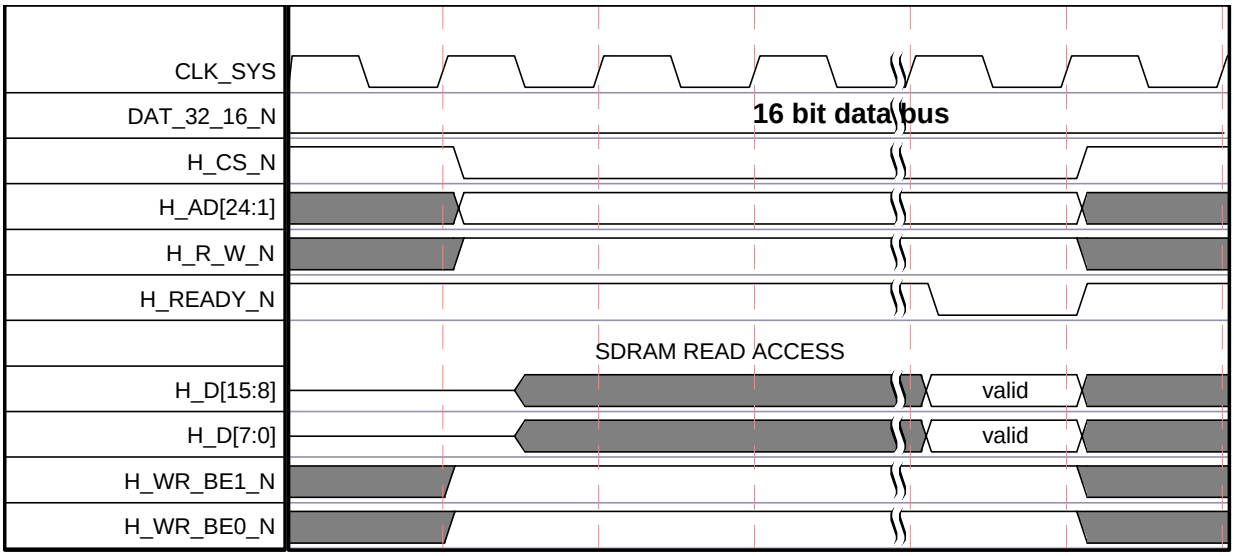


Figure 4-36. 16-bit Bus CPU Read Access to the SDRAM

The CPU Interface also generates a single interrupt. The Intpend register (detailed in Memory Map section) indicates the source of the interrupt. If one of the Intpend bits is set, it can be cleared only by writing '1' to this bit. At reset, all interrupts are disabled due to the Intmask register default values. Writing '0' to the Intmask relevant bit enables the corresponding interrupt.

The interrupts are divided into two groups. The first group consists of all interrupts generated by a single source. The second group consists all interrupts generated by several interrupt sources. The second group contains the Eth_MAC, the Rx_Cas_Change_Pn (n= 1 through 4) and the JB_underrun_Pn (n=1 through 4) interrupts.

The JB_underrun_n (n=1 through 4) registers can be masked per bit (representing a timeslot) by writing to the JB_underrun_mask_Pn (n=1 through 4) registers.

The CPU needs to read the Intpend register to identify the interrupt source. Then it should proceed as follows:

Interrupt Type	Interrupt Procedure
Group 1	1. Clear the pending bit(s) by writing '1' to the corresponding Intpend bit(s). 2. Take care of the interrupt cause.
Rx_CAS_Change_Pn	3. Clear the Rx_CAS_Change _Pn bit by writing '1' to this bit in the Intpend register. 4. Read the Rx_CAS_change_n register to find out which timeslots have been changed. 5. Clear the set bits in Rx_CAS_change_n register by writing '1' to them.
JB_underrun_Pn	6. Clear the JB_inderrun_Pn bit by writing '1' to this bit in the Intpend register. 7. Read the JB_underrun_n register to find out which timeslots have been changed. 8. Clear the set bits in JB_underrun_n register by writing '1' to them.
Eth_MAC	9. Read the MAC Interrupt Status Register (all bits are reset to '0' on read). 10. Clear the Eth_MAC bit by writing '1' to this bit in the Intpend register.

Note: If a bit in INTPEND is set but the corresponding interrupt source is masked, the CPU receives an interrupt immediately after disabling this mask. To change this behavior, the CPU should 'clear' the interrupt from the INTPEND register before disabling this interrupt mask.

4.12 QUEUE MANAGER

The data flows through the PacketTrunk-4 in the following directions:

- From TDM to Ethernet (implemented in HW, see "TDM to Ethernet Flow on page 222")
- From Ethernet to TDM (implemented in HW, see "Ethernet to TDM Flow on page 222")
- From TDM to TDM (cross-connect, implemented in HW, see "TDM to TDM Flow on page 223")
- From TDM to CPU (see "TDM to CPU Flow on page 224")
- From CPU to TDM (see "CPU to TDM Flow on page 225")
- From CPU to Ethernet (see "CPU to Ethernet Flow on page 226")
- From Ethernet to CPU (see "Ethernet to CPU Flow on page 227").

The PacketTrunk-4 data flow is shown in Figure 4-37. PacketTrunk-4 Data Flow. Each flow of data is described in subsequent sections.

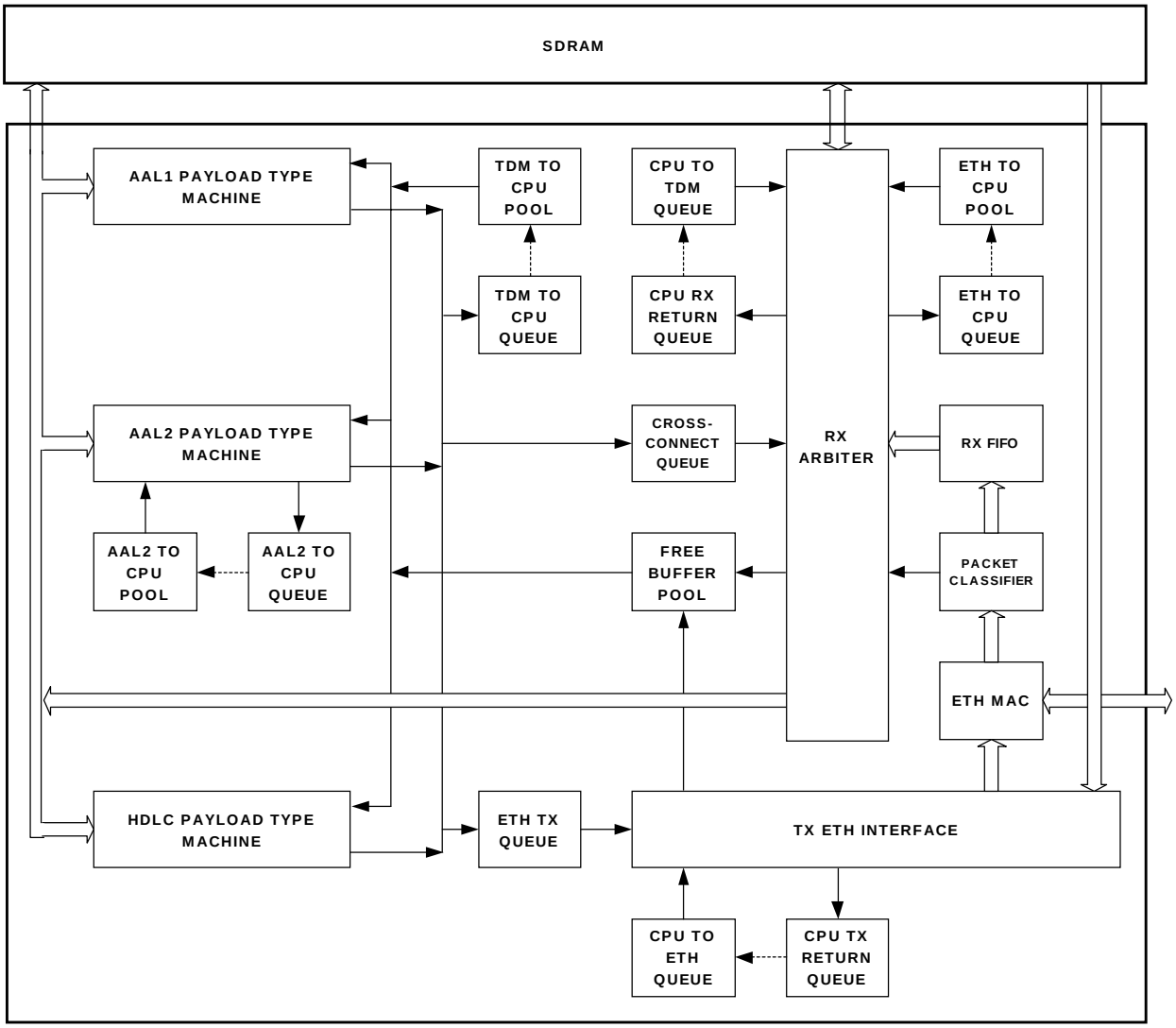


Figure 4-37. PacketTrunk-4 Data Flow

Buffer Descriptor

Data is transferred between the ETH MAC, internal chip Payload Type machines and external CPU by means of external SDRAM buffers. Payload data is stored in 2-kbyte SDRAM buffers, together with a buffer descriptor, located in the buffer's first 4 bytes (see Table 4-20. Buffer Descriptor Data Elements section). The buffer pointers are managed inside the chip and are stored in queues, pools, or other chip internal blocks. Queues store pointers to SDRAM buffers containing packet data to be processed, while pools store pointers to empty buffers. The pointers are passed from one block to another. Only the block owning the pointer can access the pointed buffer. As these pointers are the only means by which data in the SDRAM can be accessed, the pointers are owned by one internal block at a time and must never be lost.

Table 4-20. Buffer Descriptor Data Elements

Bits	Data Element	Description
[31]	VLAN tag	Indicates whether the packet header includes a VLAN tag; Used to locate the Originate Transmit Timestamp and the Echoer Transmit Timestamp for CPU→ ETH control packets.
[30]	RST	RX Reset command (the bundle is in reset process). For ETH→TDM: used by the Packet Classifier or by the CPU to inform the next blocks in flow that the bundle was reset. The buffer contains no real data.
[29-27]	Buffer contents	Buffer contents: 000: Backwards Compatible (Experimental) Format packet destined to the AAL1 Payload Type machine 001: Standard format packet destined to the AAL1 Payload Type machine 010: Standard format packet destined to the AAL2 Payload Type machine 011: Non-TDMoIP/MPLS packet 100: Standard format packet destined to the HDLC Payload Type machine 101: CPS-PDU packets coming from the AAL2 Payload Type Machine 110: Echo request (query) type control (OAM) packet 111: Echo reply type control (OAM) packet Used by the Payload Type machines to identify packets received from the Packet Classifier and by the CPU to identify echo packets and echo reply control packets received from the Packet Classifier; Used also by the CPU to identify ETH → CPU packet types.

Table 4-20. Buffer Descriptor Data Elements (Cont.)

Bits	Data Element	Description
[26-16]	Length/Rst_Ts	Packet length or payload length; For TDM→CPU, TDM→TDM, CPU→TDM and ETH→TDM: payload length (received bytes and control word, if exists) in bytes For TDM→ETH, ETH→CPU and CPU→ETH: packet length in bytes, without CRC For Buffer Contents =101: total length of packets concatenated in the buffer, in bytes For RST packets: the reset timeslot number
[15]	Reserved	
[14-8]	Offset	For ETH→CPU, TDM→ETH and CPU→ETH: offset in bytes from start of buffer to start of packet For ETH→TDM, TDM→CPU, CPU→TDM: offset in bytes from start of buffer to start of payload For TDM→TDM: bits 13-8 hold the bundle number from which the buffer has been transmitted
[7]	HW/SW Type	The pool the buffer has been extracted from and should be returned to. 0: HW buffers pool 1: SW buffers pool For packets coming from Ethernet: 0: destination = Payload Type machines 1: destination = CPU
[6]	Tx/Rx Source	The pool from which the buffer has been extracted and should returned to : 0: Tx buffers pool 1: Rx buffers pool Can be used also by the CPU to indicate the buffer source (PCM/Ethernet).
[5-0]	Bundle number	For TDM→ TDM: destination bundle number; For any other bundle: Packet Bundle Number

RX Arbiter

The RX Arbiter constantly checks for available packets in RX FIFO, CPU to TDM queue and the cross-connect queue. It can do one of the following:

- Pass a packet from the RX FIFO into the Payload Type machines
- Pass a packet from the RX FIFO into the external SDRAM and insert its pointer into the ETH to CPU queue
- Extract a pointer from the cross-connect queue and pass a packet from the external SDRAM into the Payload Type machines
- Extract a pointer from the CPU to TDM queue and pass a packet from the external SDRAM into the Payload Type machines.

In general, the Rx Arbiter handles packets according to the following priorities:

Cross-connect queue

RX FIFO (i.e., packets that arrived from the Ethernet port)

CPU to TDM queue.

A configuration parameter, Rx_fifo_priority_lvl, enables the user to adjust the load of the different sources of data handled by the RX Arbiter. If the RX FIFO level exceeds the Rx_fifo_priority_lvl, then the RX Arbiter switches priorities between the Cross-connect queue and the RX FIFO (i.e., passes packets from the RX FIFO until its level goes below Rx_fifo_priority_lvl).

TX Ethernet Interface

The TX Ethernet interface first checks the TX Ethernet queue. If the queue is not empty, it extracts a pointer, passes the buffer data from the SDRAM to the Ethernet MAC and returns the pointer to the Free Buffer pool. If the TX Ethernet queue is empty, the TX Ethernet Interface checks the status of the CPU to Ethernet queue. If the queue is not empty, it extracts a pointer, transfers buffer data to the Ethernet MAC and returns the buffer to the CPU TX Return queue.

Free Buffer Pool

The free buffer pool mechanism explained below is relevant to all flows starting from the TDM side (to Ethernet, CPU and TDM).

Before the Payload Type machines can process any data, the SW needs to initialize the Free Buffer pool. The Free Buffer pool contains pointers to SDRAM buffers that will be used by the Payload Type machines to store packets. There are a total of 512 SDRAM buffers. The CPU needs to pre-assign (statically) these SDRAM buffers to each bundle. The number of buffers allocated per specific bundle depends on the number of timeslots in the bundle. It is recommended to assign 4 buffers per timeslot.

The buffers are located in a continuous area in the SDRAM. The buffer address consists of the base address, the buffer number and the displacement within the buffer. The base address is determined by the SW. Free buffer numbers are contained in a linked list, with a Head (see Memory Map section) pointing to the first buffer, each buffer pointing to the next buffer and the last buffer pointing to itself. There are 64 Heads (one per bundle), each one containing a validity indication bit (MSB) and another 9 bits pointing to the first free buffer in the linked list.

The SW defines the number of buffers for each bundle, by initializing the linked list of the bundle. The CPU prepares these buffers by writing the Ethernet, IP/MPLS headers in advance, so that the Payload Type machines need only to write the packet payload. Since the headers contain bundle specific data (e.g., destination address), the same buffers will be used for the same bundle until the bundle is closed by the SW.

When closing a bundle, the SW should check that all buffers have been returned, by following the linked list from the Head to the last buffer. The buffers of a closed bundle may be used for a different new bundle.

The linked list operation is depicted below.

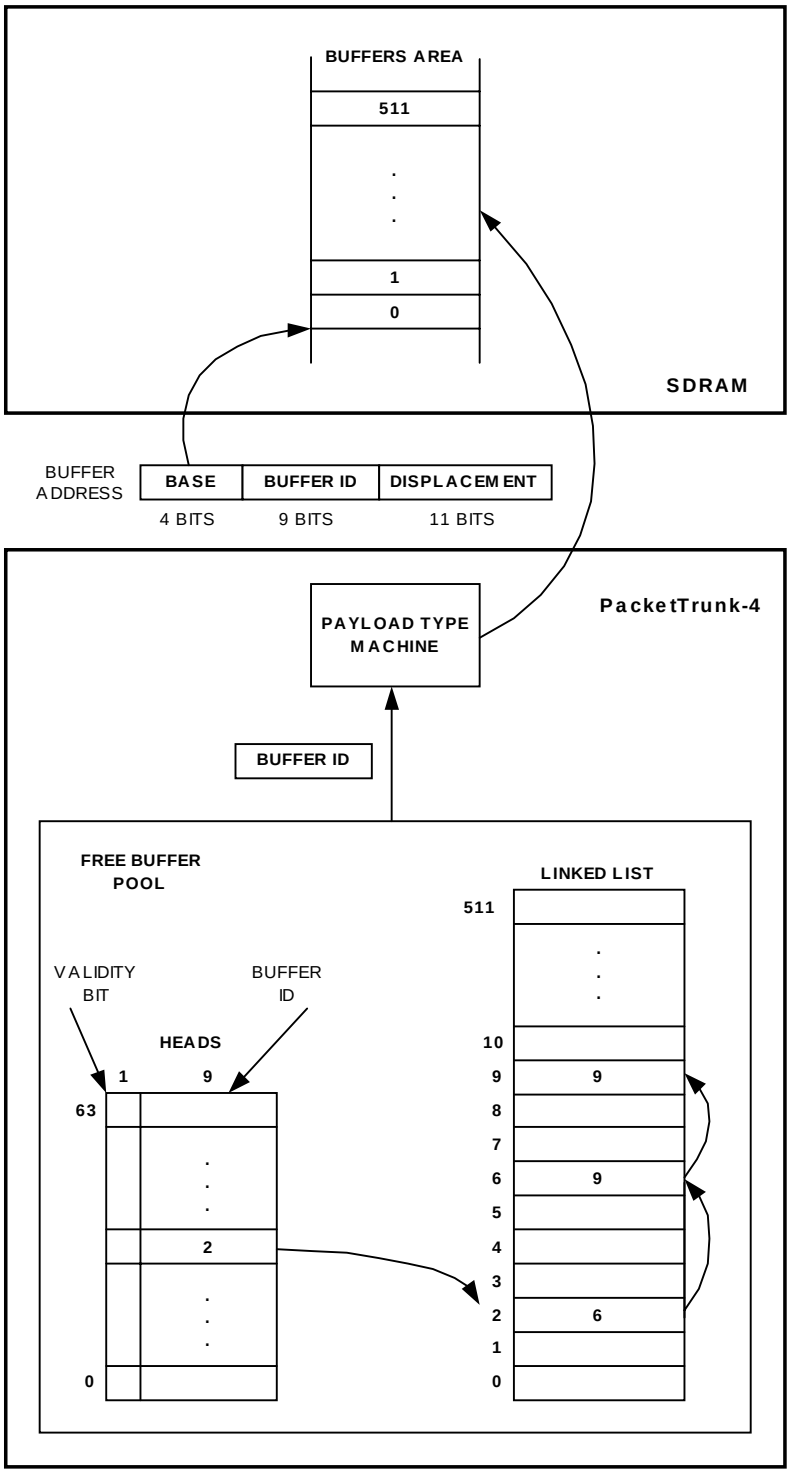


Figure -38. Free Buffer Pool Operation

TDM to Ethernet Flow

Each Payload Type machine receives data of specific bundle timeslots and transforms it into Ethernet packets. To store a new packet in preparation, the machine extracts a pointer from the Free Buffer pool and fills it as it processes the TDM data timeslots, one by one. When a packet is completed in a buffer, the machine places its pointer in the Ethernet TX queue. The TX Ethernet interface polls the queue, extracts the pointer, transfers the packets in the buffer to the Ethernet MAC block, to be sent over the Ethernet network. Then, it returns the pointer to the Free Buffer pool. The buffer can now be used again by the Payload Type machine to store TDM data for this bundle.

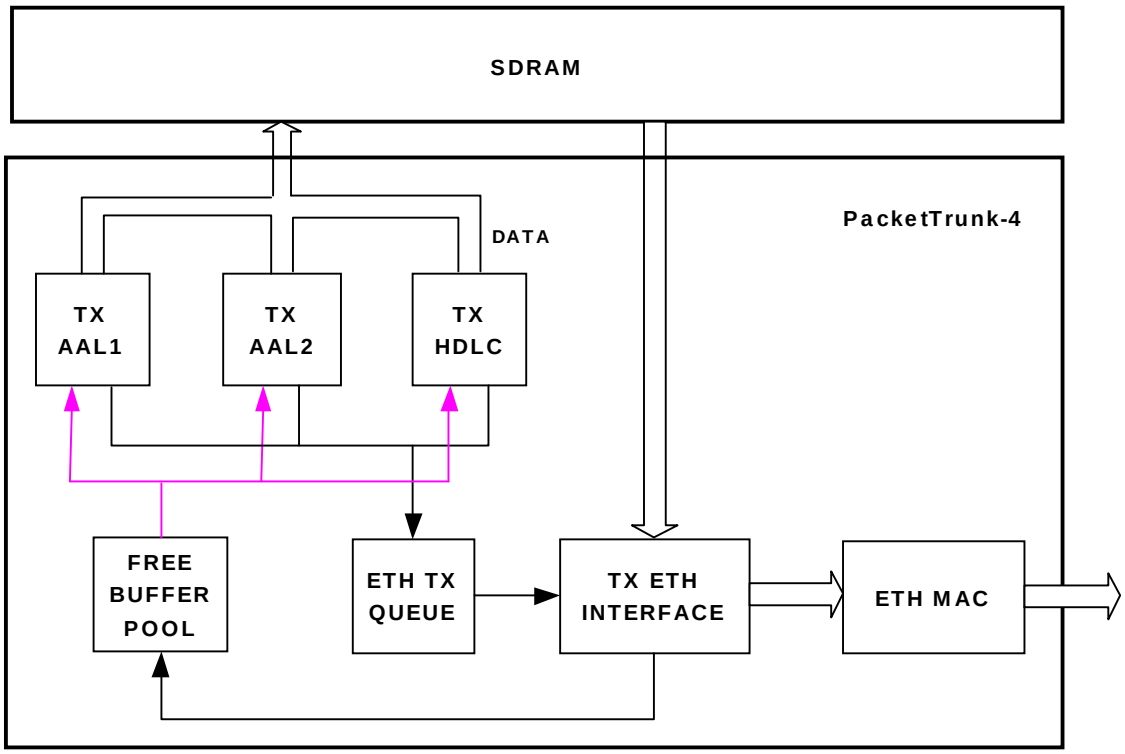


Figure 4-39. TDM to Ethernet Flow

Ethernet to TDM Flow

An Ethernet packet arriving from the Ethernet port passes through the ETH MAC block. The ETH MAC block does not store the packet, just calculates its CRC to verify it arrived correctly. If the packet is faulty, the ETH MAC signals this to the Packet Classifier on the last word of the packet and the Packet Classifier discards it.

The Packet Classifier examines the packet header and decides to either discard the packet or transfer it into the chip according to its internal CPU-configured tables. The Packet Classifier tags the buffer descriptor for one of the following destinations: ETH to CPU queue or Payload Type machines. The Packet Classifier stores the packet payload preceded by the buffer descriptor in the RX FIFO and notifies the RX Arbiter.

If the packet is destined to TDM, the RX Arbiter passes it to one of the Payload Type machines. If the packet is destined to the CPU, the RX Arbiter passes it to the SDRAM and places its pointer in the ETH to CPU queue.

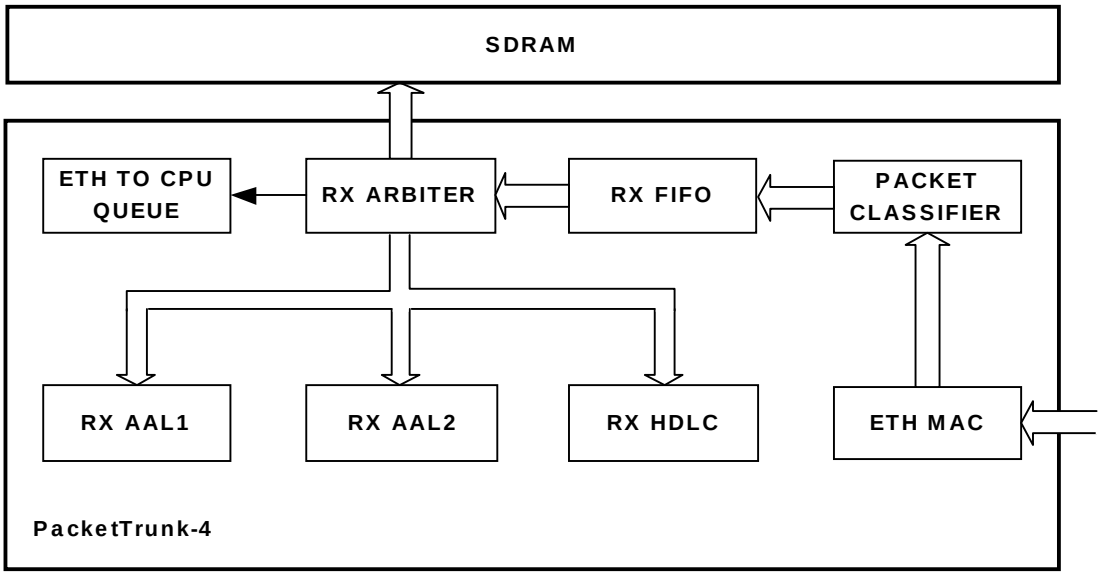


Figure 4-40. Ethernet to TDM Flow

TDM to TDM Flow

Each Payload Type machine receives data of specific bundle timeslots and transforms it into Ethernet packets. To store a packet, the machine needs an SDRAM buffer. To store a new packet in preparation, the machine extracts a pointer from the Free Buffer pool and fills it as it processes the TDM data timeslots, one by one. When a packet is completed in a buffer, the machine places its pointer in the Cross-connect queue. The RX Arbiter polls the cross-connect queue, extracts the pointer, and transfers the buffer data to the appropriate Payload Type machine, according to its descriptor. Then it returns the pointer to the Free Buffer pool. The Payload Type machine extracts the TDM data and inserts it into the Jitter buffer in the SDRAM. From there, the data is transferred later on over the TDM line.

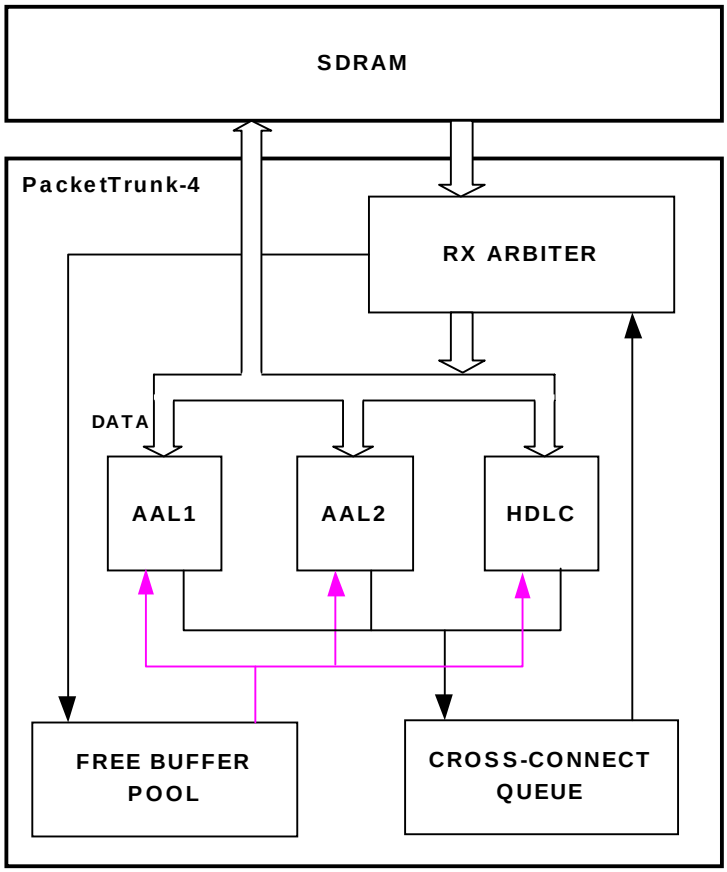


Figure 4-41. TDM to TDM Flow

TDM to CPU Flow

The Payload Type machines identify the destination of their packets according to the per bundle configuration. Upon getting the first byte of a packet (in a bundle destined to the CPU), the machine needs a buffer to store the packet. It checks whether a buffer is available in the TDM to CPU pool. If the pool is empty, the machine discards the current data. If a buffer is available, the machine uses it for storing the packet payload and then adds the pointer to the TDM to CPU queue. The CPU polls this queue to look for packets that need to be processed. After processing the packet, the CPU closes the loop by returning the pointer to the TDM to CPU pool.

The TDM to CPU pool and queue can contain up to 128 pointers each.

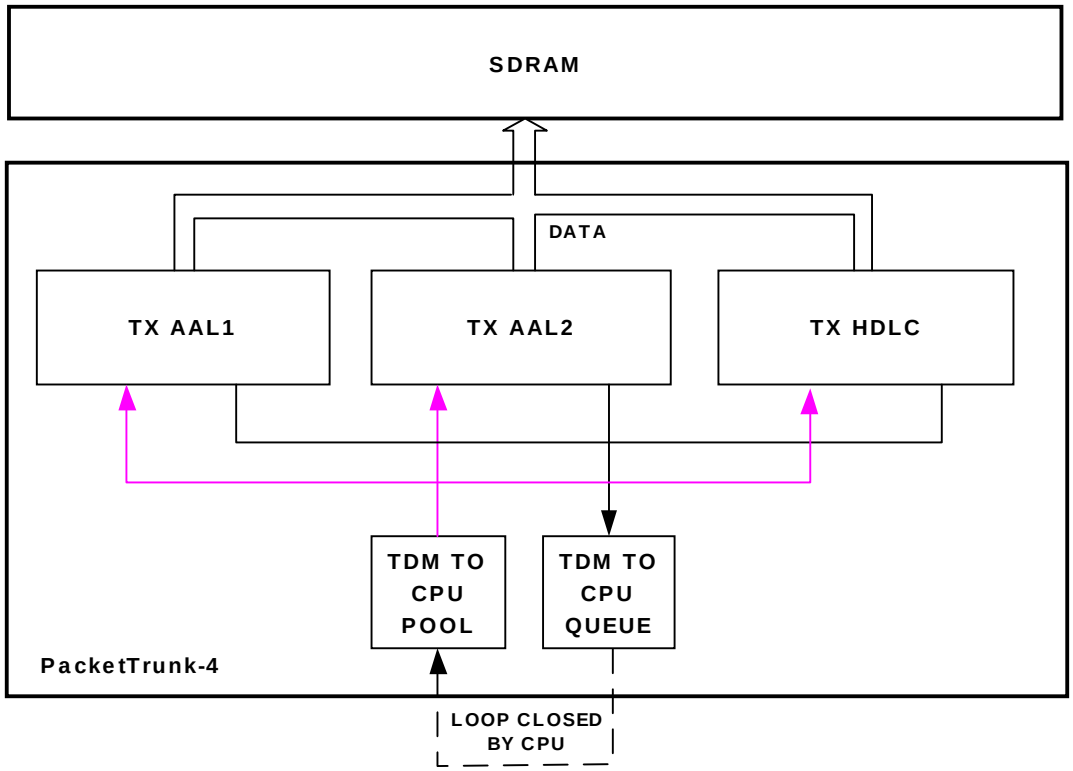


Figure 4-42. TDM to CPU Flow

CPU to TDM Flow

The RX Arbiter polls the CPU to TDM queue for new packets waiting in the SDRAM to be processed. If the queue level is greater than zero and there are no buffers pending in the RX FIFO or the Cross—connect queue, the RX Arbiter extracts the pointer and copies the relevant data from the SDRAM to the appropriate Payload Type machine (according to the buffer descriptor). Then it checks whether the CPU RX Return queue is not full to return the pointer. If the return queue is full, the pointer will remain in the Rx Arbiter and the Rx Arbiter will not poll the CPU to TDM queue until it succeeds to return the previous pointer. After returning the pointer to the CPU RX Return queue for reuse, the RX Arbiter is ready to take another pointer from the CPU to TDM queue.

Both queues can hold up to 32 pointers.

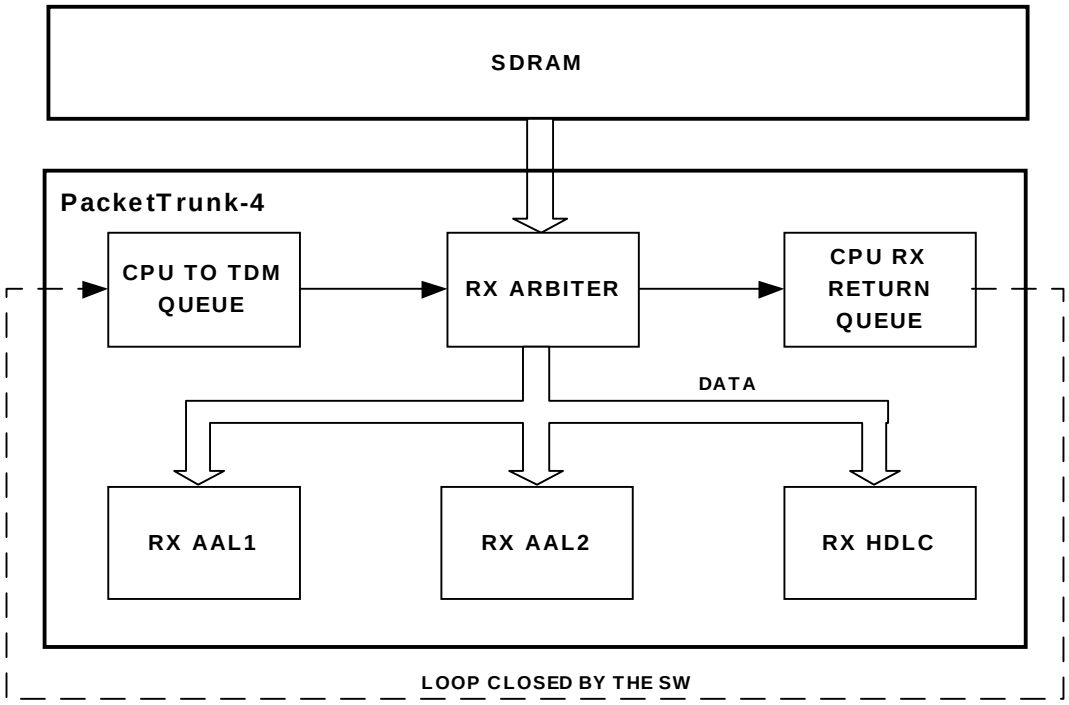


Figure 4-43. CPU to TDM Flow

CPU to Ethernet Flow

The TX ETH Interface polls the CPU to ETH queue for new packets waiting in the SDRAM to be processed. If the queue level is greater than zero and no buffers from the Payload Type machines are waiting, the TX ETH Interface extracts the pointer and copies the relevant data from the SDRAM to the ETH MAC block (according to the buffer descriptor). Then it checks whether the CPU TX Return queue is not full to return the pointer. If the return queue is full, the pointer will remain in the TX ETH interface and the interface will not poll the CPU to ETH queue until it succeeds to return the previous pointer. After returning the pointer to the CPU TX Return queue for reuse, the TX ETH Interface is ready to take another pointer from the CPU to ETH queue.

Both queues can hold up to 32 pointers.

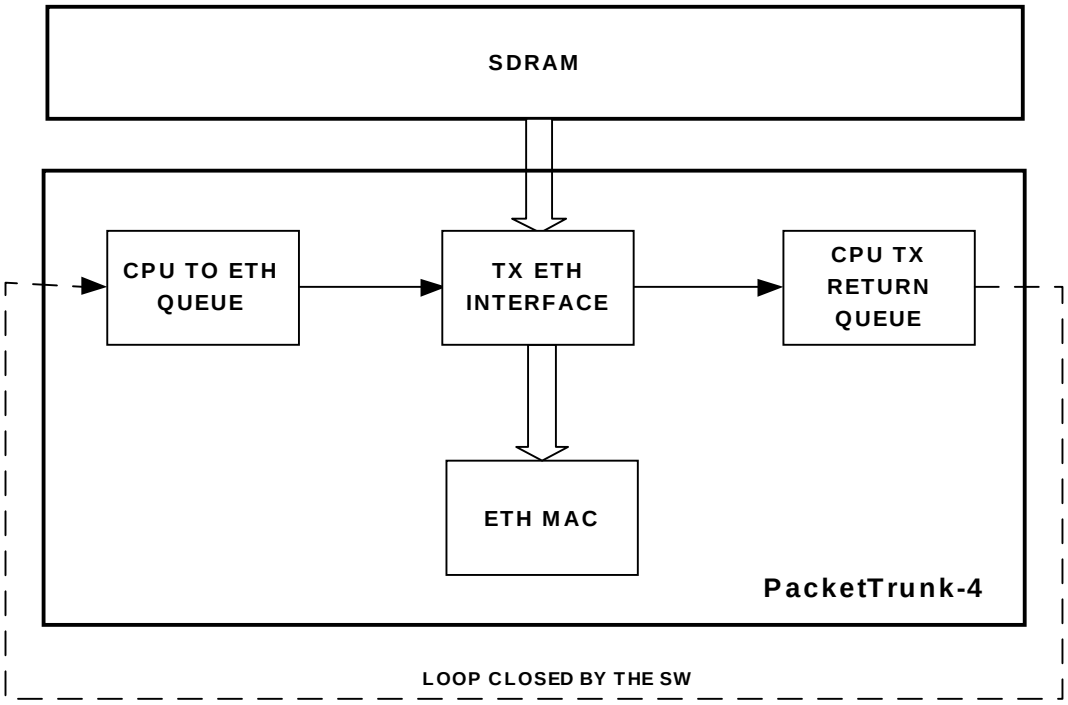


Figure 4-44. CPU to Ethernet Flow

Ethernet to CPU Flow

Ethernet packets enter the chip via the ETH MAC block and the Packet Classifier into the RX Arbiter. When the RX Arbiter identifies that a packet is destined to the CPU, it extracts a pointer from the Ethernet to CPU pool (if the pool is empty, the RX Arbiter discards the packet) and stores the packet data into the SDRAM in the buffer indicated by the pointer. Then, it sends the pointer to the Ethernet to CPU queue (processed by the CPU). If the queue is full, the RX Arbiter keeps the pointer for itself for future use. The Ethernet to CPU queue and pool contain up to 64 pointers each.

If the RX Arbiter receives an Ethernet packet destined to the AAL2 Payload Type machine (which may contain AAL2-CPS packets destined to the CPU), it passes the packet to the AAL2 Payload Type machine. The machine extracts the AAL2-CPS packets from the Ethernet packet and checks each AAL2-CPS packet header as configured to determine its destination. If a header check of an AAL2-CPS packet fails, the AAL2-CPS packet is ignored and dropped. If a legal AAL2-CPS packet is detected as destined to the CPU, the machine extracts a pointer from the AAL2 to CPU pool (unless it already owns a pointer, or the pool is empty), and writes the AAL2-CPS packet data into the SDRAM into the buffer indicated by the pointer. The AAL2 Payload Type machine keeps the pointer and concatenates new incoming AAL2-CPS packets destined to the CPU, until the buffer is full or a time-out of approximately 1 msec has passed. Then it returns the pointer to the AAL2 to CPU queue. The AAL2 to CPU pool and queue contain up to 32 pointers each.

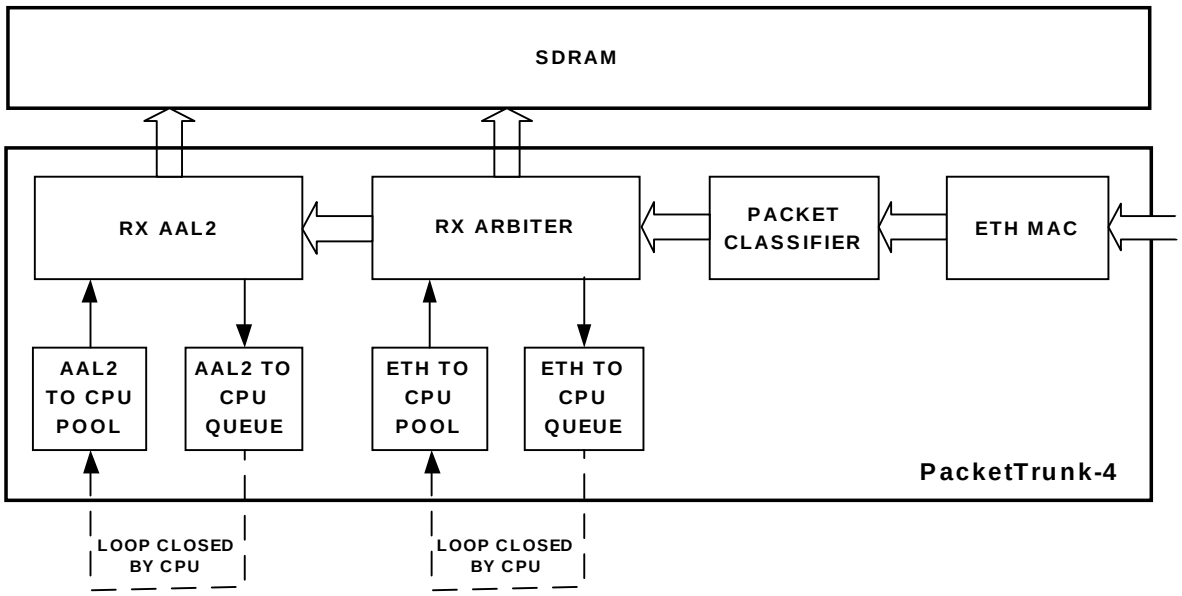


Figure 4-45. Ethernet to CPU Flow

AAL2 to CPU Buffer Format

The AAL2 to CPU Buffer consists of a buffer descriptor in the first four bytes. The first byte preceding each AAL2-CPS packet is the internal bundle number with which the following AAL2-CPS packet is associated.

AAL2 TO CPU BUFFER DESCRIPTOR	AAL2-CPS PACKET #1 INTERNAL BUFFER NO.	AAL2-CPS PACKET #1	...	AAL2-CPS PACKET #n INTERNAL BUFFER NO.	AAL2-CPS PACKET #n
4 BYTES	1 BYTE	1-64 BYTES		1 BYTE	1-64 BYTES

Figure 4-46. AAL2 to CPU Buffer Format

AAL2 to CPU Buffer Descriptor Format

The AAL2 to CPU Buffer Descriptor Format is shown in , "Table 4-21. AAL2 to CPU Buffer Descriptor Format," on page 229. Refer to , "Table 4-20. Buffer Descriptor Data Elements," on page 218 for description of Buffer Descriptor Data Elements.

Table 4-21. AAL2 to CPU Buffer Descriptor Format

Bits	Data Element	Value
31	N/A	0
30	N/A	0
29–27	Buffer contents	101
26–16	Length/Rst_Ts	Total bytes, including AAL2-CPS Packets and preceding internal bundle number bytes
15	N/A	0
14–8	Offset	Sw_packet_offset (in bytes, as configured in registers)
7	HW/SW Type	1
6	Tx/Rx Source	1
5–0	N/A	000000

4.13 ETHERNET MAC

The Ethernet MAC can operate at 10 or 100 Mbps. It supports MII, RMII (Reduced MII), SMII (Serial MII) and SSMII (source synchronous serial MII). The MAC interface to the physical layer should be configured by the user.

The UNH tested Ethernet MAC complies with IEEE 802.3. Its counters enable the software to generate network management statistics compatible with IEEE 802.3 Clause 5.

The Ethernet MAC supports physical layer management through MDIO interface. The control registers drive the MDIO interface and select modes of operation, such as full or half duplex. Half-duplex flow control is achieved by forcing collisions on incoming packets. Full duplex flow control supports recognition of incoming pause packets.

In the receive path, the block checks the incoming packets for valid preamble, FCS, alignment and length, and presents received packets to the Packet Classifier. Although packets with physical errors are discarded as a default, they can be ignored by configuration.

In the transmit path, the Ethernet MAC takes data from the TX Ethernet interface, adds preamble and, if necessary, pad and FCS, then transmits data according to the CSMA/CD (carrier sense multiple access with collision detect) protocol. The start of transmission is deferred if MII_CRS (carrier sense) is active. If MII_COL (collision) becomes active during transmission, a jam sequence is asserted and the transmission is retried after a random back off. MII_CRS and MII_COL have no effect in full duplex mode.

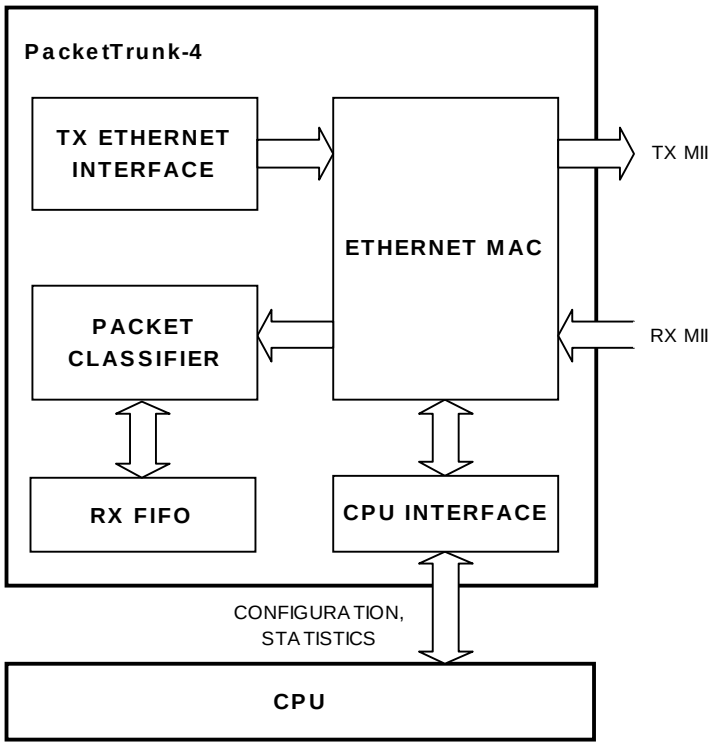


Figure 4-47. Ethernet MAC

4.14 Pause Packet Support

The MAC network configuration register contains a Pause Enable bit (13).

Table 4-22. Start of an 802.3 Pause Packet

Destination Address	Source Address	Ether Type (MAC Control Frame)	Pause opcode	Pause Time
0x0180C2000001	6 bytes	0x8808	0x0001	2 bytes

If a valid pause packet is received, the Pause Time register is updated with the packet's pause time regardless of its current contents and regardless of the state of the MAC Network Configuration register bit 13. Pause_packet_received interrupt (MAC Interrupt Status Register bit 12) is triggered when a pause packet is received (assuming it is enabled in the MAC interrupt mask register).

If Pause Enable bit is set and the value of the pause time register is non-zero, no new packet is transmitted.

A valid pause packet is defined as having a destination address that matches 0x0180C2000001 and has the MAC control frame type ID of 0x8808 and has the pause opcode of 0x0001.

Pause packets that have FCS or other errors are treated as invalid and discarded. Valid received pause packets increment the Pause Packets Received OK counter.

The Pause Time register decrements every 512 bit times once transmission has stopped. For test purposes, the register decrements every rx_clk cycle once transmission has stopped if bit 12 (retry_test) is set in the

MAC network configuration register. If the pause enable bit is not set, the decrementing happens regardless of whether transmission has stopped or not.

Pause_time_zero interrupt (MAC Interrupt Status register bit 13) is asserted whenever the Pause Time register decrements to zero (assuming it is enabled in the MAC Interrupt Mask register).

Automatic transmission of pause packets is supported through the Transmit Pause Packet bits of the Network Control register. If either bit 11 or bit 12 of the MAC Network Control register is written with 1, a pause packet is transmitted only if full duplex is selected in the MAC Network Configuration register and transmit is enabled in the MAC Network Control register. Pause packet transmission takes place immediately if transmit is inactive or if transmit is active between the current packet and the next packet due to be transmitted. The Transmitted Pause packet comprises the items in the following list:

- Destination address of 01-80-C2-00-00-01
- Source address taken from the MAC specific address registers
- Type ID of 88-08 (MAC control frame)
- Pause opcode of 00-01
- Pause quantum
- Fill of 00 to take the frame to minimum frame length
- Valid FCS.

The pause quantum used in the generated packet depends on the trigger source for the packet as follows:

- If bit 11 is written with one, the pause quantum comes from the MAC Transmit Pause Quantum register. The Transmit Pause Quantum register resets to a value of 0xFFFF giving a maximum pause quantum as a default.
- If bit 12 is written with one, the pause quantum is zero.

After transmission, no interrupts are generated and the only incremented counter is the Transmitted Pause Packets Counter.

Pause packets can also be transmitted by the MAC using normal packet transmission methods. It is possible to transmit a pause packet while the transmitter is paused by resetting the Pause Enable bit in the Network Configuration register.

4.15 PACKET CLASSIFIER

The Packet Classifier is part of the Receive path, immediately following the Ethernet MAC block. It analyzes the header of each incoming packet, by comparing the header fields to the chip's configured parameters, and then decides to discard the packet, or add the buffer descriptor and forward the packet to the CPU or to one of the Payload Type machines.

- Although the chip has two IP addresses, in most cases both IP addresses should have the same value.
- Although the chip has two tdmoip_port_numbers, in most cases both tdmoip_port_numbers should have the default value (0x085E) as assigned by IANA for TDMoIP.

Packets with CRC errors are discarded regardless to their contents, unless the Ethernet MAC has been configured to ignore them (in which case they are treated as correct packets).

Packets other than TDMoIP or MPLS packets destined to the chip are not transferred to the Payload Type machines. They are either discarded or transferred to the CPU according to the following eight chip configuration switches:

Discard_Switch #0: An ARP packet whose IP destination address is not identical to any of the chip's IP addresses is discarded if Discard_Switch #0 is set. Otherwise it is transferred to the CPU.

Discard_Switch #1: An IP packet whose IP destination address is not identical to any of the chip's IP addresses is discarded if Discard_Switch #1 is set. Otherwise it is transferred to the CPU.

Discard_Switch #2: A packet whose Ether Type is different than IP, ARP or MPLS is discarded if Discard_Switch #2 is set. Otherwise it is transferred to the CPU.

Discard_Switch #3: An ARP packet whose IP destination address is identical to one of the chip's IP addresses is discarded if Discard_Switch #3 is set. Otherwise it is transferred to the CPU.

Discard_Switch #4: An IP packet destined to the chip whose protocol is different than UDP is discarded if Discard_Switch #4 is set. Otherwise it is transferred to the CPU.

Discard_Switch #5: An IP/UDP packet destined to the chip whose UDP destination port number is not identical to one of the chip's TDMoIP port numbers is discarded if Discard_Switch #5 is set. Otherwise it is transferred to the CPU.

Discard_Switch #6: An IP/UDP/TDMoIP packet destined to the chip whose bundle number is not identical to the chip's OAM Bundle Number or one of the bundle numbers assigned to the chip's internal bundles, is discarded if Discard_Switch #6 is set. Otherwise it is transferred to the CPU.

Discard_Switch #7: An IP/UDP/TDMoIP packet destined to the chip whose bundle number is identical to the chip's OAM Bundle Number is discarded if Discard_Switch #7 is set. Otherwise it is transferred to the CPU.

A packet is identified as a TDMoIP packet destined to the chip if it meets the following conditions:

- It is a unicast with its destination address identical to the chip's MAC address, multicast or broadcast
- It has either no VLAN tags, one VLAN tag or two VLAN tags (supports VLAN stacking)
- Its protocol is IP/UDP
- Its IP address is identical to one of the IP addresses of the chip
- Its UDP destination port number is identical to one of the chip's TDMoIP port numbers
- Its bundle number is identical to the chip's OAM Bundle Number (TDMoIP OAM packets) or to one of the bundle numbers assigned to the chip's internal bundles.

A packet is identified as a TDMoMPLS packet destined to the chip if it meets the following conditions:

- It is a unicast with its destination address identical to the chip's MAC address, multicast or broadcast
- It has either no VLAN tags, one VLAN tag or two VLAN tags (VLAN stacking)
- Its Ether Type is MPLS unicast or MPLS multicast
- The 7 most significant bits of the inner MPLS label are identical to the chip's Last_label_msbits
- The bundle number located at the 13 least significant bits of the inner MPLS label are identical with the chip's OAM bundle number (TDMoMPLS OAM packets) or to one of the bundle numbers assigned to the chip's internal bundles.

The structure of packets identified as TDMoIP packets destined to a specific bundle of the chip or as OAM packets destined to the chip is shown below.

DA MAC_add/ Broadcast/ Multicast	**SA**	**Eth Type** IP	**IP Header** Dst. IP = IP_Add1/ IP_Add2	**UDP Header** Bundle no. = Rx_Ext_Bundle_Num/ OAM_bundle_num	**Control Word** Optional	**Payload Type** AAL1/AAL2/ HDLC/OAM	**CRC32**

Figure 4-48. TDMoIP Packet without VLAN Tag

DA MAC_add/ Broadcast/ Multicast	**SA**	**Vlan Tag** up to 2 tags	**Eth Type** IP	**IP Header** Dst. IP = IP_Add1/ IP_Add2	**UDP header** Bundle no. = Rx_Ext_Bundle_Num/ OAM_bundle_num	**Control Word** Optional	**Payload Type** AAL1/AAL2/ HDLC/OAM	**CRC32**

Figure 4-49. TDMoIP Packet with VLAN Tag

DA MAC_add/ Broadcast/ Multicast	**SA**	**Eth Type** MPLS	**Up to 2 MPLS Labels** Optional	**MPLS Label** Bundle no. = Rx_Ext_Bundle_Num/ OAM_bundle_num	**Control Word**	**Payload Type** AAL1/AAL2/ HDLC/OAM	**CRC32**

Figure 4-50. TDMoMPLS Packet without VLAN Tag

DA MAC_add/ Broadcast/ Multicast	**SA**	**Vlan Tag** up to 2 tags	**Eth Type** MPLS	**Up to 2 MPLS Labels** Optional	**MPLS Label** Bundle no. = Rx_Ext_Bundle_Num/ OAM_bundle_num	**Control Word**	**Payload Type** AAL1/AAL2/ HDLC/OAM	**CRC32**

Figure 4-51. TDMoMPLS Packet with VLAN Tag

Packets that passed the classification process are temporarily stored in the RX FIFO. This FIFO is used to buffer momentary bursts from the network if the internal hardware is busy. The RX Arbiter transfers the packets in the RX FIFO into the Payload Types machines or external SDRAM.

4.16 COUNTERS AND STATUS REGISTERS

For information about counters and registers, refer to Memory Map section.

4.17 CONNECTION LEVEL REDUNDANCY

The PacketTrunk-4 provides optional connection level redundancy for AAL1 bundles. In the TDM to Ethernet direction, on a bundle basis, each packet may be transmitted once with certain ETH/IP/UDP headers, or twice, each time with different ETH/IP/UDP headers. When transmitted twice, the packets have the same payload, but may have different packet headers (including layer-2, -3 and -4 headers). This way, the traffic may be transmitted via two different routes over the network.

For example, the chip can duplicate a bundle's packets on transmission where the only difference between the duplicated packets is their bundle number. On the receive side, the chip can be configured to receive only the packets with the first bundle number or the packets with the second bundle number.

To enable this feature, the SW must initialize the transmit buffers of a bundle with both headers. By changing the Protection_mode configuration field of the AAL1 bundle, the user can choose (per bundle) whether to transmit each of the packets once with the first or the second header, or twice, each time with a different header.

On the receive side, only the packets with their bundle number configured in the Rx_ext_bundle_num field of the specific AAL1 bundle, are forwarded. The SW may change this value dynamically, in order to switch to the redundant connection at any time.

4.18 OAM SIGNALING

TDMoIP bundles require a signaling mechanism to provide feedback regarding problems in the communications environment. In addition, such signaling can be used to collect statistics relating to the performance of the underlying PSN. The OAM procedures detailed below are ICMP-like.

Connectivity Check Messages

In most conventional IP applications, a server sends some finite amount of information over the network after an explicit request from a client. With TDMoIP, the source sends a continuous stream of packets towards the destination, without knowing whether the destination device is ready to accept them, leading to flooding of the PSN. The problem may occur when a TDMoIP gateway fails or is disconnected from the PSN, or the bundle is broken. After an aging time, the destination gateway disappears from the routing tables, and intermediate routers may flood the network with the TDMoIP packets in an attempt to find a new path.

The solution to this problem is to significantly reduce the number of TDMoIP packets transmitted per second when bundle failure is detected, and to return to full rate only when the bundle is restored. The detection of failure and restoration is made possible by the periodic exchange of one-way connectivity check messages. Connectivity is tested by periodically sending OAM messages from the source gateway to the destination gateway, and having the destination reply to each message.

The connectivity check mechanism can also be useful during setup and configuration. Without OAM signaling, one must ensure that the destination gateway is ready to receive packets before starting to send them. Since TDMoIP gateways operate full duplex, both must be set up and properly configured simultaneously to avoid flooding. By using the connectivity mechanism, a configured gateway waits until it can detect its destination before transmitting at full rate. In addition, errors in configuration can be readily discovered by using the service specific field.

Performance Measurements

In addition to one-way connectivity, the OAM signaling mechanism can be used to request and report on various PSN metrics, such as one-way delay, round trip delay, packet delay variation, etc. It can also be used for remote diagnostics, and for unsolicited reporting of potential problems (e.g. dying gasp messages).

Processing OAM Packets

In the Ethernet to TDM direction, the PacketTrunk-4 identifies incoming packets as OAM if their bundle number is equal to the “OAM_bundle_num” value in Packet_classifier_cfg_reg5. An OAM packet is identified as query message or as reply message according to its “OAM Msg Type” field.

The chip HW timestamps Query and Reply messages as follows:

Table 4-23. Timestamping OAM Messages

Direction	Message Type	Contents	Field
Ethernet ⇒ TDM	Query	Reception time	Destination Receive Timestamp
	Reply	Reception time	Source Receive Timestamp
TDM ⇒ Ethernet	Query	Transmission time	Source Transmit Timestamp
	Reply	Transmission time	Destination Transmit Timestamp

When the CPU transmits an OAM packet, the buffer descriptor must identify the packet either as query message or reply message, so that the HW can update the source Transmit Timestamp field or the destination Transmit Timestamp field respectively.

When the CPU receives an OAM packet with reply message, it may calculate the round trip delay according to the following formula:

Src. Rx Timestamp - Src. Tx Timestamp – (Dst. Tx Timestamp – Dst. Rx Timestamp)

To disable time stamping, clear “En_timestamp” in Packet_classifier_cfg_reg5.

5.0 APPLICATIONS

5.1 CONNECTING A SERIAL INTERFACE TRANSCEIVER

Figure 5-1. Connecting Port One of the PacketTrunk-4 to a Serial Transceiver below shows the connection of the chip to a serial interface transceiver such as RS-530. The figure shows the first port in a DCE (Data Communications Equipment) application. All other ports can be connected in the same way.

Each direction (Tx and Rx) has its own clock. However, TDM1_RCLK is optional, as the PacketTrunk-4 may work in one clock mode in which both directions are clocked by TDM1_TCLK. The clock source of RCLK or TCLK can be either:

- Internal (from the local oscillator)
- External
- Recovered from the packet network (provided by the PacketTrunk-4).

The control input signal TDM1_RSIG_RTS does not affect the data reception.

The TDM1_TSIG_CTS and TDM1_TX_MF_CD outputs are set by software.

The TDM1_RSIG_RTS value can be read by software.

A maskable interrupt may be asserted when RTS value changes.

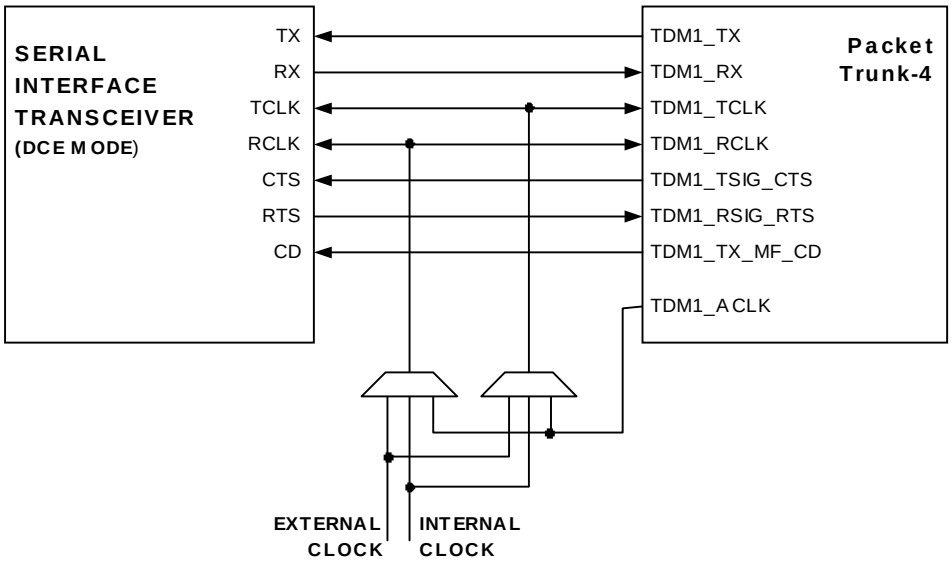


Figure 5-1. Connecting Port One of the PacketTrunk-4 to a Serial Transceiver

5.2 CONNECTING AN E1/T1 FRAMER

Figure 5-2. Connecting the First Port of the PacketTrunk-4 to an E1/T1 Framer in Applications with CAS and Figure 5-3. Connecting the First Port of the PacketTrunk-4 to an E1/T1 Framer in Applications without CAS below show the connection of the first port of the PacketTrunk-4 to an E1/T1 framer. All other ports can be connected in the same way.

Both Tx and Rx directions are clocked by TCLK.

The clock source of TCLK can be either:

- Internal (from the local oscillator)
- External
- Recovered from the packet network (provided by the PacketTrunk-4)
- RCLK (from the framer).

TDM1_TX_MF_CD and TDM1_RX_MF can be left unconnected or connected to ground if the framer cannot drive them. The chip has an internal free running counter, which generates these signals internally when not driven by an external source. These internally generated multiframe sync signals are synchronized to the sync input pulse.

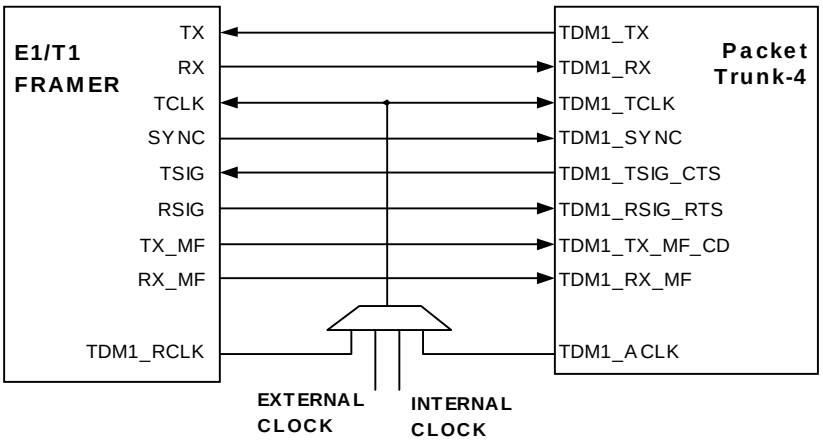


Figure 5-2. Connecting the First Port of the PacketTrunk-4 to an E1/T1 Framer in Applications with CAS

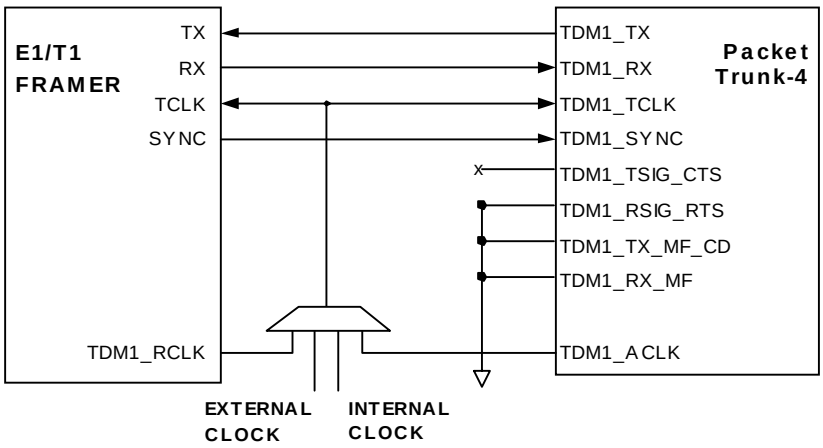


Figure 5-3. Connecting the First Port of the PacketTrunk-4 to an E1/T1 Framer in Applications without CAS

5.3 CONNECTING AN ETHERNET MAC OR PHY

Figure 5-4. Connecting the PacketTrunk-4's Ethernet Port to a PHY in MII Mode through Figure 5-10. Connecting the PacketTrunk-4's Ethernet Port to a PHY in SMII Mode below show the connection of the PacketTrunk-4's Ethernet port to a MAC or PHY device, in MII, RMII, SSMII and SMII modes.

Notes:

1. The chip cannot be connected to a MAC in SMII mode.
2. In SMII mode, when the chip works opposite a PHY with timing specifications meeting the SMII specifications, the trace delay between the chip and the PHY should be kept as low as possible and must be equal to or less than 1 ns. In SSMII mode there are two clock signals, one for each direction (Rx and Tx), routed together with the sync and data signals. As the delay between the clock and these signals is lower, the designer can apply a longer trace delay in this mode (compared to the SMII mode).

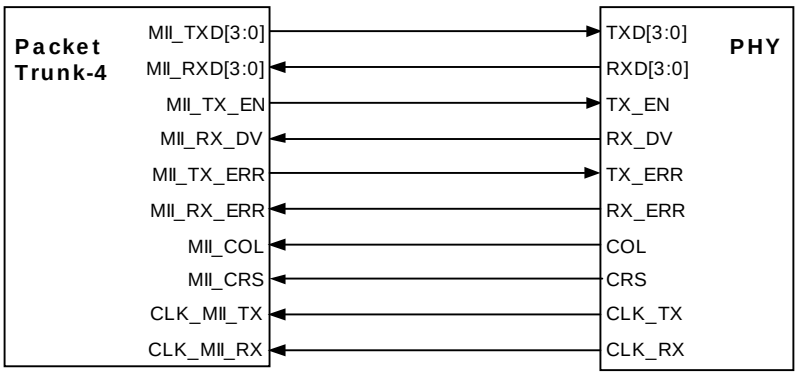


Figure 5-4. Connecting the PacketTrunk-4's Ethernet Port to a PHY in MII Mode

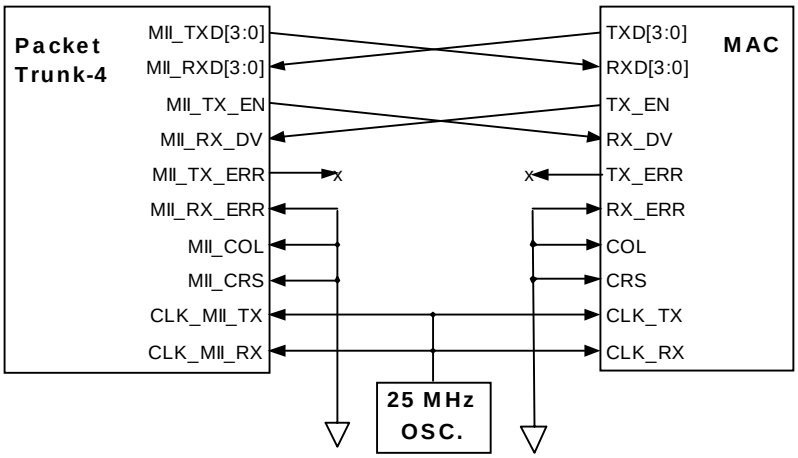


Figure 5-5. Connecting the PacketTrunk-4's Ethernet Port to a MAC in MII Mode

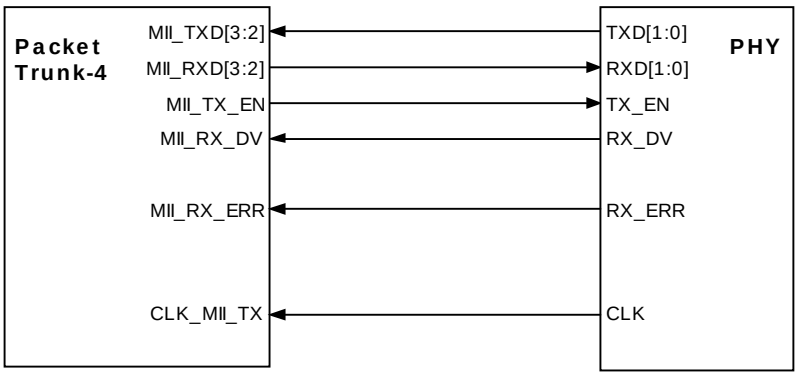


Figure 5-6. Connecting the PacketTrunk-4's Ethernet Port to a PHY in RMII Mode

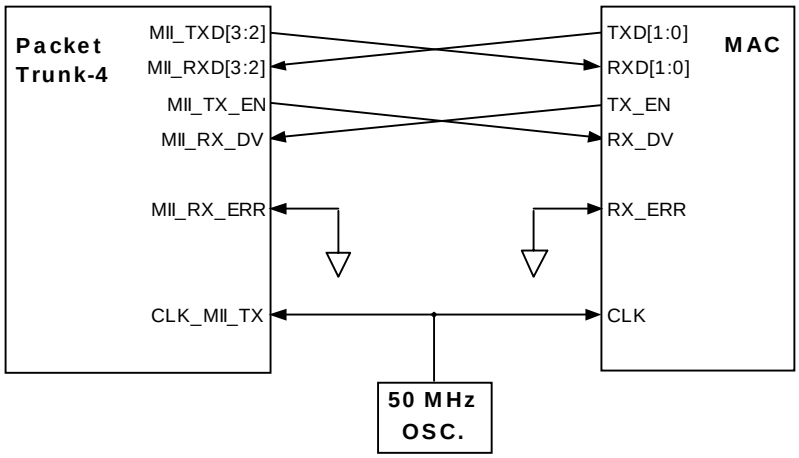


Figure 5-7. Connecting the PacketTrunk-4's Ethernet Port to a MAC in RMII Mode

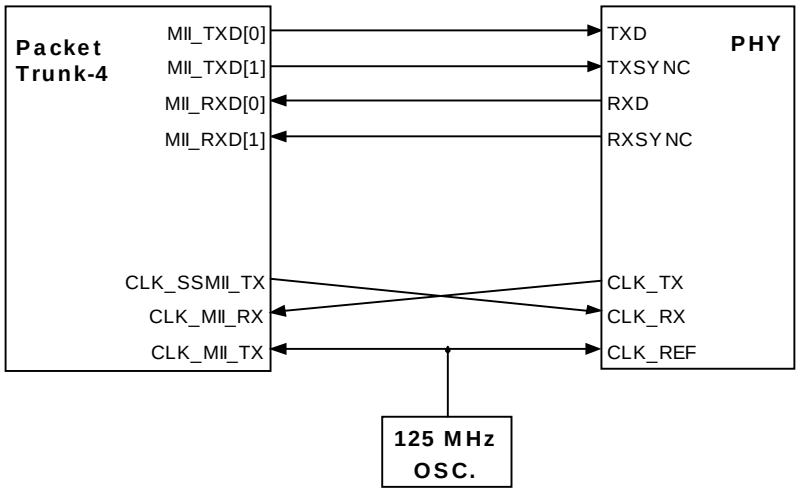


Figure 5-8. Connecting the PacketTrunk-4's Ethernet Port to a PHY in SSMII Mode

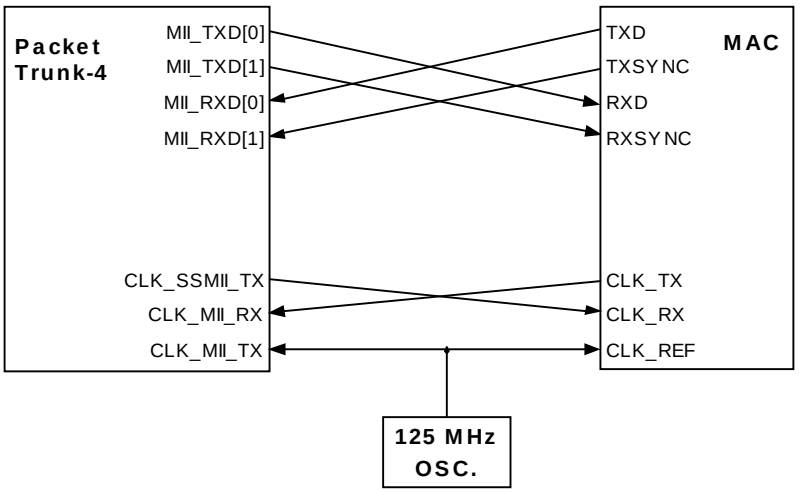


Figure 5-9. Connecting the PacketTrunk-4's Ethernet Port to a MAC in SSMII Mode

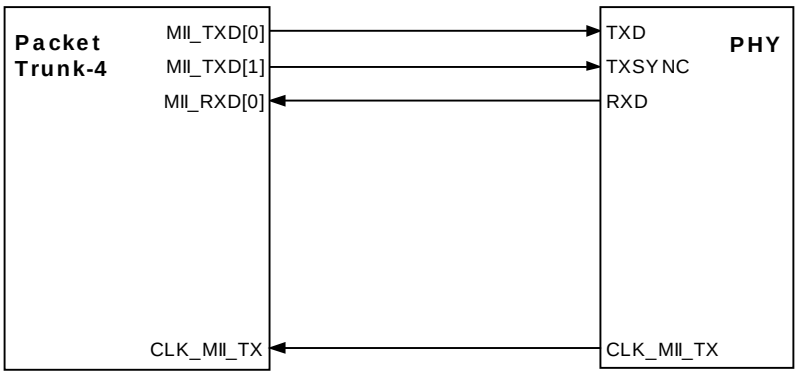


Figure 5-10. Connecting the PacketTrunk-4's Ethernet Port to a PHY in SMII Mode

For the applications above, apply the following layout considerations:

- Provide termination on all high-speed interface signals and clock lines.
- Provide impedance matching on long traces to prevent reflections.
- Keep the clock traces away from all other signals to minimize mutual interference.
- In SMII and RMII modes, a very low skew clock buffer/driver is recommended to maximize the timing budget. In these modes it is recommended to keep all traces as short as possible.
- In SSMII mode, keep data/sync traces and clock traces at the same length to maximize the timing budget.

5.4 IMPLEMENTING CLOCK RECOVERY IN HIGH SPEED APPLICATIONS

For the high-speed interface (up to 44.736 MHz), an external VCXO synthesizes the recovered clock frequency, using the chip's external DAC interface. Synthesizing frequency for the high-speed interface requires an external DAC (Digital to Analog Converter) device and a VCXO (Voltage Controlled Crystal Oscillator) connected to the chip's external 10-pin DAC interface (D2A_CS_N, D2A_WR_N and D2A[7:0]), as depicted below:

Figure 5-11. External Frequency Synthesis

Note: D2A[7] is the MSB and D2A[0] is the LSB.

The DAC converts a range of 0-255 digital values into analog voltage to drive the VCXO. The VCXO output is the recovered clock of the high-speed interface.

In this mode, the CLK_HIGH input should be tied to GND.

5.5 CONNECTING A MOTOROLA MPC860 HOST

The PacketTrunk-4 can be easily connected to a Motorola MPC860 host by means of the MPC860 GPCM (General Purpose Chip Select Machine) module.

Connecting the Bus Signals

Since the MPC860 bus MSB is always '0' while in the PacketTrunk-4 '0' is always the LSB, the signal order should be switched as depicted in the following figures.

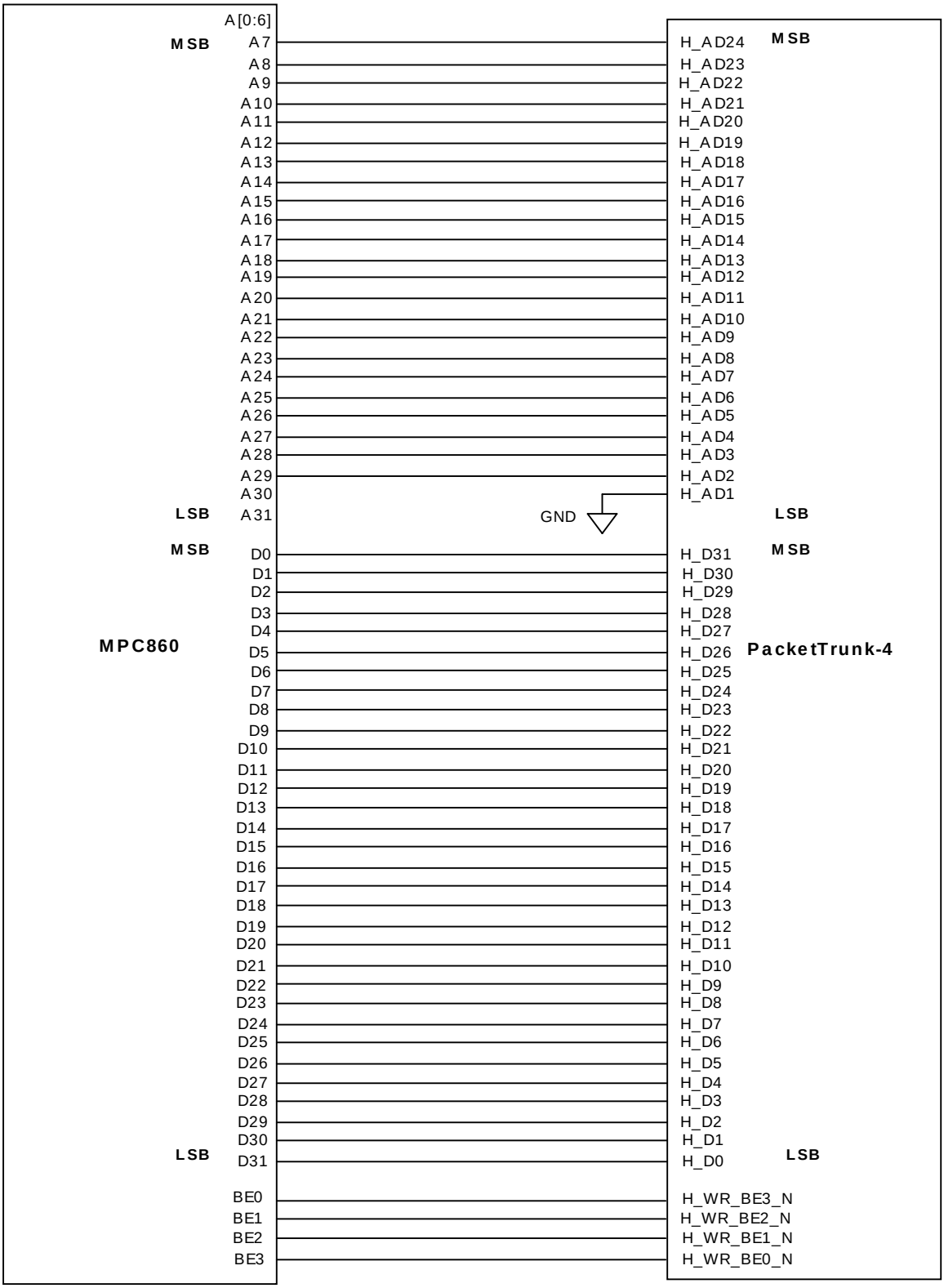


Figure 5-12. 32-Bit CPU Bus Connections

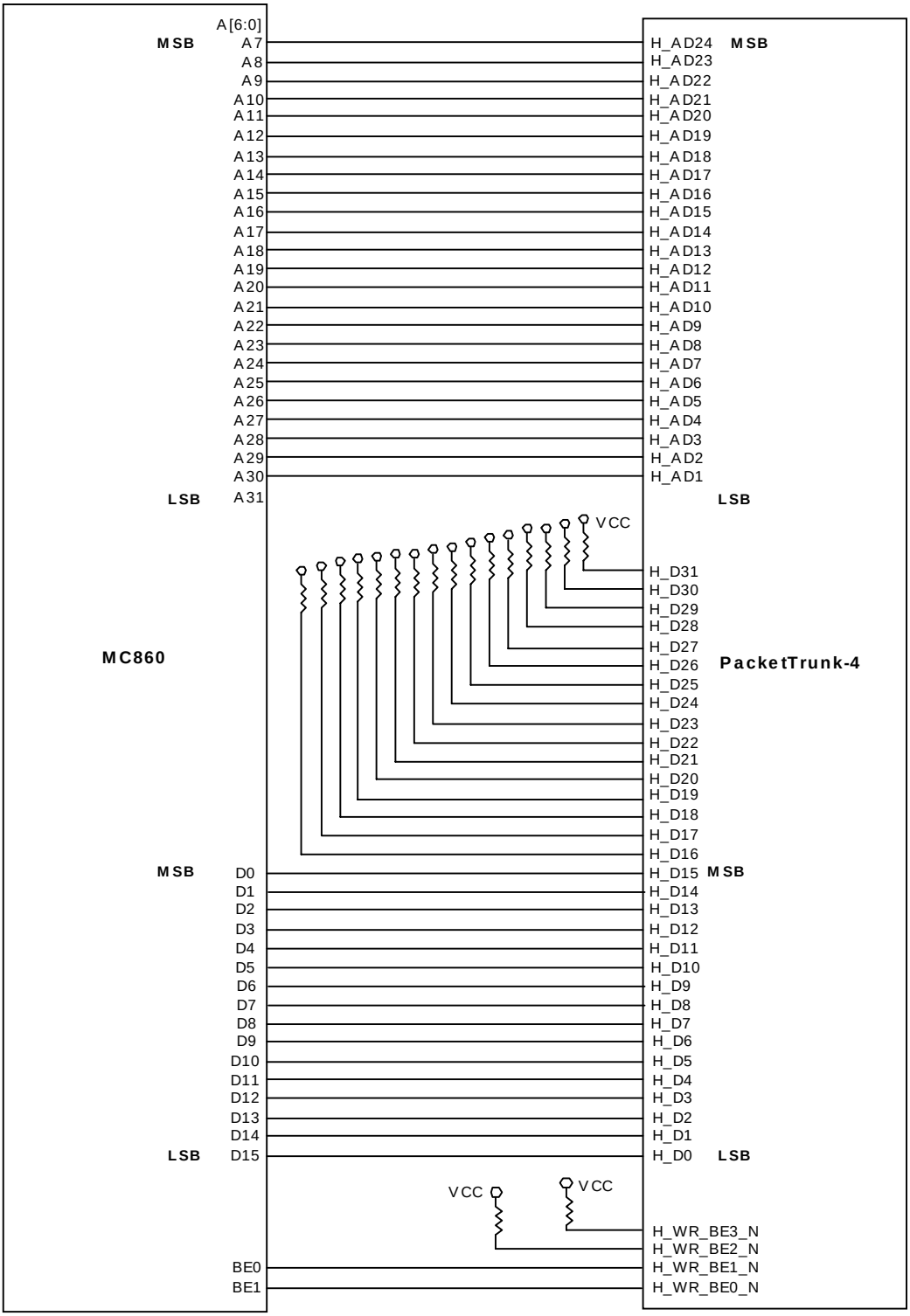


Figure 5-13. 16-Bit CPU Bus Connections

Note: Unused data bus pins (H_D[31:16]) of the PacketTrunk-4 should be pulled up by using external resistors. H_WR_BE2_N and H_WR_BE3_N should be pulled up (can be tied to Vcc directly).

Connecting the H_READY_N Signal

The PacketTrunk-4 H_READY_N output should be connected to the MPC860 TA input. However, the connection is different depending whether the PacketTrunk-4 works in synchronous or asynchronous mode.

In synchronous mode the connection is straight, as depicted below.

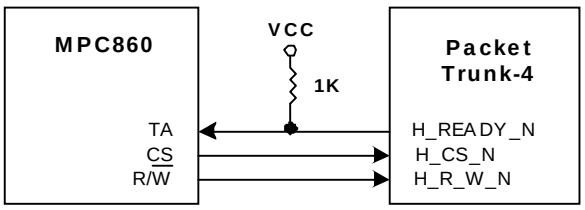


Figure 5-14. Synchronous Mode – Connection of H_READY_N to MPC860 TA

Note: The designer should take care for the timing budget of his design to meet the Tsu/Th requirements of the MPC860 and PacketTrunk-4. If these requirements cannot be met, asynchronous mode should be chosen (even though the same clock is driving the host and the PacketTrunk-4).

In asynchronous mode, since the TA of the MPC860 is a synchronous input (i.e., needs to meet Tsu/Th), the designer should synchronize H_READY_N to the MPC860 clock by means of a CPLD, which uses the MPC860 reference clock. The internal logic in the CPLD also uses the MPC860 CS output. Both H_READY_N output and MPC860 TA input should have a 1 k pull-up resistor.

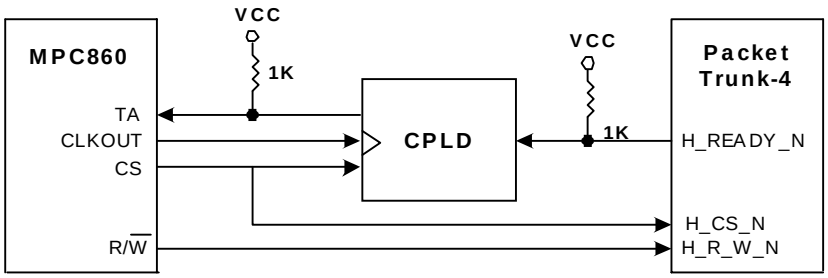


Figure 5-15. Asynchronous Mode – Connection of H_READY_N to MPC860 TA

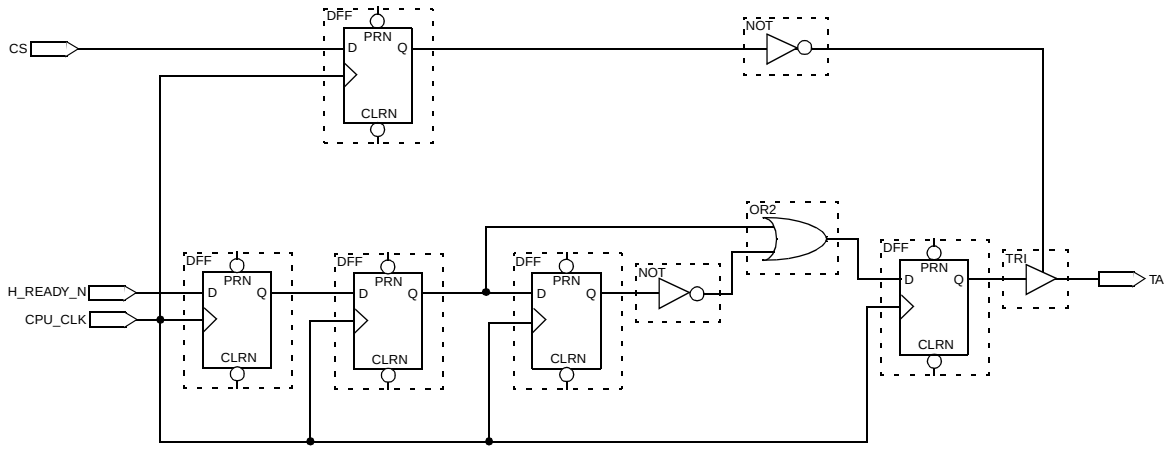


Figure 5-16. Internal CPLD Logic, Synchronizing H_READY_N to the MPC860 Clock

Note: Another alternative for connecting the H_READY_N in asynchronous mode is using the MPC860 UPM. In this option the H_READY_N output should be connected to the MPC860 UPWAIT (GPL4) signal, and no external timing adjustment is needed. The H_READY_N output should have a 1 kΩ pull-up resistor. Refer to the MPC860 user manual for additional details.

5.6 CONNECTING SDRAM DEVICES

The following table lists suggested SDRAM devices to use in conjunction with the PacketTrunk-4:

Table 5-1. List of Suggested SDRAM Devices

Vendor	64 Mb Device	128 Mb Device
Micron	MT48LC2M32B2TG-6	MT48LC4M32B2TG-6
Samsung	K4S643232F-TC/L60	K4S283232E-TC/L60
Hynix	HY57V653220BTC-6 or HY57V643220CT-6	HY57V283220T-6
Elpida	N/A	EDS1232AATA-60
Winbond	W986432DH-6	N/A
ICSI	IC42S32200/L-6T or IC42S32200/L-6TI	N/A

When connecting the PacketTrunk-4 to an external SDRAM, it is advised to connect SD_CLK through a serial termination resistor.

When connecting the PacketTrunk-4 to a 64 Mb external SDRAM, it is advised to connect SD_A[11] through a serial resistor (that will not be connected) to the SDRAM “NC” pin, that is A11 in a 128 Mb SDRAM. Thus, the 128 Mb SDRAM can be used on the same board, if needed.

ORDERING INFORMATION

Part Number: TXC-05870AIBG 256-lead plastic ball grid array package

RELATED PRODUCTS

TXC-04236, EtherMap®-3 *Plus* Device (Ethernet into STS-3/STM-1 SONET/SDH Mapper). The EtherMap-3 *Plus* is a highly integrated device that provides for mapping of 10/100/1000 Mbit/s Ethernet into SONET/SDH STS-3/STM-1 Transport payloads. The device supports connection for up to eight 10/100 Mbit/s Ethernet ports, using SMII interfaces, or a single 1000 Mbit/s Ethernet port, using a GMII interface. In the transmit direction, for each port, received Ethernet frames are encapsulated using either GFP, LAPS, LAPF or PPP protocol.

TXC-06830, TEPro VLSI Device (Channelized DS3 Access Solution). TEPro is a system-on-chip (SoC) device that supports the requirements of next-generation channelized DS3/E1 access systems.

TXC-06840, Envoy-8FE Device (Octal Fast Ethernet Controller). The Envoy-8FE Device is a Serial Media Independent Interface (SMII) to POS-PHY Level 2/3 interface converter transporting Ethernet packets. Envoy-8FE has 8 SMII ports. Packet data from the 8 ports are aggregated onto the POS-PHY interface.

STANDARDS DOCUMENTATION SOURCES

Telecommunication technical standards and reference documentation may be obtained from the following organizations:

ANSI (U.S.A.):

American National Standards Institute
25 West 43rd Street
New York, New York 10036
Tel: (212) 642-4900
Fax: (212) 398-0023
Web: www.ansi.org

The ATM Forum (U.S.A., Europe, Asia):

404 Balboa Street
San Francisco, CA 94118
Tel: (415) 561-6275
Fax: (415) 561-6120
Web: www.atmforum.com

ATM Forum Europe Office

Kingsland House - 5th Floor
361-373 City Road
London EC1 1PQ, England
Tel: 20 7837 7882
Fax: 20 7417 7500

ATM Forum Asia-Pacific Office

Hamamatsucho Suzuki Building 3F
1-2-11, Hamamatsucho, Minato-ku
Tokyo 105-0013, Japan
Tel: 3 3438 3694
Fax: 3 3438 3698

Bellcore (See Telcordia)

CCITT (See ITU-T)

EIA (U.S.A.):

Electronic Industries Association
Global Engineering Documents
15 Inverness Way East
Englewood, CO 80112
Tel: (800) 854-7179 (within U.S.A.)
Tel: (303) 397-7956 (outside U.S.A.)
Fax: (303) 397-2740
Web: www.global.ihs.com

ETSI (Europe):

European Telecommunications Standards Institute
650 route des Lucioles
06921 Sophia-Antipolis Cedex, France
Tel: 4 92 94 42 00
Fax: 4 93 65 47 16
Web: www.etsi.org

GO-MVIP (U.S.A.):

The Global Organization for Multi-Vendor Integration Protocol (GO-MVIP)
3220 N Street NW, Suite 360
Washington, DC 20007
Tel: (800) 669-6857 (within U.S.A.)
Tel: (903) 769-3717 (outside U.S.A.)
Fax: (903) 769-3818
Web: www.mvip.org

IEEE (Corporate Office):

American Institute of Electrical Engineers
3 Park Avenue, 17th Floor
New York, New York 10016-5997 U.S.A.

Tel: (212) 419-7900 (within U.S.A.)
Tel: (800) 678-4333 (Members only)
Fax: (212) 752-4929
Web: www.ieee.org

ITU-T (International):

Publication Services of International Telecommunication Union
Telecommunication Standardization Sector
Place des Nations, CH 1211
Geneve 20, Switzerland

Tel: 22 730 5852
Fax: 22 730 5853
Web: www.itu.int

JEDEC (International):

Joint Electron Device Engineering Council
2500 Wilson Boulevard
Arlington, VA 22201-3834

Tel: (703) 907-7559
Fax: (703) 907-7583
Web: www.jedec.org

MIL-STD (U.S.A.):

DODSSP Standardization Documents Ordering Desk
Building 4 / Section D
700 Robbins Avenue
Philadelphia, PA 19111-5094

Tel: (215) 697-2179
Fax: (215) 697-1462
Web: www.dodssp.daps.mil

PCI SIG (U.S.A.):

PCI Special Interest Group
5440 SW Westgate Dr., #217
Portland, OR 97221

Tel: (800) 433-5177 (within U.S.A.)
Tel: (503) 291-2569 (outside U.S.A.)
Fax: (503) 297-1090
Web: www.pcisig.com

Telcordia (U.S.A.):

Telcordia Technologies, Inc.
Attention - Customer Service
8 Corporate Place Rm 3A184
Piscataway, NJ 08854-4157

Tel: (800) 521-2673 (within U.S.A.)
Tel: (732) 699-2000 (outside U.S.A.)
Fax: (732) 336-2559
Web: www.telcordia.com

TTC (Japan):

TTC Standard Publishing Group of the Telecommunication Technology Committee
Hamamatsu-cho Suzuki Building
1-2-11, Hamamatsu-cho, Minato-ku
Tokyo 105-0013, Japan

Tel: 3 3432 1551
Fax: 3 3432 1553
Web: www.ttc.or.jp

- NOTES -

- NOTES -

TranSwitch Corporation • 3 Enterprise Drive • Shelton, CT 06484 USA • Tel: 203-929-8810 • Fax: 203-926-9453 • www.transwitch.com