PMC-SIERRA

PM73124 / PM73123

AAL1GATOR-4/8

ATM Adaptation Layer 1 Segmentation and Reassembly Processor-4/8

Data Sheet

Proprietary and Confidential

Released

Issue No. 1: June 2002

Legal Information

Copyright

Copyright 2002 PMC-Sierra, Inc. All rights reserved.

The information in this document is proprietary and confidential to PMC-Sierra, Inc., and for its customers' internal use. In any event, no part of this document may be reproduced or redistributed in any form without the express written consent of PMC-Sierra, Inc.

PMC-2000098 (P1)

Disclaimer

None of the information contained in this document constitutes an express or implied warranty by PMC-Sierra, Inc. as to the sufficiency, fitness or suitability for a particular purpose of any such information or the fitness, or suitability for a particular purpose, merchantability, performance, compatibility with other parts or systems, of any of the products of PMC-Sierra, Inc., or any portion thereof, referred to in this document. PMC-Sierra, Inc. expressly disclaims all representations and warranties of any kind regarding the contents or use of the information, including, but not limited to, express and implied warranties of accuracy, completeness, merchantability, fitness for a particular use, or non-infringement.

In no event will PMC-Sierra, Inc. be liable for any direct, indirect, special, incidental or consequential damages, including, but not limited to, lost profits, lost business or lost data resulting from any use of or reliance upon the information, whether or not PMC-Sierra, Inc. has been advised of the possibility of such damage.

Trademarks

PMC-Sierra is a registered trademark of PMC-Sierra, Inc. PMC-Sierra. Other product and company names mentioned herein may be the trademarks of their respective owners.

Patents

Granted

The technology discussed in this document is protected by one or more of the following patent grants:

U.S. Patent No. 5,844,901

Other relevant patent grants may also exist.

Contacting PMC-Sierra

PMC-Sierra
8555 Baxter Place
Burnaby, BC
Canada V5A 4V7

Tel: +1 (604) 415-6000
Fax: +1 (604) 415-6200

Document Information: document@pmc-sierra.com
Corporate Information: info@pmc-sierra.com
Technical Support: apps@pmc-sierra.com
Web Site: http://www.pmc-sierra.com

Revision History

Issue No.	Issue Date	Details of Change
1	June 2002	Created data sheet from issue 3 of eng doc.

Table of Contents

List of Registers

List of Figures

List of Tables

1 Features

The AAL1 Segmentation And Reassembly (SAR) Processor (AAL1gator-4/8) is a monolithic single chip device that provides DS1, E1, E3, or DS3 line interface access to an ATM Adaptation Layer One (AAL1) Constant Bit Rate (CBR) ATM network. It arbitrates access to an external SRAM for storage of the configuration, the user data, and the statistics. The device provides a microprocessor interface for configuration, management, and statistics gathering. PMC-Sierra also offers a software device control package for the AAL1gator-4/8 device.

- Compliant with the ATM Forum's Circuit Emulation Services (CES) specification (AF-VTOA-0078), and the ITU-T I.363.1

- Provides AAL1 segmentation and reassembly of 8/4 individual E1 or T1 lines, 1 H-MVIP lines at 8 MHz, or 1 E3 or DS3 or STS-1 unstructured lines.

- Provides a standard UTOPIA level 2 Interface which optionally supports parity and runs up to 52 MHz. Only Cell Level Handshaking is supported. In MPHY mode, can act like a single port or 4 port device. The following modes are supported:
  o 8/16-bit Level 2, Multi-Phy Mode (MPHY)
  o 8/16-bit Level 1, SPHY
  o 8-bit Level 1, ATM Master

- Provides an optional 8/16-bit Any-PHY slave interface.

- Supports up to 256(A8)/128(A4) Virtual Channels (VC).

- Supports n x 64 (consecutive channels) and m x 64 (non-consecutive channels) structured data format.

- Provides transparent transmission of Common Channel Signaling (CCS) and Channel Associated Signaling (CAS). Provides for termination of CAS signaling.

- Allows the CAS nibble to be coincident with either the first or second nibble of the data.

- Provides per-VC data and signaling conditioning in the transmit cell direction and per DS0 data and signaling conditioning in the transmit line direction. Data and signaling conditioning can be individually enabled. Includes DS3 AIS conditioning support in both directions. Transmit line conditioning options include programmable byte pattern, pseudo-random pattern or old data. Conditioning automatically occurs on underruns.

- In Cell Transmit direction, provides per-VC configuration of time slots allocated, CAS signaling support, partial cell size, data and signaling conditioning, ATM Cell header definition. Generates AAL1 sequence numbers, pointers and SRTS values in accordance with ITU-T I.363.1. Multicast connections are supported.

- In Cell Transmit direction provides counters for:
  o Conditioned cells transmitted for each queue
  o Cells which were suppressed for each queue
  o Total number of cells transmitted for each queue

- In Cell Receive direction, provides per-VC configuration of time slots allocated, CAS signaling support, partial cell size, sequence number processing options, cell delay variation tolerance buffer depth, maximum buffer depth. Processes AAL1 headers in accordance with ITU-T I.363.1.

- In Cell Receive direction, supports the Fast Sequence Number processing algorithm on all types of connections and Robust Sequence Number processing on Unstructured Data Format (UDF) connections. Cells are inserted/dropped to maintain bit integrity on lost or misinserted cells. Bit integrity is maintained through any single errored cell or up to six lost cells. Bit integrity can also optionally be maintained even if an underrun occurs. Pointer bytes, signaling bytes are taken into account. Cell insertion options include a programmable single byte pattern, pseudo-random data, or old data .

- In Cell Receive direction provides counters for the following events which include all counters required by the ATM Forum's CES-IS 2.0 MIB:
  - Incorrect sequence numbers per queue
  - Incorrect sequence number protection fields per queue
  - Total number of received cells per queue
  - Total number of dropped cells per queue
  - Total number of underruns per queue
  - Total number of lost cells per queue
  - Total number of overruns per queue
  - Total number of reframes per queue
  - Total number of pointer parity errors per queue
  - Total number of misinserted cells per queue
  - Total number of OAM or non-data cells received
  - Total number of OAM or non-data cells dropped.

- For each receive queue the following sticky bits are maintained:
  - Cell received
  - Structured pointer rule error detected
  - Cell dropped due to blank allocation table
  - Cells dropped due to pointer search
  - Cell dropped due to forced underrun
  - Cell dropped due to sequence number processing algorithm
  - Valid pointer was received
  - Pointer parity error detected
  - SRTS resume from an underrun condition
  - SRTS underrun occurred
  - Resume occurred from an underrun condition
  - Pointer reframe occurred
  - Overrun condition detected
  - Cell received while in an underrun

- Supports AAL0 mode, selectable on a per VC basis.

- Provides system side loopback support. When enabled and the incoming VCI matches the programmable loopback VCI, the cell received on the Receive UTOPIA interface is looped back to the Transmit UTOPIA interface. Alternatively the UTOPIA interface can be put into remote loopback mode where all incoming cells are looped back out. Provides line side loopback, enabled on a per queue basis, which can loop a single channel or any group of channels which can be mapped to a single queue.

- Provides a patented frame based calendar queue service algorithm with anti-clumping add-queue mechanism that produces minimal Cell Delay Variation (CDV). In UDF mode uses non-frame based scheduling to optimize CDV. In addition, four internal cell generation engines work in parallel to further insure low CDV.

- Queues re added by making entries into an add-queue FIFO to minimize queue activation overhead. An offset can be configured when queue is added to distribute cell build times to minimize CDV due to clumping.

- Provides single maskable, open-collector interrupt with master interrupt register to facilitate interrupt processing. The master interrupt register indicates the following conditions each of which can be masked:
  - Error/status condition with one of four AAL1 blocks
  - Ram parity error
  - UTOPIA parity error
  - Transmit UTOPIA FIFO is full
  - Transmit UTOPIA transfer error
  - UTOPIA loopback FIFO is full
  - UTOPIA runt cell is detected

- For each AAL1 block the following conditions can cause an interrupt, each of which can be masked. Separate 64 entry FIFOs per AAL1 block are used to track receive and transmit status.
  - A receive queue sticky bit was just set (individual mask per sticky bit)
  - Receive queue entered underrun state
  - Receive queue exited underrun state
  - Receive Status FIFO overflow
  - Transmit Frame Advance FIFO full
  - Reception of OAM cells
  - CAS change occurred
  - Line frame resync event
  - Transmit ATM Layer Processor (TALP) FIFO full

- Provides a 16-bit microprocessor interface to internal registers, and an external 128K x 16(18) (10 ns) Pipelined Single-Cycle Deselect Synchronous SRAMs, or Synchronous ZBT SRAMs.

- Provides a transmit buffer which can be used for Operations, Administration and Maintenance (OAM) cells as well as any other user-generated cells such as AAL5 cells for ATM signaling. A corresponding receive buffer exists for the reception of OAM cells or non-AAL1 data cells.

- Integrates an internal T1/E1 SRTS Clock recovery circuit per T1/E1 line.

- Integrates an internal T1/E1 Adaptive Clock recovery circuit per T1/E1 line.

- Includes an internal E1/T1 clock synthesizer for each line which can generate a nominal E1/T1 clock or be controlled via Synchronous Residual Time Stamp (SRTS) clock recovery method in Unstructured Data Format (UDF) mode or a programmable weighted moving average adaptive clocking algorithm. DS3 and E3 SRTS or adaptive clocking is supported using an external clock synthesizer and the clock control port.

- The clock synthesizers can also be controlled externally to provide customization of SRTS or adaptive algorithms. SRTS can also be disabled via a hardware input. Adaptive and SRTS information is output to a port for external processing for both low speed and high speed mode, if needed. Buffer depth is provided in units of bytes. The synthesizer can be set to 256 discrete frequencies between either +/-100 ppm for E1 or +/-200 ppm for T1.

- Low-power 2.5 Volt CMOS technology with 3.3 Volt, 5 Volt tolerant I/O.

- 324-pin fine-pitch plastic ball grid array (PBGA) package.

2 Applications

- Multi-service ATM Switch

- ATM Access Concentrator

- Digital Cross Connect

- Computer Telephony Chassis with ATM infrastructure

- Wireless Local Loop Back Haul

- ATM Passive Optical Network Equipment

3 References

Applicable Recommendations and Standards.

1. ANSI T1 Recommendation T1.403, Network-to-Customer Installation – DS1 Metallic Interface, NY, NY, 1995.

2. ANSI T1 Recommendation T1.630, Broadband ISDN-ATM Adaptation Layer for Constant Bit Rate Services, Functionality and Specification, NY, NY, 1993.

3. ATM Forum, ATM User Network Interface (UNI) Specification, V 3.1, Foster City, CA USA, September 1994.

4. ATM Forum, Circuit Emulation Service – Interoperability Specification (CES-IS), V. 2.0, Foster City, CA USA, August 1996.

5. ATM Forum, UTOPIA, an ATM-PHY Layer Specification, Level 1, V. 2.01, Foster City, CA USA, March 1994.

6. ATM Forum, UTOPIA, an ATM-PHY Layer Specification, Level 2, V. 1.0, Foster City, CA USA, June 1995.

7. ITU-T Recommendation G.703, Physical/Electrical Characteristics of Hierarchical Digital Interfaces, April 1991.

8. ITU-T Recommendation I.363.1, B-ISDN ATM Adaptation Layer (AAL) Specification, July 1995.

9. ITU-T Recommendation G.823, The Control of Jitter and Wander within Digital Networks Which Are Based on the 2048 kbit/s Hierarchy, March 1993.

10. ITU-T Recommendation G.824 The Control of Jitter and Wander within Digital Networks Which Are Based on the 1544 kbit/s Hierarchy, March 1993.

11. PMC-971268, "High density T1/E1 framer with integrated VT/TU mapper AND M13 multiplexer" (TEMUX), 2000, Issue 5.

12. GO-MVIP, "MVIP-90 Standard" Release 1.1, October 1994.

13. GO-MVIP, "H-MVIP Standard" Release 1.1a, January 1997.

4 Application Examples

An essential function for ATM networks is to emulate existing Time Division Multiplexing (TDM) circuits. Since most voice and data services are currently provided by TDM circuits, seamless interworking between TDM and ATM has become a system requirement. The ATM Forum has standardized an internetworking function that satisfies this requirement in the Circuit Emulation Service (CES) Specification. The AAL1gator-4/8 is a direct implementation of that service specification in silicon, including the complex Nx64 channelized service and support of CAS.

4.1 ATM Multi-service Switch

An Integrated Access Device (IAD) consolidates voice, data, Internet, and video wide-area network services using ATM over shared T1/E1 lines. IADs can also unify the functions of many different types of equipment including CSUs, DSUs and multiplexers. Figure 1 shows the AAL1gator-4/8 connected to PM4354 COMET-QUADs, a PM7329 S/UNI-APEX-1K800 Traffic Manager, a PM7328 S/UNI-ATLAS-1K800 ATM Layer device and the PM7347 S/UNI-JET.

Figure 1   AAL1gator-4/8 in an Integrated Access Device (IAD) Application.
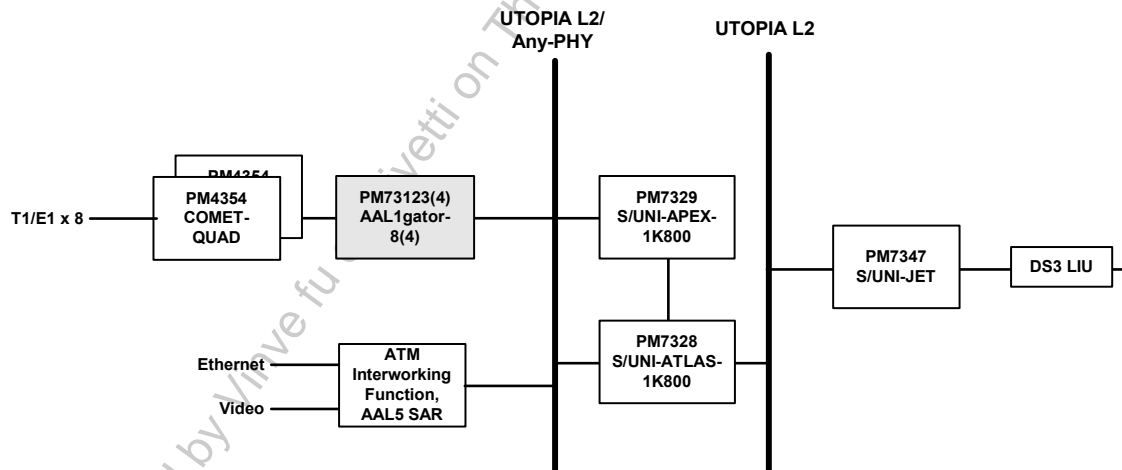
4.2 ATM Passive Optical Networks (APON)

The general architecture of a Passive Optical Network (PON) access network consists of two key elements: the Optical Line Termination (OLT) and the Optical Network Unit (ONU). The OLT is connected to the ONU through a point-to-multipoint Passive Optical Network that consists of fiber, splitters and other passive components. Typically, up to 32 ONUs are connected to a single OLT, depending on the splitting factor. OLTs are typically located in local exchanges and ONUs on street locations, in buildings or even in homes. Figure 2 shows the use of the AAL1gator-4/8 in an ONU application supporting CES functions.

Figure 2   AAL1gator-4/8 in an APON ONU Application

Release
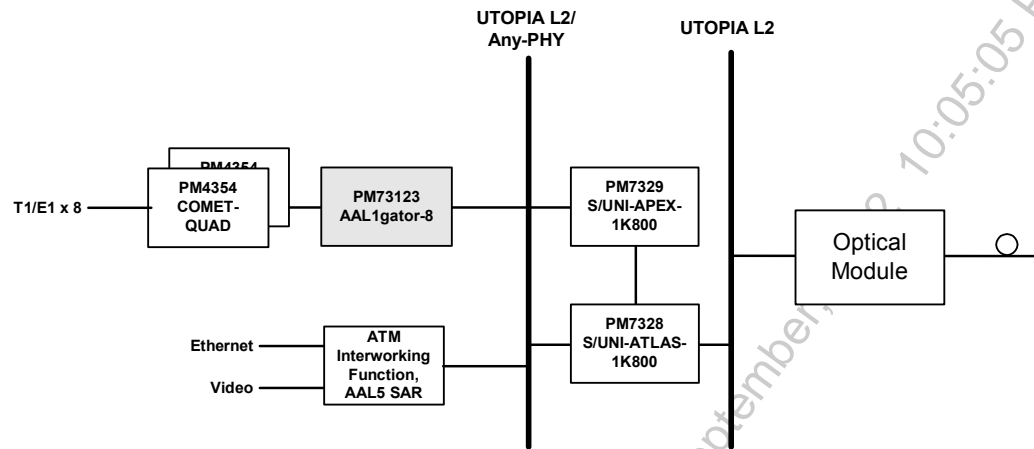
5   Block Diagram

The AAL1gator-4/8 contains an AAL1 SAR Processor block (A1SP) which work in parallel. The A1SP block interfaces to a common UTOPIA interface on one side and a Line Interface block on the other side which can be configured to support several different line protocols. The A1SP block connects to the ram interface. The processor Interface block which also contains the external clock control interface is shared by all blocks. The AAL1gator-4 has 4 serial lines.

Figure 3   AAL1gator-4/8 Internal Block Diagram
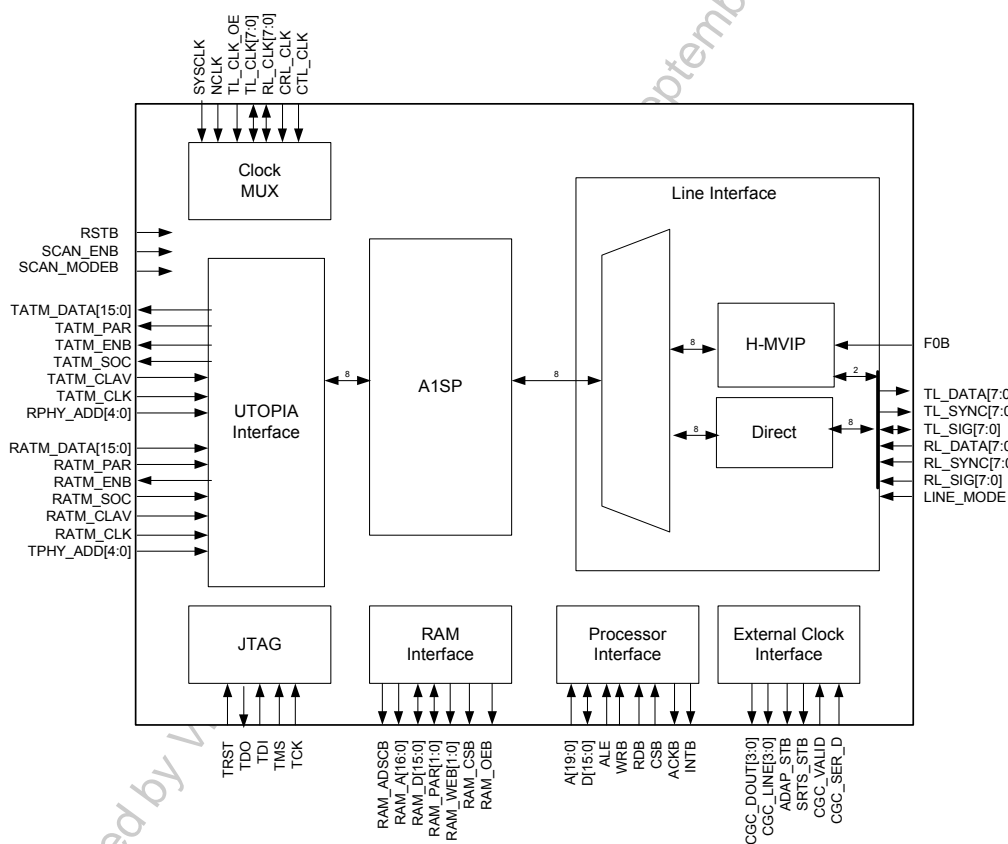
6   Description

The AAL1 Segmentation And Reassembly (SAR) Processor (AAL1gator-4/8) is a monolithic single chip device that provides DS1, E1, E3, or DS3 line interface access to an ATM Adaptation Layer One (AAL1) Constant Bit Rate (CBR) ATM network. It arbitrates access to an external SRAM for storage of the configuration, the user data, and the statistics. The device provides a microprocessor interface for configuration, management, and statistics gathering. PMC-Sierra also offers a software device control package for the AAL1gator-4/8 device.

PMC-SIERRA

7 Pin Diagram

The AAL1gator-4/8 is manufactured in a 324 pin, fine pitch, plastic ball grid array (PBGA) package. (23mm x 23 mm). Note that all center pins are thermal ground pins and should be connected to Ground.

Bottom View of AAL1gator-4

	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	
A	PQH	nc	nc	nc	PPH	RAM_D[15]	RAM_OEB	RAM_D[10]	SCAN_ENB	RAM_D[4]	PPL	RAM_WEB[1]	RAM_PAR[0]	RAM_ADDR[14]	RAM_ADDR[10]	PCH	RAM_ADDR[6]	RAM_ADDR[3]	RAM_ADDR[1]	TMS	SYSCLK	NC	**A**
B	nc	nc	nc	nc	LINE_MODE	RAM_D[9]	RAM_D[14]	RAM_D[11]	RAM_D[8]	RAM_D[5]	RAM_D[1]	RAM_ADDR[8]	RAM_WEB[0]	RAM_ADDR[13]	RAM_CSB	RAM_ADDR[7]	RAM_ADDR[4]	TDO	PPL	TCLK	NC	TATM_DATA[14]	**B**
C	nc	nc	nc	PPL	nc	PPL	CTL_CLK	RAM_D[13]	PCH	RAM_D[6]	RAM_D[2]	PCL	RAM_ADDR[16]	RAM_ADDR[12]	RAM_ADDR[9]	RAM_ADDR[5]	RAM_ADDR[2]	RAM_ADDR[0]	TDI	PPL	TATM_DATA[15]	RPHY_ADD_RSX	**C**
D	PQL	nc	nc	PCL	CRL_CLK	RAM_D[12]	PQH	PPL	RAM_D[7]	RAM_D[3]	RAM_D[0]	PPH	RAM_PAR[1]	RAM_ADDR[15]	PPL	RAM_ADDR[11]	RAM_ADDR_SCB	PQL	PPH	TATM_PAR	TATM_DATA[13]	TATM_DATA[11]	**D**
E	nc	nc	nc	nc														PCH	TATM_DATA[12]	TATM_DATA[10]		PPH	**E**
F	nc	nc	PPH	nc														TATM_CLK	TATM_DATA[9]	TATM_DATA[8]		PCL	**F**
G	nc	nc	nc	PPL														PPL		RPHY_ADD[2]	RPHY_ADD[3]	RPHY_ADD[1]	**G**
H	TL_CLK[4]	nc	nc	nc														TATM_DATA[7]	TATM_ENB	TATM_SOC		RPHY_ADD[0]	**H**
J	RL_CLK[4]	nc	nc	nc					GND	GND	GND		GND	GND				TATM_CLAV	TATM_DATA[6]	TATM_DATA[5]		TATM_DATA[4]	**J**
K	TL_CLK[3]	TL_SYNC[3]	nc	TL_SIG[3]					GND	GND	GND	GND	GND	GND				TATM_DATA[0]	PQL	TATM_DATA[2]		TATM_DATA[1]	**K**
L	RL_CLK[3]	PCL	TL_DATA[3]	PPH						GND	GND	GND	GND	GND				nc	TATM_DATA[3]	RATM_DATA[0]		PPH	**L**
M	RL_SIG[3]	PCH	RL_SYNC[3]	PPL					GND		GND	GND	GND	GND				PPL	RATM_DATA[2]	RATM_ENB		RATM_DATA[1]	**M**
N	TL_SYNC[2]	RL_CLK[2]	TL_SIG[2]	RL_DATA[3]					GND		GND	GND	GND	GND					RATM_DATA[3]	RATM_DATA[6]	RATM_DATA[5]	RATM_DATA[4]	**N**
P	RL_SIG[2]	RL_CLK[2]	RL_SYNC[2]	RL_DATA[2]					GND	GND	GND	GND	GND	GND					RATM_DATA[7]	TPHY_ADD[0]	RATM_CLAV	RATM_SOC	**P**
R	TL_CLK[1]	PPH	TL_SIG[1]	RL_DATA[2]														PPH		RATM_PAR	TPHY_ADD[2]	RATM_CLK	**R**
T	RL_SIG[1]	RL_CLK[1]	RL_DATA[1]	TL_SYNC[1]															TPHY_ADD[1]	RATM_DATA[9]	PCL	TPHY_ADD[3]	**T**
U	RL_SYNC[1]	TL_CLK[0]	PCH	TL_DATA[1]														PCH		RATM_DATA[12]	RATM_DATA[11]	RATM_DATA[8]	**U**
V	PCL	PPL	TL_DATA[0]	TL_SYNC[0]															RATM_DATA[10]	RATM_DATA[15]	PPH	TPHY_ADD[4]	**V**
W	RESERVED_IN	PPL	RL_SYNC[0]	CGC_LINE[1]	CGC_VALID	PCH	PPH	INTB	CSB	A[0]	A[3]	D[0]	D[3]	A[7]	PPH	D[7]	A[10]	D[9]	PQH	SCAN_MODEB	A[19]	RATM_DATA[13]	**W**
Y	TL_SIG[0]	RL_SIG[0]	PPH	TRSTB	PQH	PPL	TL_CLK_OE	CGC_DOUT[1]	PCL	RDB	A[1]	PPL	A[6]	D[6]	A[9]	D[8]	A[13]	PPH	D[15]	A[12]	A[18]	RATM_DATA[14]	**Y**
AA	RL_CLK[0]	RSTB	SRTS_STBH	ADAP_STBH	CGC_LINE[2]	CGC_SERD	NCLK	CGC_DOUT[0]	ACKB	WRB	A[2]	PCH	A[5]	D[5]	A[8]	PCL	D[10]	A[14]	D[13]	PPL	PPL	A[17]	**AA**
AB	RL_DATA[0]	PQL	RESERVED_OUT	CGC_LINE[3]	CGC_LINE[0]	PPH	CGC_DOUT[3]	PPL	CGC_DOUT[2]	ALE	PPL	D[1]	A[4]	D[4]	PPL	D[2]	A[11]	D[11]	A[15]	D[12]	D[14]	A[16]	**AB**
	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	

Bottom View of AAL1gator-8

	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	
A	PQH	RL_CLK[7]	TL_CLK[7]	TL_SYNC[7]	PPH	RAM_D[15]	RAM_OEB	RAM_D[10]	SCAN_ENB	RAM_D[4]	PPL	RAM_WEB[1]	RAM_PAR[0]	RAM_ADDR[14]	RAM_ADDR[10]	PCH	RAM_ADDR[6]	RAM_ADDR[3]	RAM_ADDR[1]	TMS	SYSCLK	NC	A
B	TL_SYNC[6]	RL_SYNC[7]	RL_SIG[7]	TL_DATA[7]	LINE_MODE	RAM_D[9]	RAM_D[14]	RAM_D[11]	RAM_D[8]	RAM_D[5]	RAM_D[1]	RAM_ADDR[8]	RAM_WEB[0]	RAM_ADDR[13]	RAM_CSB	RAM_ADDR[7]	RAM_ADDR[4]	TDO	PPL	TCLK	NC	TATM_DATA[14]	B
C	RL_SIG[6]	RL_CLK[6]	RL_SYNC[6]	PPL	TL_SIG[7]	PPL	CTL_CLK	RAM_D[13]	PCH	RAM_D[6]	RAM_D[2]	PCL	RAM_ADDR[16]	RAM_ADDR[12]	RAM_ADDR[9]	RAM_ADDR[5]	RAM_ADDR[0]		TDI	PPL	TATM_DATA[15]	RPHY_ADD_RSX	C
D	PQL	TL_SIG[6]	RL_DATA[7]	PCL	CRL_CLK	RAM_D[12]	PQH	PPL	RAM_D[7]	RAM_D[3]	RAM_D[0]	PPH	RAM_PAR[1]	RAM_ADDR[15]	PPL	RAM_ADDR[11]	RAM_ADDR SCB	PQL	PPH	TATM_PAR	TATM_DATA[13]	TATM_DATA[11]	D
E	RL_DATA[6]	RL_CLK[6]	TL_DATA[6]	TL_CLK[5]														PCH		TATM_DATA[12]	TATM_DATA[10]	PPH	E
F	TL_DATA[5]	TL_SYNC[5]	PPH	RL_SYNC[5]														TATM_CLK		TATM_DATA[9]	TATM_DATA[8]	PCL	F
G	RL_CLK[5]	RL_SIG[5]	TL_SIG[5]	PPL														PPL		RPHY_ADD[2]	RPHY_ADD[3]	RPHY_ADD[1]	G
H	TL_CLK[4]	TL_SYNC[4]	RL_DATA[5]	TL_SIG[4]														TATM_DATA[7]		TATM_ENB	TATM_SOC	RPHY_ADD[0]	H
J	RL_CLK[4]	RL_SIG[4]	TL_DATA[4]	RL_SYNC[4]					GND	GND	GND	GND	GND	GND				TATM_CLAV		TATM_DATA[6]	TATM_DATA[5]	TATM_DATA[4]	J
K	RL_CLK[3]	TL_SYNC[3]	RL_DATA[4]	TL_SIG[3]					GND	GND	GND	GND	GND	GND				TATM_DATA[0]		PQL	TATM_DATA[2]	TATM_DATA[1]	K
L	RL_CLK[3]	PCL	TL_DATA[3]	PPH					GND	GND	GND	GND	GND	GND				nc		TATM_DATA[3]	RATM_DATA[0]	PPH	L
M	RL_SIG[3]	PCH	RL_SYNC[3]	PPL					GND	GND	GND	GND	GND	GND				PPL		RATM_DATA[2]	RATM_ENB	RATM_DATA[1]	M
N	TL_SYNC[2]	TL_CLK[2]	TL_SIG[2]	RL_DATA[3]						GND	GND	GND	GND	GND				RATM_DATA[3]		RATM_DATA[6]	RATM_DATA[5]	RATM_DATA[4]	N
P	RL_SIG[2]	RL_CLK[2]	RL_SYNC[2]	TL_DATA[2]					GND	GND	GND	GND	GND	GND				RATM_DATA[7]		TPHY_ADD[0]	RATM_CLAV	RATM_SOC	P
R	TL_CLK[1]	PPH	TL_SIG[1]	RL_DATA[2]														PPH		RATM_PAR	TPHY_ADD[2]	RATM_CLK	R
T	RL_SIG[1]	RL_CLK[1]	RL_DATA[1]	TL_SYNC[1]														TPHY_ADD[1]		RATM_DATA[9]	PCL	TPHY_ADD[3]	T
U	RL_SYNC[1]	TL_CLK[0]	PCH	TL_DATA[1]														PCH		RATM_DATA[12]	RATM_DATA[11]	RATM_DATA[8]	U
V	PCL	PPL	TL_DATA[0]	TL_SYNC[0]														RATM_DATA[10]		RATM_DATA[15]	PPH	TPHY_ADD[4]	V
W	RESERVED_IN	PPL	RL_SYNC[0]	CGC_LINE[1]	CGC_VALID	PCH	PPH	INTB	CSB	A[0]	A[3]	D[0]	D[3]	A[7]	PPH	D[7]	A[10]	D[9]	PQH	SCAN_MODEB	A[19]	RATM_DATA[13]	W
Y	TL_SIG[0]	RL_SIG[0]	PPH	TRSTB	PQH	PPL	TL_CLK_OE	CGC_DOUT[1]	PCL	RDB	A[1]	PPL	A[6]	D[6]	A[9]	D[8]	A[13]	PPH	D[15]	A[12]	A[18]	RATM_DATA[14]	Y
AA	RL_CLK[0]	RSTB	SRTS_STBH	ADAP_STBH	CGC_LINE[2]	CGC_SER_D	NCLK	CGC_DOUT[0]	ACKB	WRB	A[2]	PCH	A[5]	D[5]	A[8]	PCL	D[10]	A[14]	D[13]	PPL	PPL	A[17]	AA
AB	RL_DATA[0]	PQL	RESERVED_OUT	CGC_LINE[3]	CGC_LINE[0]	PPH	CGC_DOUT[3]	PPL	CGC_DOUT[2]	ALE	PPL	D[1]	A[4]	D[4]	PPL	D[2]	A[11]	D[11]	A[15]	D[12]	D[14]	A[16]	AB
	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	

8 Pin Description

8.1 UTOPIA Interface Signals (52)

Pin Name	Type	Pin No.	Function
Note signals have different meanings depending on whether the UTOPIA bus is in ATM master mode, PHY mode or Any-PHY mode. The mode is controlled by the UTOP_MODE and ANY-PHY_EN fields in the UI_SRC_CFG and UI_SNK_CFG registers. All outputs are tri-state when the chip is in reset or when UI_EN is disabled in the UI_COMN_CFG register. All outputs have a maximum output current (IMAX) = 8 mA.			
TATM_CLK/RPHY_CLK	Input	F4	**ATM**: Transmit UTOPIA ATM Layer Clock is the synchronization clock input for the TATM interface. **PHY**: Receive UTOPIA/Any-PHY PHY Layer Clock is the synchronization clock input for the RPHY interface Maximum frequency is 52 MHz.
TATM_SOC/RPHY_SOC /RSOP	Output	H2	**ATM**: Transmit UTOPIA ATM Layer Start-Of-Cell is an active high signal asserted by the AAL1gator-8 when TATM_D contains the first valid byte of the cell. **PHY**: Receive Any-PHY/UTOPIA PHY Layer Start-Of-Cell is an active high signal asserted by the AAL1gator-4/8 when RPHY_D[15:0] contains the first valid word of the cell. AAL1gator-4/8 drives this signal only when the ATM layer has selected it for a cell transfer. **Any-PHY**: This pin is the Receive Start of Packet (RSOP) signal which functions just like RPHY_SOC.

Pin Name	Type	Pin No.	Function
TATM_D[15]/RPHY_D[15] TATM_D[14]/RPHY_D[14] TATM_D[13]/RPHY_D[13] TATM_D[12]/RPHY_D[12] TATM_D[11]/RPHY_D[11] TATM_D[10]/RPHY_D[10] TATM_D[9]/RPHY_D[9] TATM_D[8]/RPHY_D[8] TATM_D[7]/RPHY_D[7] TATM_D[6]/RPHY_D[6] TATM_D[5]/RPHY_D[5] TATM_D[4]/RPHY_D[4] TATM_D[3]/RPHY_D[3] TATM_D[2]/RPHY_D[2] TATM_D[1]/RPHY_D[1] TATM_D[0]/RPHY_D[0]	Output	C2 B1 D2 E3 D1 E2 F3 F2 H4 J3 J2 J1 L3 K2 K1 K4	**ATM**: Transmit UTOPIA ATM Layer Data Bits 7 to 0 form the byte-wide data driven to the PHY layer. Bit 0 is the Least Significant Bit (LSB). Bit 7 is the Most Significant Bit (MSB) and is the first bit received for the cell from the serial line. Note that only the lower 8 bit of the bus are used in ATM master mode. **PHY**: Receive UTOPIA/Any-PHY PHY Layer Data Bits 15 to 0 form the word-wide data driven to the ATM layer. This bus is only driven when the ATM layer has selected the UI_SRC_INTF for a cell transfer. The upper byte is only used if 16_BIT_MODE is set in the UI_SRC_CFG register. Otherwise the upper byte is driven to 0's. Bit 0 is the LSB. Bit 7 is the MSB of the first byte and is the first bit received for the cell from the serial line.
TATM_PAR/ RPHY_PAR	Output	D3	**ATM**: Transmit UTOPIA ATM Layer Parity is a byte parity bit covering TATM_D(7:0). **PHY**: Receive UTOPIA/Any-PHY PHY Layer Parity is either a byte parity covering RPHY_D(7:0) or word parity covering RPHY_D(15:0) depending on the value of 16_BIT_MODE.
TATM_ENB/RPHY_ENB /RENB	Bidi	H3	**ATM**: Transmit UTOPIA ATM Layer Enable is an active low signal asserted by the AAL1gator-4/8 during cycles when TATM_D contains valid data. It is not asserted until the AAL1gator-4/8 is ready to send a full cell. **PHY**: Receive UTOPIA/Any-PHY PHY Layer Enable is an active low signal asserted by the ATM layer to indicate RPHY_D and RPHY_SOC will be sampled at the end of the next cycle. If UTOP_MODE in UI_SRC_CFG is set to UTOPIA Level 2 Mode then the AAL1gator-4/8 will drive data only if RPHY_ADD matches CFG_ADDR in the UI_SRC_ADD_CFG register the cycle before RPHY_ENB goes low. **Any-PHY:** This pin is the RENB input signal, which functions the same as RPHY_ENB. The only difference is that data is driven two cycles after selection instead of just one cycle.

Pin Name	Type	Pin No.	Function
TATM_CLAV/RPHY_CLAV /RPA	Bidi	J4	**ATM**: Transmit UTOPIA ATM Layer Cell Available is an active high signal from the PHY layer device to indicate that there is sufficient room to accept a cell. **PHY**: Receive UTOPIA/Any-PHY PHY Layer Cell Available is an active high signal asserted by the AAL1gator-4/8 to indicate it is ready to deliver a complete cell. In Utopia Level 2 mode, this signal is driven only when MPHY_ADD matches CFG_ADDR in the UI_SRC_ADD_CFG register in the previous cycle. A pulldown resistor is recommended. **Any-PHY**: This pin is the Receive Packet Available (RPA) signal which functions the same as RPHY_CLAV except for it is activated two cycles after a matching address instead of one.

Pin Name	Type	Pin No.	Function
RPHY_ADD[4]/RSX RPHY_ADD[3]/RCSB RPHY_ADD[2] RPHY_ADD[1] RPHY_ADD[0]	I/O Input Input Input Input	C1 G2 G3 G1 H1	**ATM**: These signals are not used in ATM mode. **PHY**: Receive UTOPIA PHY Layer Address (Bits 4 to 0) which selects the UTOPIA receiver. These inputs are used as an output enable for RPHY_CLAV and to validate the activation of RPHY_ENB. There are internal pull-up resistors. These pins are compared with CFG_ADDR[5:0] in the UI_SRC_CFG_ADDR register. Note that these are compared in both UTOPIA Level 1 and UTOPIA Level 2 modes. In UTOPIA level 1 it is expected that these pins will be tied low. **ANY-PHY:** Receive Start Transfer(RSX) is an active high output which indicates the start of an Any-PHY packet which identifies the location of the prepended address. ANY-PHY_EN in UI_SRC_CFG register needs to be set for this function. Receive Chip Select Bar (RCSB) is an active low input which is used to select the AAL1gator-4/8 when polling in Any-PHY mode. This input is used to decode any Any-PHY address bits greater than RPHY_ADD[2]. This input goes low one cycle after Any-PHY address is valid. ANY-PHY_EN and CS_MODE_EN in UI_SRC_CFG register needs to be set for this function. Otherwise this bit functions as RPHY_ADD[3]. RPHY_ADD[2:0] is the bottom three bits of the Any-PHY address and is used to select the device when polling. These pins are compared with CFG_ADDR[2:0] in the UI_SRC_CFG_ADDR register. Note these pins must be tied to ground when not used.
RATM_CLK/ TPHY_CLK	Input	R1	**ATM**: Receive UTOPIA ATM Layer Clock is the synchronization clock input for synchronizing the RATM interface. **PHY**: Transmit UTOPIA/Any-PHY PHY Layer Clock is the synchronization clock input for synchronizing the TPHY interface. Maximum frequency is 52 MHz.

Pin Name	Type	Pin No.	Function
RATM_SOC/ TPHY_SOC /TSOP	Input	P1	This signal has two definitions depending on whether the UTOPIA is in ATM mode or PHY mode. **ATM**: Receive UTOPIA ATM Layer Start-Of-Cell is an active high signal asserted by the PHY layer when RATM_D contains the first valid byte of a cell. **PHY**: Transmit UTOPIA/Any-PHY PHY Layer Start-Of-Cell is an active high signal asserted by the ATM layer when TPHY_D contains the first valid byte of a cell. **Any-PHY**: This pin is the Transmit Start of Packet (TSOP) signal which functions just like TPHY_SOC. . This signal is optional in this mode. If unused, tie low.
RATM_D[15]/TPHY_D[15] RATM_D[14]/TPHY_D[14] RATM_D[13]/TPHY_D[13] RATM_D[12]/TPHY_D[12] RATM_D[11]/TPHY_D[11] RATM_D[10]/TPHY_D[10] RATM_D[9]/TPHY_D[9] RATM_D[8]/TPHY_D[8] RATM_D[7]/TPHY_D[7] RATM_D[6]/TPHY_D[6] RATM_D[5]/TPHY_D[5] RATM_D[4]/TPHY_D[4] RATM_D[3]/TPHY_D[3] RATM_D[2]/TPHY_D[2] RATM_D[1]/TPHY_D[1] RATM_D[0]/TPHY_D[0]	Input	V3 Y1 W1 U3 U2 V4 T3 U1 P4 N3 N2 N1 N4 M3 M1 L2	**ATM**: Receive UTOPIA ATM Layer Data Bits 7 to 0 form the byte-wide data from the PHY layer device. Bit 0 is the LSB. Bit 7 is the MSB. This is the first bit of the cell, which will be transmitted on the serial line. The upper byte is not used in ATM mode. **PHY**: Transmit UTOPIA/Any-PHY PHY Layer Data Bits 15 to 0 form the word-wide data from the ATM layer device. Bit 0 is the LSB. Bit 7 is the MSB of the first byte. This is the first bit of the cell, which will be transmitted on the serial line. The upper byte is only used if 16_BIT_MODE is set in the UI_SNK_CFG register.
RATM_PAR/ TPHY_PAR	Input	R3	**ATM**: Receive UTOPIA ATM Layer Parity is a byte odd parity bit covering RATM_D(7:0) or word odd parity covering RATM_D(15:0) depending on the value of 16_BIT_MODE. **PHY**: Transmit UTOPIA/Any-PHY PHY Layer Parity is either a byte odd parity covering TPHY_D(7:0) or word odd parity covering TPHY_D(15:0) depending on the value of 16_BIT_MODE.

Pin Name	Type	Pin No.	Function
RATM_ENB/TPHY_ENB	Bidi	M2	**ATM**: Receive UTOPIA ATM Layer Enable is an active low signal asserted by the AAL1gator-4/8 to indicate RATM_D and RATM_SOC will be sampled at the end of the next cycle. It will not be asserted until the AAL1gator-4/8 is ready to receive a full cell. **PHY**: Transmit UTOPIA/Any-PHY PHY Layer Enable is an active low signal asserted by the ATM layer device during cycles when TPHY_D[15:0] contain valid data. The AAL1gator-4/8 will accept data only if TPHY_ADD matches CFG_ADDR in the UI_SNK_CFG register the cycle before TPHY_ENB goes low **Any-PHY**: This pin is the TENB input signal, which functions the same as TPHY_ENB.
RATM_CLAV/TPHY_CLAV	Bidi	P2	**ATM**: Receive UTOPIA ATM Layer Cell Available is an active high signal asserted by the PHY layer to indicate that there is a cell available to send. **PHY**: Receive UTOPIA/Any-PHY PHY Layer Cell Available is an active high signal asserted by the AAL1gator-4/8 to indicate there is a cell-space available. The AAL1gator-4/8 drives this signal only when TPHY_ADD matches CFG_ADDR in the UI_SNK_CFG register in the previous cycle. A pulldown resistor is recommended. **Any-PHY**: This pin is the Transmit Packet Available (TPA) signal which functions the same as TPHY_CLAV except for it is activated two cycles after a matching address instead of one.

Pin Name	Type	Pin No.	Function
TPHY_ADD[4]/TSX TPHY_ADD[3]/TCSB TPHY_ADD[2] TPHY_ADD[1] TPHY_ADD[0]	Input	V1 T1 R2 T4 P3	**ATM**: These signals are not used in ATM mode. **PHY**: Transmit UTOPIA PHY Layer Address (Bits 4 to 0) which selects the UTOPIA transmitter. These inputs are used as an output enable for TPHY_CLAV and to validate the activation of TPHY_ENB. There are internal pull-up resistors. These pins are compared with CFG_ADDR[5:0] in the UI_SNK_CFG_ADDR register. Note that these are compared in both UTOPIA Level 1 and UTOPIA Level 2 modes. In UTOPIA level 1 it is expected that these pins will be tied low. **ANY-PHY:** Transmit Start Transfer(TSX) is an active high input which indicates the start of an Any-PHY packet which identifies the location of the prepended address. ANY-PHY_EN in UI_SNK_CFG register needs to be set for this function. Transmit Chip Select Bar (TCSB) is an active low input which is used to select the AAL1gator-4/8 when polling in Any-PHY mode. This input is used to decode any Any-PHY address bits greater than TPHY_ADD[2]. This input goes low one cycle after Any-PHY address is valid. ANY-PHY_EN and CS_MODE_EN in UI_SNK_CFG register needs to be set for this function. Otherwise this bit functions as TPHY_ADD[3]. TPHY_ADD[2:0] is the bottom three bits of the Any-PHY address and is used to select the device when polling. These pins are compared with CFG_ADDR[2:0] in the UI_SNK_CFG_ADDR register. Note these pins must be tied to ground when not used.

8.2 Microprocessor Interface Signals (42)

Pin Name	Type	Pin No.	Function
D[15] D[14] D[13] D[12] D[11] D[10] D[9] D[8] D[7] D[6] D[5] D[4] D[3] D[2] D[1] D[0]	I/O	Y4 AB2 AA4 AB3 AB5 AA6 W5 Y7 W7 Y9 AA9 AB9 W10 AB7 AB11 W11	The bi-directional data signals (D[15:0]) provide a data bus to allow the AAL1gator-4/8 device to interface to an external micro-processor. Both read and write transactions are supported. The microprocessor interface is used to configure and monitor the AAL1gator-4/8 device. Maximum output current (IMAX) = 6 mA.
A[19] A[18] A[17] A[16] A[15] A[14] A[13] A[12] A[11] A[10] A[9] A[8] A[7] A[6] A[5] A[4] A[3] A[2] A[1] A[0]	Input	W2 Y2 AA1 AB1 AB4 AA5 Y6 Y3 AB6 W6 Y8 AA8 W9 Y10 AA10 AB10 W12 AA12 Y12 W13	The address signals (A[19:0]) provide an address bus to allow the AAL1gator-4/8 device to interface to an external micro-processor.
ALE	Input	AB13	The address latch enable signal (ALE) latches the A[19:0] signals during the address phase of a bus transaction. When ALE is set high, the address latches are transparent. When ALE is set low, the address latches hold the address provided on A[19:0]. ALE has an internal pull-up resistor.
WRB	Input	AA13	The write strobe signal (WRB) qualifies write accesses to the AAL1gator-4/8 device. When CSB is set low, the D[15:0] bus contents are clocked into the addressed register on the rising edge of WRB. Note that if CSB, WRB and RDB are all low, all chip outputs are tristated. Therefore WRB and RDB should never be active at the same time during functional operation.

Pin Name	Type	Pin No.	Function
RDB	Input	Y13	The read strobe signal (RDB) qualifies read accesses to the AAL1gator-4/8 device. When CSB is set low, the AAL1gator-4/8 device drives the D[15:0] bus with the contents of the addressed register on the falling edge of RDB. Note that if CSB, WRB and RDB are all low, all chip outputs are tristated. Therefore WRB and RDB should never be active at the same time during functional operation.
CSB	Input	W14	The chip select signal (CSB) qualifies read/write accesses to the AAL1gator-4/8 device. The CSB signal must be set low during read and write accesses. When CSB is set high, the microprocessor interface signals are ignored by the AAL1gator-4/8 device. If CSB is not required (register accesses controlled only by WRB and RDB) then CSB should be connected to an inverted version of the RSTB signal. Note that if CSB, WRB and RDB are all low, all chip outputs are tristated.
ACKB	Open-Drain Output	AA14	The ACKB is an active low signal which indicates when processor read data is valid or when a processor write operation has completed. When inactive this signal is tristated. ACKB is an open drain output and should be pulled high externally with a fast resistor. Maximum output current (IMAX) = 6 mA
INTB	Open-Drain Output	W15	The interrupt signal (INTB) is an active low signal indicating that an enabled bit in the MSTR_INTR_REG register was set. When INTB is set low, the interrupt is active and enabled. When INTB is tristate, there is no interrupt pending or it is disabled. INTB is an open drain output and should be pulled high externally with a fast resistor. Maximum output current (IMAX) = 6 mA

8.3 Ram Interface Signals(40)

Pin Name	Type	Pin No.	Function
RAM_D[15] RAM_D[14] RAM_D[13] RAM_D[12] RAM_D[11] RAM_D[10] RAM_D[9] RAM_D[8] RAM_D[7] RAM_D[6] RAM_D[5] RAM_D[4] RAM_D[3] RAM_D[2] RAM_D[1] RAM_D[0]	I/O	A17 B16 C15 D17 B15 A15 B17 B14 D14 C13 B13 A13 D13 C12 B12 D12	The bi-directional data signals (RAM_D[15:0]) provide a data bus to allow the AAL1gator-4/8 device to access an external 128Kx16(18) RAM. Maximum output current (IMAX) = 6 mA
RAM_A[16] RAM_A[15] RAM_A[14] RAM_A[13] RAM_A[12] RAM_A[11] RAM_A[10] RAM_A[9] RAM_A[8] RAM_A[7] RAM_A[6] RAM_A[5] RAM_A[4] RAM_A[3] RAM_A[2] RAM_A[1] RAM_A[0]	Output	C10 D9 A9 B9 C9 D7 A8 C8 B11 B7 A6 C7 B6 A5 C6 A4 C5	The address signals (RAM_A[16:0]) provide an address bus to allow the AAL1gator-4/8 device to address an external 128Kx16(18) RAM. Maximum output current (IMAX) = 6 mA.
RAM_OEB	Output	A16	RAM Output Enable is an active low signal that enables the SRAM to drive data. Maximum output current (IMAX) = 6 mA.
RAM_WEB[1]	Output	A11	RAM Write Enable One is an active low signal for the high-byte write. Maximum output current (IMAX) = 6 mA.
RAM_WEB[0]	Output	B10	RAM Write Enable Zero is an active low signal for the low-byte write. Maximum output current (IMAX) = 6 mA.
RAM_CSB	Output	B8	RAM Chip Select is an active low chip-select signal for external memory. Maximum output current (IMAX) = 6 mA.

Pin Name	Type	Pin No.	Function
RAM_ADSCB/RAM_R/WB	Output	D6	This signal has different meanings depending upon the type of SSRAM that the AAL1gator-4/8 is programmed to interface to. **Pipelined Single-Cycle Deselect SSRAM**: RAM Address Status Control is an active low output for external memory and is used to cause a new external address to be loaded into the RAM. **Pipelined ZBT SSRAM**: RAM R/W indicates the direction of the transfer. Maximum output current (IMAX) = 6 mA.
RAM_PAR[1] RAM_PAR[0]	I/O	D10 A10	RAM Parity is a two bit bi-directional signal that indicates odd parity for the upper and lower byte of RAM_D[15:0]. Maximum output current (IMAX) = 6 mA

NOTE: For different modes of the line interface the I/O is redefined.

For Direct mode there are 4/8 separate bi-directional lines which can support lines up to 15 Mbps each with an aggregate bandwidth of 20 Mbps.Or line 0 can be put into highspeed mode and support data rates up to 52 Mbps. For H-MVIP mode there is one/two 8 Mbps lines which is compatible with the H-MVIP specification.

Table 1 defines which signal tables need to be used for each possible mode. Select the mode of the line interface that will be used and refer to the tables listed. Table 1 shows how pins are shared between the different modes.

Table 1   Line Interface Signal Table Selection

Line Mode	Line Interface Table
Direct	Direct
H-MVIP	H-MVIP

8.4   Line Interface Signals(Direct)(67)

Pin Name	Type	Pin No.	Function
LINE_MODE	Input	B18	Determines the mode of operation for the line interface: 0)Direct Mode 1)H-MVIP Mode

Pin Name	Type	Pin No.	Function
TL_SYNC[7] TL_SYNC[6] TL_SYNC[5] TL_SYNC[4] TL_SYNC[3] TL_SYNC[2] TL_SYNC[1] TL_SYNC[0]	I/O	A19 B22 F21 H21 K21 N22 T19 V19	Transmit Line Synchronization 7 to 0 are the transmit frame synchronization indicators used in SDF-MF and SDF-FR modes. Depending on the value of MF_SYNC_MODE in the LI_CFG_REG register for the line, these signals either indicate a frame boundary or a multi-frame boundary. Depending on the value of GEN_SYNC in the LIN_STR_MODE register for that line, the sync signal is either received from the corresponding framer device 0 to 7 or it is generated internally The Default mode of this signal is to be a frame sync input. When the MVIP_EN bit is set in LS_Ln_CFG_REG then TL_SYNC[0] is the F0B pin; the common frame sync. Maximum output current (IMAX) = 6 mA. TL_SYNC[7:4] do not exist on AAL1gator-4.
TL_DATA[7] TL_DATA[6] TL_DATA[5] TL_DATA[4] TL_DATA[3] TL_DATA[2] TL_DATA[1] TL_DATA[0]	Output	B19 E20 F22 J20 L20 P19 U19 V20	Transmit Line Serial Data 7 to 0 carry the received data to the corresponding framer devices. Maximum output current (IMAX) = 6 mA. TL_DATA[7:4] do not exist on AAL1gator-4.
TL_SIG[7] TL_SIG[6] TL_SIG[5] TL_SIG[4] TL_SIG[3] TL_SIG[2] TL_SIG[1] TL_SIG[0]	Output	C18 D21 G20 H19 K19 N20 R20 Y22	Transmit Line Signal 7 to 0 are the CAS signaling outputs to the corresponding framer devices in SDF-MF mode. This is the default function for this pin. Maximum output current (IMAX) = 6 mA. TL_SIG[7:4] do not exist on AAL1gator-4.

Pin Name	Type	Pin No.	Function
TL_CLK[7]/TSM[7] TL_CLK[6]/TSM[6] TL_CLK[5]/TSM[5] TL_CLK[4]/TSM[4] TL_CLK[3]/TSM[3] TL_CLK[2]/TSM[2] TL_CLK[1]/TSM[1] TL_CLK[0]/TSM[0]	I/O	A20 C21 E19 H22 K22 N21 R22 U21	Transmit Line Channel Clock 7 to 0 are the clock lines for the sixteen lines. They clock the data from the AAL1gator-4/8 to the corresponding framer devices. Depending on the value of the TL_CLK_OE pin and the CLK_SOURCE_TX field in the LIN_STR_MODE memory register, these pins are either outputs or inputs. If TLCLK_OUTPUT_EN is high, these pins are outputs and the clock is sourced internally at power up. This can later be changed by the CLK_SOURCE_TX field. Note that if CLK_SOURCE_TX /= "000" then this pin is an output, even if it is not driving a clock. A clock will only be driven if in E1 or T1 mode and either the internal clock synthesizer is being used or the clock is being looped. CLK_SOURCE_TX = "001", "010, "011", "100", or "101") Note that if UDF_HS=1 in the HS_LIN_REG, TL_CLK[7:1] should be tied high. Transmit Signaling Mirror is a copy of the TL_SIG output. In Direct mode, if CLK_SOURCE_TX="111" then signaling is output on this pin. This option is used with devices that share the same pin for clock and signaling. In this mode CTL_CLK is used as the line clock. **If CTL_CLK is used, then the TL_CLK[7:0] output pins do not drive valid clock signals and should be ignored.** Maximum output current (IMAX) = 6 mA.
			TL_CLK[7:5] do not exist on AAL1gator-4. TL_CLK4 is only used as a test clock on AAL1gator-4 and should be left unconnected. There is an internal pullup on this pin.
CTL_CLK	Input	C16	Common Transmit Line Clock is a transmit line clock which can be shared across all lines. Whether this clock is used or not for a given line is dependent on the value of CLK_SOURCE_TX in the LINE_STR_MODE memory register for that line.
RL_SYNC[7] RL_SYNC[6] RL_SYNC[5] RL_SYNC[4] RL_SYNC[3] RL_SYNC[2] RL_SYNC[1] RL_SYNC[0]	Input	B21 C20 F19 J19 M20 P20 U22 W20	Receive Line Synchronization 7 to 0 are the receive frame synchronization indicators used in SDF-MF and SDF-FR modes. Depending on the value of MF_SYNC_MODE in the LI_CFG_REG register for the line, these signals either indicate a frame boundary or a multi-frame boundary. Tie to ground if unused. RL_SYNC[7:4] do not exist on AAL1gator-4.

Pin Name	Type	Pin No.	Function
RL_DATA[7] RL_DATA[6] RL_DATA[5] RL_DATA[4] RL_DATA[3] RL_DATA[2] RL_DATA[1] RL_DATA[0]	Input	D20 E22 H20 K20 N19 R19 T20 AB22	Receive Line Serial Data 7 to 0 carries the receive data from the corresponding framer devices. RL_DATA[7:4] do not exist on AAL1gator-4.
RL_SIG[7] RL_SIG[6] RL_SIG[5] RL_SIG[4] RL_SIG[3] RL_SIG[2] RL_SIG[1] RL_SIG[0]	Input	B20 C22 G21 J21 M22 P22 T22 Y21	Receive Line Signaling 7 to 0 carries the CAS data from the corresponding framer devices. Signaling is sampled during the last frame of a multi-frame for insertion into AAL1 cells. If CAS change detection is used, signaling must be static for the entire multiframe as signaling is sampled at multiple points during the multi-frame. RL_SIG[7:4] do not exist on AAL1gator-4.
RL_CLK[7] RL_CLK[6] RL_CLK[5] RL_CLK[4] RL_CLK[3] RL_CLK[2] RL_CLK[1] RL_CLK[0]	Input	A21 E21 G22 J22 L22 P21 T21 AA22	Receive Line Clock 7 to 0 is the clock received from the corresponding framer device used to clock in RL_DATA, RL_SIG, and RL_SYNC. RL_CLK[7:5] do not exist on AAL1gator-4. RL_CLK4 is only used as a test clock on AAL1gator-4 and should be left unconnected. There is an internal pullup on this pin.
CRL_CLK	Input	D18	Common Receive Line Clock is a receive line clock which can be shared across all lines. Whether this clock is used or not for a given line is dependent on the value of CLK_SOURCE_RX in the LIN_STR_MODE memory register for that line. When the MVIP_EN bit is set in LS_Ln_CFG_REG then this is the C4B input; the common 4.096 MHz clock.

8.5   Line Interface Signals(H-MVIP)(12)

Pin Name	Type	Pin No.	Function
LINE_MODE	Input	B18	Determines the mode of operation for the line interface: 0)Direct Mode 1)H-MVIP Mode
F0B	Input	V19	Frame Sync 0 is the active low frame synchronization input signal used to indicate the start of a frame.
TL_DATA[1] TL_DATA[0]	Output	U19 V20	Transmit Line Serial Data 1 to 0 carry the received data to the corresponding framer devices. or an H_MVIP backplane. For AAL1gator-4 only line [0] is supported. Maximum output current (IMAX) = 6 mA.

Pin Name	Type	Pin No.	Function
TL_SIG[1] TL_SIG[0]	Output	R20 Y22	Transmit Line Signal 1 to 0 are the CAS signaling outputs to the corresponding framer devices in SDF-MF mode. H-MVIP does not support signaling directly, but these signals can be used to transport signaling if needed. For AAL1gator-4 only line [0] is supported. Maximum output current (IMAX) = 6 mA
C16B	Input	C16	Clock 16 MHz is the clock used to transfer data across the H-MVIP bus. The clock runs twice as fast as the data rate. This common clock is used in both the receive and transmit direction.
RL_DATA[1] RL_DATA[0]	Input	T20 AB22	Receive Line Serial Data 1 to 0 carries the receive data from the corresponding framer devices or H-MVIP backplane. For AAL1gator-4 only line [0] is supported.
RL_SIG[1] RL_SIG[0]	Input	T22 Y21	Receive Line Signaling 1 to 0 carries the CAS data from the corresponding framer devices. H-MVIP does not support signaling directly, but these signals can be used to transport signaling if needed. Signaling needs to be valid for the entire MF. For AAL1gator-4 only line [0] is supported.
C4B	Input	D18	Clock 4 MHz is the clock used for generating and sampling F0B. This common clock is used in both the receive and transmit direction.

8.5.1 Summary of Line Interface Signals

The following table shows all modes at the same time and shows how pins are redefined for the different modes.

Line Interface Summary

Direct	H-MVIP	AAL1gator-8 Pin	AAL1gator-4 Pin
TL_SYNC[7]		A19	NC
TL_SYNC[6]		B22	NC
TL_SYNC[5]		F21	NC
TL_SYNC[4]		H21	NC
TL_SYNC[3]		K21	K21
TL_SYNC[2]		N22	N22
TL_SYNC[1]		T19	T19
TL_SYNC[0]	F0B	V19	V19
TL_DATA[7]		B19	NC
TL_DATA[6]		E20	NC
TL_DATA[5]		F22	NC
TL_DATA[4]		J20	NC
TL_DATA[3]		L20	L20

Direct	H-MVIP	AAL1gator-8 Pin	AAL1gator-4 Pin
TL_DATA[2]		P19	P19
TL_DATA[1]	TL_DATA[1] (AAL1gator-8 Only)	U19	U19
TL_DATA[0]	TL_DATA[0]	V20	V20
TL_SIG[7]		C18	NC
TL_SIG[6]		D21	NC
TL_SIG[5]		G20	NC
TL_SIG[4]		H19	NC
TL_SIG[3]		K19	K19
TL_SIG[2]		N20	N20
TL_SIG[1]	TL_SIG[1] (AAL1gator-8 Only)	R20	R20
TL_SIG[0]	TL_SIG[0]	Y22	Y22
TL_CLK[7]		A20	NC
TL_CLK[6]		C21	NC
TL_CLK[5]		E19	NC
TL_CLK[4]		H22	NC
TL_CLK[3]		K22	K22
TL_CLK[2]		N21	N21
TL_CLK[1]		R22	R22
TL_CLK[0]		U21	U21
CTL_CLK	C16B	C16	C16
RL_SYNC[7]		B21	NC
RL_SYNC[6]		C20	NC
RL_SYNC[5]		F19	NC
RL_SYNC[4]		J19	NC
RL_SYNC[3]		M20	M20
RL_SYNC[2]		P20	P20
RL_SYNC[1]		U22	U22
RL_SYNC[0]		W20	W20
RL_DATA[7]		D20	NC
RL_DATA[6]		E22	NC
RL_DATA[5]		H20	NC
RL_DATA[4]		K20	NC
RL_DATA[3]		N19	N19
RL_DATA[2]		R19	R19
RL_DATA[1]	RL_DATA[1] (AAL1gator-8 Only)	T20	T20
RL_DATA[0]	RL_DATA[0]	AB22	AB22
RL_SIG[7]		B20	NC

Direct	H-MVIP	AAL1gator-8 Pin	AAL1gator-4 Pin
RL_SIG[6]		C22	NC
RL_SIG[5]		G21	NC
RL_SIG[4]		J21	NC
RL_SIG[3]		M22	M22
RL_SIG[2]		P22	P22
RL_SIG[1]	RL_SIG[1] (AAL1gator-8 Only)	T22	T22
RL_SIG[0]	RL_SIG[0]	Y21	Y21
RL_CLK[7]		A21	NC
RL_CLK[6]		E21	NC
RL_CLK[5]		G22	NC
RL_CLK[4]		J22	NC
RL_CLK[3]		L22	L22
RL_CLK[2]		P21	P21
RL_CLK[1]		T21	T21
RL_CLK[0]		AA22	AA22
CRL_CLK	C4B	D18	D18

8.6 Clock Generation Control Interface(14)

Pin Name	Type	Pin No.	Function
CGC_DOUT[3] CGC_DOUT[2] CGC_DOUT[1] CGC_DOUT[0]	Output	AB16 AB14 Y15 AA15	External Clock Generation Control Data Out Bits 3 to 0 form the SRTS correction code when SRTS_STBH is asserted; otherwise CGC_DOUT[3:0] bits form the channel status and frame difference when ADAP_STBH is asserted.
CGC_LINE[3] CGC_LINE[2] CGC_LINE[1] CGC_LINE[0]	Output	AB19 AA18 W19 AB18	CGC Line Bits 3 to 0 form the line CGC_DOUT corresponds to when SRTS_STBH is asserted; otherwise CGC_LINE[3:0] bits form the adaptive state machine index when ADAP_STBH is asserted.
SRTS_STBH	Output	AA20	SRTS Strobe indicates that an SRTS value is present on CGC_DOUT[3:0]. CGC_LINE[3:0] indicates the line the SRTS code controls.
ADAP_STBH	Output	AA19	Adaptive Strobe indicates that the channel status and byte difference are being played out on the CGC_DOUT[3:0]. The nibbles are identified by the values on CGC_LINE[3:0].
NCLK/ SRTS_DISB	Input	AA16	Network Clock is the ATM network-derived clock used for SRTS. If this signal is tied low, SRTS is disabled. Internally this clock can be divided down to a lower frequency. The resulting clock should be 2.43 MHz for T1 and E1mode, 38.88 MHz for E3 mode and 77.76 MHz for DS3 mode.

Pin Name	Type	Pin No.	Function
TL_CLK_OE	Input	Y16	Transmit Line Clock Output Enable controls whether or not the TL_CLK lines are inputs or outputs between the time of hardware reset and when the CLK_SOURCE_TX bits are read. If high, all TL_CLK pins are outputs. If low, all TL_CLK pins are inputs. There is an internal pull-down resistor, so all TL_CLK pins are inputs if the pin is not connected. The value of this input is overwritten by the CLK_SOURCE_TX bits in the LIN_STR_MODE memory register.
CGC_SER_D	Input	AA17	External Clock Generation Control Serial Data is an input used to allow external clock control circuitry to pass frequency information into the internal clock synthesizer.
CGC_VALID	Input	W18	External Clock Generation Control Valid signal is an active high input indicating that the data on CGC_SER_D is valid. This signal must transition from a low to a high at the first valid data on CGC_SER_D and must stay high through the whole clock control word.

8.7   JTAG/TEST Signals(7)

Pin Name	Type	Pin No.	Function
TCLK	Input	B3	The test clock signal provides timing for test operations tat can be carried out using the JTAG test access port.
TMS	Input Internal Pull-up	A3	The test mode select signal controls the test operations that can be carried out using the JTAG test access port. Maintain TMS tied high when not using JTAG logic.
TDI	Input Internal Pull-up	C4	The test data input signal is JTAG serial input data
TDO	Output	B5	The test data output signal is JTAG serial output data.
SCAN_ENB	Input Internal Pull-up	A14	An active low signal which, in SCAN mode, is used to shift data. This signal should be tied high for normal operation.
SCAN_MODEB	Input Internal Pull-up	W3	When tied low enable SCAN mode. This signal should be tied high for normal operation.
TRSTB	Schmitt Trigger Input Internal Pull-up	Y19	The active low test reset signal is an asynchronous reset for the JTAG circuitry. If JTAG logic will be used, one option is to connect TRSTB to the RSTB input, and keep TMS tied high while RSTB is high; this maintains the JTAG logic in reset during normal operation. If JTAG logic will not be used, use the option described above, or simply ground TRSTB.
RESERVED_OUT		AB20	Not used, leave unconnected
RESERVED_IN		W22	Not used, tie to ground.

8.8 General Signals(3+power/gnd)

Pin Name	Type	Pin No.	Function
RSTB	Schmitt Trigger Input Internal Pull-up	AA21	Reset is an active low asynchronous hardware reset. When RSTB is forced low, all of the AAL1gator's internal registers are reset to their default states.
SYS_CLK	Input	A2	System Clock. The maximum frequency is 45 MHz. This clock is used to clock the majority of the logic inside the chip and also determines the speed of the memory interface and the external clock control interface. This clock is also used for clock synthesis. When clock synthesis is enabled this clock must be 38.88 MHz.
VDD3.3 (PPH, PQH)	Power	E1 L1 R4 W4 V2 Y5 W8 W16 AB17 Y18 Y20 R21 L19 F20 A22 A18 D16 D11 D4	**Power (VDD3.3).** The VDD3.3 pins should be connected to a well decoupled +3.3V DC power supply. These pins power the output ports of the device. PQH pins are "quiet" power pads.
VDD2.5 (PCH)	Power	E4 U4 AA11 W17 U20 M21 C14 A7	**Power (VDD2.5).** The VDD2.5 pins should be connected to a well decoupled +2.5V DC power supply. These pins power the core of the device.

Pin Name	Type	Pin No.	Function
VSS (PPL, PQL, PCL)	Ground	C3 F1 G4 K3 M4 T2 AA2 AA3 AA7 AB8 Y11 AB12 Y14 AB15 Y17 AB21 W21 V21 V22 M19 L21 G19 D22 C19 C17 D19 D15 A12 C11 D8 D5 B4	**Ground (VSS).** The VSS pins should be connected to GND. PPL pins are ground pins for ports. PQL pins are "quiet" ground pins for ports. PCL pins are core ground pins. All grounds should be connected together.
Thermal Grounds	Ground	J9-J14 K9-K14 L9-L14 M9-M14 N9-N14 P9-P14	Thermal grounds for heat dissapation. Connect to ground plane.
No Connects	Unconnected	A1, B2, L4	These are unbonded pins. Do not connect.

Notes on Pin Description:

1. All AAL1gator-4/8 inputs and bi-directionals present minimum capacitive loading and are 5V tolerant.

2. The AAL1gator-4/8 UTOPIA/Any-PHY outputs and bi-directional pins have 8 mA drive capability. TDO has a 4 mA drive capability. Any other outputs and bi-directional pins have 6 mA drive capability.

3. All AAL1gator-4/8 outputs can be tristated under control of the IEEE P1149.1 test access port, even those which do not tristate under normal operation. All outputs and bi-directionals are 5 V tolerant when tristated.

4. All clock inputs (except TL_CLK) are Schmitt triggered. Inputs RPHY_ADD[4]/RSX, RL_DATA[7:0], RPHY_ADDR[3:0], TPHY_ADDR[4:0], RL_CLK[7:0], RL_SYNC[3:1], TL_CLK[7:0], RATM_DATA[15:0], RATM_PAR, RATM_CLK, RATM_SOC, TATM_CLK, D[15:0], RAM_PAR[1:0], WRB, CSB, RDB, NCLK, CRL_CLK, CTL_CLK, SCAN_ENB, SCAN_MODEB, CGC_SER_D, CGC_VALID, RSTB, ALE, TL_CLK_OE, TMS, TCLK, TDI and TRSTB have internal pull-up resistors.

5. Power to the VDD3.3 pins should be applied *before* power to the VDD2.5 pins is applied. Similarly, power to the VDD2.5 pins should be removed *before* power to the VDD3.3 pins is removed.

9 Functional Description

The AAL1gator-4/8 is divided into the following major blocks, all of which are explained in this section:

- UTOPIA Interface Block (UTOPIAI)

- AAL1 SAR Processing Block (A1SP)

- Processor Interface Block (PROCI)

- RAM Interface Block (RAMI)

- Line Interface Block (LINEI)

- JTAG

9.1 UTOPIA Interface Block (UI)

The UI manages and responds to all control signals on the UTOPIA bus and passes cells to and from the UTOPIA bus and the A1SP block.   Both 8-bit and 16-bit UTOPIA interfaces with an optional single parity bit are supported. Each direction can be configured independently and has its own address configuration register.

The following UTOPIA modes are supported:

- UTOPIA Level One Master (8-bit only)

- UTOPIA Level One PHY

- UTOPIA Level Two PHY

- Any-PHY PHY

In the sink direction, the UI uses a 8-cell deep FIFO for buffering cells as they wait to be sent to the A1SP block. In addition, the A1SP contains an 8-cell deep FIFO. In the source direction, the UI uses a 4-cell deep FIFOs for holding cells before they are sent out onto the UTOPIA bus. Also, the A1SP contains an 8-cell deep FIFO. The data flow showing the FIFOs is shown in Figure 4.

Figure 4   Data Flow and Buffering in the UI and the A1SP Block
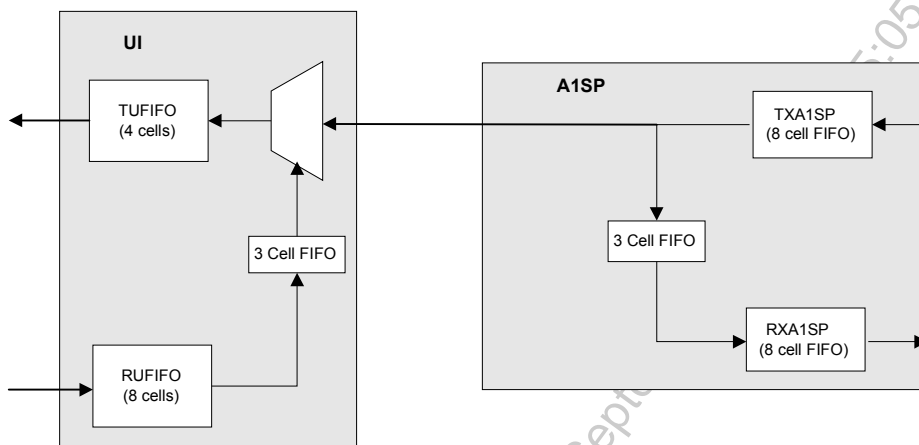
In UTOPIA Level Two mode, the AAL1gator-4/8 responds on the UTOPIA bus as a single port device.

For UTOPIA to UTOPIA loopback, there is a 3-cell FIFO in the UI Block. Line-side to Line-side loopback is done in the A1SP Block.

The UI_EN bit in the UI_COMN_CFG register enables both the source side and sink side UTOPIA interface. This bit resets to the disabled state so that the chip resets with all UTOPIA outputs tristated. Once the modes have been configured and the interface enabled, then the outputs will drive to their correct values.

The UI block consists of 7 functions: UI Data Source Interface (SRC_INTF), UI Data Sink Interface(SNK_INTF), 8-cell FIFO (FF8CELL), 4-cell FIFO (FF4CELL), 3-cell FIFO (FF3CELL), UMUX, and UI_REG. See Figure 5 for the block diagram of the AAL1_UI block.

PMC-SIERRA

Figure 5   UI Block Diagram

UTOPIA Interface (UI) Block
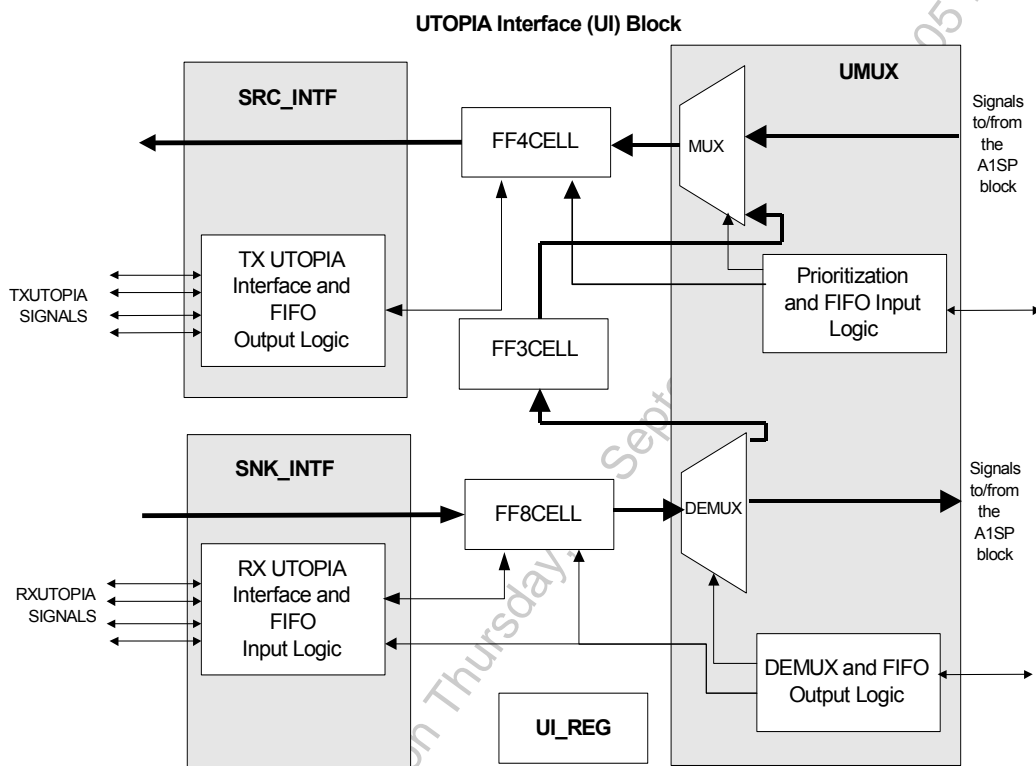
9.1.1   UTOPIA Source Interface (SRC_INTF)

The SRC_INTF block (shown in Figure 5) conveys the cells received from the UMUX block to the UTOPIA interface. Depending on the value of UTOP_MODE field in the UI_SRC_CFG register, the UTOPIA interface will either act as an UTOPIA master (controls the write enable signal) or as an UTOPIA PHY device (controls the cell available signal). As a PHY device, the SRC_INTF can either be a UTOPIA Level One device, where it is the only device on the UTOPIA bus, or a UTOPIA Level Two device where other devices can coexist on the UTOPIA bus. As a master device, the SRC_INTF can only function as a UTOPIA Level One device.

If 16_BIT_MODE is set in the UI_SRC_CFG register then all 16 bits of the UTOPIA data bus are used. 16_BIT_MODE must be '0' in UTOPIA master mode.

In master mode, the SRC_INTF block sources TATM_D, TATM_PAR, TATM_SOC, and TATM_ENB while receiving TATM_CLAV. The Start-Of-Cell (SOC) indication is generated coincident with the first word (only 8-bit mode is supported) of each cell that is transmitted on TATM_D. TATM_D, TATM_PAR and TATM_SOC are driven at all times. The TATM_ENB signal indicates which clock cycles contain valid data for the UTOPIA bus. The device will not assert the TATM_ENB signal until it has a full cell to send and the target device has activated TATM_CLAV. The TATM_CLAV signal indicates whether the target device is able to accept cells or not. Only cell level handshaking is supported. If the target device is unable to accept any additional cells it must deactivate TATM_CLAV no later than byte 49 of the current cell. No additional cells will be sent until TATM_CLAV is activated.

In PHY mode, the SRC_INTF block sources RPHY_D[15:0], RPHY_PAR, RPHY_SOC, and RPHY_CLAV, while receiving RPHY_ENB. The SOC indication is generated coincident with the first word (8-bit or 16-bit) of each cell that is transmitted on RPHY_D[15:0]. In PHY mode, the RPHY_D[15:0], RPHY_PAR, and RATM_SOC signals are driven only when valid data is being sent; otherwise they are tristated.

In UTOPIA Level 1 PHY mode, RPHY_CLAV is activated whenever a complete cell is available to be sent. It remains active until the last byte has been read of the last available complete cell. A cell is sent one cycle after RPHY_ENB goes low. If RPHY_ENB goes high during the cell transfer, data is not sent each cycle following one where RPHY_ENB is high.

RPHY_ADD[4:0] is an input and is used only in UTOPIA mode. Any bus cycle following one where RPHY_ADD[4:0] matches CFG_ADDR(4:0) in the UI_SRC_ADD_CFG register, the UI Block will drive RPHY_CLAV. Otherwise RPHY_CLAV is tri-stated. If, in addition, during the previous cycle RPHY_ENB was high and it is low in the current cycle, then the device is selected and the SRC_INTF begins transmitting a cell the next cycle. Note that in UTOPIA Level One there is no polling, and RPHY_ADD[4:0] must be tied low.

Parity is driven on TATM_PAR(RPHY_PAR) whenever TATM_D(RPHY_D[15:0]) is driven. EVEN_PAR will determine whether even parity or odd parity is generated. Since odd parity is required by the ATM Forum, EVEN_PAR is intended to be used for error checking only.

The AAL1gator-4/8 can tolerate temporary de-assertions of TATM_CLAV/RPHY_ENB), but it is assumed that enough UTOPIA bandwidth is present to accept the cells that the AAL1gator-4/8 can produce in a timely manner. Once the 4-Cell FIFO fills up in the UI, cells will begin filling up in the 8-cell FIFO in the A1SP block. Anytime the UTOPIA FIFO fills up the T_UTOP_FULL interrupt will go active in the MSTR_INTR_REG if it is enabled. This FIFO can fill during normal operation and is not usually an indication of an error. However, the A1SP FIFO should not normally fill. If it does fill it indicates there is some congestion, which is impacting the UTOPIA interface and the TALP_FIFO_FULL bit will go active in A1SP_INTR_REG. When the TALP FIFO fills, then TALP is no longer able to build cells and data will start building up in the transmit buffer and the frame_advance_fifo will fill. If this continues so that the FR_ADV_FIFO_FULL bit goes active then data has been lost and the transmit queues need to be reset. The T_UTOP_FULL indicator can be used to determine when the UTOPIA Interface clears. It may also be desirable to disable UI_EN so that the stored cells can be flushed.

The SRC_INTF circuit controls when a cell is transmitted from the internal 4 cell FIFO. Since the UTOPIA can transmit cells at higher speeds than the TALP, and since it is expected to see applications in a shared UTOPIA environment, cell transmission from the SRC_INTF commences only when there is a full cell worth of data available to transmit. The cell is then transmitted to the interface at the UTOPIA TATM_CLK rate, in accordance with the TATM_FULLB/RPHY_ENB) input. The maximum supported clock rate is 52 MHz.

Any-PHY Mode

If ANY-PHY_EN is set in the UI_SRC_CFG register then the SRC_INTF operates as a single port Any-PHY slave device. In Any-PHY mode the RPHY_ADDR(4) pin becomes the RSX pin and depending on the value of CS_MODE_EN, the RPHY_ADDR(3) pin may become the RCSB signal instead.

In Any-PHY mode in-band addressing is used to allow more than the 32 possible addresses available in UTOPIA mode. One extra word is prepended to the front of each cell that is transmitted. The prepended word indicates the port address sending the cell. The SRC_INTF uses CFG_ADDR(15:0) in the UI_SRC_ADD_CFG register for the address prepend. If 16_BIT_MODE is low then only the lower 8 bits are used.

During the cycle that the prepend address is active on the bus, RSX pulses high.

Because of the large number of possible ports, in the source direction, device addresses are used for polling and device selection, instead of port addresses. (Each device may control many ports) When a device is selected to send a cell, the PHY device prepends the port address in front of the cell. Since, in this direction the AAL1gator-4/8 is only a single port, the device address and port address are the same. However, the AAL1gator-4/8 has only a limited number of address pins. To accommodate systems, which are using a mix of different port density Any-PHY devices, the RCSB signal is available to handle any additional external decoding that is required. In Any-PHY mode, PHY devices respond with RPHY_CLAV 2 cycles after their address is on the bus instead of the one cycle required in UTOPIA mode. However, the timing of RCSB matches UTOPIA timing so that a full cycle for external decoding is available.

Table 2 shows how the CFG_ADDR field is used in different modes.

Table 2   CFG_ADDR and PHY_ADDR Bit Usage in SRC direction

	Polling		Selection	
MODE	PHY_ADDR Pins	CFG_ADDR	PHY_ADDR Pins	CFG_ADDR
UTOPIA-2 Single-Addr	[4:0]=device	[4:0]=device	[4:0]=device	[4:0]=device
Any-PHY with CSB	[2:0]=device	[2:0]=device	[2:0]=device CFG_ADDR is prepended	[2:0]=device
Any-PHY without CSB	[3:0]=device	[3:0]=device	[3:0]=device CFG_ADDR is prepended	[3:0]=device

Notes:

1.   In Any-PHY mode, in the SRC direction the AAL1gator-4/8 will prepend the cell with CFG_ADDR[15:0]. In 8-bit mode the cell will be prepended with CFG_ADDR[7:0]

2.   In Any-PHY mode, if CS_MODE_EN='1' then you must program CFG_ADDR[4:3] = "00".

3.   In Any-PHY mode, if CS_MODE_EN='0' then you must program CFG_ADDR[4]="0".

9.1.2   UTOPIA Sink Interface (SNK_INTF)

The SNK_INTF block receives cells from the UTOPIA interface and sends them to the UMUX interface. Depending on the value of the UTOP_MODE field in the UI_SNK_CFG register, the UTOPIA interface acts either as an UTOPIA master (controls the read enable signal) or as an UTOPIA PHY device (controls the cell available signal). As a PHY device the SNK_INTF can either be a UTOPIA Level One device, where it is the only device on the UTOPIA bus, or a UTOPIA Level Two device where other devices can coexist on the UTOPIA bus. As a master device the SNK_INTF can only function as a UTOPIA Level One device.

If 16_BIT_MODE is set in the UI_SNK_CFG register then all 16 bits of the UTOPIA data bus are used. 16_BIT_MODE must be '0' in UTOPIA master mode.

In master mode, the SNK_INTF block receives RATM_D, RATM_PAR, RATM_SOC, and RATM_CLAV while driving RATM_ENB. Once the UI is enabled in this mode, and, if the RATM_CLAV input signal is asserted, the SNK_INTF block waits for an RATM_SOC signal from the PHY layer. Once the RATM_SOC signal arrives, the cell is accepted as soon as possible. The Start-Of-Cell (SOC) indication is received coincident with the first word (only 8-bit mode is supported) of each cell that is received on RATM_D. An 8 cell FIFO allows the interface to accept data at the maximum rate. If the FIFO fills, the RATM_ENB signal will not be asserted again until the device is ready to accept an entire cell. The RATM_ENB signal depends only on the cell space and is independent of the state of the RATM_CLAV signal. The RATM_CLAV signal indicates whether the target device has a cell to send or not. Only cell level handshaking is supported.

In PHY mode, the SNK_INTF block receives TPHY_D[15:0], TPHY_SOC, and TPHY_ENB while driving TPHY_CLAV. The cell available (TPHY_CLAV) signal indicates when the device is ready to receive a complete cell. In UTOPIA Level One mode, TPHY_CLAV is always driven.

In UTOPIA Level Two mode, SNK_INTF responds as a single address device. TPHY_CLAV is driven the cycle following ones in which TPHY_ADDR(4:0) matches CFG_ADDR(4:0) in UI_SNK_ADD_CFG register. Otherwise TPHY_CLAV is tri-stated. If, in addition to an address match, during the previous cycle TPHY_ENB was high and it is low in the current cycle, then the device is selected and the SRC_INTF begins accepting the cell that is being received. Note that in UTOPIA Level One there is no polling, and TPHY_ADD[4:0] must be tied low.

The SNK_INTF block waits for an SOC. When an SOC signal arrives, a counter is started, and 53 bytes are received. If a new SOC occurs within a cell, the counter reinitializes. This means that the corrupted cell will be dropped and the second good cell will be received. The SNK_INTF block stores the cell in the receive FIFO. If the receive FIFO becomes full, it stops receiving cells. The bytes are written to the FIFO with RATM_CLK. RATM_CLK is an input to the AAL1gator-4/8. The maximum supported clock rate is 52 MHz.

Parity is always checked and a parity error will cause an interrupt if the UTOP_PAR_ERR_EN bit is set in the MSTR_INTR_EN_REG. FORCE_EVEN_PARITY will determine whether even parity or odd parity is checked. Since odd parity is required by the ATM Forum, FORCE_EVEN_PARITY is intended to be used for error checking only. If an error is detected the UTOP_PAR_ERR bit in the MSTR_INTR_REG is set, and the corresponding enable bit is set in the MSTR_INTR_EN_REG then INTB will go active. Any cell received with bad parity will still be processed as normal.

Any-PHY Mode

If ANY-PHY_EN is set in the UI_SNK_CFG register then the SNK_INTF operates as a multi port Any-PHY slave device. In Any-PHY mode the TPHY_ADDR[4] pin becomes the TSX pin and depending on the value of CS_MODE_EN, the TPHY_ADDR(3) pin may become the TCSB signal instead.

In Any-PHY mode in-band addressing is used to allow more than the 32 possible addresses available in UTOPIA mode.   One extra word is prepended to the front of each cell that is transmitted. The prepended word indicates the port address to receive the cell. The SNK_INTF uses CFG_ADDR(15:2) in the UI_SNK_ADD_CFG register to match with the address prepend. If 16_BIT_MODE is low then CFG_ADDR(7:2) is used. In both cases, polling should only be done with PHY_ADDR[1:0] equal to "00".

During the cycle that the prepend address is active on the bus, the TSX input pulses high.

In the sink direction, port addresses are used for polling and device selection, instead of device addresses. However the AAL1gator-4/8 has only a limited number of address pins. To accommodate systems, which are using a mix of different port density Any-PHY devices, the TCSB signal is available to handle any additional external decoding that is required. In Any-PHY mode, PHY devices respond with TPHY_CLAV 2 cycles after their address is on the bus instead of the one cycle required in UTOPIA mode. However the timing of TCSB matches UTOPIA timing so that a full cycle for external decoding is available.

Table 3 shows how the CFG_ADDR field is used in different modes.

Table 3   CFG_ADDR and PHY_ADDR Bit Usage in SNK direction

	Polling		Selection	
MODE	**PHY_ADDR Pins**	**CFG_ADDR**	**PHY_ADDR Pins**	**CFG_AD DR**
UTOPIA-2 Single-Addr	[4:0]=device	[4:0]=device	[4:0]=device	[4:0]=devi ce
Any-PHY with CSB	[2]=device [1:0]="00"	[2]=device	[15:2]=device [1:0]="00" PHY_ADDR is not used. Addr is prepended.	[15:2]=dev ice
Any-PHY without CSB	[3:2]=device [1:0]="00"	[3:2]=device	[15:2]=device [1:0]="00" PHY_ADDR is not used. Addr is prepended.	[15:2]=dev ice

Notes:

1. In Any-PHY mode, if CS_MODE_EN='1' then you must program CFG_ADDR[4:3] = "00". Else if CS_MODE_EN='0' then you must program CFG_ADDR[4]="0".

2. In Any-PHY mode the upper 14 bits of the prepended address are compared with CFG_ADDR[15:2]. The bottom two bits must be "00". If in 8-bit mode CFG_ADDR[7:2] is used instead.

9.1.3   UTOPIA Mux Block (UMUX)

The UMUX serves as the bridge between the A1SP block and the SNK_INTF and SRC_INTF blocks.

In the source direction, the UMUX polls the A1SP block and the Loopback FIFO using a least recently serviced algorithm to determine cell availability. In this algorithm, once a particular source is serviced, it is put at the lowest priority of the two sources.

If the SRC_INTF FIFO has room for a cell, the UMUX polls the A1SP block or Loopback FIFO to determine if it has a cell. If so, a cell is read from the selected A1SP block or Loopback FIFO and transferred into the SRC_INTF FIFO. If the highest priority source does not have a cell, the other source is examined. The A1SP block has an 8-cell FIFO. The Loopback FIFO is 3 cells.

In the sink direction, the UMUX waits until the SNK_INTF FIFO has a cell to send. Once the SNK_INTF FIFO has a cell to send, the UMUX polls the A1SP associated with the cell for availability. Once the A1SP has room for the cell, the UMUX reads the cell out of the SNK_INTF FIFO and places it in the A1SP FIFO.

The UMUX also supports two forms of UTOPIA to UTOPIA loopback; global loopback, where all cells are looped, and VC based loopback, where only a specific VC is used to loopback cells. In global loopback all cells received by the UTOPIA block are sent back out onto the UTOPIA bus. Global loopback is enabled by setting the U2U_LOOP bit in the UI_COMN_CFG register. In VC based loopback mode, any cell received with a VC that matches the loopback VC is sent back out onto the UTOPIA bus. VC based loopback is enabled by setting the VCI_U2U_LOOP bit in the UI_COMN_CFG register. The loopback VC is programmable by writing the U2U_LOOP_VCI register. The 3-cell FIFO is used for loopback.

9.2   AAL1 SAR Processing Block (A1SP)

The A1SP block is the main AAL1 SAR processing block. The block processes 4/8 E1/T1 lines.
The remainder of this section refers to the AAL1gator-8 A1SP block with 8 links, but for all
cases for the AAL1gator-4 there are 4 lines in the A1SP block. This block has the following
major components.

- Transmit Frame Transfer Controller (TFTC) block

- Cell Service Decision (CSD) block

- Transmit Adaptation Layer Processor (TALP) block

- TALP FIFO (TFIFO) block

- Local Loopback Block (LOC_LPBK) block

- Receive Frame Transfer Controller (RFTC) block

- Receive Adaptation Layer Processor (RALP) block

- RALP FIFO (RFIFO) block

Figure 6 shows a block diagram of the AAL1gator-4/8 and the sequence of events used to
segment and reassemble the CBR data.

Figure 6   A1SP Block Diagram
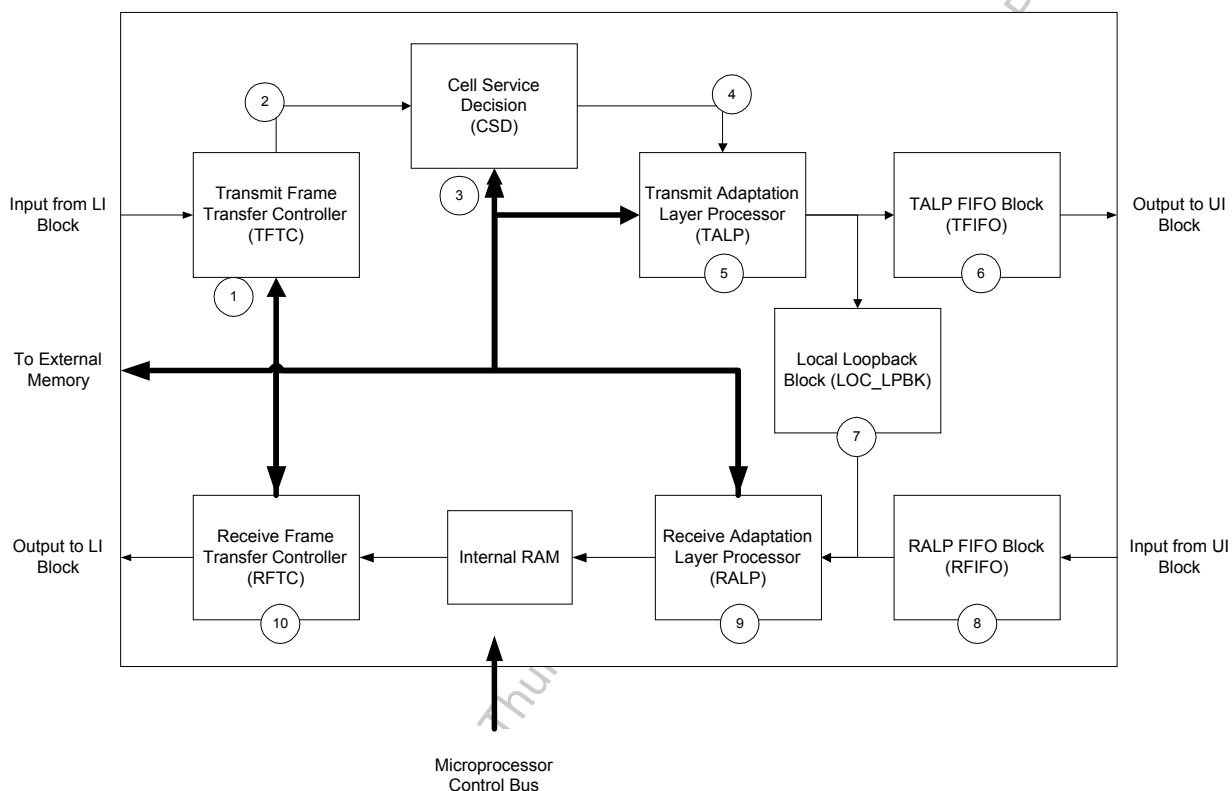
1. TFTC stores line data into the memory 16 bits at a time.

2. When the TFTC finishes writing a complete frame into the memory, it notifies the CSD of a frame completion by writing the line and frame number into a FIFO.

3. The CSD checks a frame-based table for queues having sufficient data to generate a cell. For each queue with enough data to generate a cell, the CSD schedules the next cell generation occurrence in the table.

4. The CSD commands the TALP to generate a cell from the available data for each of the ready queues identified in step 3.

5. The TALP generates the cell from the data and signaling buffers and writes the cell into the TALP FIFO.

6. The TFIFO block buffers cells which will be transmitted out to the UTOPIA Interface block.

7. If local loopback is enabled the cell is looped to RALP.

8. The RFIFO block buffers cells received from the UTOPIA Interface block.

9. The RALP performs pointer searches, checks for overrun and underrun conditions, detects SN mismatches, checks for OAM cells, and extracts the line data from the cells, and places the data into the receive buffer.

10. The RFTC plays the receiver buffer data onto the lines.

Four types of data are supported by the A1SP.

11. UDF-ML (Unstructured Data Format- Multi-Line). Unstructured bit stream for line speeds < 15 Mbps. (supports 8 lines per A1SP) if all are under 2.5 Mbps.

12. UDF-HS (Unstructured Data Format- High Speed). Unstructured bit stream for line speeds under 45 Mbps. (Only one line supported per A1SP).

13. SDF-FR (Structured Data Format- Frame). Channelized data without CAS signaling. (Frame based structure).

14. SDF-MF (Structured Data Format- Multi-Frame). Channelized data with CAS signaling. (Multi-Frame based structure).

9.2.1 AAL1 SAR Transmit Side (TxA1SP)

Transmit Frame Transfer Controller (TFTC)

The TFTC accepts deframed data from Line Interface Block. For structured data, the TFTC uses the synchronization supplied by the Line Interface Block to perform a serial-to-parallel conversion on the incoming data and then places this data into a multiframe buffer in the order in which it arrives.

The TFTC monitors the frame sync signals and will realign when an edge is seen on these signals that does not correspond to where it expects it to occur. It is not necessary to provide an edge at the beginning of every frame or multiframe. The AAL1gator-4/8 reads signaling during the last frame of every multiframe. For T1 mode, the AAL1gator-4/8 reads signaling on the 24th frame of the multiframe. For E1 mode, the AAL1gator-4/8 reads signaling on the 16th frame of the multiframe.

Note for lines using CAS change detection mode, CAS must be static for the entire multiframe structure as the AAL1gator will sample CAS at other points within the multiframe.

A special case of E1 mode exists that permits the use of T1 signaling with E1 framing. Normally an E1 multiframe consists of 16 frames of 32 timeslots, where signaling changes on multiframe boundaries. When E1_WITH_T1_SIG is set in LIN_STR_MODE and the line is in E1 mode, the TFTC will use a multiframe consisting of 24 frames of 32 timeslots. In this mode, the AAL1gator-4/8 reads signaling on the 24th frame of the multiframe.

The AAL1gator-4/8 reads the signaling nibble for each channel when it reads the last nibble of each channel's data unless the SHIFT_CAS bit in the LIN_STR_MODE register is set. If the SHIFT_CAS bit is set then the AAL1gator-4/8 reads the signaling nibble for each channel when it reads the first nibble of each channel's data. See Figure 7 for an example of a T1 frame. See Figure 8 for an example of an E1 frame.

Figure 7   Capture of T1 Signaling Bits (SHIFT_CAS=0)
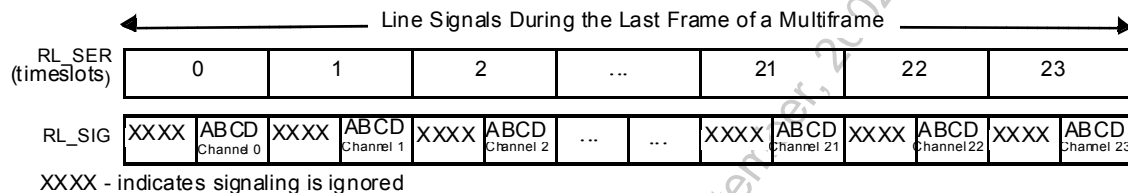
Figure 8   Capture of E1 Signaling Bits (SHIFT_CAS=0)
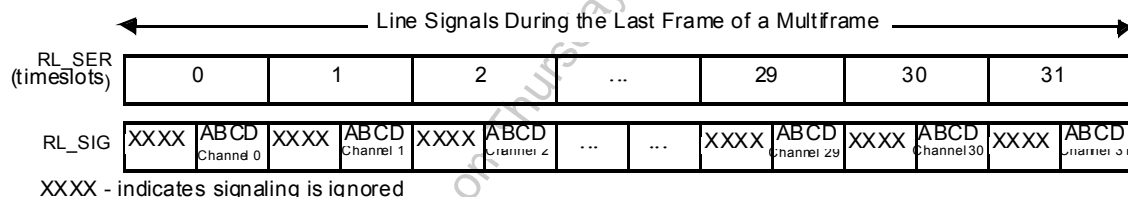
Note:

1. AAL1gator-4/8 treats all 32 timeslots identically. Although E1 data streams contain 30 timeslots of channel data and 2 timeslots of control (timeslots 0 and 16), data and signaling for all 32 timeslots are stored in memory and can be sent and received in cells.

Unstructured data is received without regard to the byte alignment of data within a frame and is placed in the frame buffer in the order in which it arrives. Figure 9 shows the basic components of the TFTC.

PMC-SIERRA

Figure 9   Transmit Frame Transfer Controller
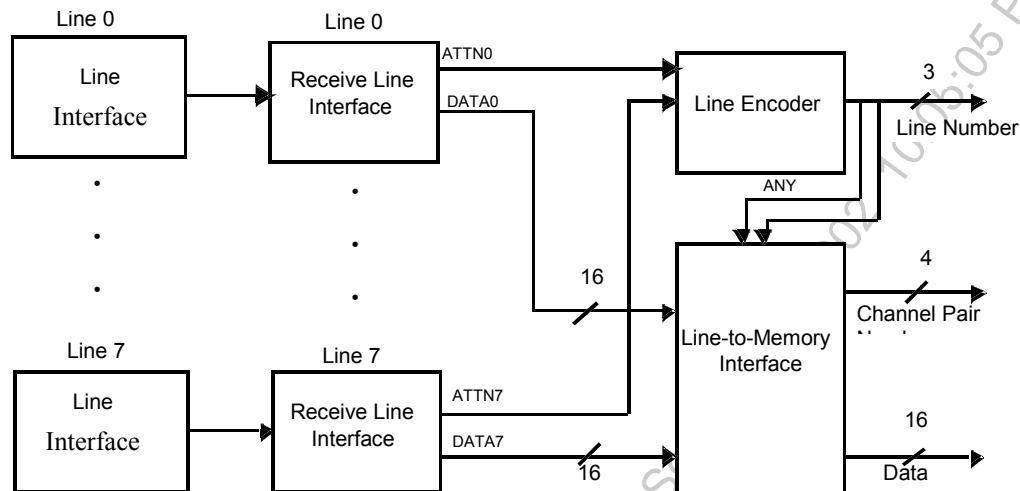
The receive line interface is primarily a serial-to-parallel converter. Serial data, which is derived from the RL_DATA signal from the LI Block, is supplied to a shift register. The shift register clock is the RL_CLK input from the external framer. When the data has been properly shifted in, it is transferred to a 2-byte holding register by an internally derived channel clock. This clock is derived from the line clock and the framing information.

The channel clock also informs the line-to-memory interface that two data bytes are available from the line. When the two bytes are available, a line attention signal is sent to the line encoder block. However, because the channel clock is an asynchronous input to the line-to-memory interface, it is passed through a synchronizer before it is supplied to the line encoder. Since there are eight potential lines and each of them provides its own channel clock, they are synchronized before being submitted to the line encoder.

The TFTC accommodates the T1 Super Frame (SF) mode by treating it like the Extended Super Frame (ESF) format. The TFTC ignores every other frame pulse and captures signaling data only on the last frame of odd SF multiframes. The formatting of data in the signaling buffers is highly dependent on the operating mode. Refer to section 7.6.6 "RESERVED (Transmit Signaling Buffer)" on page 136 for more information on the transmit signaling buffer.

Figure 10 shows the format of the transmit data buffer for ESF-formatted T1 data for lines that are in the SDF-MF mode.

Figure 10   T1 ESF SDF-MF Format of the T_DATA_BUFFER
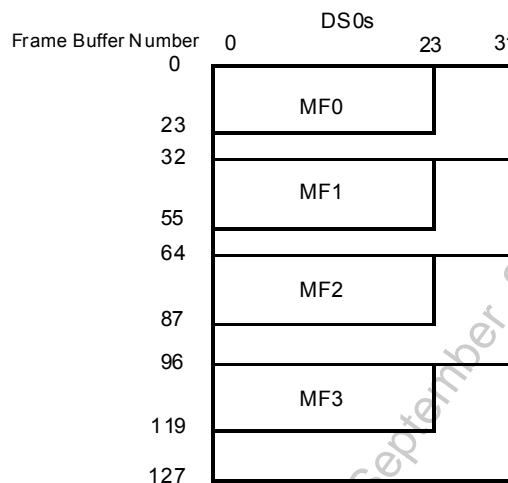
Figure 11 shows the format of the transmit data buffer for SF-formatted T1 data for lines that are in the SDF-MF mode.

Figure 11   T1 SF-SDF-MF Format of the T_DATA_BUFFER
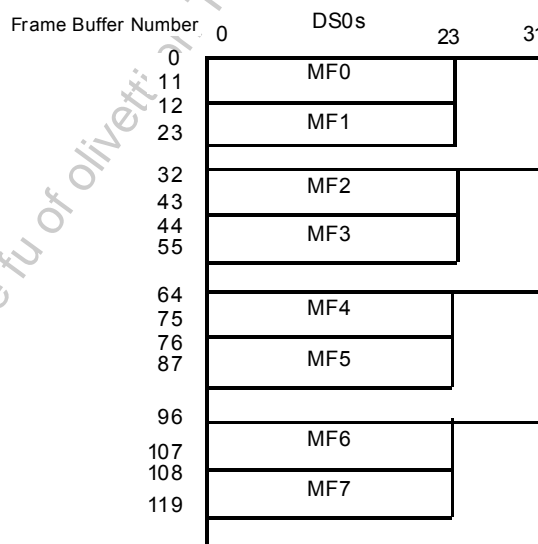
Figure 12 shows the format of the transmit data buffer for T1 data for lines that are in the SDF-FR mode.

Figure 12   T1 SDF-FR Format of the T_DATA_BUFFER
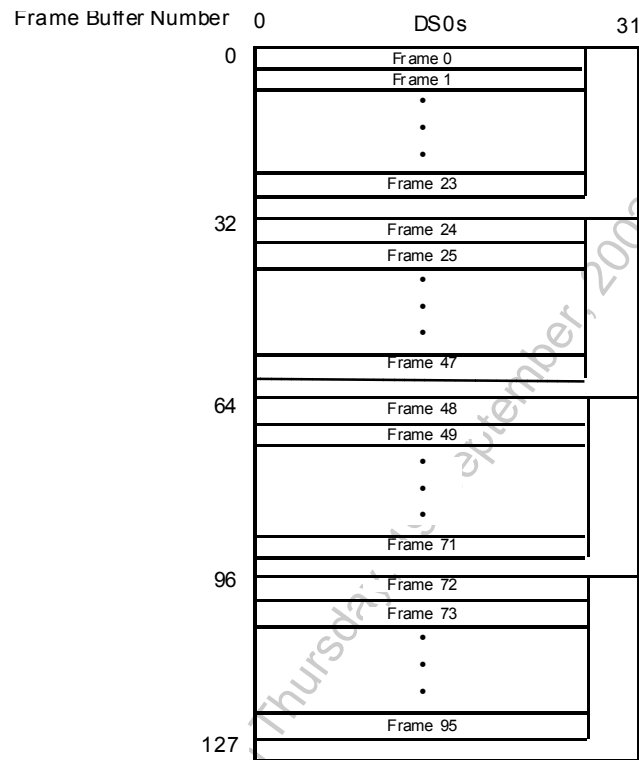
Figure 13 shows the format of the transmit data buffer for E1 data for lines that are in the SDF-MF mode.

Figure 13   E1 SDF-MF Format of the T_DATA_BUFFER
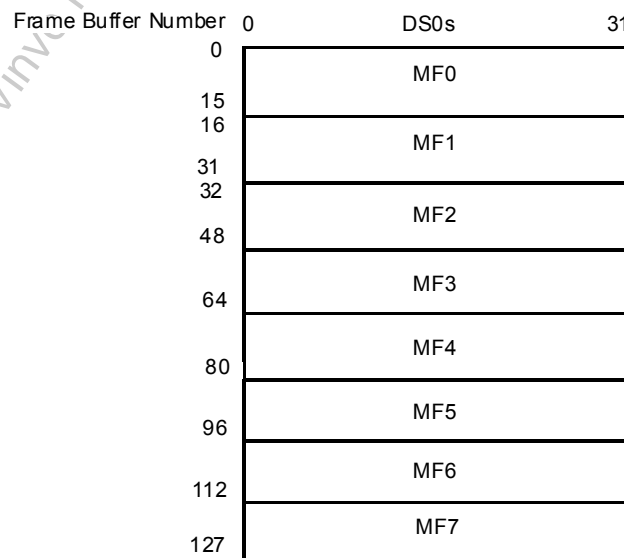
Figure 14 shows the format of the transmit data buffer for E1 data using T1 signaling, for lines that are in SDF-MF mode

Figure 14   E1 SDF-MF with T1 Signaling Format of the T_DATA_BUFFER
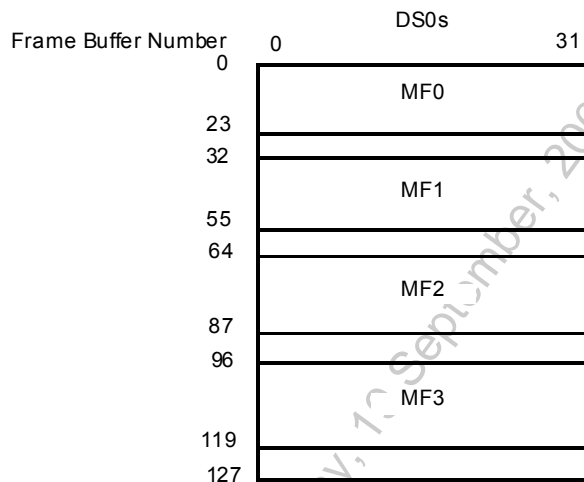
Figure 15 shows the format of the transmit data buffer for E1 data for lines that are in the SDF-FR mode.

Figure 15   E1 SDF-FR Format of the T_DATA_BUFFER
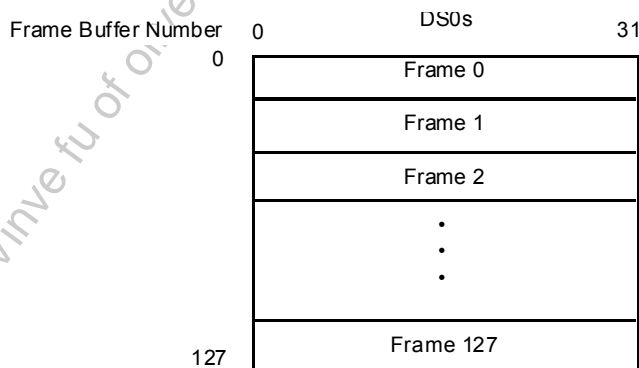
Figure 16 shows the format of the transmit data buffer for lines that are in UDF-ML mode.

Figure 16   Unstructured Format of the T_DATA_BUFFER

Frame Buffer Number

Data Bits

0	256-Bit Internal Frame 0	255
	256-Bit Internal Frame 1	
	• • •	
127	256-Bit Internal Frame 127	
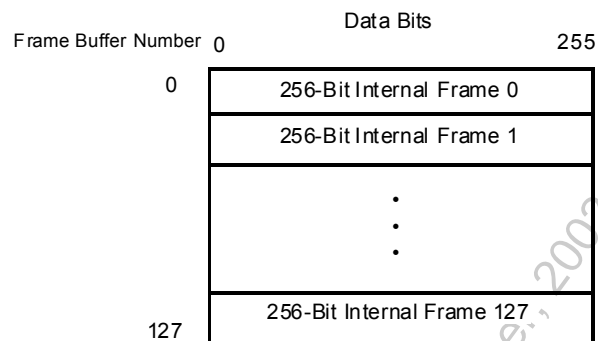
Figure 17, Figure 18, Figure 19 and Figure 20 show the contents of the transmit signaling buffer for the different signaling modes. In all cases the upper nibble of each byte is "0000".

Figure 17 SDF-MF T1 ESF Format of the T_SIGNALING_BUFFER

Byte Address

Multiframe	Channel 0 ABCD	Channel 1 ABCD	• • •	Channel 22 ABCD	Channel 23 ABCD	Channel 24 Not Used	• • •	Channel 31 Not Used
0								
1								
2								
3								

Figure 18   SDF-MF T1 SF Format of the T_SIGNALING BUFFER

Byte Address

Multiframe/2	Channel 0 ABAB	Channel 1 ABAB	• • •	Channel 22 ABAB	Channel 23 ABAB	Channel 24 Not Used	• • •	Channel 31 Not Used
0								
1								
2								
3								

Figure 19   SDF-MF E1 Format of the T_SIGNALING_BUFFER
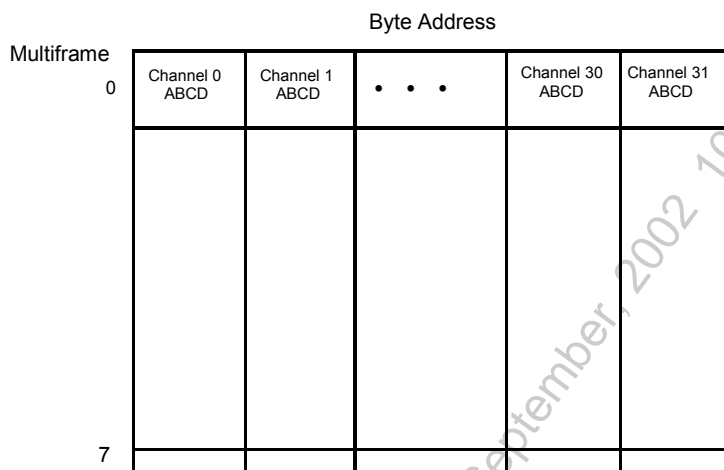
Figure 20   SDF-MF E1 with T1 Signaling Format of the T_SIGNALING_BUFFER
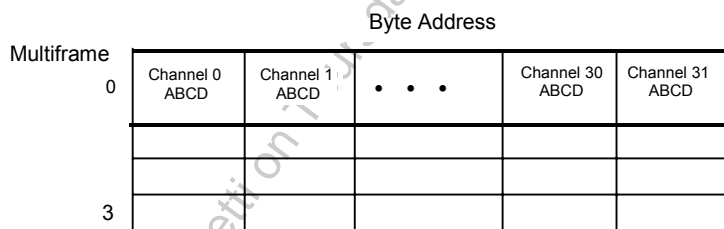
Transmit Conditioning

The T_COND_DATA structure allows conditional data to be defined on a per-DS0 basis and the T_COND_SIG structure allows conditioned signaling to be defined on a per-DS0 basis. The TX_COND bit in the T_QUEUE_TBL allows the cell building logic (described in Section 0 Transmit Adaptation Layer Processor (TALP) on page 71) to be directed to build cells from the conditioned data and signaling. To control whether conditioned data, conditioned signaling, or both is used, set TX_COND_MODE in the TX_CONFIG register to the appropriate value. The TX_COND bit and TX_COND_MODE bits can be set on a per-queue basis.

By having independent control over whether signaling or data is conditioned, it is possible to substitute the signaling which is carried in the CAS bits across the ATM network while still passing the data received off the line. This is useful for applications that may not be receiving the signaling with the data.

For HS_UDF mode the HS_TX_COND bit needs to be set in the HS_LIN_REG register. When this bit is set cells with an all ones pattern will be generated. The CMD_REG_ATTN bit needs to be written to a '1' after the HS_TX_COND bit is set for this function to take affect.

Under certain alarm conditions such as Loss of Signal (LOS), an Alarm Indication Signal (AIS) needs to be transmitted downstream. This means that cells need to be generated which carry an AIS pattern. The AAL1gator-4/8 does not do any alarm processing and is dependent on the external framer for this functionality. The framer would notify the processor of any alarm conditions and then the processor would switch a particular queue from normal mode to a conditioned mode by setting the TX_COND bit in the T_QUEUE_TBL.

Most AIS signals are an all ones pattern, so cells with this pattern can be generated by setting T_COND_DATA to "FF"x and T_COND_SIG to "F"x. In E3 mode this can be done by setting HS_TX_COND bit to a '1'. However a DS3 AIS signal is a framed "1010" pattern. This signal can be generated by setting the HS_GEN_DS3_AIS bit in the HS_LIN_REG register. The CMD_REG_ATTN bit needs to be written to a '1' after the HS_GEN_DS3_AIS bit or the HS_TX_COND bit is set for this function to take affect.

Transmit Signaling Freezing

Signaling freezing is a required function when transporting CAS. This function holds the signaling unchanged when the incoming line fails. The PMC-Sierra framers provide this function. If a framer is used that does not support signaling freezing, this function must be provided externally.

SRTS for the Transmit Side

The transmit side supports SRTS only for unstructured data formats on a per-line basis. SRTS support requires an input reference clock, NCLK. The input reference frequency is defined as $155.52/2^n$ MHz, where n is chosen such that the reference clock frequency is greater than the frequency being transmitted, but less than twice the frequency being transmitted (2 x RL_CLK > NCLK > RL_CLK). For T1 or E1 implementation, the input reference clock frequency must be 2.43 MHz. The transmit side can accept a reference clock speed of up to 77.76 MHz, which is required for DS3 applications. Figure 21 on page 65 shows the process implemented for each UDF line enabled for SRTS, regardless of the reference frequency. One bit of resulting 4-bit SRTS code is then inserted into the CSI bit of each of the odd numbered cells for that line. There are four odd cells in each 8 cell sequence, so each one carries a different SRTS bit. If the line does not supply SRTS, then all odd CSI bits are set to 0. The 3008 divider is the number of data bits in eight cells (8 x 8 x 47). The divider is aligned on the first cell generation after a reset or a resynchronization to the cell generation process.

Figure 21   Transmit Side SRTS Function
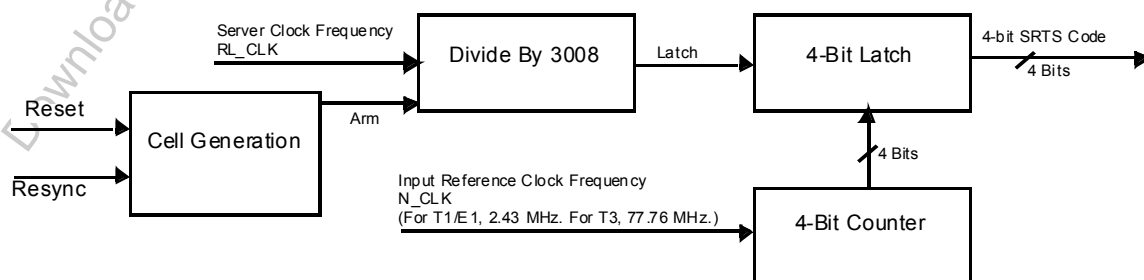
CAS Change Detection

If CAS change detection is enabled a word is written to the CAS Interrupt FIFO every time the value of the CAS nibble changes and then remains stable for one additional multiframe. CASCHG_FIFO_EMPB is set as long as this FIFO contains any unread entries, which will result in a maskable interrupt. The structure of the word contained in the FIFO is shown in Figure 22. The first eight bits indicate the channel, which encountered a change in the value of CAS. The next four bits indicate the RX CAS value and the final four bits indicate the TX CAS value.

TX refers to the CAS incoming on the TDM interface (RL_SIG), and RX refers to CAS incoming in ATM cells at the UTOPIA Cell Sink interface.

The CAS values are de-bounced internally one time, and any additional debounce must be done external to the chip.

Figure 22   CAS Change Interrupt Word

15 13	12 Channel 8	7 RX CAS 4	3 TX CAS 0
Line	Channel	RX CAS	TX CAS

The processor will also be able to mask out portions of the CAS and therefore receive interrupts only when particular bits of the CAS change. Figure 23 shows the structure of AUTO_CONFIG_n field in the CAS Change Configuration Table.

Figure 23   CAS Change Configuration Register Structure

15 RX MASK	11 TX MASK	7 Reserved	3 Reserved 0
RX MASK	TX MASK	Reserved	Reserved

Cell Service Decision (CSD) Circuit

The CSD circuit determines which cells are to be sent and when. It determines this by implementing Transmit Calendar bit tables. When the TALP builds a cell, the CSD circuit performs a complex calculation using credits to determine the frame in which the next cell from that queue should be sent. The CSD circuit schedules a cell only when a cell is built by the TALP. If SUPPRESS_TRANSMISSION bit in the TX_CONFIG word is set, then the cell is scheduled, however, the cell is not transmitted.

The following steps (as well as Figure 24 on page 68) describe how the CSD circuit schedules cells for the TALP to build.

1) Once the TFTC writes a complete frame into external memory, it writes the line number and frame number of this frame into the FR_ADVANCE_FIFO. The CSD circuit reads the line-frame number pair from the FR_ADVANCE_FIFO and uses it as an index into the Transmit Calendar. The Transmit Calendar is composed of eight-bit tables, one per line. Each bit table consists of 128 entries, one per frame buffer. Each entry consists of 32 bits, one per queue. For each bit set in the indexed entry in the Transmit Calendar, the CSD will schedule the frame in which the next cell can be built for the corresponding queue, and notify the TALP that enough data is available to build a cell for that queue.

2) The CSD circuit processes all queues from the Transmit Calendar entry starting with the lowest queue number and proceeding to the highest. The processing steps are as follows:

a) The CSD circuit obtains the QUE_CREDITS, and subtracts the average number of credits per cell from it. The average number of credits, AVG_SUB_VALU, is the number of credits that will be spent sending the current cell. For structured lines, the average number of credits per cell is 46 7/8. For unstructured lines, the average number of credits per cell is 47.

b) Next, the CSD circuit computes the frame location for the next service by subtracting the remaining credits from 47. It divides the result by the number of channels, NUM_CHAN, dedicated to that queue. The number of channels is calculated based upon the channels allocated to the queue. NUM_CHAN is equal to the number of channels allocated to the queue.

c) The CSD then adds this frame differential to the present frame location to determine the frame number of the next frame in which the TALP can build a cell. The CSD circuit then sets a bit in the corresponding entry in the Transmit Calendar and writes to the QUEUE_CREDITS.

d) The CSD circuit then adds the new credits back to the credit total for the frame increment number. The number of new credits is equal to the frame differential computed earlier, multiplied by the number of channels for that queue. Once a queue is identified as requiring service, its identity is written to the NEXT_SERV location.

e) The CSD circuit obtains the next queue for that frame and repeats steps a. through d. The CSD circuit continues this process until there are no more active queues for that frame.

3) After servicing all the queues for that frame, the CSD circuit advances to the next active line located in the line queue. If there are no active lines, the CSD circuit returns to the idle state to wait for the next line to request service.

Figure 24 shows how the CSD assigns credits to determine in which frames cells should be sent.

Figure 24 Frame Advance FIFO Operation
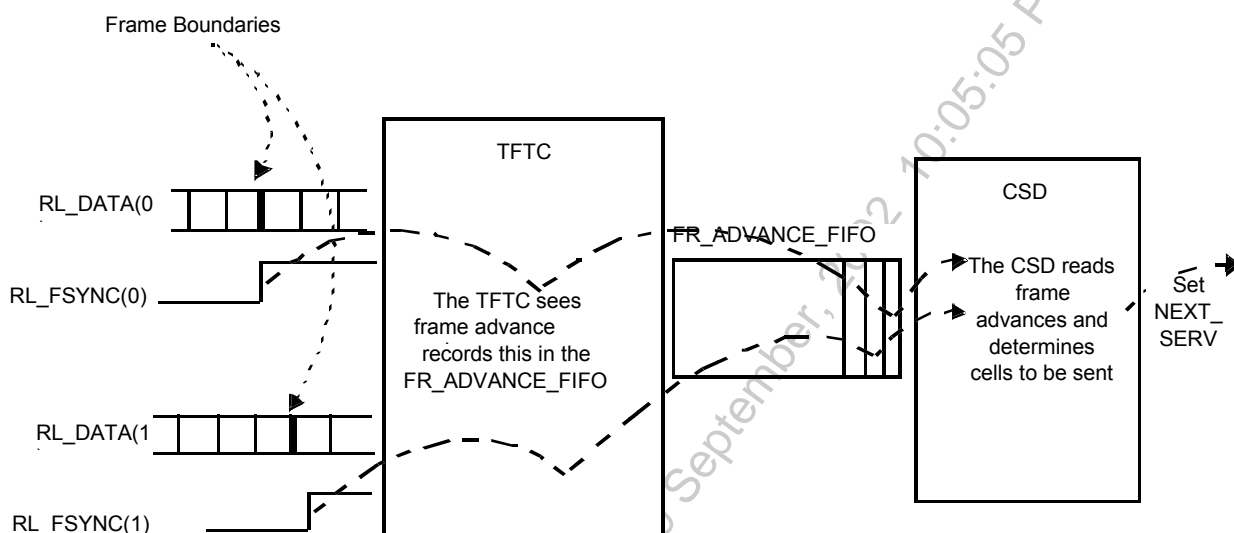
The following is an example of the calculations the CSD circuit performs. This example assumes a structured line with four channels allocated to one queue.

1) The TFTC writes Line 3 and Frame 4 to the FR_ADVANCE_FIFO.

2) The CSD circuit determines the queue for which a cell is ready by finding a set bit in the Transmit Calendar. In this example, it is queue number 100.

3) The CSD circuit reads the number of credits for queue number 100. The number of credits is always greater than 47 because it is ready for service. In this example, QUE_CREDITS = 59.375.

4) The CSD circuit subtracts AVG_SUB_VALU, the average number of credits spent per cell. (Remember: For structured lines, the average number of credits per cell is 46-7/8. For unstructured lines, the average number of credits per cell is 47.)

Credits = 59.375 – 46.875

Credits = 12.5

5) The frame differential for the next service is computed from the number of credits needed to exceed 47 and NUM_CHAN, the number of channels allocated per frame.

47 – 12.5 = 34.5

34.5 / 4 = 8.625

Round 8.625 up, so the frame differential is 9.

6) Therefore, the next cell will be sent nine frames ahead of the current cell.

Next frame = present frame number + 9

7) The CSD circuit computes the number of credits for those nine frames and adds the result to the total.

New credits = 9 x 4 = 36

QUE_CREDITS = 36 + 12.5 = 48.5

If the queue is on a line in SDF-MF mode, the CSD makes a signaling adjustment to the QUE_CREDITS before writing this value to memory. (If the queue is not in SDF-MF mode, the signaling adjustment is not made and the QUE_CREDITS calculated in Step 7 is written to memory.)

The calculation determines the number of signaling bytes in the structure, then generates an average number of signaling bytes inserted into cells per frame, and finally multiplies this average number by the frame differential to adjust the QUE_CREDITS.

8) The CSD converts the frame differential from units of frames to units of one-eighth of multiframes.

In performing this calculation, the CSD also uses the FRAME_REMAINDER value from the QUE_CREDITS location in the T_QUEUE_TBL. This example assumes that FRAME_REMAINDER = 1 from the previous calculation on this queue.

E1: Frame differential (in eighths of a multiframe) = (frame differential + FRAME_REMAINDER) / 2

T1: Frame differential (in eighths of a multiframe) = (frame differential + FRAME_REMAINDER) / 3

Frame differential = (9 + 1) / 3 = 10, or three-eighths of a multiframe, remainder 1.

The CSD writes the remainder of this division into the FRAME_REMAINDER location for use in the next calculation on this queue.

9) The CSD calculates the signaling credit adjustment by multiplying the frame differential expressed in eighths of a multiframe by the number of signaling bytes in a structure.

Number of signaling bytes in the structure = 4 channels x 0.5 bytes per channel = 2 bytes per multiframe

Signaling adjustment = three eighths x 2 = 0.75 bytes

10) Then the CSD adds the signaling credit adjustment to the total and writes the result to memory, in preparation for the next service on this queue.

QUEUE_CREDITS = 48.5 + 0.75 = 49.25 bytes

Unstructured lines use a different procedure. In the case of unstructured lines, a cell will be sent every time 47 bytes are received. This assumes that no partial cells are used for UDF mode.

Transmit CDV

The ideal minimum transmit CDV for all queue configurations is as follows. In each case, the frame rate is assumed to be 125us.

- UDF-ML/LOW_CDV bit set: 0 us.

- SDF-FR/Single-DS0-with-no-pointer: 0 us.

- SDF-FR Partial cell configurations with bytes_per_cell/num_chan = an integer: 0 us.

- All other configurations: 125 us.

The following items affect transmit CDV:

- Cell scheduling

- Contention with other cells scheduled at same time

- Actual cell build time

- UTOPIA contention

- OAM cell generation

1) The scheduler has a resolution of 125 µs. In other words, it works off a frame-based clock to determine whether or not a cell should be sent during the current frame. Therefore, if the ideal rate of cell transmission is not a multiple of 125 µs, there will be 125 µs of CDV. The scheduler will never add more than 125 µs of CDV.

For example, a single DS0 queue with no signaling and using full cells, will need to build a cell every 47 frames. Therefore, a cell will be scheduled every 47 frames, and the scheduler will add no CDV. Also in UDF mode all cells are sent every time 47 bytes are received so no CDV is added.

However, if signaling were added to the single DS0 queue, the extra byte that occurs every 24 bytes (assuming T1 mode) requires compensation. In this case, a cell will be sent every 46 or 47 frames. Therefore, there will be up to 125 µs of CDV due to the scheduler.

Note that for UDF lines there is a LOW_CDV bit which can be set in the LIN_STR_MODE memory register which will cause cells to be scheduled every 47 bytes instead of frame based. This eliminates the CDV caused by the scheduler. This mode can only be used in UDF-ML mode when BYTES_PER_CELL is 47. In High Speed mode cells are always scheduled every 47 bytes which assumes that partial cells are never used in HS mode.

2) Only one cell can be built at a time. Thus if multiple queues are scheduled to send cells during the same frame, additional delay will be incurred. If queues are activated and deactivated so that the number of queues scheduled ahead of a specific queue in the same frame changes, the resulting change in delay translates to CDV. The scheduling of multiple cells at the same time is known as clumping. It takes approximately 8 µs to build a cell normally but can take up to 15 us under worst case traffic and processor activity (Note this assumes a 38.88 MHz SYS_CLK). Therefore, each cell that is waited for could add up to 15 µs of delay. When multiple queues are scheduled to send cells at the same time, the cells will be built in sequential order, starting with 0 and going to 256. Therefore, in a system that will be adding and dropping queues, the higher number queues will experience more CDV than the lower number queues, depending on how many queues are active at the time, and are scheduled within the same frame. The AAL1gator-4/8 minimizes the effects of clumping by offsetting the schedule point of each line by 1/8th of a frame. Also when queues are added, an offset field can be supplied which will force multiple cells on the same line to be scheduled at different times. See Add Queue FIFO section in the Processor Interface section for more details.

3) For configurations that will require sending a cell every n frames where n is an integer divisor of 128 (for E1) or 96 (for T1), the cells will always be scheduled in the same frame unless the offset field is set differently for each cell.

4) The actual build time of a cell depends on microprocessor activity and contention with other internal state machines for the AAL1gator-4/8 memory bus. Therefore there will be some minor CDV that is added on a per cell basis, based on current microprocessor/memory traffic. This CDV is usually less than 4 µs and is not very noticeable.

5) If there is backpressure on the UTOPIA bus, cells will not be able to be sent which also causes CDV.

6) Since OAM cells have higher priority than data cells the transmission of OAM cells should be distributed. An OAM cell can add up to 8 us of CDV assuming a 38.88 MHz SYS_CLK under worst case processor load.

Transmit Adaptation Layer Processor (TALP)

OAM Cell Generation

When an OAM cell transmission is requested, it is sent at the first available opportunity. Transmit OAM cells have higher priority than cells scheduled by the CSD circuit. Because of this, care should be taken to ensure that OAM cells do not overwhelm the transmitter to such an extent that data cells are starved of adequate opportunities. The rate of OAM cells must be limited for the AAL1gator-4/8 to maintain its maximum CSD data rate.

To send an OAM cell, the microprocessor writes OAM cells into one of two dedicated cell buffers located in external memory. When the cell is assembled in the buffer, the microprocessor must set the appropriate bit in the Command register The TALP sends the cell as soon as possible, then clears the appropriate attention bit to indicate the requested cell has been sent. If requests for both OAM cells are active at the time the command register is read by the AAL1gator-4/8, OAM cell 0 will always be sent because it is assigned a higher priority. Therefore, to control the order of OAM cell transmission, the microprocessor should set only one OAM attention bit at a time and wait until it is cleared before setting the other attention bit.

OAM cells can optionally have the 48-byte OAM payload CRC-10 protected. This is accomplished by a CRC circuit that monitors the OAM cell as it is sent to the TUTOPIA and computes the CRC on the fly. It then substitutes the 10 bit resultant CRC, preceded by six 0s, for the last two bytes of the cell. The CRC generation is enabled by setting Bit 0 in Word 2 of the T_OAM_CELL.

Data Cell Generation

If the TALP receives a request to send a CSD-scheduled data cell and there are no OAM cell requests pending, it will do so as soon as it is free. It will look up the predefined ATM header from the T_QUEUE_TBL (refer to section 7.6.8 "T_QUEUE_TBL" on page 122). It will then obtain a sequence number for that queue from memory, and a structure pointer if necessary. After these bytes are written to the TUTOPIA interface, the TALP will then go to the data and the signaling frame buffers, locate the data bytes for the correct channels, and write them in the correct order to the UTOPIA interface. This cell building process is described in more detail in the following section.

Header Construction

The entire header is fixed per queue. Headers are maintained in the memory, one per queue. These headers include a Header Error Check (HEC) character for the fifth byte. The queue should be deactivated during header replacement to prevent cells from being constructed with incorrect header values. A queue can be paused by setting the SUPPRESS_TRANSMISSION bit in TX_CONFIG register. Emissions are still scheduled, just the transmissions are suppressed. For any cells that are suppressed, the T_SUPPRESSED_CELL_CNT is incremented.

Payload Construction

Payload construction is the most complex task the TALP circuit performs. The AAL1 requirements define much of the process, which is as follows:

1) The first byte of the payload is provided by a lookup into the T_QUEUE_TBL. This first byte consists of the CSI bit, a 3-bit sequence number, and a 4-bit sequence number protection field. The CSI bit is set depending on SRTS and pointer requirements. The sequence number is incremented every time a new cell is sent for the same VPI/VCI. If the queue has been configured for AAL0 mode, this step is not done and an additional data byte is loaded instead.

2) If the line is in one of the two structured modes, a structure pointer is needed in one of the even-numbered cells. The TALP inserts structure pointers according to the following rules:

- Only one pointer is inserted in each 8-cell sequence.

- A pointer is inserted in the first possible even-numbered cell of every 8-cell sequence.

- A pointer value of 0 is inserted when the structure starts in the byte directly after the pointer itself.

- A pointer value of 93 is inserted when the end of the structure coincides with the end of the 93-octet block of AAL-user information.

- A dummy pointer value of 127 is inserted in cell number six if no start-of-structure or end-of-structure occurs within the 8-cell sequence.

3) This algorithm supplies a constant number of structure pointers and, therefore, data bytes, regardless of the structure size. The pointer is inserted in the seventh byte location of the cell. To force the TALP to build a structure consisting of a single DS0 with no signaling nibble and no pointer, set T_CHAN_UNSTRUCT = 1 in the QUEUE_CONFIG word of the T_QUEUE_TBL.

4) The TALP fills the rest of the cell payload with data and/or signaling information. The T_CHAN_ALLOC table in the transmit queue table determines which channels are dedicated to which queue. If a bit is set, the channel represented by that bit is assigned to that queue. The TALP successively writes the data from the marked channels into the UTOPIA interface. If the LOOPBACK_ENABLE bit is set in the TX_CONFIG register then the cell is written into a separate FIFO to be looped back to RALP. A queue-based parameter, BYTES_PER_CELL, decides when enough payload bytes have been obtained. If this number is fewer than 47, then the remaining bytes for the cell are loaded with P_FILL_CHAR. This implies that because of the presence of the structure pointer, the number of fill bytes will not be constant for structured data queues.

5) The structure used for signaling is determined by the mode of the line and the value of E1_WITH_T1_SIG. Normally the signaling structure will follow the mode of the line. However, if the line is in E1 mode and E1_WITH_T1_SIG is set, then a T1 signaling structure is used. This means that for a single DS0, signaling is inserted after 24 data bytes instead of after 16 data bytes. If data is to be sent from the data queue, this process continues byte-by-byte while updating pointers and counters until one of the following occurs:

- The cell is complete.
- The last data byte for the last frame of the multiframe has been set.

6) When signaling information is to be sent, data is obtained from the signaling locations of the multiframe, with the help of the channel allocation table (T_CHAN_ALLOC). This process proceeds byte-by-byte until one of the following occurs:

- The cell is complete.
- All signaling nibbles for all channels assigned to the queue have been sent.

Figure 25 shows an example of the payload generation process.

Figure 25   Payload Generation
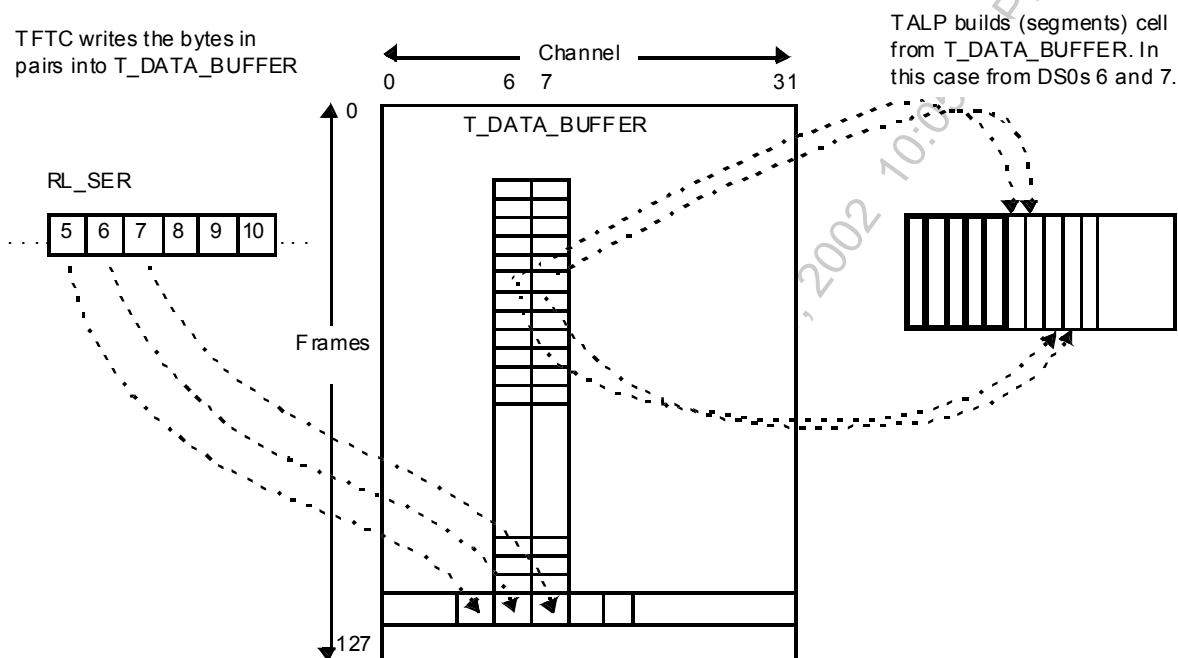
For AAL0 mode the cell build process takes 48 bytes of line data and does not add any AAL1 overhead bytes.

Peak Cell Rates (PCRs)

For purposes of discussion, the following PCR information is assumed:

- Full cells are used,

- The PCR numbers are per line, and

- The SYS_CLK is 38.88 MHz.

Peak Cell Rates (PCRs) for Structured Cell Formats

- For structured connection without CAS, PCR <= 171 x n cells per second per connection where $1 <= n <= 32$ (assuming completely filled cells).

- For structured connection with CAS, PCR <= 182 x n cells per second per connection where $1 <= n <= 32$ (assuming completely filled cells).

- Each AAL1 cell is either 46 or 47 bytes, depending upon whether or not the cell contains a structure pointer.

Peak Cell Rates (PCRs) for Unstructured Cell Formats

- PCR <= 4,107 cells per second for T1 (assuming 47 bytes for each AAL1 cell).

- PCR <= 5,447 cells per second for E1 (assuming 47 bytes for each AAL1 cell).

- PCR <= 118,980 cells per second for T3 (assuming 47 bytes for each AAL1 cell).

- PCR <= 91,405 cells per second for E3 (assuming 47 bytes for each AAL1 cell).

- If all eight lines in an A1SP are at the same rate, totaling 20 Mbps throughput for the A1SP, then the aggregate A1SP PCR <= 53,191 cells per second for multiple-line unstructured data format (assuming 47 bytes for each AAL1 cell). If all lines are not at the same rate, the aggregate A1SP PCR <= 46,542.

- PCR <= 1,000 cells per second per A1SP for OAM cells. This rate of OAM cells is calculated on the basis of up to four cells per second per VC. Transmitting and receiving OAM cells at this rate consumes 20% of the microprocessor accesses.

Peak Cell Rates and Partial Cells

Partial Cells can be used to minimize the amount of delay required to assemble a cell. However, the amount of overhead required for the same amount of TDM data increases when partial cells are used. This overhead increases as the number of data bytes per cell decreases. The following table shows the minimum partial cell sizes that can be supported for the different modes of operation if all connections are active on the device. If a smaller partial cell size is desired, either some time slots/links must not be used or only a subset of the connections must use that partial cell size. In any case the total PCR of the device should not exceed 100,000 cells per second with a 38.88 MHz SYSCLK or 110,000 cells per second with a 45 MHz SYSCLK.

Table 4   Minimum Partial Cell Size Permitted If All Connections Are Active

MODE	SYS_CLK=38.88 MHz	SYS_CLK = 45 MHz
T1 SDF-FR	16	14
T1 SDF-MF	16	15
E1 SDF-FR	21	19
E1 SDF-MF	22	20

Transmit FIFO (TALP_FIFO)

This block contains an 8 cell FIFO which buffers cells going out to the UTOPIA Interface block.

The UTOPIAI polls the A1SP blocks to determine if the A1SP has cells available in it's TALP_FIFO. If the TUTOPIA FIFO has room for a cell, the A1SP block is selected, a cell is read from the A1SP's TALP FIFO, and moved into the TUTOPIA FIFO.

Local Loopback Block(LOC_LPBK)

The AAL1gator-4/8 supports local loopback of cells. Local loopback is when a cell generated by TALP is looped back to RALP instead of being sent out to the UTOPIA bus via the TALP FIFO.

When LOOPBACK_ENABLE is set in the TRANSMIT_CONFIG register, cells for that queue will be looped back to the RALP block instead of being transmitted toward the UTOPIA bus. Since this bit is configurable on a per queue basis it can be used during normal operation to loop back individual time slots, groups of time slots on a given line or even entire lines. The cells, which are looped back, are stored in a separate 3 cell FIFO that RALP can read. The loopback FIFO will have higher priority to prevent the transmitter from backing up. See Figure 26 for architecture of the local loopback.

Note: Remote loopback is also supported in the AAL1gator-4/8, but it is performed in the UTOPIAI block.

Figure 26   Local Loopback
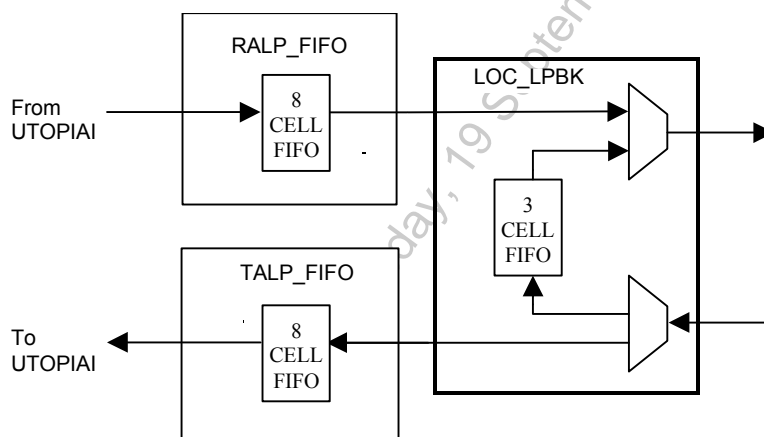
9.2.2   AAL1 SAR Receive Side (RxA1SP)

RALP FIFO (RFIFO)

This block contains an 8 cell FIFO, which buffers cells received from the UTOPIA Interface block.

Once the RUTOPIA FIFO in the UTOPIAI block has a cell to send, the UTOPIAI polls the A1SP to determine if the RALP FIFO has room for the cell. Once the RALP FIFO has room for the cell, the UTOPIAI reads the cell out of the RUTOPIAI FIFO and places it in the RALP FIFO.

Receive Adaptation Layer Processor (RALP)

The RALP receives ATM cells from the RALP_FIFO and removes and checks the AAL1 encapsulation as well as providing a VCI to queue mapping. It performs pointer searches, detects SN mismatches, and extracts the line data from the cells and stores it in the receive buffers located in external memory. It also checks for overrun and underrun conditions, and checks for OAM cells. The RALP does not verify the HEC because it expects a PHY layer device to verify the HEC before presenting the cell to the AAL1gator-4/8.

VCI Mapping

The AAL1gator-4/8 supports two options for VCI to queue mapping. Nine bits of the VCI are always used. The nine-bit field can be either located in the least significant bits of the VCI or shifted up 4 bits to occupy bits 12 down to 4 of the VCI field.

There are two methods of detecting a cell to be redirected to the OAM queue. The first method uses a data indicator bit in the header. The second method uses the customary PTI field. If the data indicator is not set (VCI(8) = 0 when SHIFT_VCI=0) or Payload Type Indicator (PTI) = 4 to 7, the cells are sent to the OAM queue and are stored using the pointers located in the OAM receive queue table. The head pointer is the address to the first cell received for each queue, and is usually maintained by the microprocessor. The tail pointer is the address of the last cell of the queue, and is maintained by the RALP. The RALP updates the tail pointer upon each OAM cell arrival.

If the cell is a data cell (cells received with VCI(8) = 1 when SHIFT_VCI=0 and PTI = 0 to 3), the cell is sent to the queue with VCI(7:0). Bits 7:5 determine the line, and bits 4:0 determine the queue. The A1SP block ignores VCI(15:9) and VPI(11:0). If SHIFT_VCI = 1, the interpretation of the VCI bits is shifted by four bits.

There is also a special mode when VP_MODE_EN is set in the UI_COMN_CFG register. When set the VPI field is used to select the Line number. Queue 0 is always assumed. So this mode is only available if all lines are configured with one queue. Normally this mode can only be used if all lines are configured in UDF-ML mode. In this mode the VCI field is used to detect an OAM cell. If the VCI is less than or equal to 31 then the cell will go to the OAM cell buffer.

Figure 27 shows the interpretation of the incoming cell header once the cell arrives at the A1SP.

PMC-SIERRA

Figure 27   Cell Header Interpretation

SHIFT_VCI=0
VP_MODE_EN=0

11	10	9	8	7	6	5	4	3	2	1	0
Ignored											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Ignored							Data	Line			Queue MOD 32				

SHIFT_VCI=1
VP_MODE_EN=0

11	10	9	8	7	6	5	4	3	2	1	0
Ignored											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Ignored			Data	Line			Queue MOD 32					Ignored			

SHIFT_VCI=X
VP_MODE_EN=1

11	10	9	8	7	6	5	4	3	2	1	0
Ignored									Line		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Ignored															
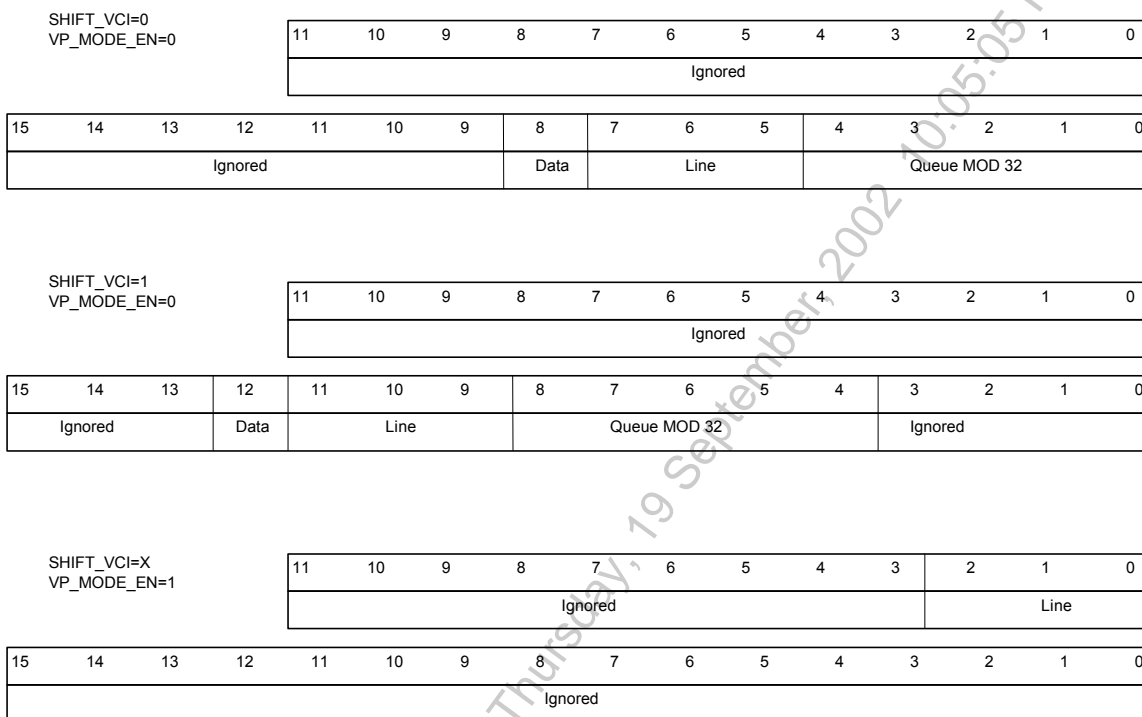
Sequence Number Processing

When the cell is a data cell, the RALP verifies the SN CRC-3 code and parity bit. If the SN CRC-3 and parity do not verify, the RALP attempts to correct it as specified in ITU-T Recommendation I.363.1 and increments the R_INCORRECT_SNP counter. If the RALP cannot correct the SN byte, it identifies the cell as invalid.

If the DISABLE_SN bit in the R_SN_CONFIG register has been set for this queue, then no further sequence number processing occurs. There are two modes of sequence number processing supported: Fast Sequence Number processing algorithm on all types of connections and Robust Sequence Number processing on Unstructured Data Format (UDF) connections. Both the Fast and Robust algorithms are done in accordance with the algorithms described in ITU-T Recommendation I.363.1.

For the "Fast SN Algorithm", the RALP acts upon the current cell based on the value of the current SN and SNP and the previous SN and SNP. The RALP either accepts the current cell, drops the current cell, or accepts the current cell and inserts cells.

For the "Robust SN Algorithm", the RALP makes decisions on the previous cell based on the value of the current SN and SNP and the previous SN and SNP. The RALP either accepts the previous cell, drops the previous cell, or inserts cells and accepts the previous cell. The Robust SN algorithm adds delay by requiring the following cell to arrive prior to accepting the previous cell. When a cell is accepted, it means that the contents of the cell are available to the RFTC for transmission onto the line. This requires that two write pointers be maintained, one that is at the end of the last accepted cell and one at the end of the last stored/received cell.

Inserted cells have the following properties:

- If the queue is in an unstructured mode or if the queue is in structured mode and the inserted cell should not contain a pointer, each inserted cell contains the number of payload bytes as determined by R_BYTES_CELL field in R_MP_CONFIG.

- If the queue is in a structured mode and the inserted cell should contain a pointer and R_BYTES_CELL is equal to 47, then the inserted cell contains 46 payload bytes.

- If the queue is in a structured mode and the inserted cell should contain a pointer and R_BYTES_CELL is less than 47, then the inserted cell contains R_BYTES_CELL payload bytes.

- The determination on whether or not the inserted cell should contain a pointer is based on the pointer generation rules defined by ITU-T Recommendation I.363.1. A pointer will be assumed if the queue is structured and the following conditions are met:

  o The SN is even AND there has been no other pointer in the group of eight cells so far AND (the sequence number = 6 OR the structure ends within the current inserted cell or next cell).

- If the queue is in SDF-MF mode and the inserted cell should contain signaling data, the number of payload bytes is adjusted but the signaling information is copied from the last valid signaling and is written to the signaling buffer. Therefore, signaling information is frozen during the playout of the frame with invalid signaling information.

Note: In SDF-FR mode, the CES specification states that if the queue contains data for only one DS0, no pointer is used. If a queue is configured in this manner, set R_CHAN_UNSTRUCT in the R_BUF_CFG receive queue table memory register.

- The value of the payload data depends on the value of INSERT_DATA in the R_SN_CONFIG receive queue table memory register.. The default is to load the value of 0xFF. Other options are to use conditioned data as defined by R_CONDQ_DATA, old data, or pseudorandom data. If old data is chosen, no data will be written and the old data in the receive buffer will be used. The receive buffer write pointer will be adjusted to the correct location. If the pseudorandom data option is chosen, the data played out will be the value of R_CONDQ_DATA with the MSB replaced by the current value of the pseudorandom number generator $x^{18} + x^{7} + 1$.

Notes:

1. All DS0s within the replaced cell will use the same algorithm. To minimize the overhead of generating the inserted cells, use the old data option whenever possible. The old data option still needs to do internal processing on a byte-by-byte basis, but since it does not have to write any data, it is about twice as fast as the other options.

2. The "Fast SN Algorithm" will, under certain situations, allow bad cells to pass through. When this occurs the cells are marked as potentially bad. Any cells marked as potentially bad will not have pointer verification done on them and any signaling data or SRTS information they contain will not be written. However, if these cells should contain pointers or signaling data, adjustments are made to the amount of payload data written so bit integrity is maintained.

3. The pseudorandom option is not available for UDF-HS mode.

The RALP will maintain bit integrity if there is no more than one consecutive errored cell, or if there are up to six lost. If the receive buffer underruns and BITI_UNDERRUN is set, then the amount of time that has passed since the last cell is checked and if it appears that less than 6 cell times have passed then cells will be inserted if no more than MAX_INSERT cells are detected as lost. Otherwise the queue will resync and realign with the structure if one exists.

MAX_INSERT controls the maximum number of cells that will be inserted when cells are lost. If more cells than MAX_INSERT are lost, then the queue will be forced into an underrun condition unless the lost cells are a result of coming out of underrun. The default value of MAX_INSERT, "000", is equivalent to "111" which means that up to seven cells will be inserted. MAX_INSERT is in the R_SN_CONFIG register and can be specified on a per-queue basis.

If MAX_INSERT and R_MAX_BUF are configured such that inserting the maximum number of cells into a buffer which is near it's R_MAX_BUF limit will cause the number of frames in the buffer to exceed 1FE, the "Robust SN Algorithm" may cause overruns due to misinserted cells and failure of the SN error protection mechanism. This will occur when a cell arrives which causes the SN processing to go to the OUT OF SEQUENCE state and the buffer is too full to allow the cell to be treated as a lost cell and stored as if cells had been lost.

If seven cells are lost, this will appear as a misinserted cell and will not be handled correctly. Likewise, if more than seven cells are lost, it will appear as if fewer than seven cells were lost because the SN repeats every eight cells. If seven or more cells were lost, there is high probability that the queue will underrun. If the queue has not underrun, the RALP takes the following steps to minimize the impact:

- Any time cells are inserted, if the next received pointer mismatches, it will immediately create a forced underrun to realign to the structure instead of waiting for two consecutive mismatches.

- No signaling information will be updated until a valid and correct pointer is received.

If the DISABLE_SN bit in the R_SN_CONFIG receive queue table memory register (refer to "DISABLE_SN" on page 172) is set, then both the Fast and Robust Sequence Number Processing algorithms are disabled.. That is, RALP will neither insert nor drop cells due to sequence number errors, but will update both the R_INCORRECT_SN and R_INCORRECT_SNP counters.

If R_AAL0_MODE bit is set in the R_MP_CONFIG register then no SN processing is done and byte six of the cell is handled as a data byte.

Fast Sequence Number Processing State Machine

The RALP sequence number processing state machine begins in the START state. Once a cell is received with a valid SN, the OUT_OF_SYNC state is entered. Any cells received while in the START state, including one with a valid SN are dropped, unless NODROP_IN_START, in the R_SN_CONFIG receive queue table memory register, is set. If this bit is set and the cell has a valid SN, the cell will be accepted.

Note: If it is important to not dump the first cell received, make sure NODROP_IN_START is set.

While in the OUT_OF_SYNC state, if another cell is received with a valid SN and in the correct sequence, then the SYNC state is entered and the cell is accepted. Otherwise, if a cell with an invalid SN is received, then the START state is re-entered and the cell is dropped. Otherwise, if the cell has a valid SN but is in the incorrect sequence, then the cell is dropped and the RALP remains in the OUT_OF_SYNC state.

While in the SYNC state, if another cell is received with a valid and correct SN, the RALP remains in the SYNC state. Otherwise, if an invalid SN is received, the RALP goes to the INVALID state; or if a valid but incorrect SN is received, the OUT_OF_SEQUENCE state is entered. All cells received while in the SYNC state are accepted.

While in the INVALID state, four possibilities can occur.

1. If the cell received has an invalid SN, the START state is re-entered and the cell is dropped.

2. If the cell received has a valid SN and is in sequence with the last valid SN, then a misinserted cell is detected and RALP returns to the SYNC state, but the cell is dropped to keep SN integrity because the previous cell has already been sent.

3. If the cell received has a valid SN and is equal to SN + 2 with respect to the last valid SN, then the RALP returns to the SYNC state and the cell is accepted.

4. Otherwise, if the SN is valid but does not meet any of the previous criteria, then the cell is dropped and the OUT_OF_SYNC state is entered.

While in the OUT_OF_SEQUENCE state, five possibilities can occur.

5. If the cell received has an invalid SN, the START state is re-entered and the cell is dropped.

6. If the SN is valid and in sequence with the last valid, in sequence SN, then a misinserted cell is detected and RALP returns to the SYNC state, but the cell is dropped to keep SN integrity because the previous cell has already been sent.

7. If the SN is valid and the SN is in sequence with the SN of the previous cell, the RALP assumes cells were lost; it inserts a number of dummy cells identical to the number of lost cells, accepts the cell and returns to SYNC. If the number of lost cells is greater than MAX_INSERT, then no cells are inserted and a forced underrun occurs. If an underrun occurred when cells were lost, no cells are inserted unless bit integrity through underrun is enabled.

    Note that if lost cells are detected, but the queue is currently in underrun and BITI_UNDERRUN is not set, then lost cells will not be inserted and a normal underrun recovery will be executed. If lost cells are detected, the queue is in underrun, and BITI_UNDERRUN is set, and less then six cells were lost, then lost cell will be inserted and processing will occur as if the underrun did not happen.

8. If the received SN is valid and the SN has a value equal to SN + 2 with respect to the last SN received in sequence, then the cell is accepted and the RALP returns to the SYNC state.

9. And finally, if the SN is valid but does not meet any of the previous criteria, then the cell is dropped and the RALP enters the OUT_OF_SYNC state.

See Figure 28 on page 83 for the flow of the "Fast SN Algorithm".

Anytime a cell is dropped, the R_DROPPED_CELLS , in the receive queue table memory register,is incremented and the SN_CELL_DROP sticky bit is set. Anytime a cell is detected lost, the R_LOST_CELLS is incremented by the number of lost cells. Anytime the SN state machine transitions from the SYNC state to the OUT_OF_SEQUENCE state, the R_SEQUENCE_ERR counter is incremented. Anytime a misinserted cell is detected the R_MISINSERTED counter is incremented. In all these cases the RCVN_STAT_FIFO will be written with the error that occurred and the queue number for which the error occurred, if the corresponding mask bit is not set or this is not the first unmasked sticky bit to be set for this queue since the sticky bit register was last cleared.

Figure 28   Fast SN Algorithm
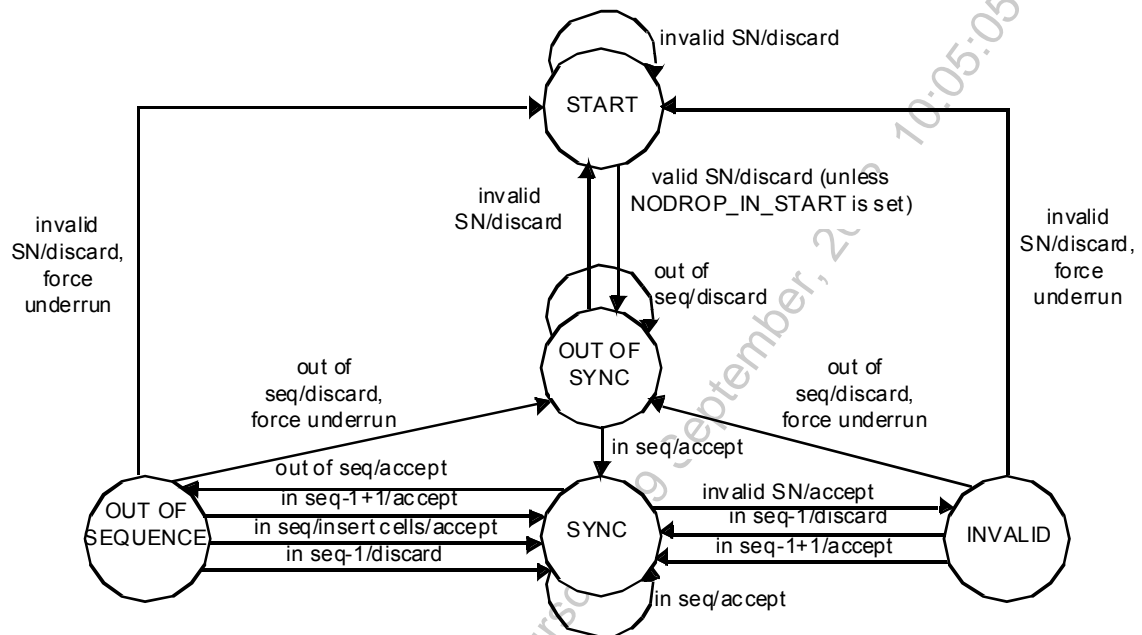
All cells received while in the SYNC state are accepted whether or not they are good. Any errored cells received while in the SYNC state are marked as potentially bad cells. These marked cells will not have their pointers checked, or bitmask checked, if they contain one; or if they contain signaling data, the signaling data will not be written to memory.

If the cell is accepted, the RALP then transfers the cell to the external memory using the R_CHAN_ALLOC fields in the R_QUEUE_TBL. Figure 29 shows this receive cell process. If a valid cell is not received in time, the queue may enter an underrun condition.

Note that during an underrun, the RALP 'Fast' sequence number processing state machine freezes in its current state. For example, if the state machine is in the SYNC state when underrun occurs, when the next cell arrives, potentially some time later, the state machine will still be in the SYNC state.

Figure 29   Receive Cell Processing for Fast SN
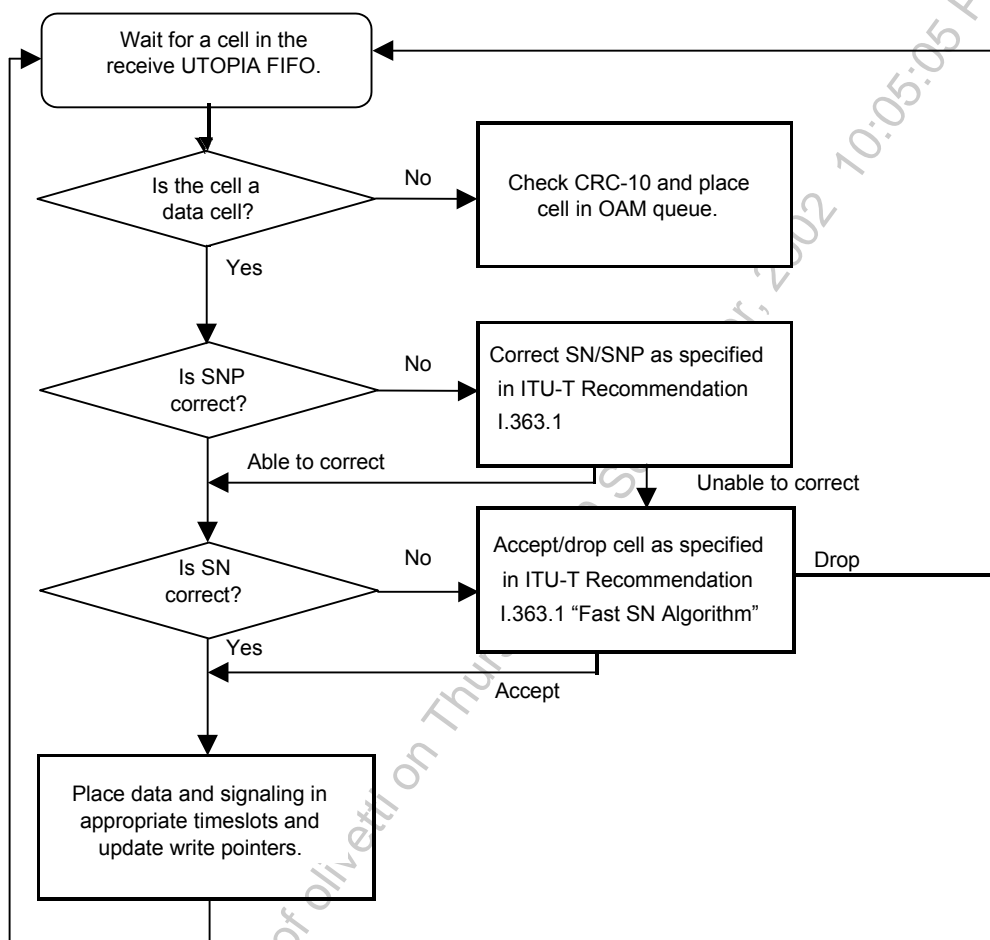
Robust Sequence Number Processing State Machine

The Robust SN processing State Machine is the same as the Fast SN processing state machine except that all actions are taken on the previous cell versus the current new cell. Robust Sequence Number processing is only supported in the UDF-ML and UDF-HS modes with full cells.

The RALP robust sequence number processing state machine begins in the START state. While in the START state two possibilities can occur:

1. If a cell is received with a valid SN, the OUT_OF_SYNC state is entered and the cell is stored.

2. If a cell is received with an invalid SN, the cell is dropped and the START state is maintained.

While in the OUT_OF_SYNC state, three possibilities can occur.

3. If a cell is received with a valid SN and in the correct sequence, then the SYNC state is entered and the previously stored cell is accepted.

4. If a cell with an invalid SN is received, then the START state is re-entered and the previously stored cell is dropped.

5. If the cell has a valid SN but is in the incorrect sequence, then the previously stored cell is dropped, the new cell is stored and the RALP remains in the OUT_OF_SYNC state.

While in the SYNC state, three possibilities can occur.

1. If a cell is received with a valid and correct SN, the RALP remains in the SYNC state. The stored cell is accepted and the new cell is stored.

2. If an invalid SN is received, the RALP goes to the INVALID state. The stored cell is accepted and the new cell is stored

3. If a valid but incorrect SN is received, the OUT_OF_SEQUENCE state is entered. The stored cell is accepted and the new cell is stored as the next cell and also in the position it would reside if cells had been lost . If there is not physically enough space in the buffer to store the second copy of the cell an overrun condition is declared.

While in the INVALID state, four possibilities can occur.

1. If the cell received has an invalid SN, the START state is re-entered and the stored cell is dropped.

2. If the cell received has a valid SN and is in sequence with the last valid SN, then a misinserted cell is detected and RALP returns to the SYNC state, the stored cell with invalid SN is dropped and the new cell is stored.

3. If the cell received has a valid SN and is equal to SN + 2 with respect to the last valid SN, then the RALP returns to the SYNC state and the stored cell is accepted and the new cell is stored.

4. Otherwise, if the SN is valid but does not meet any of the previous criteria, then the stored cell is dropped, the new cell is stored and the OUT_OF_SYNC state is entered.

While in the OUT_OF_SEQUENCE state, five possibilities can occur.

1. If the cell received has an invalid SN, the START state is re-entered and the stored cell is dropped.

2. If the SN is valid and in sequence with the last valid, in sequence SN, then a misinserted cell is detected and the stored cell is dropped. The RALP returns to the SYNC state and stores the new cell.

3. If the SN is valid and the SN is in sequence with the SN of the previous cell, the RALP assumes cells were lost; it inserts a number of dummy cells identical to the number of lost cells, accepts the second copy of stored cell and returns to SYNC. If the number of lost cells is greater than MAX_INSERT, then no cells are inserted and a forced underrun occurs. If an underrun occurred when cells are lost, no cells are inserted unless bit integrity through underrun is enabled.

4. If the received SN is valid and the SN has a value equal to SN + 2 with respect to the last SN received in sequence, then the stored cell is accepted, the new cell is stored and the RALP returns to the SYNC state.

5. And finally, if the SN is valid but does not meet any of the previous criteria, then the stored cell is dropped, the new cell is stored and the RALP enters the OUT_OF_SYNC state.

See Figure 30 on page 87 for the flow of the "Robust SN Algorithm".

Anytime a cell is dropped, the R_DROPPED_CELLS counter is incremented and the SN_CELL_DROP sticky bit is set. Anytime a cell is detected lost, the R_LOST_CELLS counter is incremented by the number of lost cells. Anytime the SN state machine transitions from the SYNC state to the OUT_OF_SEQUENCE state, the R_SEQUENCE_ERR counter is incremented. Anytime a misinserted cell is detected the R_MISINSERTED counter is incremented. In all these cases the RCV_STAT_FIFO will be written with the error that occurred and the queue number for which the error occurred, if the corresponding mask bit is not set or this is not the first unmasked sticky bit to be set for this queue since the sticky bit register was last cleared.

Figure 30   Robust SN Algorithm
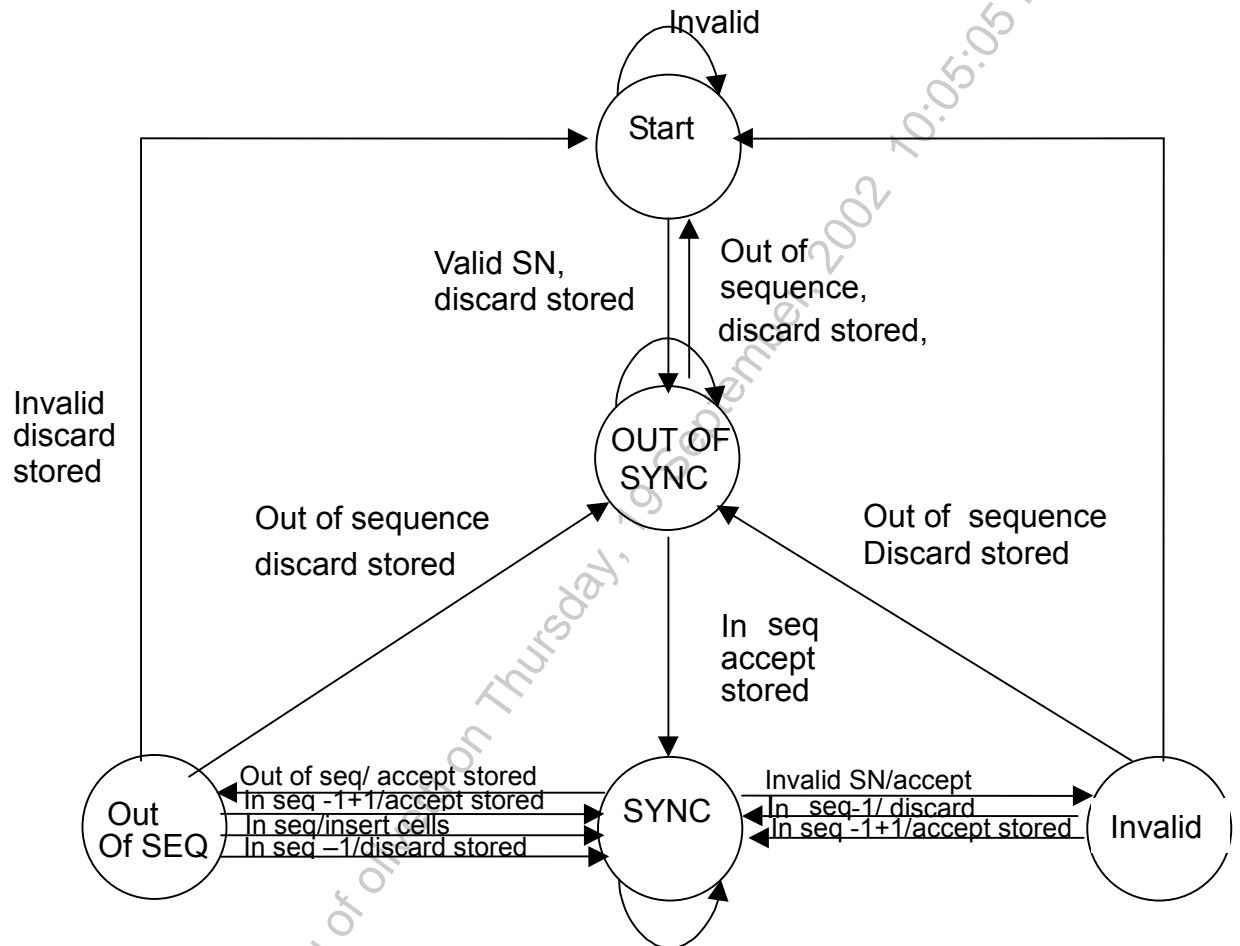
If the cell is stored, the RALP then transfers the cell to the external memory using the R_CHAN_ALLOC fields in the R_QUEUE_TBL. If a cell is accepted, the wr_ptr for the RFTC is advanced to make the cell available to the RFTC. . If a valid cell is not received in time, the queue may enter an underrun condition.

The CDVT value allows the receiver to be configured to store a variable amount of data for that queue before data is emitted. This storage permits the cells to arrive with variable delays without causing errors on the line outputs. This CDVT value is used when the first cell is received after an underrun. The AAL1gator-4/8 also provides protection from buffer overrun and pointer misalignment.

Note that during an underrun, the RALP 'Robust' sequence number processing state machine freezes in its current state. For example, if the state machine is in the SYNC state when underrun occurs, when the next cell arrives, potentially some time later, the state machine will still be in the SYNC state.

Line Data Storage

The RALP reads the ATM cell from the RALP_FIFO, verifies the header, and determines the queue to which the cell belongs. It then locates the data bytes of the cell and writes them into frame buffers provided for that line. Receive queues exist in the external memory, and 128 kBytes of external memory are allocated for receive queues for each A1SP. Since there are eight lines per A1SP, each line is allocated 16 kBytes of memory. This provides for 32 multiframes of E1 data (16 multiframes if T1 signaling is used) or 16 multiframes of T1. The queues are used in a wrap-around fashion. Read and write pointers are used at the frame and multiframe level to access the receive queue.

Figure 31 on page 88 shows cell reception. Read and write pointers are used at the frame and multiframe level to access the correct data byte location. The RALP writes sequential cell data bytes into successive DS0 locations assigned to that queue.

When the RALP encounters signaling bytes, it places them in the receive signaling buffer. The buffers are organized in a multiframe format. There is one signaling nibble per multiframe allocated to each DS0 channel. Therefore, 32 bytes of signaling storage are allocated for each multiframe worth of data buffer. The signalling bytes are not used for UDFmodes.

Figure 31 and the figures that follow illustrate these points and identify how different data formats are stored in the data and signaling buffers.

Figure 31   Cell Reception
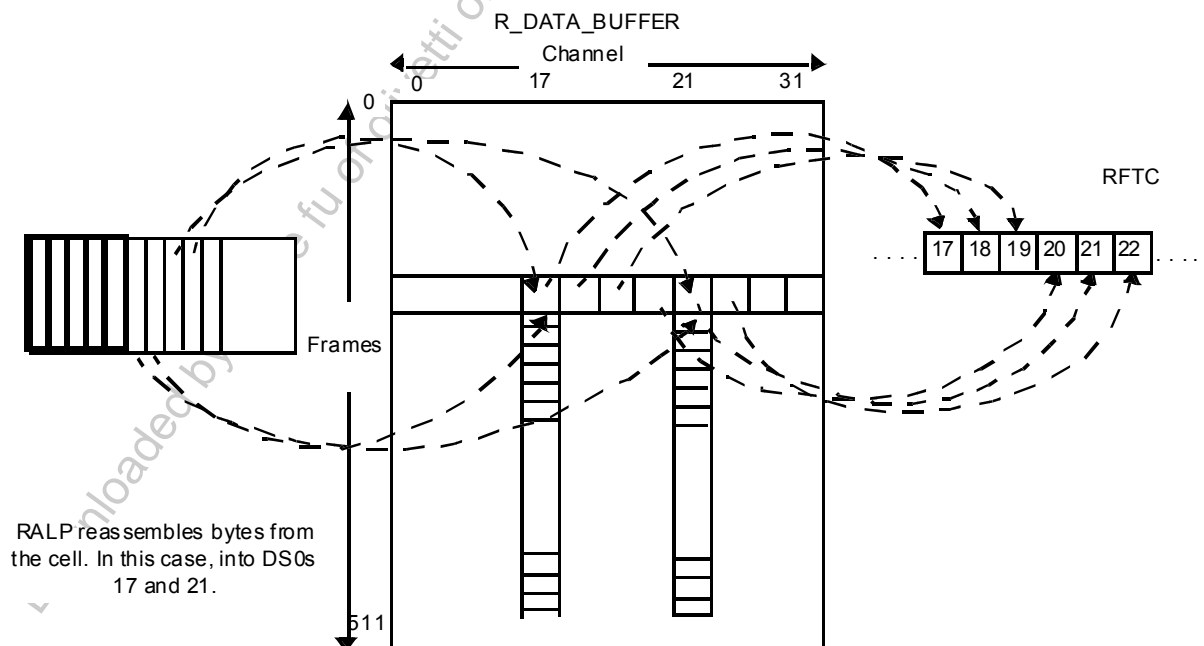
The RALP determines channels by reading from the R_CHAN_ALLOC table and then storing the data in the corresponding timeslots of successive frame buffers in the R_DATA_BUFFER.

Figure 32 shows the contents of the receive data buffer for ESF-formatted T1 data for lines in the SDF-MF mode. Only the first 24 bytes of each frame buffer and the first 24 frame buffers of every 32 are used. This provides storage for 384 frames or 16 multiframes of T1 data.

Figure 32   T1 ESF SDF-MF Format of the R_DATA_BUFFER
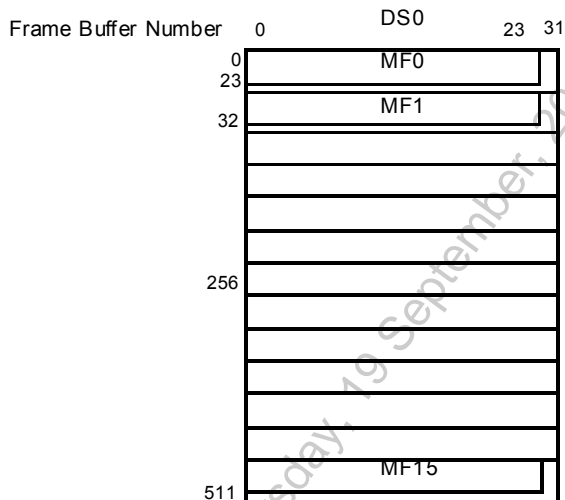
Figure 33 shows the contents of the receive data buffer for SF-formatted T1 data for lines in the SDF-MF mode. Only the first 24 bytes of each frame buffer and the first 24 frame buffers of every 32 are used. This provides storage for 384 frames of T1 data.

Figure 33   T1 SF SDF-MF Format of the R_DATA_BUFFER
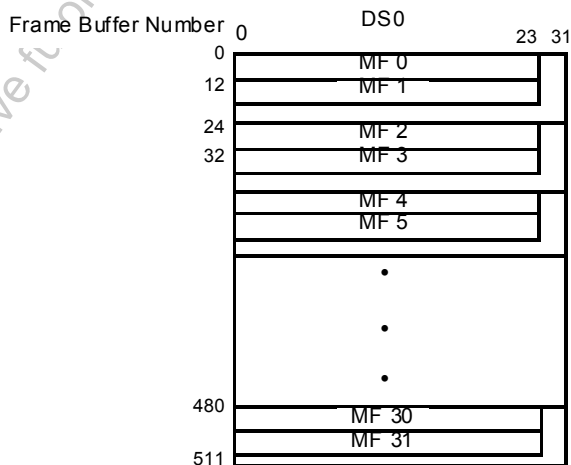
Figure 34 shows the contents of the receive buffer with T1 data for lines in the SDF-FR mode. Only the first 24 bytes of each frame buffer and the first 24 frame buffers of every 32 are used. This provides storage for 384 frames of T1 data.

Figure 34   T1 SDF-FR Format of the R_DATA_BUFFER
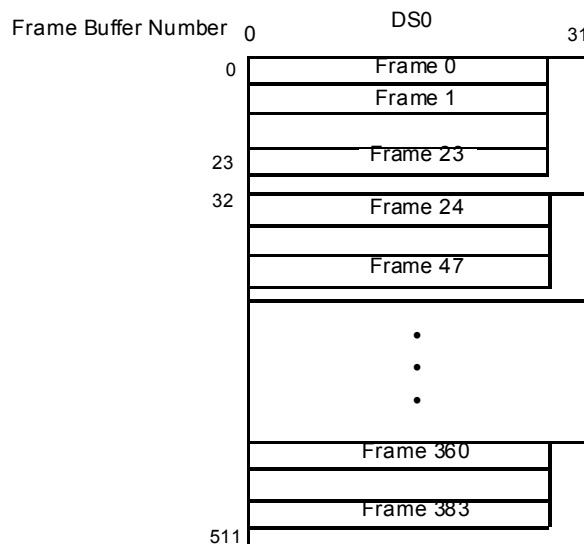
Figure 35 shows the contents of the receive buffer with E1 data for lines in the SDF-MF mode.

Figure 35   E1 SDF-MF Format of the R_DATA_BUFFER
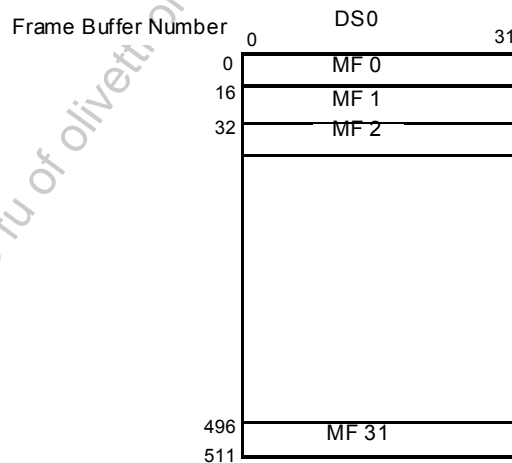
Figure 36 shows the contents of the receive data buffer for E1 SDF-MF data using T1 signaling. In this case a 24 frame multiframe is used.

Figure 36   E1 SDF-MF with T1 Signaling Format of the R_DATA_BUFFER
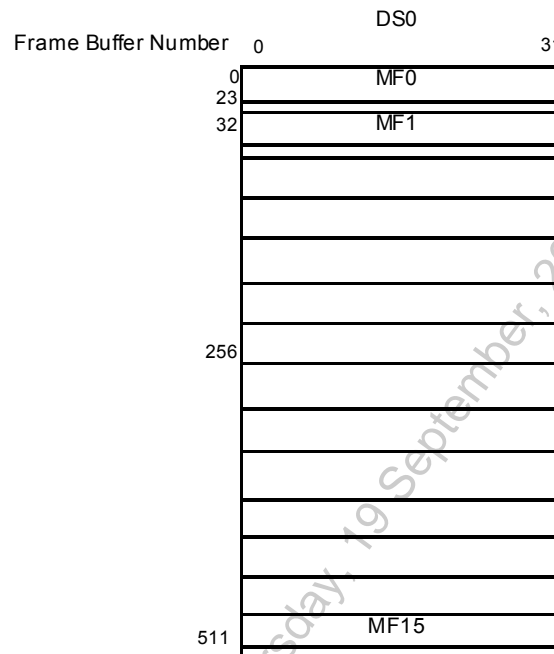
Figure 37 shows the contents of the receive data buffers with E1 data for lines in the SDF-FR mode.

Figure 37   E1 SDF-FR Format of the R_DATA_BUFFER
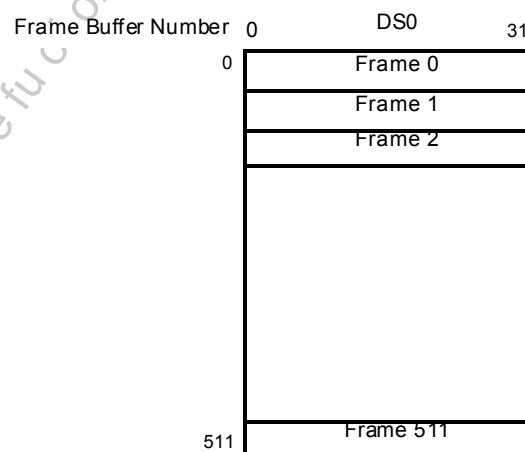
Figure 38 shows the contents of the receive data buffer for lines in the UDF-ML or UDF-HS mode, including T1 unstructured mode. In unstructured modes, each frame buffer contains 256 bits. In the case of unstructured T1, each frame of T1 data also consists of 256 bits. Therefore, each frame buffer contains 1.33 frames of T1 data. This must be considered when determining CDVT numbers.

Figure 38   Unstructured Format of the R_DATA_BUFFER

Frame Buffer Number 0   Data Bits   255

Frame Buffer Number		
0	256 bits	
1	256 bits	
2	256 bits	
511	256 bits	
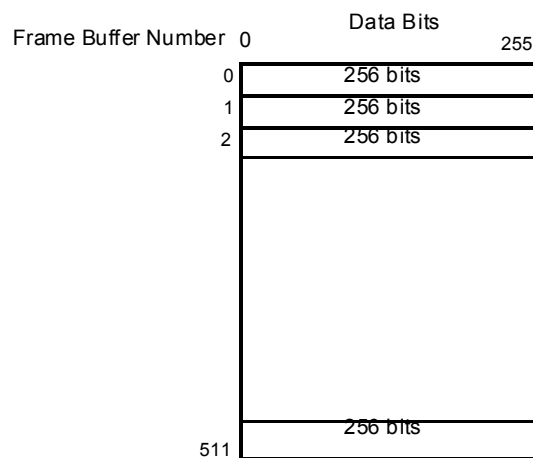
Figure 39 shows the contents of the receive signaling buffer with an ESF-formatted T1 line in the SDF-MF mode. Even channel bytes are stored in the low-byte end of the data words.

Figure 39   T1 ESF SDF-MF Format of the R_SIG_BUFFER

	0					Byte Address			31
0	Channel 0 ABCD	Channel 1 ABCD	• • •	Channel 22 ABCD	Channel 23 ABCD	Channel 24 Not Used	• • •	Channel 31 Not Used
1								
2								
3								
Multiframe								
15								

Figure 40 shows the contents of the receive signaling buffer with an SF-formatted T1 line in the SDF-MF mode. Even channel bytes are stored in the low-byte end of the data words.

Figure 40   T1 SF SDF-MF format of the R_SIG_BUFFER
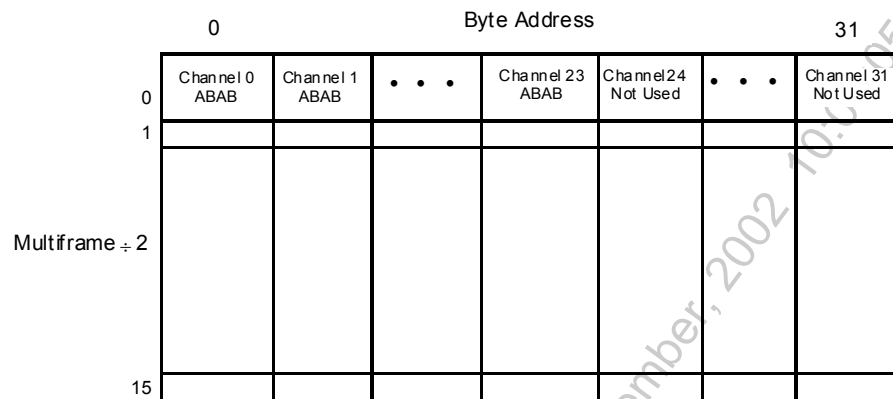
Figure 41 shows the contents of the receive signaling buffer with an E1 line in the SDF-MF mode. Even channel bytes are stored in the low-byte end of the data words.

Figure 41   E1 SDF-MF Format of the R_SIG_BUFFER
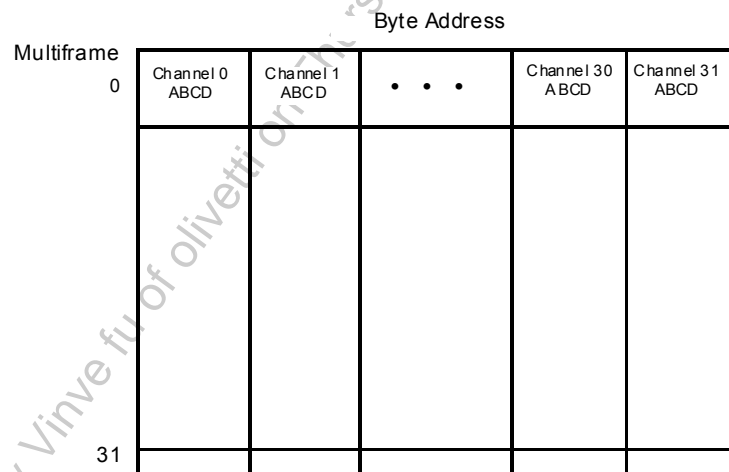
Figure 42 shows the contents of the receive signaling buffer with an E1 line in the SDF-MF mode with T1 signaling. Even channel bytes are stored in the low-byte end of the data words.

Figure 42   E1 SDF-MF with T1 Signaling Format of the R_SIG_BUFFER

Byte Address

Multiframe	Channel 0 ABCD	Channel 1 ABCD	• • •	Channel 30 ABCD	Channel 31 ABCD
0					
15					
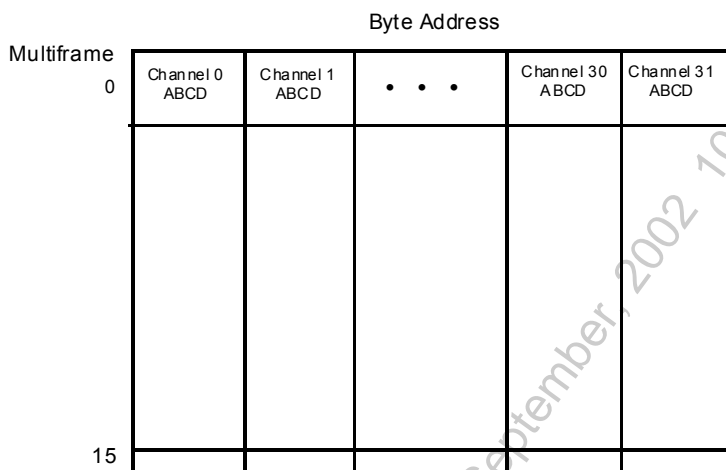
Handling Data and Signaling Bytes in a Structure

A data structure consists of all of the data bytes for that structure, followed by all of the signaling bytes for that structure, if any. In order to locate the data and signaling bytes, the following memory structures are used:

- The R_TOT_SIZE structure provides the number of data and signaling bytes in a structure. An element can be a data byte or a signaling nibble. This value is initialized by the microprocessor.

- The R_TOT_LEFT structure provides the running count of the number of bytes remaining in the structure.

- The R_DATA_LAST structure identifies the last byte of the data structure. As the data bytes are stored in memory, the R_TOT_LEFT structure is decremented by one. When R_TOT_LEFT is equal to R_DATA_LAST, it indicates the end of the data structure. The remaining bytes are stored in the R_SIG_QUEUE. The signaling nibbles are written to the memory until the R_TOT_LEFT equals zero. Once R_TOT_LEFT is zero, R_TOT_SIZE is copied to R_TOT_LEFT and the storing of data and signaling structures is repeated. R_TOT_SIZE and R_DAT_LAST are initialized by the microprocessor.

- The structure used for signaling is determined by the mode of the line and the value of E1_WITH_T1_SIG in the LIN_STR_MODE register. Normally the signaling structure will follow the mode of the line. However if the line is in E1 mode and E1_WITH_T1_SIG is set, then a T1 signaling structure is used. This means that for a single DS0, signaling is updated after 24 data bytes instead of after 16 data bytes.

- If the line is in SDF-MF mode and R_CHAN_NO_SIG = 1 in the R_BUF_CFG word, then the queue is handled as if it is in SDF-FR mode. The structure should be adjusted from a multiframe structure to a frame structure. For example, a channel with two DS0s would have a structure size of two bytes when R_CHAN_NO_SIG is set and a structure size of 49 bytes (include two signaling nibbles) if in T1 mode when R_CHAN_NO_SIG is off.

Underrun

The AAL1gator-4/8 declares an underrun condition for a VC when no data is present in the VC receive buffer. When this situation occurs, the AAL1gator-4/8 plays out conditioned data and frozen signaling onto the timeslots assigned to the VC experiencing underrun. Timeslots generated by other VCs are unaffected. Each time a cell is received after a queue has entered an underrun condition, the UNDERRUN sticky bit is set. RALP does not know about an underrun until a cell is received for the queue that underflowed. To make sure that each underrun is counted only once, RALP will increment the R_UNDERRUN counter when exiting the underrun state and entering the resume state. The initial underrun caused by reset is not counted. Forced underruns due to other errors are not counted by the underrun counter.

Each time a queue enters or exits the Underrun state an entry will be made to the RCV_STAT_FIFO.

If the underrun counter is read and the queue is currently in underrun, the present underrun condition will not be accounted for until the queue exits underrun. To determine if the queue is in underrun, check the value of the R_UNDERRUN bit in the R_LINE_STATE register. When not in UDF-HS mode, the choice of conditioning data while in underrun depends on the value of RX_COND in the R_CH_QUEUE_TBL. Three choices for data exist:

1. Play out the data from R_COND_DATA (Default).

2. Play out pseudorandom data. (For applications that are sensitive to constant data.) The pseudorandom data option uses the data from R_COND_DATA and then replaces the most significant bit with the result of an 18th order polynomial, specifically $x^{18} + x^7 + 1$.

3. Play out old data. (Also for applications that are sensitive to constant data.) The old data option replays out the contents of the data in the receive buffer for that channel. Data is played out from the location of the read pointer. Therefore, the oldest data is played out first.

While in underrun, the signaling data will be frozen if signaling data was not already conditioned.

If in UDF-HS mode and in underrun, the data played out is the conditioned data defined for line 0 for the A1SP, channel 0 or DS3 AIS. There are no old data or pseudorandom data options available for UDF-HS mode.

Bit integrity may be maintained through an underrun condition if at least one cell is lost and less than 6 cells are lost if the BITI_UNDERRUN is set. In order to detect the amount of lost cells, whenever a cell is received, the value of the line rd_ptr is stored. When a cell is received with a SN error when the queue is in underrun, or cell is received when the SN state machine is in the OUT_OF_SEQUENCE state, the current value of the line rd_ptr is compared against the stored value; if the time that has passed since the last cell was received is less than time it should have taken to play out 6 cells worth of data on the line (stored rd_ptr – current rd_ptr < 7 * FRAMES_PER_CELL), less than 7 cells have been lost and the new incoming cell is processed normally as if an underrun never happened. If the new updated wrt_ptr is greater than the read_ptr, the resume bit is set to indicate to the RFTC that the underrun condition is ending and the end-of underrun is set at wrt_ptr. One problem is that underun can persist for a long time. To detect this, a R_LONG_UNDERRUN bit will be set by the RFTC whenever it detects that the rd_ptr is wrapping (rd_ptr+64= wr_ptr and the queue is already in underrun)

To exit an underrun condition, if the BITI_UNDERRUN is not set or more than 6 cells have been lost, the RALP queues up data for one CDVT worth of time before exiting the underrun condition. The R_UNDERRUN bit in R_LINE_STATE word of R_QUEUE_TBL indicates if the queue is in an underrun state. The UNDERRUN sticky bit is set each time a cell is received during the underrun condition. If the data is structured, the RALP searches for a new pointer, and finds the start of the structure. Cells received while the pointer and start of structure are being located are dropped and the POINTER_SEARCH sticky bit is set. The DROPPED_CELL counter is also incremented. If the underrun condition persists, the microprocessor should set the conditioned bits to derive both the data and the signaling from the conditioned areas. The RALP can optionally maintain MF alignment when exiting underrun, when this is done, the new data is written at the first multiframe boundary after the minimum CDVT buffering requirement. This option may add more delay.

By default, in SDF-MF mode, only the CDVT value is taken into account when determining when to play out data. This provides predictable delay but causes a difference in MF alignment on both sides of the ATM network. To align the MF structure in the cell with the external MF pulse, ALIGN_MF should be set. If MF_ALIGN_MODE is set to ALIGN_MF then RALP will not only queue up one CDVT worth of time of data but will also delay the write pointer to the next MF boundary. This method will insure MF alignment across the ATM network but will add variable delay which could be between 0 and 3 ms(T1)/2 ms(E1).

If enabled, when a queue enters UNDERRUN or exits UNDERRUN an entry is made into the RCVN_STAT_FIFO indicating the QUEUE number and setting either the ENTER_UNDERRUN bit or the EXIT_UNDERRUN bit. These conditions can be disabled from causing an entry in the RCVN_STAT_FIFO. When an entry is made in the RCVN_STAT_FIFO, this event may cause the INTB line to go low if the A1SP interrupt is enabled and the RSTAT_FIFO_EMPB_EN is set and A1SPn_INTR_EN is setB.

Refer to line items 5 and 6 in Figure 44 on page 100 that show how a start up after an underrun occurs.

Pointer Processing

When an incoming cell has a cell pointer, the cell pointer is checked against the local pointer value maintained by the RALP. A single pointer mismatch causes no corrective actions. The pointer is ignored and the cell is used as if it contained valid data. If two consecutive pointer mismatches occur, then the RALP forces an underrun condition that causes the receiver to realign to the next incoming pointer. The proper R_COND_DATA and frozen signaling are played out until the new pointer is found and one CDVT worth of time has passed. The PTR_MISMATCH sticky bit is set when the pointer mismatch occurs. The FORCED_UNDERRUN sticky bit is set each time a cell is received and dropped during the forced underrun condition. Then, the underrun and pointer search bits are set, as described previously in section 0 "Underrun" on page 95.

Only one pointer is supposed to be received within a sequence of 8 cells (SN 0 – SN 7). If more than one pointer is received or if no pointers are received in an otherwise error free sequence of 8 cells, then the PTR_RULE_ERROR sticky bit will be set. Pointers are also supposed to be less than or equal to 93 or equal to the dummy value 127. If a pointer is received that does not fall within the legal range and no other SN error occurred the PTR_RULE_ERROR sticky bit will be set.

If the received cell is potentially bad as determined by SN processing, but should contain a pointer, the pointer is not checked against the local pointer. However, in this situation, and when cells are inserted, if the next received pointer mismatches, then a PTR_MISMATCH error is reported and a forced underrun occurs. By not waiting for the second pointer mismatch in these situations where bit integrity may have been lost, a quicker detection and recovery will result.

When a PTR_MISMATCH error occurs, the R_POINTER_REFRAMES counter is incremented. Also, when a single pointer mismatch occurs, signaling information will not be updated until a valid pointer is received to prevent corruption of the signaling information.

Parity checking is also performed on pointers if R_CHK_PARITY is set in R_MP_CONFIG. If a parity error is detected the PTR_PARITY_ERR sticky bit will be set and the R_PTR_PAR_ERR counter will increment. If two consecutive pointer parity errors occur, then the RALP forces an underrun condition and resynchronizes. This resynchronization will cause R_POINTER_REFRAMES to increment.

When any pointer error is detected an entry will be made into the RCVN_STAT_FIFO, unless the corresponding enable bit is not set or this is not the first enabled sticky bit to be set for this queue since the sticky bit register was last cleared.

Figure 43 shows the state machine that checks for valid pointers and structures.

Figure 43   Pointer/Structure State Machine
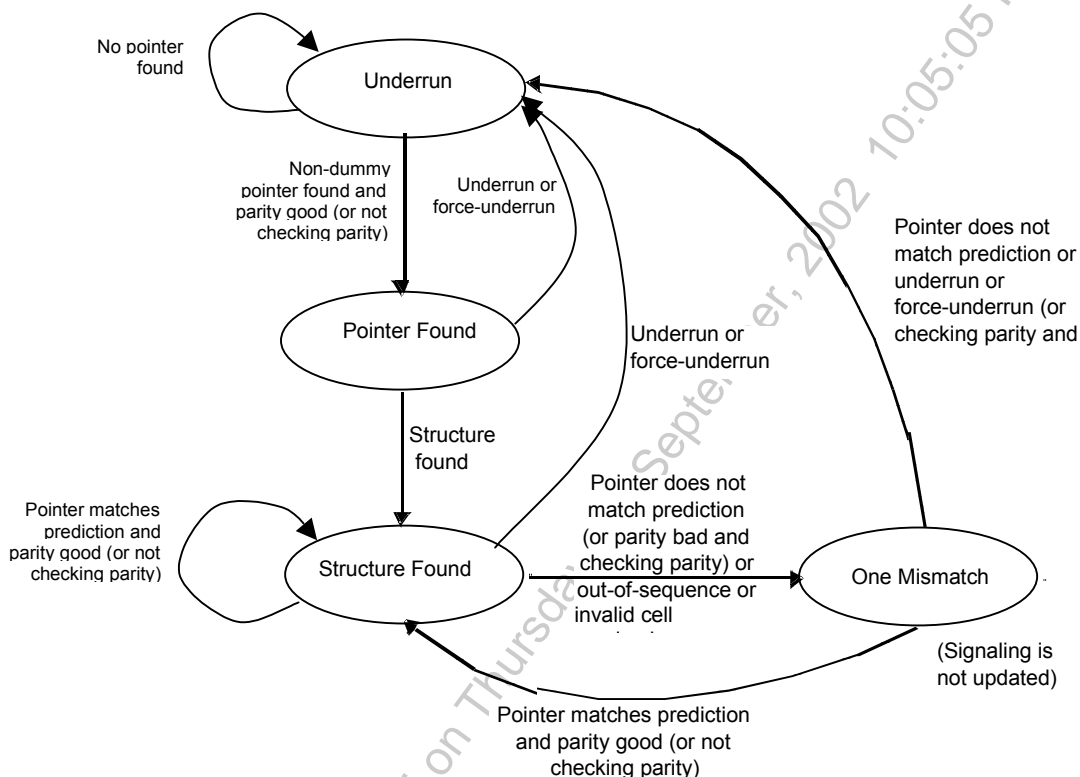
Overrun

Overrun occurs when the data in the buffer is removed at a slower rate than it is filled. However, because the AAL1gator-4/8 buffers are quite large, 16 Kbytes per line, by the time this happens, all data in the buffer can be quite old. Therefore, the buffer size is adjustable, which regulates how much data can be stored in the buffer before an overrun occurs. The R_MAX_BUF field in the R_BUF_CFG register controls the maximum size of the receive buffer. The value of R_MAX_BUF should be equal to or greater than two times CDVT, or CDVT plus two times the number of frames required per cell, whichever is greater. If MF_ALIGN is set, then extra margin should be added for the additional multi-frame of data that may be present. Therefore the value should be increased by 16 frames for E1 or 24 frames for T1.

The overrun condition is declared when the data in the receive buffer exceeds the maximum specified buffer. When a cell is received that causes the maximum buffer depth to be exceeded, the OVERRUN sticky bit is set and the AAL1gator-4/8 enters the forced underrun condition. The incoming cells for the queue are dropped until underrun occurs. Each time a cell is received and dropped in the forced underrun condition, the FORCED_UNDERRUN sticky bit is set. Once underrun occurs, the overrun flag is cleared and the same algorithm used in underrun is followed. Figure 44 on page 100 describes overrun detection, the underrun and recovery process.

Overruns can also occur due to lost/misinserted cells when robust SN processing is done if the buffer is set too small. This is because when the SN processing detects a potentially lost cell event, the cell will be written into the buffer at the correct position assuming that cells have been lost. When Robust SN processing is enabled, the R_MAX_BUF should be equal to or greater than two times CDVT or CDVT plus 9 times the number of frames required per cell to allow for the processing performed on lost/misinserted cells.

Anytime an overflow occurs, the R_OVERRUNS counter is incremented. When an overrun is detected an entry will be made into the RCVN_STAT_FIFO, unless the corresponding enable bit is not set or this is not the first enabled sticky bit to be set for this queue since the sticky bit register was last cleared.

Note:

1. Inserting cells can cause an overrun. The threshold is checked as each byte is written into memory. If an overflow occurs in the middle of a cell, the remainder of the cell will be dropped.

PMC-SIERRA

Figure 44   Overrun Detection
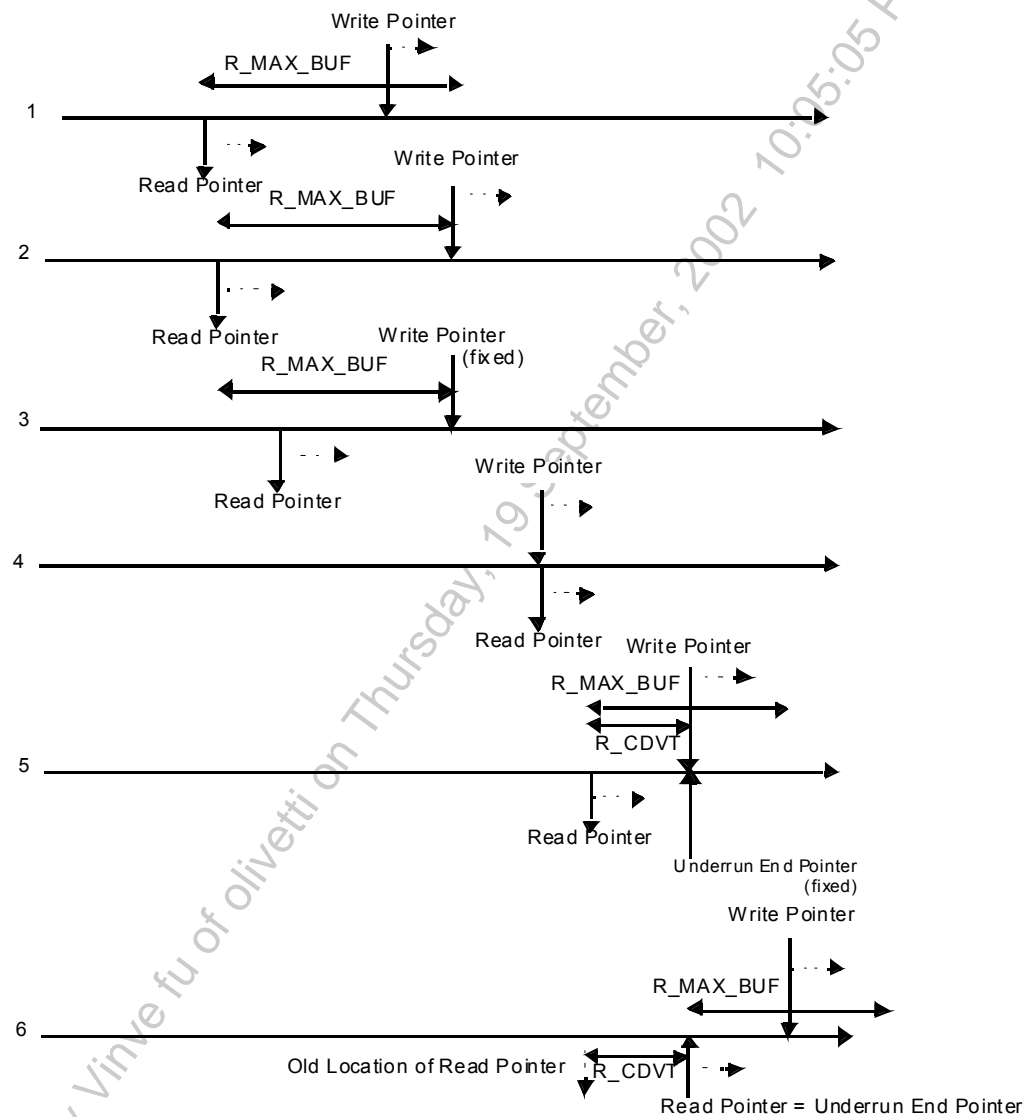
NOTES:

1. Normal operation.

2. If an overrun occurs, then the OVERRUN sticky bit is set and a forced underrun condition is set.

3. During the forced underrun condition, the write pointer is fixed, new data is dropped, and data in the buffer is played out, causing the read pointer to increment. Each time a cell is received and dropped, the FORCED_UNDERRUN sticky bit is set.

4. The read pointer catches up to the write pointer, indicating a forced underrun condition, and the underrun condition is set. Both pointers are advanced for each frame that the queue remains in the underrun condition. Conditioned underrun data is played out.

5. When the first valid cell comes in, the RESUME sticky bit is set. The write pointer and the underrun end pointer are set to the proper frame that is one R_CDVT ahead of the read pointer. Conditioned underrun data is played out.

6. Once the read pointer is equal to the underrun end pointer, then RESUME is complete and real data is now played out. Normal operation now takes place.

Counters and Sticky Bits

The RALP sets sticky bits for overrun, underrun, pointer mismatch, resume, SRTS underrun, SRTS resume, and other conditions. As with most registers, the sticky bits are located in the external RAM. They are set by the AAL1gator-4/8 and must be cleared by the microprocessor. Sticky bits provide a history of events that have occurred. Since these bits can be set with every cell, it is better to use the counters for statistics gathering purposes. The AAL1gator-4/8 increments the counters for incorrect SNs, incorrect SNPs, cells received, underflows, overflows, dropped cells, misinserted cells, lost cells, pointer parity errors, and pointer mismatches. The transfer status bits can be used to detect that a clear was overwritten by toggling transfer bit 15 with each clear of the status bits and then reading the value immediately thereafter. Refer to "R_ERROR_STKY Word Format" for a description of sticky bits.

The first time that a sticky bit is set for a queue that has the corresponding enable bit set in the RCV_STAT_EN_REG register, an entry will be made in the RCV_STAT_FIFO containing the queue number and the sticky bit which was just set. No new entries for sticky bits will be made into the RCVN_STAT_FIFO for this queue until the sticky bit register is cleared.

OAM Cells

When an OAM cell arrives, the RALP stores it in the OAM queue. The RALP notifies the microprocessor of the arrival of OAM cells by setting the OAM_INTR bit in the A1SPn_INTR_REG. The microprocessor reads the OAM cells from the OAM queue. The microprocessor maintains the OAM_HEAD value. The RALP maintains the OAM_TAIL value. The AAL1gator-4/8 also checks the CRC-10 of the cell and records the results in the receive buffer in the CRC_10_PASS parameter.

OAM Cell Interrupt Handling

The AAL1gator-4/8 has the capability to generate an interrupt on the receipt of an OAM cell.

At the end of an OAM cell, the RALP sets the OAM_INTR bit in the A1SPn_INTR_REG. If the OAM_INTR bit is enabled, the A1SPn interrupt bit will be set in the MSTR_INTR_REG which will generate an interrupt if enabled.

Receive Frame Transfer Controller (RFTC)

The RFTC moves data bytes from the receive frame buffer to the appropriate timeslot of the appropriate line. It must perform a timeslot-to-queue translation for each timeslot by reading the receive channel-to-queue table. The RFTC outputs data to the Line Interface Block. For structured data, the RFTC uses the frame pulses generated by the Line Interface Block or internally generated frame pulses to perform a parallel-to-serial conversion on the outgoing data that it reads from a multiframe buffer in the order in which it is needed

A rising edge on the appropriate frame pulse indicates the beginning of a frame or multiframe. The RFTC realigns when a rising edge is seen on these signals. It is not necessary to provide an edge every frame or multiframe. Signaling data is driven for all frames of any multiframe and will change only on multiframe boundaries. Signaling will not change when the queue is in underrun. This is called frozen signaling. The RFTC also has the option of playing out conditioned signaling.

For T1 mode, signaling data may change every 24th frame. For E1 mode, signaling may change every 16th frame. The RFTC accommodates the T1 Super Frame (SF) mode by treating it like the Extended Super Frame mode(ESF) format. The RFTC generally ignores the multiframe pulses that occur on the 12th frame in the multiframe.

A special case of E1 mode exists that permits the use of T1 signaling with E1 framing. Normally an E1 multiframe consists of 16 frames of 32 timeslots, where signaling changes on multiframe boundaries. When E1_WITH_T1_SIG is set in LIN_STR_MODE and the line is in E1 mode, the TFTC will use a multiframe consisting of 24 frames of 32 timeslots.

If GEN_SYNC is set in the LIN_STR_MODE memory register for this line then the RFTC will generate the frame pulses based on its internal counter.

The signaling nibble is valid for each channel when the last nibble of each channel's data is being driven unless the SHIFT_CAS bit is set in the LIN_STR_MODE register. If the SHIFT_CAS bit is set the signaling nibble is valid for each channel when the first channel of each channel's data is being driven. See Figure 45 for an example of signaling bits in a T1 frame. See Figure 46 for an example of signaling bits in the E1 mode.

Figure 45   Output of T1 Signaling Bits (SHIFT_CAS=0)
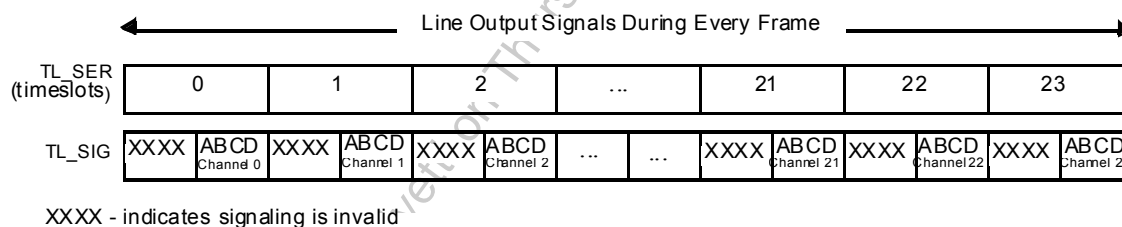
Figure 46   Output of E1 Signaling Bits (SHIFT_CAS=0)
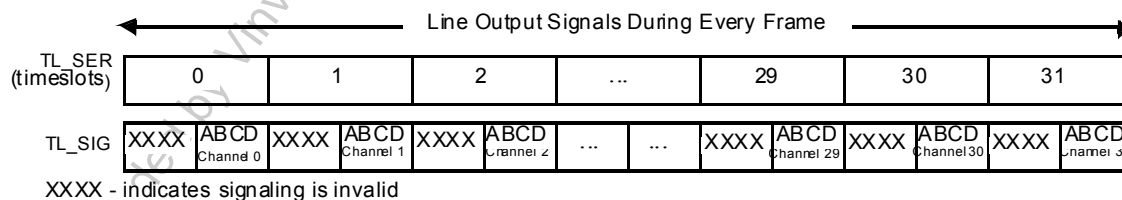
Note:

1. The AAL1gator-4/8 treats all 32 timeslots identically. Although E1 data streams contain 30 timeslots of channel data and 2 timeslots of control (timeslots 0 and 16), data and signaling for all 32 timeslots are stored in memory and can be sent and received in cells.

Each line request is serviced by the main RFTC state machine using a priority encoder (line 0 has the highest priority). The line requests two bytes at a time. This means two channels have to be serviced by the RFTC state machine. When the RFTC state machine receives an attention request from a line, it services the request according to the following pseudocode (please note that the bits discussed are maintained in R_LINE_STATE in the queue tables and should not be confused with the externally accessible sticky bits):

- If the RX_COND_H (or RX_COND_L) = "10", play out the data from the R_COND_DATA area and signaling from the R_COND_SIG area. This is the conditioned state.

- Else if either the R_UNDERRUN bit or the R_RESUME bit is set in R_LINE_STATE, then play out the data and signaling as determined by the value of RX_COND_H (or RX_COND_L). This is the frozen signaling state. The data played out can either be constant, pseudorandom, or old data.

- Else play out the data from the R_DATA_BUF and the signaling from the R_SIG_BUFFER. This is the normal operating state.

- In any of the above states, the conditioned signaling may be forces by setting the RX_SIG_CONDH(or RX_SIG_COND_L)

- When there is no data in the frame buffer, the RFTC sets an underrun bit, R_UNDERRUN. The RALP clears R_UNDERRUN and sets R_RESUME when it encounters the first valid cell in the receive buffer.

- If the R_RESUME bit is set and the R_RD_FR_PTR = R_END_UNDERRUN_PTR, then RFTC clears the R_RESUME bit. This occurs one CDVT time after the first valid cell arrives. If the line is in SDF_MF mode, then R_SIG_RESUME is set to indicate that signaling is not yet available. Once a multiframe has completed and signaling data is available, R_SIG_RESUME will be cleared.

Figure 47 shows the channel-to-queue table operation

PMC-SIERRA

Figure 47   Channel-to-Queue Table Operation

R_CH_TO_QUEUE

R_QUEUE_TBL

20   10

(Note 2)

RFTC

(Note 3)

64
74
84

(Note 1)

(Note 4)   R_DATA_BUFFER

R_ACT_TBL

(Note 6)

(Note 5)

R_COND_DATA

(Note 7)
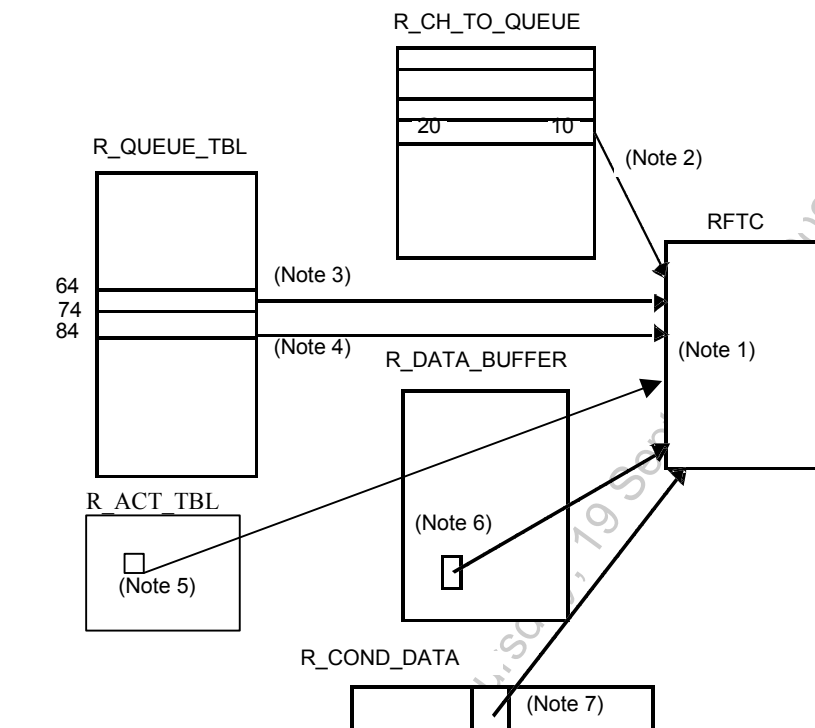
(Notes

1. RFTC fetches R_CH_TO_QUEUE entry for DS0s 6 and 7 for line 2.

2. DS0 6 is being serviced by Queue MOD 32 = 10
   Queue = line 2 x 32 + 10 = 74
   DS0 7 is being serviced by Queue MOD 32 = 20
   Queue = line 2 x 32 + 20 = 84

3. RFTC fetches the queue status and write pointer for queue 74. The queue is not in underrun.

4. RFTC fetches the queue status and write pointer for queue 84. The queue is in underrun.

5. RFTC fetches the data for DS0 6 from the R_DATA_BUFFER.

6. RFTC fetches the data for DS0 7 from R_COND_DATA.

SRTS Support

Figure 48 shows the process implemented for each UDF line enabled for SRTS. It queues the
incoming SRTS values, and, when requested, passes this SRTS value to the AAL1 Clock
Generation Control block. This value, along with a locally calculated SRTS value, is used by
the AAL1 Clock Generation Control block to determine the appropriate line frequency to
synthesize.

The RFTC supports SRTS only for unstructured data formats on a per-line basis. The SRTS_CDVT value in R_SRTS_CONFIG register must be configured correctly so the time delay value of the SRTS data matches the time delay value of the signal data. The RFTC queues the SRTS nibbles and then fetches when requested by the AAL1 Clock Generation Control block. If the SRTS queue overruns or underruns, the value is indicated as invalid.

Note SRTS can be used with the internal clock synthesizers for E1 and T1 mode. If other frequencies are used including DS3 and E3, an external clock synthesizer needs to be used.

Figure 48   Receive Side SRTS Support
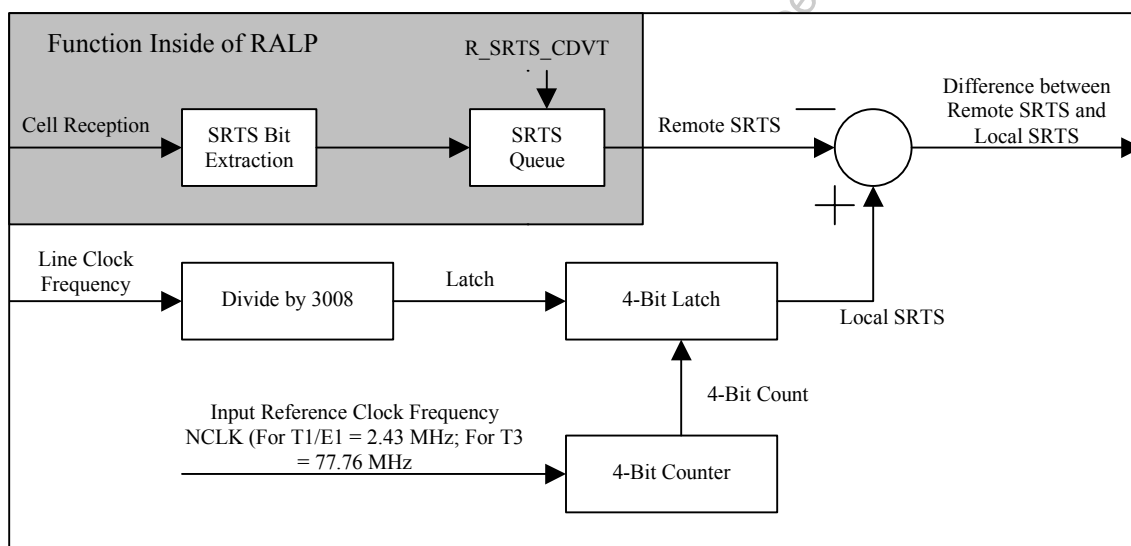
9.3   AAL1 Clock Generation Control

14.1.1 Description

The CGC block is responsible for generating the A1SP transmit line clocks. A given line clock is synthesized internally using a 38.88 MHz system clock (SYS_CLK). Only E1 or T1 clocks can be generated internally. Any other frequency clock must be generated externally and passed into the AAL1gator-4/8 as an input. The frequency of the synthesized clock can be controlled via an external input, the internal SRTS algorithm, or the internal adaptive algorithm. Alternatively a nominal E1 or T1 frequency clock can be generated. Each line clock can be controlled independently. To assist an external source in determining what frequency to use, SRTS and adaptive information is output using the external interface. The CGC block also outputs status information for channel pairs through the external interface.

The CGC consists of four major blocks: External Interface, SRTS, Adaptive, and Frequency Synthesizer. The External Interface passes SRTS and adaptive information out to the user and allows the user to control the Frequency Synthesizer. The SRTS block receives SRTS values and uses the values to determine the frequency to be synthesized. The Adaptive block receives buffer depth information and uses this information to determine the frequency to be synthesized. The Frequency Synthesizer block synthesizes any one of 256 possible frequencies centered around either the T1 or E1 nominal frequency based upon an 8-bit select value. The synthesized frequency is derived from the 38.88 MHz system clock.

The transmit line clock source is controlled by the CLK_SOURCE_TX field and the T1_MODE bit in the LIN_STR_MODE memory register. The eight possible options are:

000   Use external clock. (TL_CLK is an input).

001   LOOPED – Use RL_CLK as the clock source.

010   NOMINAL Synthesized – Generate a clock of the nominal (T1 or E1) frequency from SYS_CLK.

011   SRTS Synthesized- Generate a clock frequency based on the received SRTS values.

100   ADAPTIVE Synthesized- uses receive buffer depth to control TL_CLK

101   Externally controlled Synthesized: Generate a clock frequency based on the values provided by CGC_SER_D pin. This mode is used for external implementations of SRTS or Adaptive clocking.

110   Use common external clock (CTL_CLK) (Not valid in H-MVIP mode)

111   Use common external clock (CTL_CLK) and drive TL_SIG data onto TL_CLK pin. Not valid in H-MVIP mode.

Options "010" through "101" involve the CGC block. Each line is independently controlled. When a particular line is not in one of these modes it is held in reset.

14.1.2 CGC BLOCK DIAGRAM
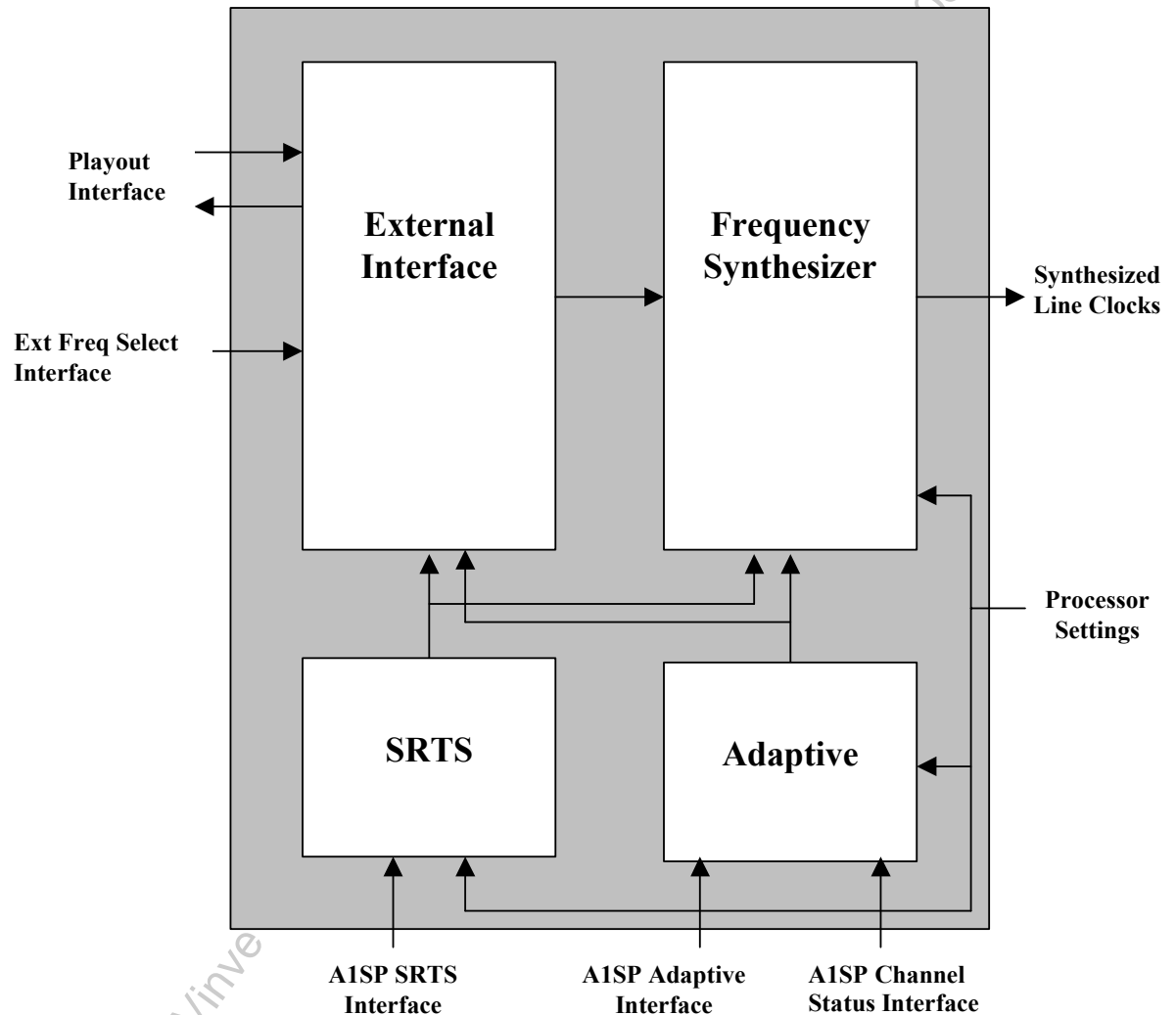
9.3.1 Functional Description

The CGC consists of four major blocks: External Interface, SRTS, Adaptive, and Frequency Synthesizer.

External Interface

The External Interface transmits and receives line clock related information that allows the line clocks to be controlled externally. It serves two functions:

1. external playout of Adaptive, SRTS, and Channel Status data

2. receipt of external frequency select data.

The External Interface block transmits either SRTS, Adaptive, or Channel Status information on a line basis which enables an external decision to made about altering the line clock frequency. The SRTS output data is the difference between the local and remote SRTS nibble. The adaptive output data is the averaged (using the adaptive algorithm described in section 0) relative buffer depth in units of bytes. If the adaptive weighting is set to 0 this value becomes the raw buffer depth in units of bytes.

Status information is played out on the External Clock Generation Control (CGC) Interface which includes the external output signals: CGC_DOUT[3:0], CGC_LINE[3:0], SRTS_STBH, and ADAP_STBH.

Information is played out for all chip modes (Direct, H-MVIP).

The interface clocking operates according to the following pseudocode:

- If a channel pair is being serviced, start a state machine to play out the channel status, as shown in Table 5, asserting ADAP_STBH as each state is played out.

- Else, if an SRTS difference value is ready, it is played out with SRTS_STBH asserted and ADAP_STBH deasserted.

- Else if a cell is received and a valid averaged relative buffer depth can be computed, start a state machine to play out the averaged relative buffer depth and queue number, as shown in Table 6, asserting ADAP_STBH as each state is played out. Note that buffer depth information is not played out if the queue is in a resume state or if cells are lost as this data would not represent the steady state buffer depth.

- Once playout of a certain data type has begun it will not be interrupted. Also if SRTS data or buffer depth data is temporarily blocked due to other data being played out, the blocked data will be played out at the next available oppurtunity.

SRTS Data Output

In SRTS mode the CGC block puts out a 4 bit value which represents the difference between the local SRTS value and the received SRTS value. Figure 49 below shows a typical CGC output in SRTS mode.

Figure 49   SRTS Data
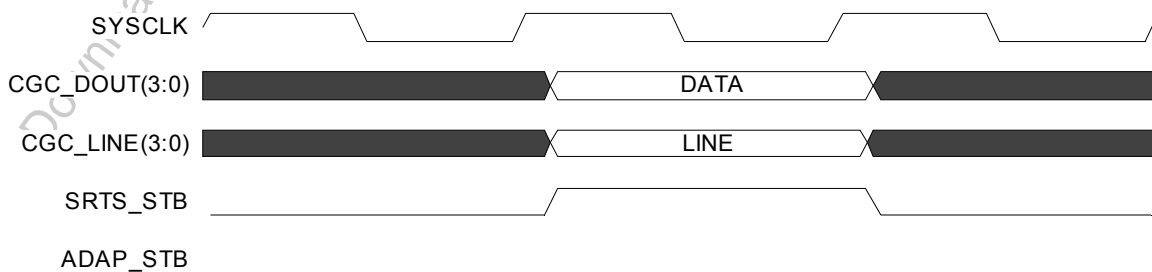
Channel Underrun Status Output

Figure 50 shows an example of the channel underrun status being reported on the CGC Interface. In this example for a structured line, the line number is 3, the channel pair is 7 (channels 14 and 15), and the high channel (channel 15) is in underrun while the low channel (channel 14) is in normal mode. For an unstructured line, there is no per-channel status, only per-line status; Channel(4:1) can be ignored, and both Underrun_h and Underrun_l will have the same value – either can be used. These values are encoded into the 4 data words following the format shown in Table 5.

Status is only reported when a channel enters or exits underrun.

Figure 50   Channel Status Functional Timing
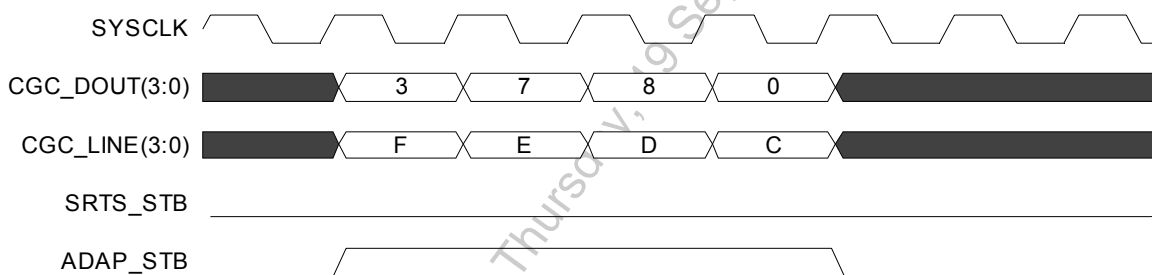
Table 5   Channel Status

CGC_LINE(3:0) Value	CGC_DOUT(3:0) Value			
	3	**2**	**1**	**0**
15	0	Line(3)	Line(2)	Line(1)
14	Channel(4)	Channel(3)	Channel(2)	Channel(1)
13	Underrun_h	0	underrun_l	0
12	0	0	0	0
	0	0	0	0

Adaptive Status Output

The AAL1gator-4/8 provides the average buffer depth in units of bytes for an external circuit to generate an adaptive clock. (If adap_filt_size is set to zero it will provide the current buffer depth with no averaging). The general mechanism is often termed "buffer centering". A clock delta value is determined externally by subtracting the nominal buffer depth (value of CDVT) from the actual buffer depth. This delta value is then transformed into the frequency selection for an external frequency synthesizer. The closed-loop action of this circuit causes the delta value to find a center point. When the delta is above the center point, there is too much data buffered and the frequency must be increased. When the delta is below the center point, the frequency must be reduced.

The AAL1gator-4/8 implements a programmable weighted moving average internally. However, if an alternative adaptive algorithm is desired then this information can be processed externally. In High Speed mode, since an E3 or DS3 clock cannot be internally synthesized, this information can be used for external synthesis of an E3 or DS3 clock.

Table 6 below shows an example of the CGC Interface for adaptive data. In this example the line number is 5, and the average buffer depth in units of bytes is 43. (For unstructured lines, the average buffer depth is given in units of bytes. For structured lines, the average buffer depth is given in units of frames, ie, buff_depth(4:0) should be ignored.) These values are encoded into the 6 data words following the format shown.

Table 6   Buffer Depth

CGC_LINE_OUT[3:0] Value	CGC_DOUT[3:0] Value			
	3	2	1	0
5	queue(7)	queue(6)	queue(5)	queue(4)
4	queue(3)	queue(2)	queue(1)	queue(0)
3	0	0	buff_depth(13)	buff_depth(12)
2	Buff_depth(11)	buff_depth(10)	buff_depth(9)	buff_depth(8)
1	Buff_depth(7)	buff_depth(6)	buff_depth(5)	buff_depth(4)
0	Buff_depth(3)	buff_depth(2)	buff_depth(1)	buff_depth(0)

Figure 51   Adaptive Data Functional Timing
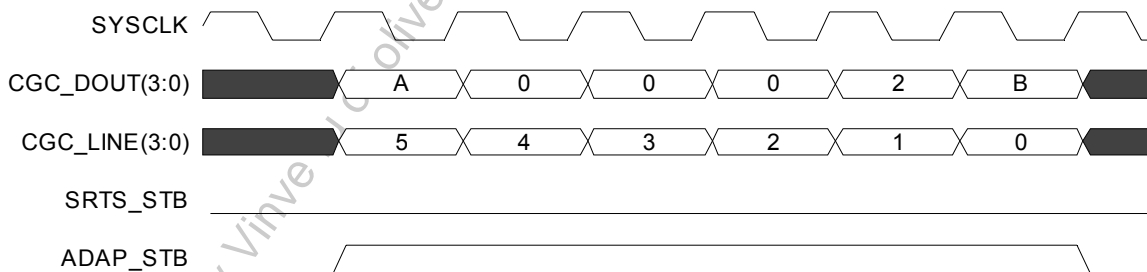
Ext Freq Select Interface

The Ext Freq Select Interface allows an external source to directly select the line clock frequency from any one of 171 T1 or 240 E1 frequencies centered around the nominal clock rate. For T1 the legal input values are –83 to 88. For E1 the legal input values are –128 to 111. Any values outside of this range will be clamped to these levels. These levels correspond to a +/-200 ppm T1 clock and a +/- 100 ppm E1 clock. See the section titled "Frequency Synthesizer" on page 114 for more detailed information.

PMC-SIERRA

The External Interface block has two input ports that allow an external source to control the frequency synthesizers internal to the CGC. These two ports are CGC_SER_D and CGC_VALID. CGC_SER_D contains the data that selects one of the 171/240 frequencies and CGC_VALID indicates when this data is valid. There should be a rising edge of CGC_VALID at the start of each timing message. And CGC_VALID should go low at the end of each timing message. CGC_VALID only needs to be deactivated for one cycle between timing messages. The CGC block decodes the incoming line number, and if the address matches one of its 32 line numbers, passes the data on to the appropriate frequency synthesizer.

Figure 52 below shows an example of where Line 19 is being programmed to run with the frequency setting of –79 (Two's complement). See the section on the Frequency Synthesizer block for a discussion of the different frequency settings which range from -83 to 88 in T1 mode and –128 to 111 in E1 mode.

Figure 52   Ext Freq Select Functional Timing
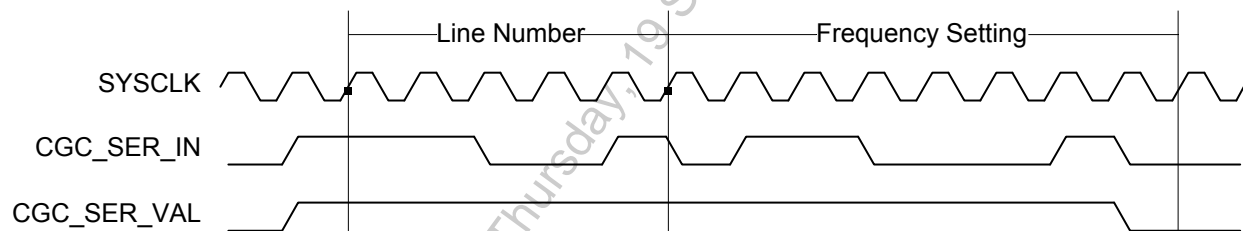
SRTS

SRTS functionality is enabled by setting the SRTS_EN bit in the LIN_STR_MODE memory register. To disable SRTS processing on the chip via hardware, tie the NCLK/SRTS_DISB signal to ground.

When enabled, the local SRTS values, which are calculated within this SRTS block, are subtracted from the SRTS values received in the cells received by RALP, which represent the remote SRTS values. This SRTS difference is sent by the SRTS block to the Frequency Synthesizer to indicate what frequencies should be synthesized for each line. The SRTS difference is also played out externally.

SRTS is supported for unstructured data formats on a per-line basis. SRTS support requires an input reference clock (NCLK). The input reference frequency is defined as $155.52 / 2^n$ MHz, where n is chosen so the reference clock frequency is greater than the frequency being transmitted, but less than twice the frequency being transmitted (2 x TL_CLK > NCLK > TL_CLK). For T1 or E1 implementations, the input reference clock frequency is 2.43 MHz and must be synchronized to the ATM network. Figure 53 shows the process implemented for each UDF line enabled for SRTS. The CGC generates a local SRTS value from the network clock (NCLK) and the local TL_CLK. It makes a request to the A1SP to read a new remote SRTS value from the SRTS queue and, at the appropriate time, generates a 4-bit two's complement code that indicates the difference between the locally generated SRTS value and the incoming SRTS value. The value of this code ranges from –8 (1000) to +7 (0111). A higher value than had been output previously indicates the remote clock is running faster than the local clock. A lower value than had been output previously indicates the remote clock is running slower than the local clock. The AAL1_CGC uses this value to synthesize the TL_CLK. Since the frequency synthesizers accept 8 bit input values, the lower 4 bits will be set to zero and the upper 4 bits will receive the SRTS nibble. The synthesizers should be set for low resolution mode by programming the HI_RES_SYNTH register bit in the LIN_STR_MODE memory register to zero.

Figure 53   Receive Side SRTS Support
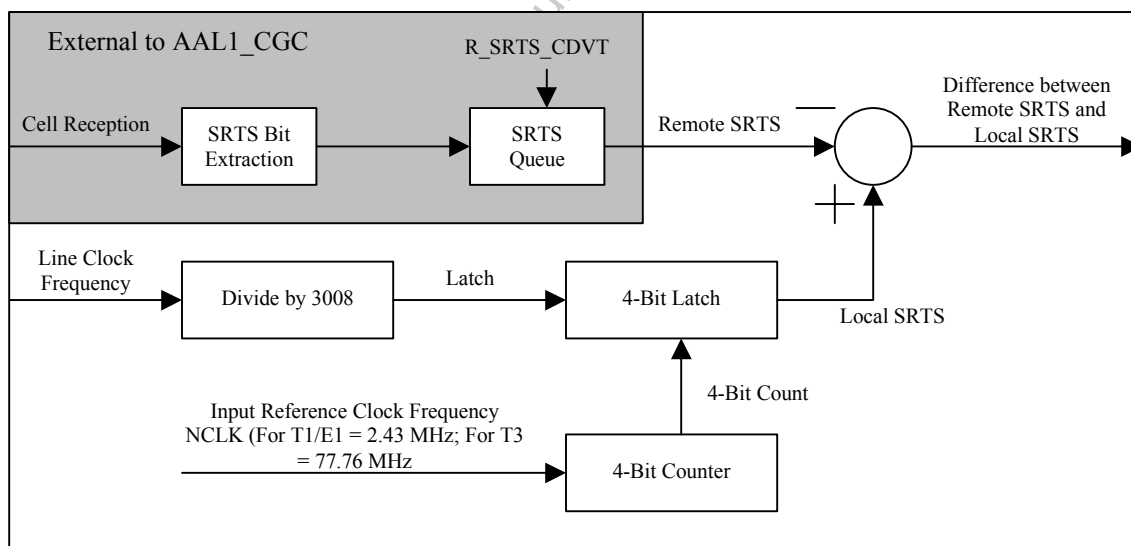
Adaptive

The Adaptive block determines the appropriate line clock frequencies based on the buffer depth received from the A1SP. Every time a cell is received on a particular line, the Adaptive block is given the current depth of the receive buffer. If the buffer depth is increasing, then the local line clock is running slower than the remote line clock. If the buffer depth is decreasing, then the local line clock is running faster than the remote line clock. Therefore, the Adaptive block will adjust the local line clock according to the buffer depth by passing the appropriate value to the Frequency Synthesizer.

When CDV is accounted for, the buffer depth will not only vary due to differences in the line clock frequencies, but also due to differences in the interarrival time of cells. In order to truly measure the difference in the remote line clock frequency and the local line clock frequency the buffer depth needs to be filtered. The method the Adaptive block uses is a weighted moving average algorithm where the amount of weighting is programmable. The following formula is used to determine the average buffer depth.

$$\frac{(PrevAvgDepth * (N-1)) + CurBufDepth}{N}$$

$N = \text{Weighting} = 2^{\wedge}\text{Adap\_Filt\_Size}$

$PrevAvgDepth = \text{Previous Average Buffer Depth}$

$CurBufDepth = \text{Current Buffer Depth}$

N must be a power of 2 and is controlled by the ADAP_FILT_SIZE field in the A_ADAP_CFG register. If ADAP_FILT_SIZE equals '0' then no filtering is done.

Note that ADAP_FILT_SIZE also controls the buffer depth information that is presented on the CGC output port.

To enable quicker lock times, the window size will start small and will grow until it matches the ADAP_FILT_SIZE parameter. The average value will reset when either the line is disabled or it goes into underrun.

The value sent to the synthesizers is a truncated value of the Relative Buffer Depth. Relative Buffer Depth is the difference between the calculated average buffer depth and the CDVT setting (R_CDVT in the receive queue table) which represents the ideal buffer depth value. The Relative Buffer Depth is limited to represent a value of +/- 8 frames (ie, +/- 256 bytes) from the CDVT setting. Relative buffer depths that exceed this value are truncated to a max/min value of +/- 8 frames. Using a signed 8 bit data value, this corresponds to 1 bit representing 2 bytes of data.

The frequency synthesizer will further cap this value by limiting the frequency range to +/- 200 ppm for T1 and +/-100 ppm for E1.

Note that adaptive clocking, in general, is not well suited for voice applications since low frequency or DC changes of the CDV will pass through most filters and cause frame slips. Adaptive clocking is only supported for unstructured connections inside the AAL1gator. If adaptive clocking is desired for structured connections, it will need to be processed externally. The buffer depth will be played out for each queue, but the information will be in frames. (the bottom 5 bits are invalid).

If an alternative algorithm is desired, this can be handled externally following an architecture similar to what is shown in Figure 54. Note that the internal synthesizers can be used via the CGC control port, so that only the adaptive algorithm has to be in external logic and not the synthesizers.

Figure 54   Direct Adaptive Clock Operation
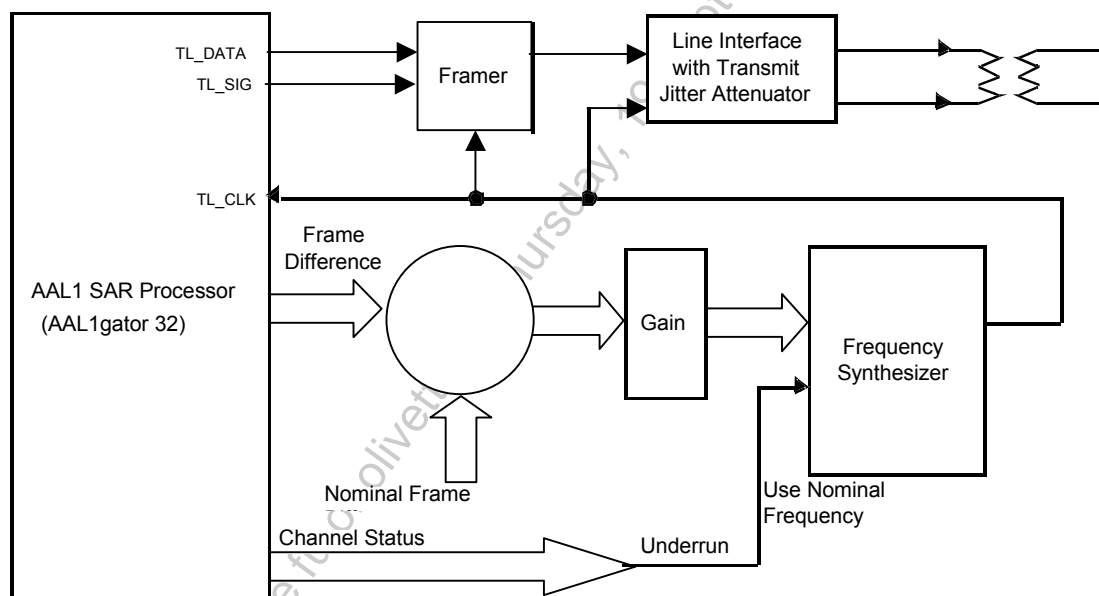
Frequency Synthesizer

The Frequency Synthesizer block receives an 8-bit two's complement number to select a frequency setting. Although possible (with 8 bits) to input a value of –128 to 127 this input is limited internally to –83 to 88 for T1 and –128 to 111 for E1. This is done in order to not exceed the frequency range specification of +/- 200ppm for T1 and +/- 100ppm for E1. Based on this value, the Frequency Synthesizer synthesizes a clock for each line. The line frequency is synthesized by dividing down the 38.88 MHz system clock. The method for dividing down the system clock is dependent on whether the line is in T1 or E1 mode.

In order for this block to work properly, the system clock must be 38.88 MHz. The accuracy of the synthesized clock is dependent on the accuracy of SYS_CLK. Therefore, if a 50 ppm T1 clock is desired, SYS_CLK needs to be a 38.88 MHz clock signal with 50 ppm accuracy. To lock the synthesized clock to a network clock, be sure SYS_CLK is derived from the network clock. This block can be accessed through the CGC serial data in port, and it is also used internally by the SRTS and adaptive algorithms

To minimize jitter, long and short cycles are distributed. The resultant synthesized clocks meet G.823 and G.824 specs for jitter. Jitter can be further reduced by running the TL_CLKs through a jitter attenuator in the framer or LIU.

The synthesizers can be set to operate in normal or high resolution mode. In normal mode only the 4 high order bits of the frequency setting value are monitored to generate 1 of 16 frequencies. In high resolution mode all 8 bits are monitored to generate 1 of 256 frequencies. SRTS requires the use of normal mode, while high resolution mode is recommended for Adaptive clock recovery.

T1 Mode

In T1 mode the 1.544 MHz nominal frequency is synthesized by maintaining a certain ratio of long cycles to short cycles. The long cycle is comprised of 26 SYS_CLK periods (38.88 MHz). The short cycle is comprised of 25 SYS_CLK periods. For the nominal frequency in high resolution mode, the synthesis period is comprised of 3088 cycles. 560 of the 3088 cycles are long cycles and the remaining 2528 are short cycles. For the nominal frequency in low resolution mode, the synthesis period is comprised of 193 cycles, 35 of those 193 cycles are long cycles and the remaining 158 are short cycles. In order to alter the frequency the number of short cycles is increased or decreased according to the frequency select value. The resultant frequency is calculated using the following equation:

$$\frac{38.88 * (M + N)}{26M + 25N}$$

M = # long cycles

N = # short cycles

Table 7 shows results of this calculation for various values of frequency select. . The average increment in frequency is ~3.6 Hz.

Table 7   Frequency Select – T1 Mode

Freq Select	M	N	Total # Cycles	Frequency
-128	560	2432	2992	1.5436433121
-127	560	2432	2992	1.5436433121
-126	560	2432	2992	1.5436433121
-125	560	2432	2992	1.5436433121
…	…	…	…	…
-96	560	2432	2992	1.5436433121
-95	560	2433	2993	1.5436471447
-94	560	2434	2994	1.5436509747
-93	560	2435	2995	1.5436548021
-92	560	2436	2996	1.5436586271
…	…	…	…	…
-4	560	2524	3084	1.5439855782
-3	560	2525	3085	1.5439891871
-2	560	2526	3086	1.5439927937
-1	560	2527	3087	1.5439963980
0	560	2528	3088	1.5440000000
1	560	2529	3089	1.5440035997
2	560	2530	3090	1.5440071970
3	560	2531	3091	1.5440107921
4	560	2532	3092	1.5440143848
…	…	…	…	…
93	560	2621	3181	1.5443251545
94	560	2622	3182	1.5443285482
95	560	2623	3183	1.5443319399
96	560	2624	3184	1.5443353293
…	…	…	…	…
124	560	2624	3184	1.5443353293
125	560	2624	3184	1.5443353293
126	560	2624	3184	1.5443353293
127	560	2624	3184	1.5443353293

* Note that the synthesizers are limited internally to a freq select range of –96 to 96 in order to maintain a +/- 230 ppm range.

E1 Mode

In E1 mode the 2.048 MHz nominal frequency is synthesized by maintaining a certain ratio of long cycles to short cycles. The long cycle is comprised of 19 SYS_CLK periods (38.88 MHz). The short cycle is comprised of 18 SYS_CLK periods. For the nominal frequency in high resolution mode, the synthesis period is comprised of 1024 cycles. 1008 of the 1024 cycles are long cycles and the remaining 16 are short cycles. For the nominal frequency in low resolution mode, the synthesis period is comprised of 64 cycles, 63 of the cycles are long and the remaining 1 is short. In order to alter the frequency the number of long cycles is increased or decreased according to the frequency select value. The resultant frequency is calculated using the following equation:

$$\frac{38.88 * (M + N)}{19M + 18N}$$

M = # long cycles

N = # short cycles

Table 8 shows results of this calculation for various values of frequency select.

Table 8   Frequency Select – E1 Mode

Freq Select	M	N	Total # Cycles	Frequency
-128	1136	16	1152	2.0478127286
-127	1135	16	1151	2.0478140301
-126	1134	16	1150	2.0478153339
-125	1133	16	1149	2.0478166399
-124	1132	16	1148	2.0478179482
-123	1131	16	1147	2.0478192589
-122	1130	16	1146	2.0478205717
-121	1129	16	1145	2.0478218869
-120	1128	16	1144	2.0478232044
-119	1127	16	1143	2.0478245242
…	…	…	…	…
-4	1012	16	1028	2.0479934413
-3	1011	16	1027	2.0479950762
-2	1010	16	1026	2.0479967142
-1	1009	16	1025	2.0479983555
0	1008	16	1024	2.0480000000
1	1007	16	1023	2.0480016477
2	1006	16	1022	2.0480032986
3	1005	16	1021	2.0480049528
4	1004	16	1020	2.0480066102
…	…	…	…	…
109	899	16	915	2.0482008175
110	898	16	914	2.0482028818
111	897	16	913	2.0482049507
112	896	16	912	2.0482070240
…	…	…	…	…
124	896	16	912	2.0482070240
125	896	16	912	2.0482070240
126	896	16	912	2.0482070240
127	896	16	912	2.0482070240

* Note that the frequency synthesizers are limited internally to a freq select range of –128 to 112 in order to maintain a +/- 100 ppm range. The average increment in frequency is ~1.66 Hz.

9.4   Processor Interface Block (PROCI)

The microprocessor interface block provides normal and test mode registers, as well as memory mapped registers and the logic required to connect to the microprocessor interface. The normal mode registers and memory mapped registers are required for normal operation, and test mode registers are used to enhance the testability of the AAL1gator-4/8. A general memory map for the register set can be seen in the following four figures:

Figure 55 shows the memory region broken into three blocks. The first block, A1SP SRAM, is the memory mapped registers which are mostly contained within SRAM. The second block, Internal Registers, is composed of configuration registers common to the entire chip that are contained internally to the chip. The final block is the test registers.

Figure 55   Memory Map

00000 1FFFF	A1SP SRAM
80000 BFFFF	Internal Registers
C0000 FFFFF	Test Registers

Figure 56 shows the structure of the A1SP block within the external SSRAM. The addresses listed in the figure represent the relative location within the A1SP block. The first block, Control Registers, contains registers which are common to both the transmit block and the receive block. The second and third blocks contain registers which are specific to the transmit block and the receive block respectively.

Figure 56   A1SP SRAM Memory Map

00000 0001F	Control Registers
00020 07FFF	Transmit Data Structures
08000 1FFFF	Receive Data Structures

Figure 57 shows the Control Registers block in more detail.

Figure 57   Control Registers Memory Map

00000	Reserved
00001	HS_LIN_REG
00002 00003	Unused
00004	P_FILL_CHAR
00005 0000F	Unused
00010	LIN_STR_MODE0
00011	LIN_STR_MODE1
00012	LIN_STR_MODE2
00013	LIN_STR_MODE3
00014	LIN_STR_MODE4
00015	LIN_STR_MODE5
00016	LIN_STR_MODE6
00017	LIN_STR_MODE7
00018 0001F	Unused

Figure 58 shows the format of the Transmit Data Structures block in more detail.

Figure 58   Transmit Data Structures Memory Map

00020 0002F	T_SEQNUM_TBL
00030 0003F	T_ADD_QUEUE
00040 003FF	Unused
00400 0047F	T_COND_SIG
00480 004FF	T_COND_DATA
00500 006FF	Unused
00700 007FF	Reserved (Frame Advance FIFO)
00800 00FFF	Reserved (Transmit Calendar)
01000 013FF	Reserved (Transmit Signaling Buffer)
01400 0143F	T_OAM_QUEUE
01440 01FFF	Unused
02000 03FFF	T_QUEUE_TBL
04000 07FFF	Reserved (Transmit Data Buffer)

Figure 59 shows the format of the Receive Data Structures block in more detail.

Figure 59   Receive Data Structures

Address	Description
08000 08001	R_OAM_QUEUE_TBL
08002	R_OAM_CELL_CNT
08003	R_DROPPED_OAM_CELL_CNT
08004 0801F	Unused
08020 0802F	Reserved (SRTS Queue Pointers)
08030 08037	Unused
08038 0803F	R_SRTS_CONFIG
08040 0807F	Unused
08080 080FF	R_CRC_SYNDROME
08100 081FF	Unused
08200 0827F	R_CH_TO_QUEUE_TBL
08280 083FF	Unused
08400 0847F	R_COND_SIG
08480 084FF	R_COND_DATA
08500 087FF	Unused
08800 08FFF	Reserved (Receive SRTS Queue)
09000 09FFF	Reserved (Receive Signaling Buffer)
0A000 0BFFF	R_QUEUE_TBL
0C000 0DFFF	Unused
0E000 0FFFF	R_OAM_QUEUE
10000 1FFFF	Reserved (Receive Data Buffer)

Figure 60 below shows the format of the Normal Mode Registers block in more detail.

Figure 60   Normal Mode Registers Memory Map

80000 800FF	Command Reigsters
80100 8011F	RAM Interface Registers
80120 801FF	UTOPIA Interface Registers
80200 80FFF	Line Interface Reigsters
81000 812FF	Interrupt and Status Registers
82000 82FFF	CAS Change Configuration and Status Registers
84000 84FFF	DLL Control and Status Registers

9.4.1   Interrupt Driven Error/Status Reporting

The interrupt logic has several layers and can be sourced from any of three blocks. The three blocks are the UTOPIA block, the RAM interface block, and the A1SP block. The top layer of the interrupt logic, the Master Interrupt Register, indicates from which block the interrupt came from. Once the block is determined the processor can access the appropriate block to determine the interrupt cause.

Figure 61 shows the registers in the interrupt tree. The microprocessor traverses the tree based on the value of individual bits within each register.

Figure 61   Interrupt Hierarchy
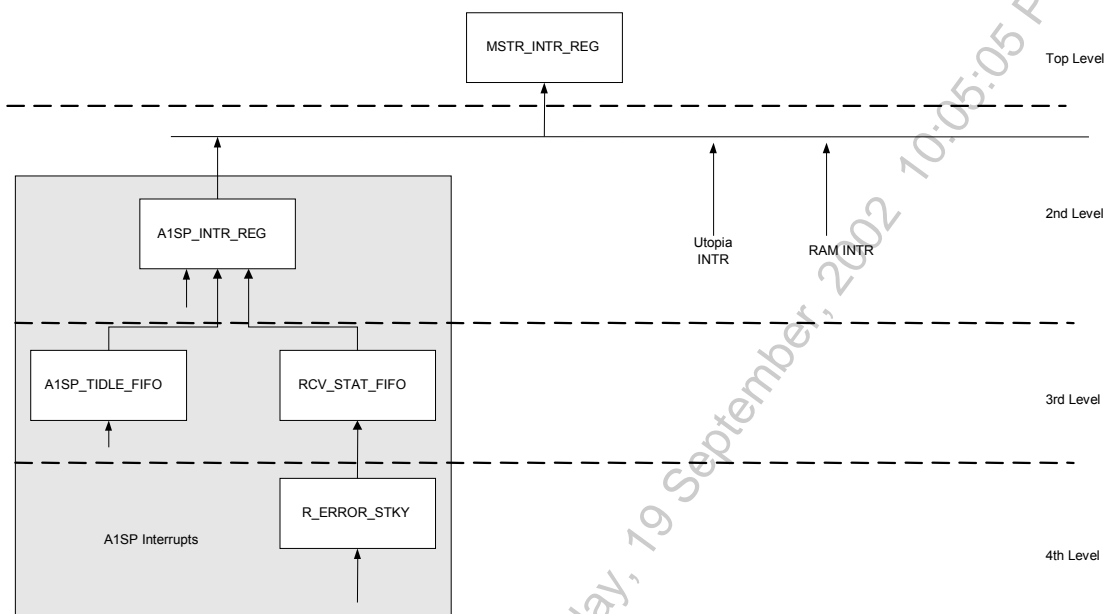
UTOPIA Interrupts

The UTOPIA block sources five interrupts directly to the Master Interrupt Register. The five interrupts are Transmit UTOPIA FIFO full, Loopback FIFO full, UTOPIA parity error, runt cell error, and UTOPIA transfer error. The first interrupt indicates that the Transmit UTOPIA four cell FIFO has filled. The second interrupt indicates that the UTOPIA loopback FIFO has filled. The third interrupt indicates there was a parity error on the data received on the UTOPIA interface. The fourth interrupt indicates cell less than 53 bytes was detected. The fifth interrupt indicates the receive UTOPIA interface was requested to send a cell when it did not have one available.

RAM Interface Interrupt

The RAM interface block sources a parity error interrupt directly to the Master Interrupt Register.

A1SP Interrupts

The A1SP block sources an interrupt to the Master Interrupt Register. The A1SP interrupt register indicates the source of the interrupt within the A1SP block. Since many indications provided by the A1SP interrupt structure are per channel or per queue, there are 2 FIFOs provided for per channel indications. There are six possible sources of an interrupt for the A1SP_INTR_REG: A1SP Receive Status FIFO not empty, A1SP CAS Change FIFO not empty, A1SP Receive Status FIFO overflow, CAS Change FIFO overflow, OAM interrupt, and Frame Advance FIFO overflow. The OAM interrupt indicates that the Receive OAM Queue is not empty. In other words, this interrupt can be thought of as an active low Receive OAM Queue Empty signal. The A1SP Receive Status FIFO overflow, A1SP CAS Change FIFO overflow, and Frame Advance FIFO overflow interrupts simply indicate that the FIFOs have overflowed. The A1SP Receive Status FIFO not empty interrupt indicates the A1SP Receive Status FIFO is not empty and the A1SP Transmit CAS Change FIFO not empty interrupts do the same. The next two sections discuss how these FIFOs operate.

Since some of the conditions are transitory, the A1SP_INTR_REG captures if the condition has occurred since the last time the register was read. The A1SP_STAT_REG reflects the current status.

A1SP Receive Status FIFO

The Receive Status FIFO (RCV_STAT_FIFO) consists of 64 entries and is contained internal to the chip. The FIFO is accessed using a single address. When the FIFO transitions from empty to not empty, the INTR_FIFO_EMPB bit in both the A1SP_INTR_REG and the A1SP_STAT_REG will go active. When there are no longer any entries in the FIFO, the INTR_FIFO_EMPB bit in the A1SP_STAT_REG will go inactive.

Each entry within the Interrupt FIFO indicates the queue responsible for the interrupt and one of three possible causes: exiting underrun, entering underrun, and receive queue error. The first two causes simply report a change in the underrun status, while the third cause indicates an error has occurred on the receive side. To find out the specific cause of the error, the processor should access the R_ERROR_STKY register for the queue responsible for the interrupt. An error entry will only occur for the latter case if this is the first unmasked sticky bit error to occur for this queue since the last time the sticky bit memory register was cleared.

A1SP CAS Change FIFO

The CAS Change FIFO consists of 64 entries and is contained internal to the chip. The FIFO is accessed using a single address port. When the FIFO transitions from empty to not empty, the CASCHG_FIFO_EMPB bit in both the A1SP_INTR_REG and the A1SP_STAT_REG will go active. When there are no longer any entries in the FIFO, the CAS Change FIFO not empty interrupt bit in the A1SP_STAT_REG will go inactive.

Each entry in the CAS Change FIFO will indicate the channel upon which CAS changed and the new CAS value. Both Transmit and Receive CAS is monitored and reported.

9.4.2 Add Queue FIFO

In order to add a queue the processor has to write the ADDQ_FIFO. The ADDQ_FIFO consists of 64 16-bit entries and is accessed using a single address. The format of the ADDQ_FIFO word is shown in Figure 62. The first byte specifies the number of the queue to be added. The next six bits represent an offset that is used to spread the scheduling of cells across multiple frames. This helps to avoid the problem of clumping, which refers to contention with other cells scheduled during the same frame (see section 0 on Transmit CDV).

The upper bit indicates whether or not the ADDQ_FIFO is empty. This bit can be polled after adding queues to find out when they all have been added. The Empty bit indicates Empty status when it is set. The amount of time it takes to empty the FIFO is dependent on how full it is, whether TALP is processing cells and whether there is back pressure on the UTOPIA bus. ADDQ_FIFO entries can only be processed when TALP is idle. If the TALP_FIFO fills and prevents TALP from processing cells, this will prevent ADDQ_FIFO entries from being processed.

Note that the Offset and Queue Number fields are write only and cannot be read. The Empty field is read only and cannot be written.

Figure 62   ADDQ_FIFO Word Structure

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Empty	Unused				Offset						Queue Number				
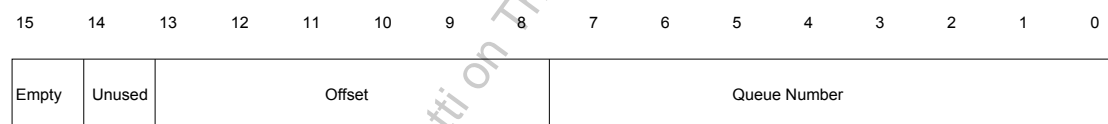
It is necessary to understand the basics of the frame-based scheduling performed by the chip to best utilize the Offset field in the ADDQ_FIFO. Read Section 0 and pay particular attention to how the Transmit Calendar works and to Section 0 and the information on clumping. The purpose of the offset field in the ADDQ_FIFO is to reduce the amount of clumping.

In frame mode (SDF-FR), the first cell of a queue can be scheduled relative to a reference value. When REF_VAL_ENABLE=0 in the LIN_STR_MODE memory mapped register, the reference value is always frame 0. When REF_VAL_ENABLE =1, the reference value is based upon the configuration consisting of all 24 (T1) or 32 (E1) queues configured identically as single DS0 with no pointer, full cells, and no CAS connections which will be scheduled to build cells during frame 0, 47, 94, 13, etc.. The reference value increments by 47 frames every time the current t_data_buffer frame write pointer is equal to the current reference value. When the microprocessor wants to add a queue, it writes the queue number and an offset to the ADDQ_FIFO. This offset is then added to the current reference value, and the first cell is scheduled in the resulting frame. For example, if two single DS0 queues are scheduled one right after the other with offsets of 0 and 1 respectively, there will never be any clumping because the first queue will be scheduled during frames 0, 47, 94, 13, etc, and the second queue will be scheduled during frames 1, 48, 95, 14, etc..

Note that the reference value is optimized to the configuration consisting of all 24 (T1) or 32 (E1) queues configured identically as single DS0 with no pointer, full cells, and no CAS. This is the configuration that is most likely to experience clumping. By using the offset field, cells can be prevented from being scheduled within the same frame even if queues were added some time ago. Software can guarantee this by assigning an offset equal to queue number(4:0), or by keeping track of which offsets are available. For non-single DS0 queues, the offset is beneficial for initial offsets if all queues have the same configuration, but won't guarantee that a newly added queue won't overlap with a queue started some time ago. However, for non-single DS0 queues, there are less queues active per line, and therefore clumping is less of an issue.

In multiframe mode (SDF-MF), the REF_VAL_ENABLE bit should be set only for lines which have all queues configured identically as single DS0 queues with full cells. When enabled, the reference value is based upon a configuration consisting of all 24 (T1) or 32 (E1) queues configured identically as single DS0 full cells with CAS. The reference value increments by 45 frames for T1 connections, and for E1 connections the reference value increments by either 44 frames or 45 frames to give an average of 44 2/17ths frames. By using different offsets, clumping can be minimized, but not completely avoided as in the frame mode (SDF-FR) description above. However, when using offsets, generation of a 0 pointer cannot be guaranteed, because cells may start on non-MF boundaries.

In order to guarantee a zero pointer for single DS0 SDF-MF connections as well as all non-single DS0 SDF-MF connections, REF_VAL_ENABLE should be disabled and the offset field used should be set equal to the FRAMES_PER_CELL field in the QUEUE_CONFIG word of the Transmit Queue Table. This will add the queue in the frame which is FRAMES_PER_CELL past frame number 0. Note that doing this will cause all queues, that have the same FRAMES_PER_CELL value, that are added at the same time, to be scheduled in the same frame. To avoid the clumping of cells, Add-Queue operations can be spread out in time so that not all queues are added at the same time.

In summary, when REF_VAL_EN is set, the generated CDV for the following two configurations will be minimized, more closely approaching the ideal minimum CDV for each case (REF_VAL_EN provides no added value for other configurations.):

- All 24(32) Queues SDF-FR, Single-DS0-with-no-pointer, full cells, Ideal minimum CDV = 0 us. Clumping is guaranteed not to occur.

- All 24(32) Queues SDF-MF, Single-DS0, full cells, Ideal minimum CDV = 125 us. Clumping is minimized.

IMPORTANT NOTE:

If a multiframe resync occurs on a line and REF_VAL_ENABLE is set, the relationship between different queues on that line will change. If this occurs, clumping may occur in the existing connections and in any future connections that are added. In H-MVIP mode, multiframe resyncs will never occur. In low speed mode, if REF_VAL_ENABLE is set it is recommended that MF_SYNC_MODE in LS_Ln_CFG_REG is disabled, as this will prevent multiframe resyncs.

9.5   RAM Interface Block (RAMI)

The RAMI is the central arbiter for all memory accesses. It provides a priority mechanism that incorporates fairness to satisfy all real-time requirements of the various blocks. All blocks requesting a data transfer with the common memory supply the address, control signals, and the data, if the requested data transfer is a write, to the RAMI. When the RAMI actually grants the transfer, it provides a grant signal to the requesting block, indicating that the transfer has been performed. The memory is arbitrated on a cycle-by-cycle basis. No device is granted the bus for an indefinite time.

The AAL1gator-4/8 has a separate SSRAM and processor interface. One 128K x 16 or 128K x 18 SSRAMs are needed, depending on the mode of operation.

Either a pipelined synchronous SRAM with a single cycle deselect or a pipelined ZBT or ZBT-compatible synchronous SRAM can be used.

For most applications the pipelined single-cycle deselect SSRAM is sufficient, but if additional performance is needed, such as in cross connect applications which need to use partial cells to lower delay, the ZBT SSRAM is recommended.

The SSRAM interface runs at the same frequency as SYS_CLK and can run up to 45 MHz.

14.2 Line Interface Block (AAL1_LI)

14.2.1 Conventions

The following conventions are used in this document:

The 8 lines, which connect the AAL1_LI to the A1SP block, are called **local links**. The lines on the external interface are called **external lines**.

The direction from the local links to the external line interface is call the **transmit** direction. The direction from the external line interface to the local links is called the **receive** direction.

14.2.2 Functional Description

The line interface block is responsible for passing the TDM data between the A1SP blocks and converting it to the appropriate protocol used on the external lines. The options available are:

Direct Mode

This is mainly a pass through mode between the external 4 lines and the lower 4 local links, which connect to the A1SP block. This mode is used to connect to any PMC E1 or T1 Framer or compatible device. This mode is also used to interface to devices, which support the MVIP-90 protocol. 8/4 lines are supported, including a clock, data, frame pulse, and signaling pin for each direction. In addition any low speed (< 2.5 MHz) clear channel data stream can be passed in this mode.

Note clocks rates up to 15 MHz are supported in this mode. However, the aggregate bandwidth cannot exceed 20 Mbps. Therefore, if all 4 lines of the A1SP are used and are the same rate, 5 MHz is the highest rate supported.

A common clock pin is also available, which can be shared across all receive lines or all transmit lines and is selectable on a per line basis.

Some framers also share a clock and signaling pin, where the clock pin becomes a signaling pin when signaling is required or remains as a clock pin when individual clocks per line are required. When this pin carries signaling information a common clock is used, which is shared across all lines. This option can be configured on a per line basis.

2 Mbps MVIP mode is also supported where the line is handled in accordance with the MVIP-90 specification. MVIP mode can be individually selected per line for all 4 lines. Tri-stating of individual time slots is not supported. There is a common 4 MHz clock and a common framing reference signal.

Note that if a mix of MVIP-90 and non MVIP-90 lines are being used, line 0 must be MVIP-90.

Line 0 can also be configured to be a highspeed (E3/DS3) line. In this mode, none of the other lines can be used.

H-MVIP Mode

This mode supports 1 separate 8 Mbps H-MVIP lines with 1 in the transmit direction and 1 in the receive direction. There is a common 16 MHz clock, a 4 MHz clock, and a common framing reference signal. Two Separate data pins and signaling pins are provided in each direction. Individual time slots cannot be disabled.

Internally each 8 Mbps stream is converted into four 2 Mbps streams in a round-robin fashion. The data is byte interleaved into/from each 2 Mbps stream. See section 14.6.2 for diagrams showing relationship between external H-MVIP data and internal 2 Mbps data.

The mode of the module is determined by the value of the LINE_MODE pin. The following encoding is used:

Table 9   LINE_MODE Encoding

LINE_MODE[0]	Line Interface Mode
"0"	Direct
"1"	H-MVIP

Figure 63 shows the block diagram for the Line Interface Block. The block consists of the H-MVIP Block and mux/demux logic.

The B0 mux/demux logic selects between the H-MVIP data or external direct links.

PMC-SIERRA

Figure 63   Line Interface Block Architecture
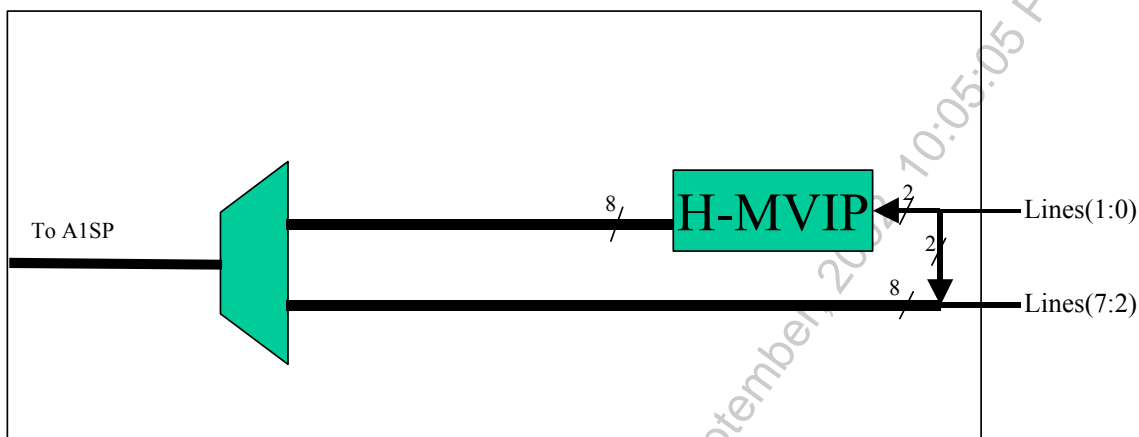
The different modes of the Line Interface Block require the pin definitions to change depending on the mode you are in. See Section 8.5.1 for exact definitions.

The AAL1gator supports several different types of links. E1 and T1 unstructured links are supported in Direct mode. E1 and T1 structured links (with and without CAS) are also supported in Direct Mode. MVIP-90 lines are only supported in Direct Mode. H-MVIP lines are only supported in H-MVIP mode. While a single high speed line is supported in Direct mode.

T1/E1 Frame Resynchronization Recovery Mechanisms

In all line modes, for any channelized line (E1, T1, H-MVIP, MVIP-90) a frame alignment and multiframe alignment exists which tells the AAL1gator where time slot, frame and multi-frame boundaries exist. The counters which track the frame and multi-frame alignment will free-run until an external pulse (frame or multi-frame sync) indicates a specific boundary. If the external pulse indicates a new frame or multiframe boundary, which is different than the device expects, a resync event will occur and the device will realign to the new boundary. Resync events are reported in the RSTAT_FIFO. Receive resync events also cause the SUPPRESSED_CELL counter to increment.

In the receive direction, frame and multi-frame resync events cause the following events to happen:

- Frame and bit counters realign to the new frame/multi-frame boundary

- Cell transmission is suppressed for 12-24ms(T1) or 16-32ms(E1).   This allows the cell scheduling to be realigned and the destination buffer to underrun and readjust to the new cell stream. Ideally the receive side should have the CDVT buffer depth less that 12(T1) or 16(E1) ms to insure an underrun will occur.

- Since cell transmission is only suppressed and not deactivated, cell processing will continue including incrementing of sequence numbers. Thus a resync event will cause sequence number errors on the remote side.

In the transmit direction, frame and multi-frame resync events cause the following events to happen:

- Frame and bit counters realign to the new frame/multi-frame boundary

- A forced underrun will occur so that the AAL1 receiver can realign correctly to new structure.

- Cells will be dropped for one or two frames (125 to 250 us).

Note that if the device is configured so that Multiframe information is passed into the device (Direct mode and MF_SYNC_MODE=1) and an external MF re-alignment occurs, the line will resync if the line is configured for SDF-FR or SDF-MF mode. It is expected that if the line is in SDF-FR mode, the external MF indicator will not move.

The following sections describe each of the line modes supported by the AAL1gator.

Direct Mode

This section defines how the Line Interface functions when in Direct Mode. The Line interface does no processing on the any of the line signals but just passes them between the 4 local links, which connect the A1SP and the 4 external lines. In this mode the H-MVIP portions of the Line Interface Block are not used.

The type of line is selected based on the value of T1_MODE in the LIN_STR_MODE memory register for each line. For unstructured data format (UDF mode), the value of T1_MODE does not matter because data is interpreted as a clear channel bit stream. If MVIP-90 mode is being used (MVIP_EN bit is set in the LS_Ln_CFG_REG register for this line), then T1_MODE should be inactive ('0').

The frame structure for E1 and T1 lines can be unstructured-multiline (UDF-ML), Structured-Frame (SDF-FR) or Structured-Multi-Frame (SDF-MF) and is determined by the value of FR_STRUCT[1:0] in the LIN_STR_MODE memory register for each line.

00   Reserved

01   SDF-FR

10   UDF-ML

11   SDF-MF

SDF-FR mode is used when making a structured connection and CAS signaling is not being transported. SDF-MF mode is used when making a structured connection and CAS signaling is being transported. If a mixture of CAS and non-CAS connections are being made on the same line, then put the line in SDF-MF mode and set R_CHAN_NO_SIG and T_CHAN_NO_SIG in the queue tables for the connections not carrying CAS.

Several clocking options exist in this mode and are controlled by the value of the CLK_SOURCE bits in the LIN_STR_MODE register for each line.

In the receive direction, the CLK_SOURCE_RX bit has two possible options. If this bit is set then the line receives its clock from the CRL_CLK pin. If this bit is not set then the line receives its clock from the RL_CLKn pin associated with that line.

In the transmit direction, eight possible options exist and are controlled by the value of CLK_SOURCE_TX bits in the LIN_STR_MODE memory register for each line. The eight options are:

000   Clock is an input on pin TL_CLK[n].

001   Clock is an input on pin RL_CLK[n] (loop timing mode).

010   Clock is internally synthesized in the CGC Block as a nominal E1 or T1 clock based on SYS_CLK and the value of T1_MODE. The clock is output on TL_CLK[n] pin.

011   Clock is internally synthesized in the CGC Block based on SRTS. The clock is output on TL_CLK[n] pin.

100   Clock is internally synthesized in the CGC Block using the adaptive algorithm. The clock is output on TL_CLK[n] pin.

101   Clock is internally synthesized based on values received on the CGC interface. (externally controlled)

110   The clock is output on TL_CLK[n] pin.The clock on CTL_CLK input pin is used.

111   The clock on CTL_CLK input pin is used and signaling data is output on TL_CLK[n] pin.

Receive Direction

The Line Interface Block accepts deframed data from the 4 external lines. The data, signaling and synchronization signals are received from the external interface. The external lines can support data rates up to 15 Mbps per line. (However the total aggregate bandwidth cannot exceed 20 Mbps). The falling edge of RL_CLK[n] is used to clock in the data and is used as the active edge for all receive logic in this mode. For structured data, the Line Interface Block uses the external synchronization input signals (RL_SYNC[n]) as either frame pulses or multi-frame pulses. Whether the Line Interface Block interprets RL_SYNC as a frame pulse or a multi-frame pulse is determined by the value of MF_SYNC_MODE in the LS_Ln_CFG_REG register for that line. If the line is configured for UDF-ML mode (unstructured), the data will be passed as a clear channel bit stream and RL_SYNC[n] will be ignored.

In normal mode (MVIP_EN bit is low in the LS_Ln_CFG_REG register for this line), the first time RL_SYNC[n] is sampled high after being low, indicates the first bit of a frame or multi-frame. In MVIP-90 mode (MVIP_EN bit is set in the LS_Ln_CFG_REG register for this line), the first time RL_SYNC[n] is sampled low after being high, indicates the first bit of a frame. For T1 structured data a frame is completed every 193 bits. For E1 or MVIP-90 structured data a frame is completed every 32 bytes.

It is not necessary to provide an edge at the beginning of every frame or multi-frame. However if a frame or multi-frame pulse is detected on a non-frame or non-multi-frame boundary, then the AAL1gator-4/8 will resync to the new boundary and cell generation will be suppressed for 12 – 32 ms.

In T1 mode a multi-frame can either be 12 or 24 frames depending on if the line is in Super Frame (SF) or Extended Super Frame (ESF) mode. An E1 multi-frame is 16 frames long. A special case of E1 mode exists that permits the use of T1 signaling with E1 framing. When E1_WITH_T1_SIG is set in LIN_STR_MODE and the line is in E1 mode, a multi-frame of 24 frames will be used.

Since the framer accumulates signaling over an entire multi-frame, signaling only has to be sampled once per multi-frame. The AAL1gator-4/8 reads signaling during the last frame of every multi-frame. If the framer supplies constant signaling for an entire multi-frame, then the multi-frame sync signal is not required unless there is a desire to synchronize the multi-frame with the data across the network. In general this is not necessary.

Note for lines using automatic CAS Change detection mode, CAS must be static for the entire multiframe structure.

The AAL1gator-4/8 reads the signaling nibble for each channel when it reads the last nibble of each channel's data. See Figure 64 for an example of a T1 frame. See Figure 65 for an example of an E1 frame.

Figure 64   Capture of T1 Signaling Bits
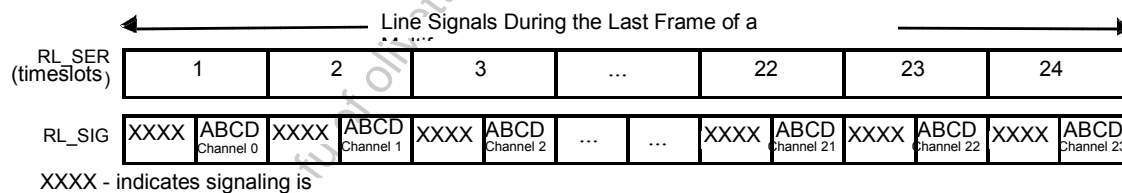
Figure 65   Capture of E1 Signaling Bits
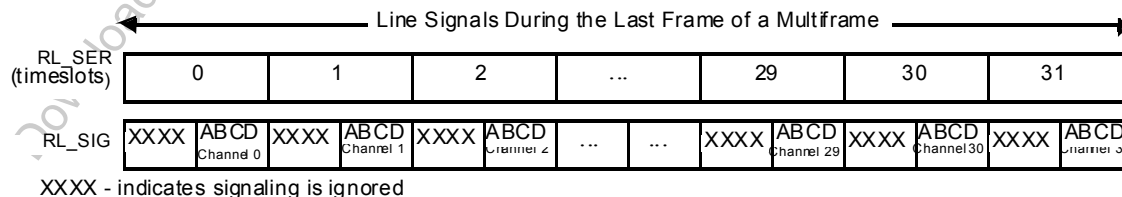
Note:

1. AAL1gator-4/8 treats all 32 timeslots identically. Although E1 data streams contain 30 timeslots of channel data, 1 timeslot of framing (timeslot 0) and one time slot that can either be signaling or data (time slot 16), data and signaling for all 32 timeslots are stored in memory and can be sent and received in cells.

Transmit Direction

In the line transmit direction, for structured data, the Line Interface Block may take the TL_SYNC input signal and depending on the value of MF_SYNC_MODE interpret the signal as either a frame pulse or multi-frame pulse. Alternatively if GEN_SYNC in LIN_STR_MODE memory register is high for that line, then the Line Interface Block will take the frame pulse or multi-frame pulse generated by the A1SP Block for the local link and output that signal to the TL_SYNC[n] pin on the external lines. Whether the Line Interface Block generates a frame pulse or a multi-frame pulse is determined by the value of MF_SYNC_MODE in the LS_Ln_CFG_REG register for that line.

In normal mode (MVIP_EN bit is low in the LS_Ln_CFG_REG register for this line), the first time TL_SYNC is sampled high after being low, indicates the beginning of a frame or multi-frame. In MVIP-90 mode (MVIP_EN bit is set in the LS_Ln_CFG_REG register for this line), the first time TL_SYNC is sampled low after being high, indicates the beginning of a frame or multi-frame. For T1 structured data a frame is completed every 193 bits. For E1 structured data a frame is completed every 32 bytes.

It is not necessary to provide an edge at the beginning of every frame or multi-frame. However if a frame or multi-frame pulse is detected on a non-frame or non-multi-frame boundary, then the AAL1gator-4/8 will resync to the new boundary and there will be a temporary disruption of data being played out on the line.

In T1 mode a multi-frame can either be 12 or 24 frames depending on if the line is in Super Frame (SF) or Extended Super Frame (ESF) mode. In either case the AAL1gator will only sample CAS on 24 frame boundaries. An E1 multi-frame is 16 frames long. A special case of E1 mode exists that permits the use of T1 signaling with E1 framing. When E1_WITH_T1_SIG is set in LIN_STR_MODE and the line is in E1 mode, a multi-frame of 24 frames will be used.

Signaling data is driven for all frames of any multi-frame and will change only on multi-frame boundaries. For T1 mode, signaling data may change every 24th frame. For E1 mode, signaling may change every 16th frame.

The signaling nibble is valid for each channel when the last nibble of each channel's data is being driven. See Figure 66 for an example of signaling bits in a T1 frame. See Figure 67 for an example of signaling bits in the E1 mode.

The line interface block just passes the signaling and data bits from the A1SP block to the external output pins.

Figure 66   Output of T1 Signaling Bits

Line Output Signals During Every

TL_SER (timeslots): 1 | 2 | 3 | … | 22 | 23 | 24

TL_SIG: XXXX ABCD Channel 0 | XXXX ABCD Channel 1 | XXXX ABCD Channel 2 | … … | XXXX ABCD Channel 21 | XXXX ABCD Channel 22 | XXXX ABCD Channel 23

XXXX - indicates signaling is
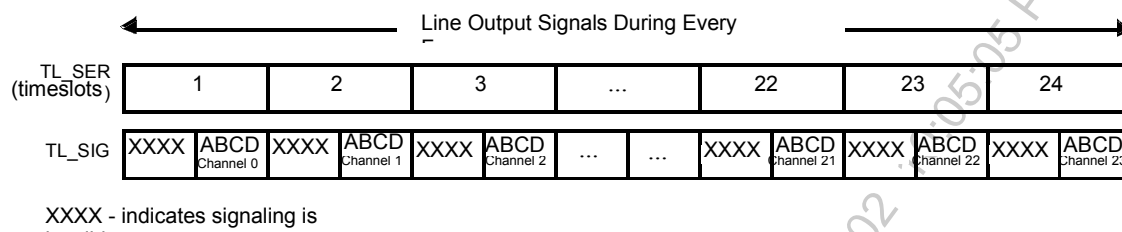
Figure 67   Output of E1 Signaling Bits
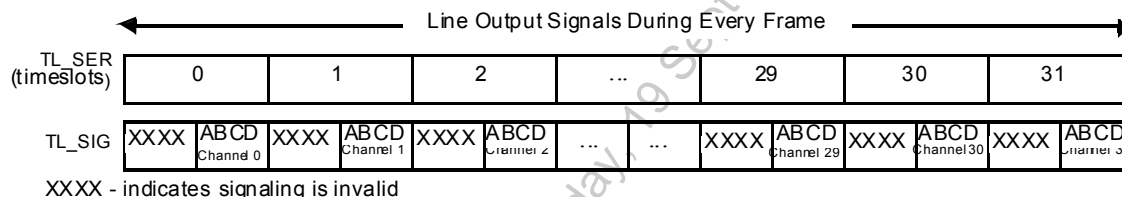
Line Output Signals During Every Frame

TL_SER (timeslots): 0 | 1 | 2 | … | 29 | 30 | 31

TL_SIG: XXXX ABCD Channel 0 | XXXX ABCD Channel 1 | XXXX ABCD Channel 2 | … … | XXXX ABCD Channel 29 | XXXX ABCD Channel 30 | XXXX ABCD Channel 31

XXXX - indicates signaling is invalid

Note:

1.   The AAL1gator-4/8 treats all 32 timeslots identically. Although E1 data streams contain 30 timeslots of channel data, 1 timeslot of framing (timeslot 0) and one time slot that can either be signaling or data (time slot 16), data and signaling for all 32 timeslots are stored in memory and can be sent and received in cells.

In high speed only line 0 is used. The Line Interface Block just passes the data and clock between the external line and the local link. The data is passed as a clear channel bit stream and data rates are supported up to 52 Mbps.

Note that if a rate > 45 MHz is used, the SYS_CLK needs to be faster than 38.88 MHz. If the line rate is 52 MHz, SYS_CLK must be 45 MHz.

The A1SP should be configured in UDF-HS mode in the HS_LIN_REG memory register. H-MVIP Block

This section defines how the Line Interface functions in H-MVIP Mode.

The H-MVIP block supports 1 lines in each direction of 8Mbps H-MVIP formatted data.

In this mode, the H-MVIP block takes each incoming external 8 Mbps H-MVIP data stream and breaks it into 4 separate local 2Mbps data streams. The bytes are taken off the bus in round robin fashion and sent to separate 2Mbps links. In the outgoing direction the H-MVIP block takes each group of four 2Mbps local links and byte interleaves them into one external 8 Mbps H-MVIP data stream.

See section 14.6.2 for diagrams showing relationship between external transmit H-MVIP data and internal 2 Mbps data.

In H-MVIP mode there is a common 16 MHz clock (HMVIP16CLK) whose every other rising edge is used to sample data on all external lines in the receive direction and whose every other falling edge is used to source data on all external lines in the transmit direction. There is also a common 4 MHz clock (HMVIP4CLK) whose falling edge is used to sample the frame pulse. Internally a 2 MHz clock is generated for each link which connects to the A1SP block.

A common frame pulse F0B is also used for all external lines. Individual local link frame pulses are derived from F0B. This signal is always an input.

Signaling is passed through as received and sent with the corresponding data byte. Signaling format is the same as in Direct mode, where the last nibble of each timeslot carries the CAS signaling bits.

Note since there is no MF indicator, CAS must be static for the entire multiframe structure.

The type of line is selected based on the value of T1_MODE in the LIN_STR_MODE memory register for each line. For H-MVIP mode T1_MODE must be off.

The frame structure for H-MVIP lines can be Structured-Frame (SDF-FR) or Structured-Multi-Frame (SDF-MF) and is determined by the value of FR_STRUCT[1:0] in the LIN_STR_MODE memory register for each line.

00   Reserved

01   SDF-FR

10   not valid in H-MVIP mode

11   SDF-MF

SDF-FR mode is used when making a structured connection and CAS signaling is not being transported. SDF-MF mode is used when making a structured connection and CAS signaling is being transported. If a mixture of CAS and non-CAS connections are being made on the same line, then put the line in SDF-MF mode and set R_CHAN_NO_SIG and T_CHAN_NO_SIG in the queue tables for the connections not carrying CAS.

Because H-MVIP lines all run off the same clock, there is only one mode of clocking available in this mode. Therefore the CLK_SOURCE values are not used. The HMVIP16CLK and HMVIP4CLK will always be used and a clock will be expected on these pins.

9.6   JTAG Test Access Port

The JTAG Test Access Port block provides JTAG support for boundary scan. The standard JTAG EXTEST, SAMPLE, BYPASS, IDCODE and STCTEST instructions are supported. The AAL1gator-4/8 identification code is 3124 hexadecimal.

10 Memory Mapped Register Description

The Chip SW_RESET state is automatically entered after a hardware reset is removed, or it can be asserted by setting the SW_RESET bit in the DEV_ID_REG. (Note memory cannot be accessed while the chip SW_RESET bit is set.) The A1SP SW_RESET state is automatically entered after a hardware reset, or chip SW_RESET, or it can be asserted by setting the A_SW_RESET bit in the A_CMD_REG for the A1SP.

The initial programming for the AAL1gator-4/8 is performed by loading the external memory with specified information while the A_SW_RESET bit is set, in the A_CMD_REG. (and the Chip SW_RESET is inactive) After the memory is initialized, the CMD_REG_ATTN bit should be set so the configuration data can be read. Then A_SW_RESET can be removed. The device then reads the data structures from memory and enters the correct operating mode.

Word data structures have the first byte located at the low-byte end of the bus, which is also the location of the even data bytes (little endian implementation).

Most AAL1gator control structures and all data buffers are stored in the external SSRAM. SSRAM accesses by the processor require the A[19] input to be equal to '0'. R_STATE_1, and R_LINE_STATE registers in the receive queue table are mapped to the memory address space but are actually located inside the chip to improve performance. All memory locations are readable and writable. Although once processing has begun, writing to some locations is restricted to prevent corruption of structures or data buffers used by the AAL1gator. Any restricted locations are designated below.

The remaining registers which are located inside the chip are accessed when A[19] is high and A[18] is low. See the Normal Mode Register section for descriptions of all the internal registers.

The address map is as follows:

Table 10 AAL1gator-4/8 Memory Map

A[19:17]	Description
"000"	A1SP memory registers
"10X"	Internal Normal Mode Registers
"11X"	Test Mode Registers

This section lists all the memory mapped registers and defines the bit fields of each register.

10.1 Initialization

The memory must be initialized to 0 (unless otherwise indicated) before the A1SP software reset is released, but the global software reset must be cleared before writes to memory can take place. A number of data structures used by the device in reserved areas depends on this initialization.

Notes:

2. All ports marked as "Reserved" must be initialized to 0 at initial setup. Software modifications to these locations after setup will cause incorrect operation.

3. All read/write port bits marked "Not used" must be written with the value 0 to maintain software compatibility with future versions.

4. All read-only port bits marked "Not used" are driven with a 0 and should be masked off by the software to maintain compatibility with future versions.

10.2 A1SP and Line Configuration Structures

Table 11   A1SP and Line Configuration Structures Summary

Note the addresses listed below are the offsets within the A1SP address space.

Addr[1]	Name	R/W	Org	Size	Description
0001h	HS_LIN_REG	R/W	1 word	2 bytes	The High Speed Line Register provides overall mode information.
0010h	LIN_STR_MODE_0	R/W	1 word	2 bytes	The Line Structure Mode register identifies which data structure type will be supported for each line. This is selectable on a line basis.
0011h	LIN_STR_MODE_1	R/W	1 word	2 bytes	
0012h	LIN_STR_MODE_2	R/W	1 word	2 bytes	
0013h	LIN_STR_MODE_3	R/W	1 word	2 bytes	
0014h	LIN_STR_MODE_4	R/W	1 word	2 bytes	not used in A4
0015h	LIN_STR_MODE_5	R/W	1 word	2 bytes	not used in A4
0016h	LIN_STR_MODE_6	R/W	1 word	2 bytes	not used in A4
0017h	LIN_STR_MODE_7	R/W	1 word	2 bytes	not used in A4

10.2.1 HS_LIN_REG

Organization: 1 word

Base address within A1SP: 1_h

Type: Read/Write

Function: Stores the configuration of the high speed line. Writes to this register will not take affect until the CMD_ATTN bit is set in the CMD_REG.

Format: Refer to the following table.

Field (Bits)	Description
Reserved (15:14)	Initialize to 0.

Field (Bits)	Description
Not used (13:6)	Write with a 0 to maintain future software compatibility.
HS_GEN_DS3_AIS (5)	When the bit is set, Send cells with a DS3 framed "1010" pattern which is a DS3 AIS signal. (line 0 only).
HS_TX_COND (4)	Send cells with all 1s when in high-speed mode (line 0 only).
HS_RX_COND (3)	Fetch receive conditioning data from the R_COND_DATA buffer for line 0, channels 0 and 1. High-speed mode (line 0 only).
UDF_HS (2)	Line 0 is in the UDF-HS mode. 0 Disables the UDF-HS (T3/E3) mode. 1 Enables the UDF-HS (T3/E3) mode. If this mode is selected, the T_QUEUE_TBL and R_QUEUE_TBL entry index 0 are used. Initialize to the proper value. This bit works in conjunction with the T1_MODE bit in LIN_STR_MODE to enable playing out of framed DS3 AIS when in underrun.)
Unused (1:0)	Write with a 0 to maintain future software compatibility

LIN_STR_MODE

Organization: Eight 16-bit words.

Base address within A1SP: 10_h

Index: 1_h

Type: Read/Write

Function: Stores the per-line configuration. Writes to this memory register will not take affect until the CMD_ATTN bit is set in the CMD_REG.

Format: Refer to the following table.

Offset	Name	Description
0_h	LIN_STR_MODE_0	Line structure mode for line 0.
1_h	LIN_STR_MODE_1	Line structure mode for line 1.
2_h	LIN_STR_MODE_2	Line structure mode for line 2.
3_h	LIN_STR_MODE_3	Line structure mode for line 3.
4_h	LIN_STR_MODE_4	Line structure mode for line 4. not used in A4
5_h	LIN_STR_MODE_5	Line structure mode for line 5. not used in A4
6_h	LIN_STR_MODE_6	Line structure mode for line 6. not used in A4
7_h	LIN_STR_MODE_7	Line structure mode for line 7. not used in A4

LIN_STR_MODE Word Format

Field (Bits)	Description
LOW_CDV (15)	This bit is used in unstructured mode to decrease the CDV. If this bit is set then the cells are scheduled every 47 bytes, otherwise the scheduling is frame based. This mode cannot be used when the queue is configured for partial cells or for AAL0 mode. By default, a high speed queue is always configured for low CDV and this bit is ignored. 1 Unstructured line is in low CDV mode (byte based scheduling) 0 Unstructured line is not in low CDV mode (frame based scheduling) See Section 13.2.1 for important information about configuration guidelines when using this option.
REF_VAL_ENABLE (14)	This bit enables the generation of the reference value for this line when adding queues.
T1_MODE (13)	Determines mode of the line 1 Line is in T1 mode. If HS_UDF is also set, the highspeed line will play out of framed DS3 AIS when in underrun. Line is in E1 mode.
E1_WITH_T1_SIG (12)	E1_WITH_T1_SIG enables T1 signaling while in E1 mode for this line. Signaling is updated every 24 frames instead of every 16 frames. This bit is valid only when the line is in E1 SDF-MF mode. AAL1 cell structures contain a signaling nibble every 25 bytes instead of every 17 bytes per single DS0. 1 Use T1 signaling. Use E1 signaling.
HI_RES_SYNTH (11)	When this bit is set the clock synthesizer for this line is in high resolution mode. In high resolution mode the synthesizer has 256 distinct frequencies to use when recovering an E1 or T1 clock. When high resolution mode is disabled the synthesizer has 16 distinct frequencies. This bit should be set for adaptive clocking and must be off for SRTS.
Reserved (10)	Reserved
MF_ALIGN_EN (9)	When operating in SDF-MF mode, MF_ALIGN_MODE controls the MF alignment options: 0) only the CDVT value is taken into account when determining when to play out data. Normally this mode is used 1) MF_ALIGN_EN, The CDVT and the multiframe boundaries are taken into account, the data is aligned with the next MF boundary after CDVT.
SHIFT_CAS (8)	Causes the CAS nibble to be shifted so as to be coincident with the upper nibble of data instead of with the lower nibble of data. 0) CAS nibble is coincident with the lower nibble of data 1) CAS nibble is coincident with the upper nibble of data
GEN_SYNC (7)	Enables the AAL1gator to generate TL_SYNC instead of receiving it as an input. Only valid in Direct Low Speed mode. 1) TL_SYNC is an output for this line 0) TL_SYNC is an input for this line

Field (Bits)	Description
CLK_SOURCE_TX (6:4)	Selects TL_CLK source. This value will override the setting defined by the TLCLK_OUTPUT_EN input. If switching from an external to an internal clock or visa versa, make sure there are not two clocks driving simultaneously. In Direct mode (low speed lines) all options are valid. For high speed lines the only valid options are "000" and "001". The field is ignored in H-MVIP mode. 000 Use external clock. (TL_CLK is an input). 001 LOOPED – Use RL_CLK as the clock source. 010 NOMINAL Synthesized – Generate a clock of the nominal (T1 or E1) frequency from SYS_CLK. 011 SRTS Synthesized- Generate a T1/E1 clock frequency based on the received SRTS values. 100) ADAPTIVE Synthesized- uses receive buffer depth to generate a T1/E1 clock. 101) Externally controlled Synthesized: Generate a T1/E1 clock frequency based on the values provided by CGC_SER_D pin. This mode is used for external implementations of SRTS or Adaptive clocking. 110) Use common external clock (CTL_CLK) (only valid in low speed mode) 111) If in Direct Mode, use common external clock (CTL_CLK) and drive TL_SIG data onto TL_CLK pin. Not valid for UDF-HS lines or H-MVIP mode.
CLK_SOURCE_RX (3)	Selects RL_CLK source. 0 Use external clock. (RL_CLK is an input). 1 Use common external clock (CRL_CLK) as the clock source.
SRTS_EN (2)	Enable SRTS for this line. (Assert only for UDF-ML or UDF-HS). 1 The insertion of the transmit SRTS bits is enabled for this line and the received SRTS bits are accumulated. 0 The CSI bits of the odd transmit AAL1 cells are set to 0 and the received SRTS bits are ignored.
FR_STRUCT (1:0)	Frame structure. 11 Enables SDF-MF. Signaling information is preserved. 01 Enables SDF-FR. No signaling information is preserved. 10 Enables UDF, clear channel, no channelization, no signaling. If this mode (UDF-ML) is selected, the T_QUEUE_TBL and R_QUEUE_TBL entries used are found at index = 32 x line. For example, line 3 uses T_QUEUE_TBL and R_QUEUE_TBL entry 96. Not used. Initialize to the proper value.

10.3 Transmit Structures Summary

Table 12   Transmit Structures Summary

Note the addresses listed below are the offsets within the A1SP address space.

Name	R/W	Org	Size	Addr	Description
P_FILL_CHAR	R/W	1 word	2 bytes	0004h	The empty bytes in a partially filled cell are filled with P_FILL_CHAR.

Name	R/W	Org	Size	Addr	Description
Reserved(AQ)	R/W	16 words	32 bytes	0030$_h$ – 003F$_h$	(Reserved (AQ))
T_SEQNUM_TBL	R/W	16 words	32 bytes	0020$_h$ – 002F$_h$	The Transmit Sequence Number Table is initialized according to a table.
T_COND_SIG	R/W	32 bytes x 8 lines	256 bytes	A 0400$_h$ - A 047F$_h$	This table stores the signaling to be used when the TX_COND bit in the T_QUEUE_TBL is set.
T_COND_DATA	R/W	32 bytes x 8 lines	256 bytes	0480$_h$- 04FF$_h$	This table stores the data to be used when the TX_COND bit in the T_QUEUE_TBL is set.
Reserved	R/W	256 words	512 bytes	0700$_h$- 07FF$_h$	Reserved (Frame Advance FIFO).
Reserved	R/W	8 x 128 x 2 words	4 Kbytes	0800$_h$- 0FFF$_h$	Reserved (Transmit Calendar).
Reserved	R/W	8 x 256 bytes	2 Kbytes	1000$_h$- 13FF$_h$	Reserved (Transmit Signaling Buffer).
T_OAM_QUEUE	R/W	2 x 32 words	128 bytes	1400$_h$- 143F$_h$	The Transmit OAM Queue contains the OAM cells to be transmitted.
T_QUEUE_TBL	R/W	256 x 32 words	16 Kbytes	2000$_h$- 3FFF$_h$	The Transmit Queue Table contains all pointers and variables that are queue-dependent.
Reserved	R/W	8 x 2 K words	32 Kbytes	4000$_h$- 7FFF$_h$	Reserved (Transmit Data Buffer).

Notes:

1. All ports marked as "Reserved" must be initialized to 0 at initial setup. Software modifications to these locations after setup will cause incorrect operation.

2. All read/write port bits marked "Not used" must be written with the value 0 to maintain software compatibility with future versions.

3. All read-only port bits marked "Not used" are driven with a 0 and should be masked off by the software to maintain compatibility with future versions.

10.3.1 P_FILL_CHAR

Organization: One word

Base address within A1SP: 4h

Type: Read/Write

Function: Contains the fill character for partially filled cells.

Format: Refer to the following table.

Field (Bits)	Description
Not used (15:8)	Write with a 0 to maintain future software compatibility.
P_FILL_CHAR (7:0)	Character used in partially filled cells. Initialize to the desired value.

10.3.2 T_SEQNUM_TBL

Organization: 16 words

Base address within A1SP: 20_h

Index: 1_h

Type: Read/Write

Function: Stores all possible first bytes in the payload: CSI, SN, and SNP. This table must be loaded into the external SSRAM on every power cycling.

Initialization: Initialize to the values in the following table

Offset	Data Value
0_h	0000_h
1_h	0017_h
2_h	$002D_h$
3_h	$003A_h$
4_h	$004E_h$
5_h	0059_h
6_h	0063_h
7_h	0074_h
8_h	$008B_h$
9_h	$009C_h$
A_h	$00A6_h$
B_h	$00B1_h$
C_h	$00C5_h$
D_h	$00D2_h$
E_h	$00E8_h$
F_h	$00FF_h$

10.3.3 T_COND_SIG

Organization: 32 bytes x 8 lines. Only first 4 lines used with A4.

Base address within A1SP: 400_h

Index: 10_h

Type: Read/Write

Function: Stores the transmit conditioned signaling.

Initialization: Initialize to the conditioned signaling value for the channel. This value typically depends on the type of channel unit that is connected. For example, a Foreign Exchange Office (FXO) needs a different conditioning value than a Foreign Exchange Subscriber (FXS).

Format: One nibble per byte, two bytes per word, 16 words per line. Refer to the following table.

Offset	Name	Description
00000$_h$	T_COND_SIG_0	Transmit conditioned signaling for line 0.
00010$_h$	T_COND_SIG_1	Transmit conditioned signaling for line 1.
00020$_h$	T_COND_SIG_2	Transmit conditioned signaling for line 2.
00030$_h$	T_COND_SIG_3	Transmit conditioned signaling for line 3.
00040$_h$	T_COND_SIG_4	Transmit conditioned signaling for line 4.
00050$_h$	T_COND_SIG_5	Transmit conditioned signaling for line 5.
00060$_h$	T_COND_SIG_6	Transmit conditioned signaling for line 6.
00070$_h$	T_COND_SIG_7	Transmit conditioned signaling for line 7.

T_COND_SIG_n Word Format

Field (Bits)	Description
Not used (15:12)	Write with a 0 to maintain future software compatibility.
T_COND_SIG_A_H (11)	Transmit conditioned A bit for: Offset = ((channe– -1) / 2) + line x 16.
T_COND_SIG_B_H (10)	Transmit conditioned B bit for: Offset = ((channe– -1) / 2) + line x 16.
T_COND_SIG_C_H (9)	Transmit conditioned C bit or A bit if T1 SF for: Offset = ((channe– -1) / 2) + line x 16.
T_COND_SIG_D_H (8)	Transmit conditioned D bit or B bit if T1 SF for: Offset = ((channe– -1) / 2) + line x 16.
Not used (7:4)	Write with a 0 to maintain future software compatibility.
T_COND_SIG_A_L (3)	Transmit conditioned A bit for: Offset = (channel / 2) + line x 16.
T_COND_SIG_B_L (2)	Transmit conditioned B bit for: Offset = (channel / 2) + line x 16.
T_COND_SIG_C_L (1)	Transmit conditioned C bit or A bit if T1 SF for: Offset = (channel / 2) + line x 16.
T_COND_SIG_D_L (0)	Transmit conditioned D bit or B bit if T1 SF for: Offset = (channel / 2) + line x 16.

10.3.4 T_COND_DATA

Organization: 32 bytes x 8 lines. Only first 4 lines used with A4.

Base address within A1SP: 480_h

Index: 10_h

Type: Read/Write

Function: Stores the transmit conditioned data.

Initialization: Initialize to the conditioned data appropriate for the channel, which typically depends on the type of channel connected to the device. For example, data usually needs an FF_h value and voice needs a small Pulse Coded Modulation (PCM) value.

Format: Two bytes per word, 16 words per line. Refer to the following table.

Offset	Name	Description
00000_h	T_COND_DATA_0	Transmit conditioned data for line 0.
00010_h	T_COND_DATA_1	Transmit conditioned data for line 1.
00020_h	T_COND_DATA_2	Transmit conditioned data for line 2.
00030_h	T_COND_DATA_3	Transmit conditioned data for line 3.
00040_h	T_COND_DATA_4	Transmit conditioned data for line 4.
00050_h	T_COND_DATA_5	Transmit conditioned data for line 5.
00060_h	T_COND_DATA_6	Transmit conditioned data for line 6.
00070_h	T_COND_DATA_7	Transmit conditioned data for line 7.

T_COND_DATA_n Word Format

Field (Bits)	Description
T_COND_DATA_H (15:8)	Transmit conditioned data offset = ((channel / 2) + 1) + line x 16.
T_COND_DATA_L (7:0)	Transmit conditioned data offset = (channel / 2) + line x 16.

10.3.5 RESERVED (Transmit Signaling Buffer)

This structure is reserved and need not be initialized to 0. Software modifications to this structure after setup will cause incorrect operation.

Organization: Eight multiframes x 32 DS0s x 8 lines. Only first 4 lines used with A4. Each of the eight lines are allocated a separate signaling buffer. Each DS0 generates one new nibble of signaling per multiframe. The data is stored in the buffer in the order it is received from the framer device. Different framers provide the signaling information in different formats, as the following illustration shows, for one multiframe worth of signaling data.

Base address: 01000_h

Index: 80_h

Type: Read/Write

Function: Stores the outgoing signaling data.

Figure 68   SDF-MF Format of the T_SIGNALING BUFFER
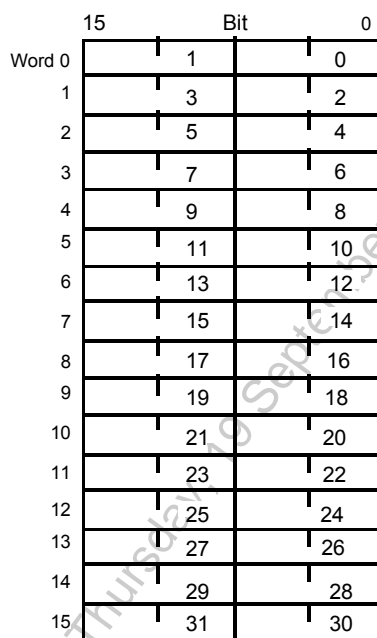
The upper nibble of each byte is 0.

10.3.6 T_OAM_QUEUE

Organization: 2 cells x 32 words

Base address within A1SP: 01400$_h$

Index: 20$_h$

Type: Read/Write

Function: Stores two transmit OAM cells.

Initialization: An optimization is to initialize to the body of an OAM cell so only the header must be modified before sending.

Format: Refer to the following table.

Offset	Name	Description
01400$_h$	T_OAM_CELL_1	Transmit OAM cell 1.
01420$_h$	T_OAM_CELL_2	Transmit OAM cell 2.

T_OAM_CELL_n Format

Offset	Bits 15:8	Bits 7:0
Word 0	Header 1	Header 2
Word 1	Header 3	Header 4
Word 2	Header 5 (HEC) (Pre-calculated by software)	Bits 7:1 Not used. Set to 0. Bit 0 0 Disables CRC-10 insertion. 1 Enables CRC-10 insertion.
Word 3	Payload 1	Payload 2
. . .	. . .	. . .
Word 26	Payload 47	Payload 48
	If CRC-10 is enabled in Word 2, set data to 0 in Word 26. Word 26 will be replaced by the computed CRC-10 result as the cell is transmitted.	

10.3.7 T_QUEUE_TBL

Organization: 256 x 32 words. Only 128 queues used with A4.

Base address within A1SP: 2000$_h$

Index: 20$_h$

Type: Read/Write

Function: Configures the VCs.

Format: Each queue will be allocated 32 consecutive words.

Offset	Name	Description
0$_h$	Reserved	(Data pointer.) Initialize to FFFF each time this queue is initialized.
1$_h$	Not used	Initialize to 0 each time this queue is initialized to maintain future software compatibility.
2$_h$	T_COND_CELL_CNT	A 16-bit rollover count of conditioned cells transmitted.
3$_h$	T_SUPPRESS_CNT	A 16-bit rollover count of cells not sent because of a line resynchronization. Or, if in UDF-HS mode, a 16-bit rollover count of cells not sent because TX_ACTIVE is not set. This counter also counts when cells are not sent because SUPPRESS_TRANSMISSION is set.
4$_h$	Not used	Initialize to 0 each time this queue is initialized to maintain future software compatibility.
5$_h$	Reserved	(Sequence number.) Initialize to 0 each time this queue is initialized.
6$_h$	QUEUE_CONFIG	The configuration of the current queue. Initialize to the proper value.
7$_h$	T_CELL_CNT	A 16-bit count of the cells transmitted.

Offset	Name	Description
8_h	TX_HEAD(1:2)	Header byte 1 in bits 15:8, header byte 2 in bits 7:0.
9_h	TX_HEAD(3:4)	Header byte 3 in bits 15:8, header byte 4 in bits 7:0.
A_h	TX_HEAD(5)	Header byte 5 (pre-calculated HEC) in bits 15:8.
B_h	QUE_CREDITS	A 10-bit quantity representing the number of byte credits accumulated for the queue.
C_h	CSD_CONFIG	Stores the average number of bytes in each cell, and carries the number of DS0s for this queue.
D_h	Not used	Initialize to 0 each time this queue is initialized to maintain future software compatibility.
E_h	T_CHAN_ALLOC(15:0)	A bit table with a bit set per DS0 allocated to this queue for DS0s 15:0 on the line defined by queue / 32.
F_h	T_CHAN_ALLOC(31:16)	A bit table with a bit set per DS0 allocated to this queue for DS0s 31:16 on the line defined by queue / 32.
10_h	T_CHAN_LEFT(15:0)	Initialize to the same value as T_CHAN_ALLOC(15:0).
11_h	T_CHAN_LEFT(31:16)	Initialize to the same value as T_CHAN_ALLOC(31:16).
12_h	TRANSMIT_CONFIG	Controls transmission of data.
13_h-$1F_h$	Not used	Initialize to 0 each time this queue is initialized.

T_COND_CELL_CNT Word Format (02_H)

Initialize to "0000" and at all other times the word is read only. The word maintained by TALP.

Field (Bits)	Description
T_COND_CELL_CNT (15:0)	A 16-bit rollover count of conditioned cells transmitted. This counter increments once, each time a cell with conditioned data is sent. If only signaling is conditioned this counter will not increment.

T_SUPPRESS_CNT Word Format (03_H)

Initialize to "0000" and at all other times the word is read only. The word is maintained by TALP.

Field (Bits)	Description
T_SUPPRESS_CNT (15:0)	A 16-bit rollover count of cells not sent because of a line resynchronization. Or, if in UDF-HS mode, a 16-bit rollover count of cells not sent because TX_ACTIVE is not set. This counter also counts when cells are not sent because SUPPRESS_TRANSMISSION is set.

QUEUE_CONFIG Word Format (06_H)

This word is maintained by the microprocessor.

Field (Bits)	Description
TX_COND (15)	Sends data and signaling from the transmit conditioned data area according to the conditioning mode selected in the TRANSMIT_CONFIG register. Initialize to the proper value.

Field (Bits)	Description
TX_ACTIVE (14)	When set, this bit enables this queue. To enable a connection for this queue: **1)Assert this bit.** **2)Add this queue to the ADDQ_FIFO Register.** To disable a connection on this queue, clear the TX_ACTIVE bit. This queue is then removed from the calendar queue the next time a cell would have been sent. Once this bit is cleared, the associated queue must not be returned to the add-queue FIFO until FRAMES_PER_CELL frames have passed by. If quick reconfiguration is required and the size of the queue is not going to change (number of allocated channels), then use SUPPRESS_XMT bit to pause queue and reconfigure instead of clearing TX_ACTIVE bit. When reactivating a previously active queue, be sure to reinitialize all the registers in the queue table for that queue.
FRAMES_PER_CELL (13:8)	A 6-bit integer specifying the maximum number of frames required to have enough data to construct a cell (round up of BYTE_PER_CELL/number of DS0s assigned) plus 1. For example, for a T1 line in SDF-FR mode with five DS0s, initialize this field to 11. In T1 SDF-MF, T1 SDF-FR, and E1_w_T1_sig modes, FRAMES_PER_CELL is encoded as the number of 24-frame multiframes required in bit 13 and the number of frames mod 24 in bits 12:8. In AAL0 mode this field is set to 48 (in T1 mode this is encoded as 1 MF and 24 frames: 111000_B). In all other modes, including unstructured T1 mode, encode this value as the maximum number of 256 bit increments required to create a cell. For unstructured mode with full cells, set this value to 3. For T1 SDF-MF single DS0 queues, encode the value 48 as one multiframe and 24 frames: 0x38. When calculating the FRAMES_PER_CELL value, do not subtract the bytes used by signaling nibbles from the value. For example, for an SDF-MF, single DS0, full cell connection, use the value 47 + 1 = 48 and not 46 + 1 = 47. For SDF-MF connections using partial cells, set FRAMES_PER_CELL to (round up of BYTE_PER_CELL/number of DS0s assigned) plus 2. This prevents scheduling more than one cell per frame. FRAMES_PER_CELL is ignored for lines with the LOW_CDV bit set in the LIN_STR_MODE location.
T_CHAN_NO_SIG (7)	Set to 1 to send cells with no signaling when in SDF-MF mode. This is the same as using this queue in SDF-FR mode, which means the structure forms on frame boundaries instead of multiframe boundaries.
T_CHAN_UNSTRUCT (6)	Set to 1 only when sending cells with a single DS0 without a pointer in the SDF-FR mode. To conform to the CES standard V 2.0 when using a single DS0 in SDF-FR mode, no pointer should be used. This bit can be ignored for AAL0 mode.

Field (Bits)	Description
BYTES_PER_CELL (5:0)	A 6-bit integer specifying how many bytes per cell are required if no structure pointers are used. For UDF_HS mode, this field is not used. This number must be set so the cell generation rate per queue is slower than once per frame. For unstructured lines, this means between 33 and 47. For structured applications, the BYTES_PER_CELL number must exceed the number of DS0 channels allocated to the queue. For example, a two channel queue may have the number set from 3 to 47. For SDF-MF connections with more than 16 channels allocated, the BYTES_PER_CELL number must exceed the number of DS0 channels allocated to the queue by two. For example, a 17 channel SDF-MF queue may have the number set from 19 to 47. For AAL0 connections this field should be set to 48. This is due to the fact that there is no sequence number byte in AAL0 cells.

T_CELL_CNT Word Format (07$_H$)

Initialize to "0000" and at all other times the word is read only. The word is maintained by TALP.

Field (Bits)	Description
T_CELL_CNT (15:0)	A 16-bit count of the data cells transmitted. Rolls to 0 from FFFFh. Initialize to 0. After initialization, do not write to this word.

TX_HEAD(1:2) Word Format (08$_H$)

This word is maintained by the microprocessor. . Note: in UDF-HS mode, TALP reads this word only once, before it generates the first cell of the connection; as a result, any writes to this word after TALP generates the first cell will have no affect. TALP simply reads the TX_HEAD locations, then writes them into the UTOPIA FIFO during cell construction.

Field (Bits)	Description
TX_HEAD(1) (15:8)	First header byte in bits 15:8. Initialize to the proper value.
TX_HEAD(2) (7:0)	Second header byte in bits 7:0. Initialize to the proper value.

TX_HEAD(3:4) Word Format (09$_H$)

This word is maintained by the microprocessor. . Note: in UDF-HS mode, TALP reads this word only once, before it generates the first cell of the connection; as a result, any writes to this word after TALP generates the first cell will have no affect.

Field (Bits)	Description
TX_HEAD(3) (15:8)	Third header byte in bits 15:8. Initialize to the proper value.
TX_HEAD(4) (7:0)	Fourth header byte in bits 7:0. Initialize to the proper value.

TX_HEAD(5) Word Format (0A$_H$)

This word is maintained by the microprocessor. . Note: in UDF-HS mode, TALP reads this word only once, before it generates the first cell of the connection; as a result, any writes to this word after TALP generates the first cell will have no affect.

Field (Bits)	Description
TX_HEAD(5) (15:8)	Fifth header byte that contains the precalculated HEC word. TALP neither calculates nor verifies the HEC. Initialize to the proper value.
Not used (7:0)	Write with a 0 to maintain compatibility with future software versions.

QUE_CREDITS Word Format (0B$_H$)

After initialization this word is read only. The word is maintained by CSD.

Field (Bits)	Description
FRAME_REMAINDER (15:14)	A 2-bit quantity representing the remainder of the division operation the CSD performs when converting the frame differential (expressed in frames) to the frame differential (expressed in eighths of multiframes). This quantity is maintained by the CSD. Initialize to
STR_PTR_SENT (13)	Indicates a structure pointer has been sent in the current set of 8 cells. Initialize to "0".
Reserved (12)	Initialize to "0".
Reserved (11)	Initialize to "0".
FIRST_CELL_SCHED (10)	Indicates the first cell has been scheduled for this queue. Initialize to 0.
QUEUE_CREDITS (9:0)	A 10-bit quantity representing the number of byte credits accumulated for the queue. It is measured in eighths (three LSBs are fractional bits). Initialize to the following: UDF-ML: 178$_H$ (47 x 8) SDF-FR, single DS0, no pointer: 178$_H$ (47 x 8) SDF-FR (except above): 177$_H$ (46.875 x 8) AAL0: 180$_H$ (48 x 8) Partial Cells: #Bytes per cell x 8 SDF-MF, E1, single DS0: 187$_H$ ((46.875 + 2) * 8) SDF-MF T1, single DS0: 17F$_H$ ((46.875 + 1) * 8) SDF-MF E1, two DS0s: 17F$_H$ ((46.875 + 1) * 8) SDF-MF (except above): 177$_H$ (46.875 * 8) UDF-HS and Low_CDV mode: NOT USED

CSD_CONFIG Word Format (0C$_H$)

This word is maintained by the microprocessor.

Field (Bits)	Description
NUM_CHAN (15:10)	A 6-bit integer specifying the number of DS0 channels being carried by this queue. If a queue serves seven DS0s, initialize this field to 7. This field has to be set to 32 in UDF-ML mode. It is not used in UDF-HS mode.

Field (Bits)	Description
AVG_SUB_VALU (9:0)	A 10-bit integer representing the average number of data bytes per cell measured in eighths. The three LSBs represent bits after the fixed decimal point. Initialize to 46.875 (0101110.111) for full cells when in SDF-FR or SDF-MF mode. Initialize to 47 (0101111.000) for full cells when in UDF-ML mode. For AAL0 cells, this value is 48 (0110000.000). For partial cells, this value is the same as the partially filled value x 8. This field is not used in UDF-HS mode.

T_CHANNEL_ALLOC(15:0) Word Format (0E$_H$)

This word is maintained by the microprocessor.

Field (Bits)	Description
T_CHANNEL_ALLOC (15:0)	A bit table with a bit set per DS0 allocated to this queue for DS0s 15 to 0 on the line defined by queue / 32. Initialize to the proper value for SDF-MF and SDF-FR modes and to FFFF$_h$ for UDF-ML mode. This field is not used for UDF-HS mode.

T_CHANNEL_ALLOC(31:16) Word Format (0F$_H$)

This word is maintained by the microprocessor.

Field (Bits)	Description
T_CHANNEL_ALLOC (31:16)	A bit table with a bit set per DS0 allocated to this queue for DS0s 31 to 16 on the line defined by queue / 32. Initialize to the proper value for SDF-MF and SDF-FR modes and to FFFF$_h$ for UDF-ML mode. This field is not used for UDF-HS mode.

T_CHANNEL_LEFT(15:0) Word Format (10$_H$)

After initialization this word is read only. The word is maintained by TALP.

Field (Bits)	Description
T_CHANNEL_LEFT (15:0)	Initialize to the same value as T_CHAN_ALLOC(15:0). This field is not used for UDF-HS mode.

T_CHANNEL_LEFT(31:16) Word Format (11$_H$)

After initialization this word is read only. The word is maintained by TALP.

Field (Bits)	Description
T_CHANNEL_LEFT (31:16)	Initialize to the same value as T_CHAN_ALLOC(31:16). This field is not used for UDF-HS mode.

TRANSMIT_CONFIG Word Format (12$_H$)

This word is maintained by the microprocessor. Note: in UDF-HS mode, TALP reads this word only once, before it generates the first cell of the connection; as a result, any writes to this word after TALP generates the first cell will have no affect.

Field (Bits)	Description
SUPPRESS_XMT (15)	Set to 1 to suppress the generation of cells for this queue. Cells are scheduled but not transmitted. Note that this bit is invalid in UDF-HS mode. A UDF-HS queue needs to be stopped by clearing the TX_ACTIVE bit and restarted by adding the queue.
LOOPBACK_ENABLE (14)	Set to 1 to loopback cell to receive side. Set VPI/VCI to corresponding receive queue number. Note to maximize throughput, this register is only read once in UDF-HS mode, when building the first cell. Therefore if loopback is desired for UDF-HS mode, this bit must be set first and cannot be changed after the queue is already running.
AAL0_MODE_ENABLE (13)	Set to 1 to build AAL0 cells instead of AAL1. Use QUEUE_CREDITS=x0180, AVG_SUB_VALU=x0180, and BYTES_PER_CELL=x30 for full AAL0 cells.
COND_MODE (12:11))	Selects the conditioning mode with the following encoding: 00 Both signaling and data are conditioned 01 Only signaling is conditioned 10 Only data is conditioned 11 reserved The chosen mode takes effect when the TX_COND bit is set in the QUEUE_CONFIG memory register. If data is conditioned the T_COND_CELL_CNT counter will increment. If only signaling is conditioned the T_CELL_CNT will increment as normal.
Reserved (10)	Initialize to "0"
Reserved (9)	Initialize to "0"
Not used (8:0)	Write with a 0 to maintain future software compatibility.

10.3.8 RESERVED (Transmit Data Buffer)

This structure is reserved and must be initialized to 0 at initial setup. Software modifications to this location after setup will cause incorrect operation.

Organization: 4 Kbytes x 4 lines.

Each line is allocated a separate 128 frame buffer memory. For E1 applications, this is large enough to store eight multiframes (32 DS0s x 16 frames x 8 multiframes = 4096 bytes). In T1 mode, 96 frames or four multiframes are stored (24 * 24 * 4 = 2304 bytes). T1 storage uses 32 bytes per frame and 32 frames per multiframe to simplify address generation.

Every data byte is stored in the multiframe line buffers in the order in which it arrives.

If E1_with_T1_SIG is set, data is arranged as if in T1 mode.

Base address within A1SP: 4000_h

Index(line): 800_h

Type: Read/Write

Function: Stores the outgoing data.

Format: Two data bytes per word, 16 words per frame.

T_DATA_BUFFER Word Format

Field (Bits)	Description
T_DATA_H (15:8)	Transmit data for: Channel = (offset mod 16) x 2 + 1. E1 offset = line x 2048 + multiframe x 256 + frame x 16 + (channe– -1) / 2. T1 offset = line x 2048 + multiframe x 512 + frame x 16 + (channe– -1) / 2.
T_DATA_L (7:0)	Transmit data for: Channel = (offset mod 16) x 2. E1 offset = line x 2048 + multiframe x 256 + frame x 16 + channel / 2. T1 offset = line x 2048 + multiframe x 512 + frame x 16 + channel / 2.

10.4 Receive Data Structures Summary

Table 25 lists the data structures unique to the receive side of the AAL1gator-4/8.

Note the addresses listed below are the offsets within the A1SP address space.

Name	Org	Size	Addr	Description
R_OAM_QUEUE_TBL	2 words	4 bytes	8000_h–8001_h	Receive OAM head and tail pointers.
R_OAM_CELL_CNT	1 word	2 bytes	8002_h	Count of received OAM cells.
R_DROP_OAM_CELL	1 word	2 bytes	8003_h	Count of dropped OAM cells.
Reserved	16 words	32 bytes	8020_h-$802F_h$	Reserved (SRTS Queue Pointers).
R_SRTS_CONFIG	2 bytes x 8 lines	16 bytes	8038_h-$803F_h$	Receive SRTS configuration.
R_CRC_SYNDROME	128 words	256 bytes	8080_h-$80FF_h$	Mask of bits. Initialized from a table.
R_CH_TO_QUE_TBL	128 words	256 bytes	8200_h-$827F_h$	Receive channel to queue table.
R_COND_SIG	16 x 8 bytes	256 bytes	8400_h-$847F_h$	Receive signaling conditioning values.
R_COND_DATA	32 x 8 bytes	256 bytes	8480_h-$84FF_h$	Receive data conditioning values.
Reserved	8 x 256 words	4 Kbytes	8800_h-$8FFF_h$	Reserved (Receive SRTS Queue).
Reserved	8 x 32 x 16 words	8 Kbytes	9000_h-$9FFF_h$	Reserved (Receive Signaling Buffer).
R_QUEUE_TBL	256 x 32 words	16 Kbytes	$A000_h$-$BFFF_h$	Receive queue table.
R_OAM_QUEUE	256 x 64 bytes	16 Kbytes	$E000_h$-$FFFF_h$	Receive OAM queue.

Name	Org	Size	Addr	Description
Reserved	8 x 512 x 32 bytes	128 Kbytes	1 0000$_h$- 1 FFFF$_h$	Reserved (Receive Data Buffer).

This section describes the structures used by the receive side of the AAL1gator-4/8.

Notes:

1. All ports marked as "Reserved" must be initialized to 0 at initial setup. Software modifications to these locations after setup will cause incorrect operation.

2. All read/write port bits marked "Not used" must be written with the value 0 to maintain software compatibility with future versions.

3. All read-only port bits marked "Not used" are driven with a 0 and should be masked off by the software to maintain compatibility with future versions.

10.4.1 R_OAM_QUEUE_TBL

Organization: 2 words

Base address within A1SP: 8000$_h$

Index: 1$_h$

Type: Read/Write

Function: OAM cells received from the ATM side are stored in a FIFO queue in the memory. Head and tail pointers are used to keep track of the read and write locations of the OAM cell buffers. There are 256 cell buffers in the OAM receive queue. Of these 256 cell buffers, 255 are usable. The 256th buffer is used to detect a full queue as follows:

When the queue is empty, OAM_HEAD = OAM_TAIL = N. When a cell is received, the cell is written into the buffer at index (OAM_TAIL + 1) mod 256, and OAM_TAIL is replaced with (OAM_TAIL + 1) mod 256. When the processor receives an interrupt, it reads the cell at the buffer index (OAM_HEAD + 1) mod 256. After completing the read, it sets OAM_HEAD to (OAM_HEAD + 1) mod 256. This process is continued until OAM_HEAD = OAM_TAIL, at which time the OAM receive queue is empty. The receive OAM interrupt can be cleared by asserting the CLR_RX_OAM_LATCH bit in the CMD_REG.

If an OAM cell arrived between the time the OAM_TAIL was last read and CLR_RX_OAM_LATCH was asserted, this OAM cell's arrival can be detected within the interrupt service routine by re-reading OAM_TAIL after CLR_RX_OAM_LATCH was asserted.

OAM Queue Format

Offset	Name	Description
0	OAM_HEAD	Head pointer
1	OAM_TAIL	Tail pointer

OAM_HEAD Word Format

Field(Bits)	Description
OAM_HEAD (7:0)	The microprocessor should increment to the next cell location when it reads a cell. Initialize to 0.

OAM_TAIL Word Format

Field(Bits)	Description
OAM_TAIL (7:0)	Incremented by the RALP after it writes a cell to the OAM cell queue. Initialize to 0.

10.4.2 R_OAM_CELL_CNT

Organization: 1 word

Base address within A1SP: 8002_h

Index: 1_h

Type: Read/Write

Function: 16-bit rollover counter that counts the number of OAM cells received. The software must initialize this counter to 0 during reset.

R_OAM_CELL_CNT Word Format

Field(Bits)	Description
R_OAM_CELL_CNT (15:0)	16-bit rollover counter that counts the number of OAM cells received. The software must initialize this counter to 0 during reset. After initialization, do not write to this word.

10.4.3 R_DROP_OAM_CELL

Organization: 1 word

Base address within A1SP: 8003_h

Index: 1_h

Type: Read/Write

Function: 16-bit rollover counter that counts the number of dropped OAM cells. The software should initialize this counter to 0 during reset.

R_DROP_OAM_CELL Word Format

Field(Bits)	Description
R_DROP_OAM_CELL (15:0)	16-bit rollover counter that counts the number of OAM cells dropped. OAM cells are dropped when more than 255 are present in the receive queue. The software must initialize this counter to 0 during reset. After initialization, do not write to this word.

10.4.4 R_SRTS_CONFIG

Organization: 2 bytes x 8 lines 4 lines.

Base address within A1SP: 8038_h

Index: 1_h

Type: Read/Write

Function: This table stores the CDVT for the SRTS channel, expressed in the number of queued SRTS nibbles.

Initialization: Initialize to the number of SRTS nibbles equivalent to the CDVT for the data by rounding up. Each frame of CDVT for unstructured applications represent 256 bits. Each SRTS nibble represents 3008 bits, which is the number of data bits in eight cells. Therefore, the number of SRTS nibbles that corresponds to the CDVT can be determined by dividing the CDVT number in frames by 3008 / 256, or 11.75, and rounding up to the next higher integer.

Format: One byte per line. Refer to the following table

R_SRTS_CONFIG Format

Offset	Name	Description
0_h	R_SRTS_CDVT_0	Receive SRTS CDVT for line 0
1_h	R_SRTS_CDVT_1	Receive SRTS CDVT for line 1
2_h	R_SRTS_CDVT_2	Receive SRTS CDVT for line 2
3_h	R_SRTS_CDVT_3	Receive SRTS CDVT for line 3
4_h	R_SRTS_CDVT_4	Receive SRTS CDVT for line 4
5_h	R_SRTS_CDVT_5	Receive SRTS CDVT for line 5
6_h	R_SRTS_CDVT_6	Receive SRTS CDVT for line 6
7_h	R_SRTS_CDVT_7	Receive SRTS CDVT for line 7

R_SRTS_CDVT_n Word Format

Field(Bits)	Description
Not used (15:5)	Write with 0 to maintain compatibility with future software versions.
R_SRTS_CDVT (5:0)	Receive SRTS CDVT

10.4.5 R_CRC_SYNDROME

Organization: 128 words

Base address within A1SP: 8080$_h$

Index: 1$_h$

Type: Read/Write

Function: This table identifies which bit of the SN/SNP byte has been corrupted, if any. Load after each power cycle. Used internally to perform CRC correction.

R_CRC_SYNDROME Word Format

Field(Bits)	Description
Not used (15:5)	Write with 0 to maintain compatibility with future software versions.
RX_CRC_SYNDROME (4:0)	Mask of bits to change.

Figure 69   R_CRC_SYNDROME Mask Bit Table Legend

```
              LEGEND

    00  No errors
    01  Correct bit 0
    02  Correct bit 1
    04  Correct bit 2
    08  Correct bit 3
    10  SNP error (no need
        to correct SN field)
```

Table 13   R_CRC_SYNDROME Mask Bit Table

Sequence Number	Offset	Data (Hex)		Sequence Number	Offset	Data (Hex)
0	00	00		4	40	08
0	01	10		4	41	10
0	02	10		4	42	04
0	03	01		4	43	02
0	04	10		4	44	10
0	05	08		4	45	00
0	06	02		4	46	01
0	07	04		4	47	10
0	08	01		4	48	02
0	09	10		4	49	04

Sequence Number	Offset	Data (Hex)		Sequence Number	Offset	Data (Hex)
0	0A	10		4	4A	10
0	0B	00		4	4B	08
0	0C	04		4	4C	10
0	0D	02		4	4D	01
0	0E	08		4	4E	00
0	0F	10		4	4F	10
1	10	02		5	50	01
1	11	04		5	51	10
1	12	10		5	52	10
1	13	08		5	53	00
1	14	10		5	54	04
1	15	01		5	55	02
1	16	00		5	56	08
1	17	10		5	57	10
1	18	08		5	58	00
1	19	10		5	59	10
1	1A	04		5	5A	10
1	1B	02		5	5B	01
1	1C	10		5	5C	10
1	1D	00		5	5D	08
1	1E	01		5	5E	02
1	1F	10		5	5F	04
2	20	04		6	60	10
2	21	02		6	61	01
2	22	08		6	62	00
2	23	10		6	63	10
2	24	01		6	64	02
2	25	10		6	65	04
2	26	10		6	66	10
2	27	00		6	67	08
2	28	10		6	68	10
2	29	08		6	69	00
2	2A	02		6	6A	01
2	2B	04		6	6B	10
2	2C	00		6	6C	08
2	2D	10		6	6D	10
2	2E	10		6	6E	04
2	2F	01		6	6F	02
3	30	10		7	70	10

Sequence Number	Offset	Data (Hex)		Sequence Number	Offset	Data (Hex)
3	31	00		7	71	08
3	32	01		7	72	02
3	33	10		7	73	04
3	34	08		7	74	00
3	35	10		7	75	10
3	36	04		7	76	10
3	37	02		7	77	01
3	38	10		7	78	04
3	39	01		7	79	02
3	3A	0		7	7A	08
3	3B	10		7	7B	10
3	3C	02		7	7C	01
3	3D	04		7	7D	10
3	3E	10		7	7E	10
3	3F	08		7	7F	00

10.4.6 R_CH_TO_QUEUE_TBL

Organization: 128 words (8 lines x 32 DS0s) 4 lines

Base address within A1SP: 8200_h

Index: 1_h

Type: Read/Write

Hardware Reset Value: 8080_h (this table is located inside device)

Function: This table associates the DS0 with the queue. It allows the transmit line interface to determine the status of the receive queue supplying bytes for the DS0s being processed. This table is located inside the chip and all time slots are initialized to play out conditioned data. The AAL1gator-4/8 processes two bytes at a time so the values in the following table are in pairs. For unstructured, low speed lines, set all of the queue values to the receive queue number mod 32. In UDF-HS mode, this table is not used. When this queue is in underrun, the AAL1gator-4/8 reads data for the line from the first word of the R_COND_DATA_0 table.

Format: Refer to the following table.

R_CH_TO_QUEUE_TBL Format

Offset	Name	Description
N	R_CH_TO_QUEUE	Queue numbers and condition bits associated with this pair of channels where: Line = N / 16. Low channel = (N mod 16) x 2. High channel = (N mod 16) x 2 + 1.

R_CH_TO_QUEUE Word Format

Field(Bits)	Description
RX_COND_H (15:14)	Determines the type of data to be played out: Options "00", "01", and "11" are executed only when the queue is in an underrun or resume state. 00_b When the queue is in underrun, freeze signaling and read the data for this channel from the R_COND_DATA table. 01_b When the queue is in underrun, freeze signaling and play out pseudorandom data, which is inserted data from R_COND_DATA, with the MSB controlled by the pseudorandom number algorithm $x^{18} + x^7 + 1$ (not valid for UDF-HS). 10_b Read signaling for this channel from the R_COND_SIG table and the data for this channel from the R_COND_DATA table. 11_b When the queue is in underrun freeze signaling and play out the contents of the buffer.
RX_SIG_COND_H (13)	Overrides the normal signaling with Conditioned signaling 0_b Read signaling as indicated by RX_COND_H 1_b Always read signaling for this channel from the R_COND_SIG table
QUEUE_H (12:8)	Five LSBs of the queue index associated with this DS0. The three MSBs are implicitly those of the line number. Offset = (channe– - 1) / 2 + line x 16. For unstructured lines, set to the receive queue number mod 32.
RX_COND_L (7:6)	Determines the type of data to be played out: Options "00", "01", and "11" are executed only when the queue is in an underrun or resume state. 00_b When the queue is in underrun, freeze signaling and read the data for this channel from the R_COND_DATA table. 01_b When the queue is in underrun, freeze signaling and play out pseudorandom data, which is inserted data from R_COND_DATA, with the MSB controlled by the pseudorandom number algorithm $x^{18} + x^7 + 1$ (not valid for UDF-HS). 10_b Read signaling for this channel from the R_COND_SIG table and the data for this channel from the R_COND_DATA table. 11_b When the queue is in underrun, freeze signaling and play out the contents of the buffer.
RX_SIG_COND_L (5)	Overrides the normal signaling with Conditioned signaling 0_b Read signaling as indicated by RX_COND_L 1_b Always read signaling for this channel from the R_COND_SIG table
QUEUE_L (4:0)	Five LSBs of the queue index associated with this DS0. The three MSBs are implicitly those of the line number. Offset = channel / 2 + line x 16.

10.4.7 R_COND_SIG

Organization: 16 words x 8 4 lines

Base address within A1SP: 8400_h

Index: 10_h

Type: Read/Write

Function: This table stores the signaling to be used when RX_SIG_COND_H or RX_SIG_COND_L equals "1" in the R_CH_TO_QUEUE_TBL.

Initialization: Initialize to the conditioned signaling value for the channel. This value typically depends on the type of channel unit that is connected. For example, an FXO channel unit needs a different conditioning value than an FXS channel unit.

Format: One nibble per byte, two bytes per word, 16 words per line. Refer to the following table

R_COND_SIG Format

Offset	Name	Description
00000_h	R_COND_SIG_0	Receive conditioned signaling for line 0.
00010_h	R_COND_SIG_1	Receive conditioned signaling for line 1.
00020_h	R_COND_SIG_2	Receive conditioned signaling for line 2.
00030_h	R_COND_SIG_3	Receive conditioned signaling for line 3.
00040_h	R_COND_SIG_4	Receive conditioned signaling for line 4.
00050_h	R_COND_SIG_5	Receive conditioned signaling for line 5.
00060_h	R_COND_SIG_6	Receive conditioned signaling for line 6.
00070_h	R_COND_SIG_7	Receive conditioned signaling for line 7.

R_COND_SIG_n Word Format

Field (Bits)	Description
Not used (15:12)	Write with a 0 to maintain future software compatibility.
R_COND_A_H (11)	Receive conditioned A signaling bit for: Offset = (channe– - 1) / 2 + line x 16.
R_COND_B_H (10)	Receive conditioned B signaling bit for: Offset = (channe– - 1) / 2 + line x 16.
R_COND_C_H (9)	Receive conditioned C signaling bit or A bit if T1 SF for: Offset = (channe– - 1) / 2 + line x 16.
R_COND_D_H (8)	Receive conditioned D signaling bit or B bit if T1 SF for: Offset = (channe– - 1) / 2 + line x 16.
Not used (7:4)	Write with a 0 to maintain future software compatibility.
R_COND_A_L (3)	Receive conditioned A signaling bit for: Offset = (channel / 2) + line x 16.

Field (Bits)	Description
R_COND_B_L (2)	Receive conditioned B signaling bit for: Offset = (channel / 2) + line x 16.
R_COND_C_L (1)	Receive conditioned C signaling bit or A bit if T1 SF for: Offset = (channel / 2) + line x 16.
R_COND_D_L (0)	Receive conditioned D signaling bit or B bit if T1 SF for: Offset = (channel / 2) + line x 16.

10.4.8 R_COND_DATA

Organization: 16 words x 8 4 lines

Base address within A1SP: 8480_h

Index: 10_h

Type: Read/Write

Function: This table stores the data to be used when RX_COND in the R_CH_TO_QUEUE_TBL equals "00_b", "01_b", or "10_b".

Initialization: Initialize to the conditioned data appropriate for the channel. This typically depends on the type of channel connected to the device. For example, data usually needs an FF value and voice needs a small PCM value.

Format: Two bytes per word, 16 words per line. Refer to the following table.

R_COND_DATA Format

Offset	Name	Description
00000_h	R_COND_DATA_0	Receive conditioned data for line 0.
00010_h	R_COND_DATA_1	Receive conditioned data for line 1.
00020_h	R_COND_DATA_2	Receive conditioned data for line 2.
00030_h	R_COND_DATA_3	Receive conditioned data for line 3.
00040_h	R_COND_DATA_4	Receive conditioned data for line 4.
00050_h	R_COND_DATA_5	Receive conditioned data for line 5.
00060_h	R_COND_DATA_6	Receive conditioned data for line 6.
00070_h	R_COND_DATA_7	Receive conditioned data for line 7.

R_COND_DATA_n Word Format

Field (Bits)	Description
R_COND_DATA_H (15:8)	Receive conditioned data for: Offset = (channe– - 1) / 2 + line x 16.
R_COND_DATA_L (7:0)	Receive conditioned data for: Offset = channel / 2 + line x 16.

10.4.9 RESERVED (Receive SRTS Queue)

This structure is reserved. Software modifications to this structure after setup will cause incorrect operation.

Organization: 64 words x 8 lines. 4 lines. Each line is allocated a separate 64-entry queue to store the SRTS receive nibbles.

Base address within A1SP: 8800_h

Index: 100_h

Type: Read/Write

Function: The receive signaling queue stores the SRTS bits received from the UTOPIA interface.

Initialization: It is not necessary to initialize this structure.

Format: One SRTS nibble per word.

R_SRTS_QUEUE_n Word Format

Field (Bits)	Description
Not used (15:8)	Write with a 0 to maintain future software compatibility.
R_SRTS_VAL (7:4)	Indicates if each SRTS bit contains valid data. When an error occurs which causes a bit to be lost the corresponding bit will be written with a 0. Each time a new entry is written, the remaining bits which haven't been received yet will also be written with a 0. So normally this field will be written with a "1000" for the first bit then "1100" for the second bit, then "1110" for the third bit and finally "1111" for the last bit.
R_SRTS (3:0)	Receive SRTS data for line = offset / 64.

10.4.10 RESERVED (Receive Signaling Buffer)

This structure is reserved. Software modifications to this structure after setup will cause incorrect operation.

Organization: 32 x 32 DS0s x 8 lines. 4 lines.   Each line is allocated a separate 32 x 32 byte memory. For E1, this allows storage of signaling information for 32 multiframes, unless E1_WITH_T1_SIG is set. T1 applications use only the first 24 bytes of every 32 to store signaling data. In addition, since the receive data buffer is only 16 multiframes in size, this structure also needs to store only 16 multiframes. Successive multiframes are stored in every other 32-byte buffer. When signaling is frozen due to an underrun, the value in multiframe 0 is used.

Base address within A1SP: 9000_h

Index (line): 200$_h$

Type: Read/Write

Function: The receive signaling queue stores the signaling that is received from the UTOPIA interface.

Initialization: The signaling buffer should be initialized to "0". Also, if R_CHAN_NO_SIG is set for some queues and a specific signaling value is desired to be driven for these queues, then the DS0s in those queues must be initialized to the desired value for all multiframes.

Format: Two signaling nibbles per word.

R_SIG_BUFFER_n Word Format

Field (Bits)	Description
Not used (15:13)	Write with a 0 to maintain future software compatibility.
R_SIG_INVALID (12)	Indicates that the stored signaling is invalid. If signaling is not valid due to lost cells, signaling will freeze.
R_SIG_H (11:8)	Receive signaling data for: Channel = (offset mod 16) x 2 + 1. Multiframe = (offset mod 512) / 16. Line = offset / 512. Offset = line x 512 + multiframe x 16 + (channe– - 1) / 2.
Not used (7:5)	Write with a 0 to maintain future software compatibility.
R_SIG_INVALID (4)	Indicates that the stored signaling is invalid. If signaling is not valid due to lost cells, signaling will freeze.
R_SIG_L (3:0)	Receive signaling data for: Channel = (offset mod 16) x 2. Multiframe = (offset mod 512) / 16. Line = offset / 512. Offset = line x 512 + multiframe x 16 + channel / 2.

10.4.11 R_QUEUE_TBL

Organization: 256 x 32 words. 128 queues.

Base address within A1SP: A000$_h$

Index: 20$_h$

Type: Read/Write

Function: Receive Queue Table contains all the structures and pointers specific to a queue. The RALP and RFTC blocks both use the R_QUEUE_TBL. Some of the words are read by both the blocks but written by only one of the blocks.

Format: Each queue is allocated 32 consecutive words. Each word is 16-bits wide. The organization of the words is as follows.

Table 14   R_QUEUE_TBL Format

Offset	Name	Description
0h	R_STATE_0	Cell receiver state 0.
1h	R_MP_CONFIG	Bytes per cell and CDVT constant.
2h	R_STATE_1	Cell receiver state 1.
3h	R_LINE_STATE	Line state.
4h	R_BUF_CFG	Receive maximum buffer size.
5h	R_SEQUENCE_ERR	16-bit rollover count of SN errors.
6h	R_INCORRECT_SNP	16-bit rollover count of cells with incorrect SNP.
7h	R_CELL_CNT	16-bit rollover count of played out cells.
8h	R_ERROR_STKY	Receive sticky bits.
9h	R_TOT_SIZE	Total bytes in structure.
Ah	R_DATA_LAST	Number of signaling bytes in structure.
Bh	R_TOT_LEFT	Number of bytes remaining in the structure. Initialize to 0 each time this queue is initialized.
Ch	Not used	Initialize to 0 each time this queue is initialized.
Dh	R_SN_CONFIG	Configures sequence number processing algorithm.
Eh	R_CHAN_ALLOC (15:0)	A bit table with a bit set per DS0 allocated to this queue for DS0s 15 to 0 on the line defined by queue / 32.
Fh	R_CHAN_ALLOC (31:16)	A bit table with a bit set per DS0 allocated to this queue for DS0s 31 to 16 on the line defined by queue / 32.
10h	Reserved (R_CHAN_LEFTL)	Initialize to 0 each time this queue is initialized.
11h	Reserved (R_CHAN_LEFTH)	Initialize to 0 each time this queue is initialized.
12h	R_DROPPED_CELLS	16-bit rollover count of cells that were received but dropped. Initialize to 0.
13h	R_UNDERRUNS	16-bit rollover count of the occurrences of underrun on this queue. Initialize to 0.
14h	R_LOST_CELLS	16-bit rollover count of the number of lost cells for this queue. Initialize to 0.
15h	R_OVERRUNS	16-bit rollover count of the occurrences of overrun on this queue. Initialize to 0.
16h	R_PTR_REFRAMES	16-bit rollover count of the occurrences of pointer reframes. Initialize to 0.
17h	R_PTR_PAR_ERR	16-bit rollover count of the occurrences of pointer parity errors. Initialize to 0.
18h	R_MISINSERTED	16-bit rollover count of the occurrences of misinserted cells. Initialize to 0.
19h	R_ROBUST_SN	Write pointer for robust SN processing
1Ah	Reserved	Initialize to 0 each time this queue is initialized.
1Bh	Reserved	Initialize to 0 each time this queue is initialized.
1Ch	R_RD_PTR_LAST	Read pointer for bit integrity through underrun
1Dh-1Fh	Not used	Initialize to 0 each time this queue is initialized.
All of these locations must be initialized whenever the queue is initialized.		

R_STATE_0 Word Format (00$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
Reserved (15)	Initialize to 0.
R_STRUCT_FOUND (14)	Indicates that the receiver structure was found. Initialize to 0.
Reservd(OLDUNDRN_N) (13)	Initialize to 0 to maintain future software compatibility.
Reservd(UNDRN_2AGO) (12)	Initialize to 0 to maintain future software compatibility.
Reserved(ACTSN) (11:9)	Initialize to 0 to maintain future software compatibility.
SN_STATE (8:6)	Specifies the state of the SN state machine. Initialize to 0.
2ND_LAST_SN (5:3)	Specifies the SN that was received two cells ago. Initialize to 0.
LAST_SN (2:0)	Specifies the last SN that was received. Initialize to 0.

R_MP_CONFIG Word Format (01$_H$)

This word is maintained by the microprocessor.

Field (Bits)	Description
R_CHK_PARITY (15)	If set, check the parity on the incoming structure pointer.
R_BYTES_CELL (14:9)	A 6-bit integer specifying how many bytes per cell are required if no structure pointers are used. For UDF-HS mode, this must be set to 47. In other modes, set this to the partially filled length. If cells are not partially filled, set this to 47.
R_AAL0_MODE (8)	If set, treats this queue as an AAL0 queue and will write all 48 bytes of payload into the allocated time slots. Use R_BYTES_CELL=x30 for full AAL0 cells.
R_CDVT (7:0)	Receive Cell Delay Variation Tolerance (R_CDVT) is a constant and is programmed by the microprocessor during initialization. It is used by the RFTC after the receipt of the first cell after an underrun. In T1 SDF-FR, T1 SDF-MF, and E1_WITH_T1_SIG, modes, R_CDVT is expressed as the number of multiframes in bits 7:5 and the number of frames in bits 4:0. In E1 and all other T1 modes, R_CDVT is the number of frames. In unstructured applications, the number of frames refers to the number of 256-bit increments. For T1 unstructured modes, this is equivalent to the number of 165.8 us periods. For Robust SN Processing, this field represents the R_CDVT desired plus the number of frames stored in the cell that is conditionally stored. The minimum recommended value is R_CDVT=2.

R_STATE_1 Word Format (02_H)

This word is read-only and is maintained by the RALP. This register is located inside the chip and is reset to "0000"

Field (Bits)	Description
Reserved (FRC_UNDRN) (15)	Initialize to 0 to maintain future software compatibility.
Reserved (SNCRCST) (14)	Initialize to 0 to maintain future software compatibility.
Reserved (PTRMMST) (13)	Initialize to 0 to maintain future software compatibility.
Reserved (FNDPTR) (12)	Initialize to 0 to maintain future software compatibility.
Reserved (FNDFRSTPTR) (11)	Initialize to 0 to maintain future software compatibility.
Reserved (10)	Initialize to 0 to maintain future software compatibility.
Not used (9)	Driven with a 0. Mask on reads to maintain future software compatibility.
R_WRITE_PTR (8:0)	Pointer to the frame to which the cell receiver is writing the last accepted cell.

R_LINE_STATE Word Format (03_H)

This word is read-only after initialization and is maintained by the RALP and RFTC. This register is located inside the chip and is reset to "9000". This register is not used in UDF-HS mode.

Field (Bits)	Description
R_UNDERRUN (15)	Indicates that this queue is currently in underrun. Initialize to 1.
R_RESUME (14)	Indicates that this queue is currently in resume state. Initialize to 0.
R_SIG_RESUME (13)	Indicates that this queue is currently in signal resume state. Initialize to 0.
R_LONG_UNDERRUN (12)	Indicates that the rd_ptr has wrapped while the queue was in underrun. Initialize to 1.
Reserved (11:9)	Initialize to 0 to maintain future software compatibility.
R_END_UNDERRUN_PTR (8:0)	Location read pointer needs to reach after an underrun to begin playing out new data. Initialize to 0 to maintain future software compatibility.

R_BUF_CFG Word Format (04_H)

This word is maintained by the microprocessor

Field (Bits)	Description
R_CHAN_UNSTRUCT (15)	Set to 1 only when receiving cells with a single DS0 without a pointer in the SDF-FR mode. This bit is valid only in SDF-FR mode. To conform to the CES standard V 2.0 when using a single DS0 in SDF-FR mode, no pointer should be used. This bit can be ignored for AAL0 mode.
R_CHAN_NO_SIG (14)	Set to 1 to receive cells without signaling when the line is in SDF-MF mode. This is the same as using this queue in SDF-FR mode, which means that the structure forms on frame boundaries instead of multiframe boundaries. The R_SIG_BUFFER will never be updated for this queue. However, the TL_SIG output will drive the value that was initialized into this timeslot in T_SIG_BUFFER.
R_CHAN_DISABLE (13)	Set to 1 to drop all cells for this queue. Set to 0 for normal operation. Cells dropped because of this bit are recorded in the ALLOC_TBL_BLANK sticky bit.
BITI_UNDERRUN (12)	Set to 1 to maintain bit count integrity through underrun. Set to 0 for normal operation. This mode is not supported for UDF-HS mode.
Reserved (11:10)	Initialize to "00."
Reserved (9)	Initialize to "0".
R_MAX_BUF (8:0)	Receiver maximum buffer size. The R_MAX_BUF is coded as the number of frames. In all structured modes, this is the number of frames. In all unstructured modes, this is the number of 256-bit increments. If the amount of data in the receive buffer exceeds R_MAX_BUF, no more data will be written, an overflow will be reported, and the queue will be forced into underrun. The maximum value of R_MAX_BUF is 1FEh for most cases. For T1 structured mode or E1 with T1 signaling, the maximum value is 17Eh because not all frames are used. The value of R_MAX_BUF should be equal to or greater than two times R_CDVT, or R_CDVT plus two times the number of frames required per cell, whichever is greater. If MF_ALIGN_EN is set in LIN_STR_MODE, then extra margin should be added for the additional multi-frame of data that may be present. Therefore, the value should be increased by 16 frames for E1 or 24 frames for T1.

R_SEQUENCE ERROR Word Format (05H)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
R_SEQUENCE_ERR (15:0)	16-bit rollover count of SN errors. This counter counts transitions from the SYNC state to the OUT_OF_SEQUENCE state. This is the atmfCESAal1SeqErrors count from the CES specification. Note that if SN processing is disabled, this counter will count all out-of-sequence cells. Initialize to 0. Once initialized, do not write to this word.

R_INCORRECT_SNP Word Format (06H)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
R_INCORRECT_SNP (15:0)	16-bit rollover count of cells with SNP errors. This is the atmfCESHdrErrors counter from the CES specification. Initialize to 0. Once initialized, do not write to this word.

R_CELL_CNT Word Format (07$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
R_CELL_CNT (15:0)	16-bit rollover count of received cells. This is the atmfCESReassCells counter from the CES specification. Initialize to 0. Once initialized, do not write to this word.

R_ERROR_STKY Word Format (08$_H$)

Receive sticky bits should be used for statistics gathering purposes only as there is no means of clearing them without the possibility of missing

Field (Bits)	Description
TRANSFER (15)	This bit is read then written with the same value each time the AAL1gator-4/8 receives a cell. This feature allows the processor to determine if the AAL1gator-4/8 was in the middle of a read then write cycle when the processor cleared the other sticky bits. To accomplish this each time the processor wants to clear sticky bits, it should complement this bit. Then, if an additional read of this bit showed it to be the wrong value, then the AAL1gator-4/8 has had its sticky word update interrupted.
CELL_RECEIVED (14)	A cell was received.
Reserved (13)	Reserved.
PTR_RULE_ERROR (12)	There was a violation of a structure pointer generation rule. An eight-cell sequence begins with an SN=0 cell and ends with an SN=7 cell. This bit will be set if no structure pointer was received, more than one structure pointer was received, or an 'out of range' structure pointer was received in the sequence. This condition will be checked only in modes where a structure pointer is expected, and no sequence number error or underrun occurred.
ALLOC_TBL_BLANK (11)	A cell was dropped because of a blank allocation table or because R_CHAN_DISABLE in the R_BUF_CFG memory register was asserted.
POINTER_SEARCH (10)	A cell was dropped because a valid structure pointer has not yet been found.
FORCED_UNDERRUN (9)	A cell was dropped because a forced underrun condition exists. A forced underrun condition can be caused by an overrun, consecutive structure pointer mismatches, and consecutive sequence number errors.

Field (Bits)	Description
SN_CELL_DROP (8)	A cell was dropped in accordance with an SN Algorithm (as specified in ITU-T Recommendation I.363.1). If Fast SN processing is used and the line is not in high speed mode, this bit will always be set for the first cell if NODROP_IN_START = 0. This bit will also be set for the first cell if ROBUST SN processing is enabled.
POINTER_RECEIVED (7)	A structure pointer was received.
PTR_PARITY_ERR (6)	A cell was received with a structure pointer parity error.
SRTS_RESUME (5)	An SRTS resume has occurred. A valid SRTS value was received and stored in the R_SRTS_QUEUE.
SRTS_UNDERRUN (4)	A cell was received while the SRTS queue was in underrun.
RESUME (3)	A resume has occurred: a valid cell was received and stored into the buffer. The data carried in this cell will be played out after R_CDVT frames.
PTR_MISMATCH (2)	A cell was dropped because of a structure pointer mismatch. This event causes a forced underrun condition.
OVERRUN (1)	A cell was dropped due to overrun. The receive buffer depth exceeded R_MAX_BUF. This event causes a forced underrun condition.
UNDERRUN (0)	A cell was received while the queue was in underrun.

R_TOT_SIZE Word Format (09$_H$)

This word is maintained by the microprocessor

Field (Bits)	Description
FRAMES_PER_CELL (15:10)	Average number of frames contained within a single cell. This field is used only if BITI_UNDERRUN=1.
R_TOT_SIZE (9:0)	Total bytes minus one in the structure (for example, for an E1 MF VC with two DS0s, R_TOT_SIZE is set to 32). This field is not used in UDF-ML or UDF-HS mode. Three formulas for R_TOT_SIZE are: For T1/E1 SDF-FR: R_TOT_SIZE = no. of DS0– - 1 For T1 SDF-MF or E1 with T1 Signaling: R_TOT_SIZE = $(24 \times \text{no. of DS0s}) + \frac{(\text{no. of DS0s} + 1)}{2} - 1$ For E1 SDF-MF: R_TOT_SIZE = $(16 \times \text{no. of DS0s}) + \frac{(\text{no. of DS0s} + 1)}{2} - 1$

R_DATA_LAST Word Format (0A$_H$)

This word is maintained by the microprocessor

Field (Bits)	Description
Not used (15:13)	Write with a 0 to maintain future software compatibility.
LAST_CHAN (12:8)	Channel number (0 to 31) of the last DS0 with a bit set in the R_CHAN_ALLOC memory registers.
Not used (7:6)	Write with a 0 to maintain future software compatibility.
Reserved (5:4)	Write with a 0 to maintain future software compatibility.
R_DATA_LAST (3:0)	Number of signaling bytes in the structure, minus one. (For example, for an E1 SDF-MF VC with six DS0s, R_DATA_LAST is set to 2. An E1-SDF-MF VC with seven DS0s is set to 3 as one signaling nibble is unused.) Not used in UDF-ML, UDF-HS, and SDF-FR modes. R_DATA_LAST = RoundUp[# of DS0's/2] –1.

R_TOT_LEFT Word Format (0B$_H$)

This word is read-only and is maintained by RALP

Field (Bits)	Description
Not used (15)	Driven with a 0. Mask on reads to maintain future software compatibility.
Reserved (14)	Initialize to "0".
Reserved (13)	Initialize to "0".
Reserved (12:11)	Initialize to "0".
Not used (10)	Driven with a 0. Mask on reads to maintain future software compatibility.
R_TOT_LEFT (9:0)	Total bytes minus one remaining in the structure. Not used in UDF-ML or UDF-HS mode. Initialize to "0".

R_SN_CONFIG Word Format (0D$_H$)

This word is maintained by the microprocessor

Field (Bits)	Description
R_CONDQ_DATA (15:8)	Value of conditioned data inserted into lost cells depending on the value of INSERT_DATA.
ROBUST_SN_EN (7)	Set to 1 to enable the "Robust SN algorithm". Set to a "0' for the "Fast SN Algorithm". Note: Do not set if DISABLE_SN is set. Only one of these bits can be set.

Field (Bits)	Description
INSERT_DATA (6:5)	Controls the format of the data inserted for lost cells: 00_b Insert xFF. 01_b Insert data from R_CONDQ_DATA. 10_b Insert old data from receive buffer. 11_b Insert data from R_CONDQ_DATA with the MSB controlled by the pseudorandom number algorithm $x^{18} + x^7 + 1$ (not valid for UDF-HS).
DISABLE_SN (4)	If set, both "Fast SN Algorithm" and "Robust SN Algorithm" are disabled. RALP will neither drop nor insert cells due to SN errors, but will maintain both R_INCORRECT_SNP and R_INCORRECT_SN counters. This bit should be set to 0 for AAL0 mode.
NODROP_IN_START (3)	In the "Fast SN Algorithm" for SN processing, the first cell received will always be dropped because a sequence has not been established yet. This bit disables the automatic dropping of cells while in the START state 0 When SN_STATE equals "000"$_b$ any received cell will be dropped. When SN_STATE equals "000"$_b$ any received cell with valid SNP will be accepted. Note that in Robust SN processing this bit has no effect.
MAX_INSERT (2:0)	The maximum number of cells that will be inserted when cells are lost. If the number of cells lost exceeds MAX_INSERT, then the queue will be forced into underrun. If this value is set to 000_b, it is interpreted the same as 111_b, which means that up to seven cells will be inserted.

R_CHAN_ALLOC(15:0) Word Format (0E$_H$)

This word is maintained by the microprocessor

Field (Bits)	Description
R_CHAN_ALLOC (15:0)	A bit table with a bit set per DS0 allocated to this queue for DS0s 15 to 0 on the line defined by queue /32. In UDF-ML mode, initialize to FFFF$_h$. (DS0 15 is in bit 15). This field is not used for UDF-HS mode.

R_CHAN_ALLOC(31:16) Word Format (0F$_H$)

This word is maintained by the microprocessor

Field (Bits)	Description
R_CHAN_ALLOC (31:16)	A bit table with a bit set per DS0 allocated to this queue for DS0s 31 to 16 on the line defined by queue /32. In UDF-ML mode, initialize to FFFF$_h$. (DS0 31 is in bit 15). This field is not used for UDF-HS mode.

R_DROPPED_CELLS Word Format (12$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
R_DROPPED_CELLS (15:0)	16-bit rollover count of dropped non-OAM cells. Initialize to 0. Once initialized, do not write to this word. Cells may be dropped due to: Pointer mismatch. Overrun. Blank allocation table SN processing. Structured cell received while in underrun but structure start has not been found yet. NOTE: This counter will always increment for the first cell received, if: FAST SN processing and NODROP_IN_START is set. ROBUST SN processing is enabled.

R_UNDERRUNS Word Format (13$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
R_UNDERRUNS (15:0)	16-bit rollover count of the occurrences of an underrun on this queue. This is the atmfCESBufUnderflows counter. Initialize to 0. Once initialized, do not write to this word. Underruns are counted by the RALP, which does not know an underrun occurred until a cell is received while in underrun. To ensure the underrun count is correct, the counter is not incremented until the queue exits the underrun state and enters the resume state underrun condition. To determine if the queue is in underrun, check the level of the R_UNDERRUN bit in R_LINE_STATE register. If this bit is set, then increment the underrun count by one to get the current count. This counter does not count underruns which are the result of a FORCED_UNDERRUN.

R_LOST_CELLS Word Format (14$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
R_LOST_CELLS (15:0)	16-bit rollover count of cells that were detected as lost. This is the atmfCESLostCells counter in the CES specification. Initialize to 0. Once initialized, do not write to this word.

R_OVERRUNS Word Format (15$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
R_OVERRUNS (15:0)	16-bit rollover count of the occurrences of an overrun on this queue. This is the atmfCESBufOverflows counter in the CES specification. Initialize to 0. Once initialized, do not write to this word.

R_POINTER_REFRAMES Word Format (16$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
R_POINTER_REFRAME (15:0)	16-bit rollover count of the occurrences of pointer reframes on this queue. This is the atmfCESPointerReframes counter in the CES specification. Initialize to 0. Once initialized, do not write to this word.

R_PTR_PAR_ERR Word Format (17$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
R_PTR_PAR_ERR (15:0)	16-bit rollover count of the occurrences of pointer parity errors on this queue. This is the atmfCESPointerParityErrors counter in the CES specification. Initialize to 0. Once initialized, do not write to this word.

R_MISINSERTED Word Format (18$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
R_MISINSERTED (15:0)	16-bit rollover count of the occurrences of misinserted cells on this queue. This is the atmfCESMisinsertedCells counter in the CES specification. Initialize to 0. Once initialized, do not write to this word.

R_ROBUST_SN Word Format (19$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
Reserved (15)	Used to indicate when the first cell is received on a RSN connection.
R_RSN_RESUME (14)	Indication that the stored cell is the first cell after an underrun.
R_RSN_CHAN_PTR (13:9)	Pointer to the channel number in which to start if dropping a previously stored cell.
R_RSN_WRT_PTR (8:0)	Pointer to the frame to which the cell receiver is writing for Robust SN processing.

R_RD_PTR_LAST Word Format (1C$_H$)

This word is read-only and is maintained by the RALP

Field (Bits)	Description
Not used (15:9)	Driven with a 0. Mask on reads to maintain future software compatibility.
R_RD_PTR_LAST (8:0)	Pointer to the frame that was last read when the last cell was received. This is used to determine whether more than 6 cells have been lost when a SN error occurs to help maintain bit integrity through underrun.

10.4.12 R_OAM_QUEUE

Organization: 256 cells x 64 bytes

Base address within A1SP: E000$_h$

Index: 20$_h$

Type: Read/Write

Function: The receive signaling queue stores the signaling received from the UTOPIA interface.

Initialization: It is not necessary to initialize this structure.

Format: Two data bytes per word

R_OAM_QUEUE Format

Offset	Name	Description
00000$_h$	R_OAM_CELL_0	Receive OAM cell 0
00010$_h$	R_OAM_CELL_1	Receive OAM cell 1
.	.	.
.	.	.
.	.	.
01FFF$_h$	R_OAM_CELL_255	Receive OAM cell 255

R_OAM_CELL_n Format

Offset	Bits (15:8)	Bits (7:0)
Word 0	Header 1	Header 2
Word 1	Header 3	Header 4
Word 2	Header 4 (HEC)	Blank
Word 3	Payload 1	Payload 2
.	.	.
.	.	.
.	.	.
Word 26	Payload 47	Payload 48
Word 27	CRC_10_PASS	

CRC_10_PASS Word Format

Field (Bits)	Description
CRC_10_PASS (15)	The CRC_10_PASS bit is set if the cell passes the CRC-10 check.
Not used (14:0)	Write with a 0 to maintain future software compatibility.

10.4.13 RESERVED (Receive Data Buffer)

This structure is reserved and must be initialized to 0 at initial setup. If RX_COND for some channels is set to "11" (insert old data during underrun), then those channels may need to be initialized to some other value if "0" data is unacceptable, since all the queues will reset to the underrun state. Software modifications to this location after setup will cause incorrect operation.

Organization: Each line has a separate receive data buffer consisting of 512 frame buffers. Each frame buffer can store 32 bytes. For E1 structured data applications, this allows storage of 512 frames or 32 multiframes of data. Structured T1 applications use only the first 24 bytes of each frame buffer for data storage. Also, only the first 24 frame buffers of every 32 are used to store T1 structured data frames. This provides 384 frames of storage, or 16 multiframes. Unstructured applications store 256 bits of data in every frame buffer. For E1 with T1 signaling, use T1 structure but with 32 channels.

Base address within A1SP: 10000_h

Index (line): 2000_h

Type: Read/Write

Function: The data buffers store receive data information. The data is stored in the buffers in the order that they will be played out to the lines.

Initialization: Initial to 0 at startup. If RX_COND for some channels is set to "11" (insert old data during underrun), then those channels may need to be initialized to some other value if "0" data is unacceptable.

Format: Two data bytes per word.

R_DATA_BUFFER_n Word Format

Field (Bits)	Description
R_DATA_H (15:8)	Receive data for: Channel = (offset mod 16) x 2 + 1. E1 frame = (offset mod 256) / 16. T1 frame = (offset mod 512) / 16. E1 multiframe = (offset mod 8192) / 256. T1 multiframe = (offset mod 8192) / 512. Line = offset / 8192. E1 offset = line x 8192 + multiframe(E1) x 256 + frame(E1) x 16 + (chan-1) / 2. T1 offset = line x 8192 + multiframe(T1) x 512 + frame(T1) x 16 + (chan-1) / 2.

Field (Bits)	Description
R_DATA_L (7:0)	Receive data for: Channel = (offset mod 16) x 2. E1 frame = (offset mod 256) / 16. T1 frame = (offset mod 512) / 16. E1 multiframe = (offset mod 8192) / 256. T1 multiframe = (offset mod 8192) / 512. Line = offset / 8192. E1 offset = line x 8192 + multiframe(E1) x 256 + frame(E1) x 16 + channel / 2. T1 offset = line x 8192 + multiframe(T1) x 512 + frame(T1) x 16 + channel / 2.

11 Normal Mode Register Description

Normal mode registers are used to configure and monitor the operation of the AAL1gator-4/8. Internal Registers are selected when A[19] is high. Normal mode registers are selected when A[18] is low. Test registers are accessed when both A[19] and A[18] are high. Note all register bits that refer to physical links 3 through 7 are reserved locations for the AAL1gator-4 device.

Notes on Normal Mode Register Bits:

1. Writing values into unused register bits has no effect. However, to ensure software compatibility with future, feature-enhanced versions of the product, unused register bits must be written with logic zero unless stated otherwise. Reading back unused bits can produce either a logic one or a logic zero; hence, unused register bits should be masked off by software when read.

2. All configuration bits that can be written into can also be read back. This allows the processor controlling the AAL1gator to determine the programming state of the block.

3. Writable normal mode register bits are cleared to logic zero upon reset unless otherwise noted.

4. Writing into read-only normal mode register bit locations does not affect AAL1gator operation unless otherwise noted.

5. Certain register bits are reserved. To ensure that the AAL1gator operates as intended, reserved register bits must be written with their default value as indicated by the register bit description.

The register memory map is arranged as follows:

Table 15 Register Memory Map

Address	Register Description
0x8000X	Command Registers
0x8010X	Ram Interface Registers
0x8012X	UTOPIA Interface Registers
0x80200-0x80FFF	Line Interface Registers
0x81000-0x812FF	Interrupt and Status Registers
0x82000-0x82FFF	CAS Change Configuration Registers
0x84000-0x84FFF	DLL Control and Status Registers

11.1 Command Registers

These registers provide the means to update line and chip configuration from memory, reset unused lines, and send OAM cells. The register to add queues is also located here. There is one register per A1SP block.

Table 16 Command Register Memory Map

Address	Register Description	Register Mnemonic
0x80000	Reset and Device ID	DEV_ID_REG
0x80010	Command Register for A1SP	A_CMD_REG
0x80020	Add Queue FIFO Register for A1SP	A_ADDQ_FIFO

Address	Register Description	Register Mnemonic
0x80030	Clock Configuration for A1SP	A_CLK_CFG

Register 0x80000   Reset and Device ID Register (DEV_ID_REG)

Bit	Type	Function	Default
15	R/W	SW_RESET	1
14:7	Rsvd	Unused	X
6:4	R	DEV_TYPE[2:0]	010
3:0	R	DEV_ID[3:0]	0011

DEV_ID[3:0]

The ID bits can be read to provide a binary number indicating the AAL1gator-4/8 feature version. These bits are incremented only if features are added in a revision of the chip.

DEV_TYPE[2:0]

The TYPE bits can be read to distinguish the AAL1gator-4/8 from the other members of the AAL1gator family of devices.

011 = AAL1gator-32

010 = AAL1gator-8

010 = AAL1gator-4

SW_RESET

When set, causes all of the device to be held in reset including all other registers. While set, the external SSRAM may not be accessed. This is a chip software reset.

0)   Chip is active

1)   Chip is in reset

Notes:

1) Software should ensure that the RUN bit in DLL_STAT_REG reads back a "1" before releasing the AAL1gator-4/8 from software reset.
2) Software should wait two clock periods of the slowest clock after clearing SW_RESET, before attempting to write to any other register. Exception to this rule is the DLL register port.

Register 0x80010   A1SP Command Register (A_CMD_REG)

Bit	Type	Function	Default
Bit 15	R	Unused	X
Bit 14	R	Unused	X
Bit 13	R	Unused	X
Bit 12	R	Unused	X
Bit 11	R	Unused	X
Bit 10	R	Unused	X
Bit 9	R	Unused	X
Bit 8	R	Unused	X
Bit 7	R	Reserved	0
Bit 6	R	Reserved	0
Bit 5	R/W	A_SW_RESET	1
Bit 4	R	Reserved	0
Bit 3	R/W	A_CMDREG_ATTN	0
Bit 2	R/W	A_SEND_OAM_A1	0
Bit 1	R/W	A_SEND_OAM_A0	0
Bit 0	R	Reserved	0

A_SEND_OAM_0

A write of 1 causes the cell in the TX OAM buffer 0 for the A1SP to be sent. Reads as a 0 when the cell has been sent.

A_SEND_OAM_1

A write of 1 causes the cell in the TX OAM buffer 1 for the A1SP to be sent. Reads as a 0 when the cell has been sent.

A_CMDREG_ATTN

When written with a 1, causes the device to read the HS_LIN_REG and the LIN_STR_MODE memory registers for the A1SP. Reads as a 0 when the operation is complete.

A_SW_RESET

When set, causes the A1SP to be held in reset. This bit also functions as a Queue reset in High Speed mode.

Register 0x80020 A1SP Add Queue FIFO Register (A_ADDQ_FIFO)

Bit	Type	Function	Default
15	RO	EMPTY	1
14	Rsvd	Unused	X
13:8	WO	OFFSET	XX$_H$
7:0	WO	QUEUE_NUM	XX$_H$

This register is the write port of a 64 word FIFO that is used to add a queue on a first come first serve basis.

QUEUE_NUM

The number of the queue being added. Note that this field is invalid on reads.

OFFSET

This field indicates the offset from the cell scheduling reference value. This offset can be used to spread the scheduling of cells across multiple frames in order to control clumping (See Add Queue description in Processor Interface section . Note that this field is invalid on reads.)

Note: In SDF-MF mode, OFFSET must be set equal to FRAMES_PER_CELL in QUEUE_CONFIG in order to insure that the first pointer generated has a value of 0.

EMPTY

This field indicates when the Add Queue FIFO is empty. It can be polled to determine when the A1SP module has finished processing the ADDQ_FIFO entries that were in the FIFO.

Register 0x80030   A1SP Clock Configuration Register (A_CLK_CFG)

Bit	Type	Function	Default
15:10	R	Unused	X
9	R/W	NCLK_DIV_EN	0
8:5	R/W	NCLK_DIV	0000_B
4:0	R/W	ADAP_FILT_SIZE	00000_B

ADAP_FILT_SIZE

When a line is configured to use the internal adaptive clocking algorithm, this field defines the size of the filtering window. The filter size is a power of 2 where ADAP_FILT_SIZE represents the exponent (ie, $2^{ADAP\_FILT\_SIZE}$). The adaptive algorithm determines the clock frequency by averaging the byte difference over $2^{ADAP\_FILT\_SIZE}$ number of samples. The maximum value is "10000"$_B$ or 16.

ADAP_FILT_SIZE=00000 disables filtering.   ADAP_FILT_SIZE can be set on a per-A1SP basis, and has the following scope:

  1) ADAP_FILT_SIZE always controls the buffer_depth played out onto the external CGC port, independent of any other configuration.

  2) for lines with CLK_SOURCE_TX=100, ADAP_FILT_SIZE controls the buffer_depth filtering applied to the internal T1/E1 clock synthesizer.

NCLK_DIV

When NCLK_DIV_EN is set, this field indicates how much to divide down the NCLK for this A1SP. The clock is divided by ((NCLK_DIV + 1) *2). For instance if this field is 000 and NCLK_DIV_EN = '1' then the NCLK is divided by 2: If this field is "1111" then NCLK is divided by 32.

NCLK_DIV_EN

When set the NCLK is divided down as specified by the NCLK_DIV field. Otherwise the NCLK is passed directly from the NCLK pin to the A1SP.

11.2 RAM Interface Registers

This register controls the configuration of the RAM interface.

Table 17   RAM Interface Registers Memory Map

Address	Register Description	Register Mnemonic
0x80100	RAM Configuration Register	RAM_CFG_REG

Register 0x80100   RAM Configuration Register (RAM_CFG_REG)

Bit	Type	Function	Default
15:2	R/W	Unused	X
1	R/W	RAM_EVEN_PAR	0
0	R/W	SSRAM_ZBT_MODE	0

This register controls the configuration of the external RAM.

SSRAM_ZBT_MODE

When set to 0, the pipelined single-cycle deselect SSRAM protocol is used on the RAM interface allowing glueless connection to pipelined single-cycle deselect SSRAMs. When set to a 1, the pipelined ZBT SSRAM protocol is used allowing glueless connection to pipelined ZBT SSRAMS.

RAM_EVEN_PAR

The RAM_EVEN_PAR bit selects even or odd parity for the RAM I/F. When set to a 1, even parity is generated and checked. When set to a 0, odd parity is used.

11.3 UTOPIA Interface Registers

These registers control the configuration of the UTOPIA interface.

Table 18   UTOPIA Interface Registers Memory Map

Address	Register Description	Register Mnemonic
0x80120	UTOPIA Common Configuration Register	UI_COMN_CFG
0x80121	UTOPIA Source Configuration Register	UI_SRC_CFG
0x80122	UTOPIA Sink Configuration Register	UI_SNK_CFG
0x80123	UTOPIA Source Address Config Register	UI_SRC_ADD_CFG
0x80124	UTOPIA Sink Address Config Register	UI_SNK_ADD_CFG
0x80125	UTOPIA to UTOPIA Loopback VCI Register	UI_U2U_LOOP_VCI

Register 0x80120   UI Common Configuration Register (UI_COMN_CFG)

Bit	Type	Function	Default
15:5	R	Unused	X
4	R/W	VP_MODE_EN	0
3	R/W	SHIFT_VCI	0
2	R/W	VCI_U2U _LOOP	0
1	R/W	U2U_LOOP	0
0	R/W	UI_EN	0

This register controls the general configuration of the Utopia Interface

UI_EN

When set, enables the UTOPIA Interface in both directions. When this bit is cleared
(disabled) all the UI logic is held in reset and all the UTOPIA FIFO's are cleared and all the
UTOPIA outputs are tristated. The AAL1gator-4/8 will not respond in the UTOPIA
interface when this bit is disabled. The registers are not affected by this bit.

0) UI is disabled

1) UI is enabled

Note: UI_EN must only be set AFTER all other UI interface registers are configured.

U2U_LOOP

When set, all cells received by UI are sent back out to the UI (regardless of single or multi-
addressing mode).

0) UI in normal mode

1) UI in remote loopback mode

VCI_U2U_LOOP

When set, all cells received by UI with a VCI which matches the VCI programmed in the
U2U_LOOP_VCI register are sent back out by Utopia. To avoid any momentary corruption
of data, this bit should only be changed when UI_EN is disabled.

0) UI in normal mode

1) UI in VCI based remote loopback mode

SHIFT_VCI

Selects the VCI Address range used for mapping to queue numbers. This bit only controls the reception of cells. This field is not used if VP_MODE_EN is set.

0) Will use VCI(7:0) as the queue number if VCI(8) = 1.

1) Will use VCI(11:4) as the queue number if VCI(12) = 1.

VP_MODE_EN

When set uses the VPI field for A1SP and Line selection and uses VCI field for OAM cell detection. This bit only controls the reception of cells. This bit can only be set if all lines are in UDF mode. In particular:

1) VPI[2:0] selects line.

2) If VCI <= 31 then interpret cell as OAM cell and place in OAM buffer.

3) Queue 0 will be assumed and no VCI bits need to be used to indicate queue number

Note that VP_MODE_EN should only be set in Utopia L1 or L2 modes.

Register 0x80121   UI Source Config Reg (UI_SRC_CFG)

Bit	Type	Function	Default
15:6	R	Unused	X
5	R/W	CS_MODE_EN	0
4	R/W	16_BIT_MODE	0
3	R/W	EVEN_PAR	0
2	R/W	ANY-PHY_EN	0
1:0	R/W	UTOP_MODE	00

This register controls the source side configuration of the Utopia Interface

UTOP_MODE(1:0)

Selects the UTOPIA operating mode for the source side interface:

00   Utopia-1 Master

01   Utopia-1 Slave

10   Utopia-2 Single Address Slave

11   Reserved

ANY-PHY_EN

Enables Any-PHY mode for the source side interface:

0) UTOPIA mode. (Use UTOP_MODE for UTOPIA type)

1) Any-PHY mode.

EVEN_PAR

Determines the calculated parity across data bytes/words sent out of the source interface.

0) Odd parity

1) Even parity.

16_BIT_MODE

When set, UI source side interface operates in 16-bit mode.

0) 8-bit mode

1) 16-bit mode

CS_MODE_EN

When set, RPHY_ADDR(3)/RCSB input pin is used as a chip select (RCSB) for the source side interface, when clear RPHY_ADDR(3)/RCSB is used as an address bit (RPHY_ADDR(3)). This bit should only be set in Any-PHY mode.

0) RPHY_ADDR(3)/RCSB input is used as RPHY_ADDR(3).

1) RPHY_ADDR(3)/RCSB input is used as RCSB

PMC-SIERRA

Register 0x80122   UI Sink Config Reg (UI_SNK_CFG)

Bit	Type	Function	Default
15:6	R	Unused	X
5	R/W	CS_MODE_EN	0
4	R/W	16_BIT_MODE	0
3	R/W	EVEN_PAR	0
2	R/W	ANY-PHY_EN	0
1:0	R/W	UTOP_MODE	00

This register controls the sink side configuration of the Utopia Interface

UTOP_MODE(1:0)

Selects the operating mode for the sink side interface:

00   Utopia-1 Master

01   Utopia-1 Slave

10   Utopia-2 Single Address Slave

11   Reserved

ANY-PHY_EN

Enables Any-PHY mode for sink side interface.

0) UTOPIA mode. (Use UTOP_MODE for UTOPIA type)

1) Any-PHY mode.

EVEN_PAR

Determines the checked parity across data bytes/words received by the sink interface.

0) Odd parity

1) Even parity.

16_BIT_MODE

When set, UI sink side interface operates in 16-bit mode.

0) 8-bit mode

1) 16-bit mode

CS_MODE_EN

When set, TPHY_ADDR(3)/TCSB input pin is used as a chip select (TCSB) for the sink side interface, when clear TPHY_ADDR(3)/TCSB is used as a regular address bit (TPHY_ADDR(3)). This bit should only be set in Any-PHY mode.

0) TPHY_ADDR(3)/TCSB input is used as TPHY_ADDR(3)

1) TPHY_ADDR(3)/TCSB input is used as TCSB

Register 0x80123   Slave Source Address Config Register (UI_SRC_ADD_CFG)

Bit	Type	Function	Default
15:0	R/W	CFG_ADDR	0000_H

CFG_ADDR(15:0)

These bits contain the configured slave address used for Utopia-2 and Any-PHY operation in the source direction. Depending on the mode of the UTOPIA/Any-PHY interface different bits of this field are used. See Table 20 for details. In the source direction the AAL1gator is always one address. In UTOPIA Level One (UL1) mode these field should be left at zero assuming RPHY_ADDR[4:0] is tied low. In UL1 mode, if RPHY_ADDR[4:0] is set to any other level, it must match the same value in this register.

Table 19   CFG_ADDR and PHY_ADDR Bit Usage in SRC direction

	Polling		Selection	
MODE	**PHY_ADDR Pins**	**CFG_ADDR**	**PHY_ADDR Pins**	**CFG_ADDR**
UTOPIA-2 Single-Addr	[4:0]=device	[4:0]=device	[4:0]=device	[4:0]=device
Any-PHY with CSB	[2:0]=device	[2:0]=device	[2:0]=device CFG_ADDR is prepended	[2:0]=device
Any-PHY without CSB	[3:0]=device	[3:0]=device	[3:0]=device CFG_ADDR is prepended	[3:0]=device

Notes:

1. In UTOPIA-2 Multi-Addr mode is only an option in the SNK direction.

2. In Any-PHY mode, in the SRC direction the AAL1gator will prepend the cell with CFG_ADDR[15:0]. In 8-bit mode the cell will be prepended with CFG_ADDR[7:0]

3. In Any-PHY mode, if CS_MODE_EN='1' then you must program CFG_ADDR[4:3] = "00".

4. In Any-PHY mode, if CS_MODE_EN='0' then you must program CFG_ADDR[4]="0".

Register 0x80124   Slave Sink Address Config Register (UI_SNK_ADD_CFG)

Bit	Type	Function	Default
15:0	R/W	CFG_ADDR	0000$_H$

CFG_ADDR(15:0)

These bits contain the configured slave address used for Utopia-2 and Any-PHY operation in the sink direction. Depending on the mode of the UTOPIA/Any-PHY interface different bits of this field are used. See Table 20 for details. In UTOPIA Level One (UL1) mode these field should be left at zero assuming TPHY_ADDR[4:0] is tied low. In UL1 mode, if TPHY_ADDR[4:0] is set to any other level, it must match the same value in this register.

Table 20   CFG_ADDR and PHY_ADDR Bit Usage in SNK direction

	Polling		Selection	
MODE	**PHY_ADDR Pins**	**CFG_ADDR**	**PHY_ADDR Pins**	**CFG_ADDR**
UTOPIA-2 Single-Addr	[4:0]=device	[4:0]=device	[4:0]=device	[4:0]=device
UTOPIA-2 Multi-Addr	[4:2]=device [1:0]="00"	[4:2]=device	[4:2]=device [1:0]= "00"	[4:2]=device
Any-PHY with CSB	[2]=device [1:0]= "00"	[2]=device	[15:2]=device [1:0]= "00" PHY_ADDR is not used. Addr is prepended.	[15:2]=device
Any-PHY without CSB	[3:2]=device [1:0]= "00"	[3:2]=device	[15:2]=device [1:0]= "00" PHY_ADDR is not used. Addr is prepended.	[15:2]=device

Notes:

1. In Any-PHY mode, if CS_MODE_EN='1' then you must program CFG_ADDR[4:3] = "00".else if CS_MODE_EN='0' then you must program CFG_ADDR[4]="0".

2. In Any-PHY mode the upper 14 bits of the prepended address are compared with CFG_ADDR[15:2]. The bottom two bits must be "00". This overrides the value of VCI[10:9] which is not used in Any-PHY mode. If in 8-bit mode CFG_ADDR[7:2] is used instead.

Register 0x80125   UI to UI Loopback VCI (U2U_LOOP_VCI)

Bit	Type	Function	Default
15:0	R/W	U2U_LOOP_VCI	0000$_H$

U2U_LOOP_VCI(15:0)

> If VCI_U2U_LOOP is set in the UI_COMN_CFG_REG, any cell received from the UI bus, with a VCI which matches this programmed VCI, will be sent back out to the UI bus. To avoid any momentary corruption of incoming cell data, the value of this register should only be changed only when VCI_U2U_LOOP in UTO_CFG_REG is disabled and UI_EN in UTO_CFG_REG s disabled.

11.4 Line Interface Registers

These registers control the configuration and report status for the Line Interface. Gaps within the address space are reserved and should not be read or written. The Line interface registers are broken into the following sections:

Table 21   Line Interface Register Memory Map Summary

Address	Register Description
0x802XX	General Line Configuration Registers

11.5 Direct Mode Registers

These registers are selected when the address = 0x802XX.

Table 22   Direct Mode Register Memory Map

Offset	Register Description	Register Mnemonic
0x00 – 0x07	Low Speed Line Configuration Registers	LS_Ln_CFG_REG
0x10	Line Mode Register	LINE_MODE_REG

Register 0x80200H, 01H …03H: Low Speed Line n Configuration Registers(LS_Ln_CFG_REG)

Bit	Type	Function	Default
Bit 15	Rsvd	Unused	0
Bit 14	Rsvd	Unused	0
Bit 13	Rsvd	Unused	0
Bit 12	Rsvd	Unused	0
Bit 11	Rsvd	Unused	0
Bit 10	Rsvd	Unused	0
Bit 9	Rsvd	Unused	0
Bit 8	Rsvd	Unused	0
Bit 7	Rsvd	Unused	0
Bit 6	Rsvd	Unused	0
Bit 5	Rsvd	Unused	0
Bit 4	Rsvd	Unused	0
Bit 3	Rsvd	Unused	0
Bit 2	Rsvd	Unused	0
Bit 1	R/W	MVIP_EN	0
Bit 0	R/W	MF_SYNC_MODE	0

This register configures how independent lines are handled by the Line Interface Block in Direct Low Speed Mode..

MF_SYNC_MODE

Controls if the line sync signal (RL_SYNC and TL_SYNC) function as frame sync signals or multi-frame sync signals.

1) Sync signals are multi-frame sync signals

0) Sync signals are frame sync signals

MVIP_EN

When set selects the MVIP-90 format for external line data instead of the normal format. On read:

0) This line is in MVIP-90 mode

0) This line is in normal mode

Register 0x80210H   Line Mode Register(LINE_MODE_REG)

Bit	Type	Function	Default
Bit 15	Rsvd	Unused	0
Bit 14	Rsvd	Unused	0
Bit 13	Rsvd	Unused	0
Bit 12	Rsvd	Unused	0
Bit 11	Rsvd	Unused	0
Bit 10	Rsvd	Unused	0
Bit 9	Rsvd	Unused	0
Bit 8	Rsvd	Unused	0
Bit 7	Rsvd	Unused	0
Bit 6	Rsvd	Unused	0
Bit 5	Rsvd	Unused	0
Bit 4	Rsvd	Unused	0
Bit 3	Rsvd	Unused	0
Bit 2	Rsvd	Unused	0
Bit 1:0	RO	LINE_MODE	Depends on value of LINE_MODE pins

This register indicates the line mode of the device…

LINE_MODE

This field shows the values of the LINE_MODE pins. It is read only.

00)   Direct Low Speed Mode

01)   Reserved

10)   H-MVIP Mode

11)   Reserved

11.6 Interrupt and Status Registers

These registers indicate the current status of the device and any conditions that might require processor attention.

Table 23   Interrupt and Status Registers Memory Map

Address	Register Description	Register Mnemonic
0x81000	Master Interrupt Register	MSTR_INTR_REG
0x81010	A1SP Interrupt Register	A1SP_INTR_REG

Address	Register Description	Register Mnemonic
0x81020	A1SP Status Register	A1SP_STAT_REG
0x81030	A1SP CAS Change FIFO	A1SP_CASCHG_FIFO
0x81040	A1SP Receive Status FIFO	A1SP_RSTAT_FIFO
0x81100	Master Interrupt Enable Register	MSTR_INTR_REG
0x81110	A1SP Interrupt Enable Register	A1SP_INTR_EN
0x81140	A1SP Receive Status FIFO Enable Register	A1SP_RSTAT_EN
0x81150	A1SP Receive Queue Error Enable	A1SP_RCV_Q_ERR_ EN

Register 0x81000   Master Interrupt Register (MSTR_INTR_REG)

Bit	Type	Function	Default
15	RO	Unused	X
14	RO	Unused	X
13	RO	Reserved	0
12	RO	Reserved	0
11	RO	Reserved	0
10	R2C	R_UTOP_RUNT_CL	0
9	R2C	UTOP_LFIFO_FULL	0
8	R2C	T_UTOP_XFR_ERR	0
7	R2C	T_UTOP_FULL	0
6	R2C	UTOP_PAR_ERR	0
5	R2C	Reserved	0
4	R2C	RAM_PAR_ERR	0
3	RO	Reserved	0
2	RO	Reserved	0
1	RO	Reserved	0
0	RO	A1SP_INTR	0

This register is the top of the Interrupt Tree. It indicates which lower level interrupt registers have interrupts pending. The UTOPIA Interface error bits and RAM parity error bits are cleared on read, the other bits are current status and will remain set as long as the underlying condition remains active.

A1SP_INTR

> When set, there is an interrupt pending from the A1SP block. Read the A1SP_INTR_REG to determine the cause of the interrupt. This bit indicates current status and will clear only when no interrupt conditions remain in A1SP_INTR_REG. On read:

> 0) No interrupt pending from the A1SP block
> 1) Interrupt pending from the A1SP block

RAM_PAR_ERR

> When set, indicates there was a parity error encountered in the external RAM interface. This bit is cleared on read. On read:

> 0) No parity error encountered in RAM interface
> 1) Parity error encountered in RAM interface

UTOP_PAR_ERR

When set, indicates there was a parity error encountered in the UTOPIA interface. This bit is cleared on read. On read:

0) No parity error encountered in UTOPIA interface
1) Parity error encountered in UTOPIA interface

T_UTOP_FULL

When set, indicates the Transmit UTOPIA FIFO was full. This bit is cleared on read. If the Transmit UTOPIA FIFO is still full, the bit will be set again. On read:

0) Transmit UTOPIA FIFO is not full
1) Transmit UTOPIA FIFO is full

T_UTOP_XFR_ERR

When set indicates that the Transmit UTOPIA Interface was requested to send a cell when it did not have one available. This bit is cleared on read.

0) No transfer error occurred
1) A UTOPIA transfer error occurred

UTOP_LFIFO_FULL

When set indicates that the UTOPIA Loopback FIFO went full. This bit is cleared on read. If the Loopback FIFIO is still full, the bit will be set again.

0) UTOPIA loopback FIFO is not full.
1) UTOPIA loopback FIFO went full.

R_UTOP_RUNT_CL

When set indicates that a short cell (less than 53 bytes) was received. This bit is cleared on read.

0) No runt cell was received
1) A runt cell was received

Register 0x81010   A1SP Interrupt Register (A1SP_INTR_REG)

Bit	Type	Function	Default
15	RO	Unused	X
14	RO	Unused	X
13	RO	Unused	X
12	RO	Unused	X
11	RO	Unused	X
10	RO	Unused	X
9	RO	Unused	X
8	RO	Unused	X
7	RO	Unused	X
6	R2C	TALP_FIFO_FULL	0
5	R2C	RSTAT_FIFO_FULL	0
4	R2C	RSTAT_FIFO_EMPB	0
3	R2C	CASCHG_FIFO_FULL	0
2	R2C	CASCHG_FIFO_EMPB	0
1	R2C	OAM_INTR	0
0	R2C	FR_ADV_FIFO_FULL	0

The bits in this register are set upon entry into the indicated condition and are cleared when this register is read. If any of these conditions still exist the corresponding bit will not be set again until the condition ends and then occurs again. Read A1SP_STAT_REG for current status. If any bit is set in this register and the corresponding enable bit is set in A1SP_INTR_EN_REG, the A1SP_INTR bit will be set in MSTR_INTR_REG.

FR_ADV_FIFO_FULL

    When set indicates the Frame Advance FIFO has entered the full state since the last time this register was read. On read:

    0) Frame Advance FIFO has not entered the full state
    1) Frame Advance FIFO has entered the full state

OAM_INTR

    When set, indicates the A1SP block has received a new OAM cell. On read:

    0) A1SP has not received a new OAM cell
    1) A1SP has received a new OAM cell

CASCHG_FIFO_EMPB

        When clear, indicates the CAS Change FIFO has remained empty. On read:

0) CAS Change FIFO has remained empty
1) CAS Change FIFO has entered the not empty state

CASCHG_FIFO_FULL

When set, indicates the CAS Change FIFO has entered the full state. On read:

0) CAS Change FIFO has not entered the full state
1) CAS Change FIFO has entered the full state

RSTAT_FIFO_EMPB

When clear, indicates the Receive Status FIFO has remained empty. On read:

0) Receive Status FIFO has remained empty
1) Receive Status FIFO has entered the not empty state

RSTAT_FIFO_FULL

When set, indicates the Receive Status FIFO has entered the full state. On read:

0) Receive Status FIFO has not entered the full state
1) Receive Status FIFO has entered the full state

TALP_FIFO_FULL

When set, indicates the TALP FIFO has entered the full state. On read:

0) TALP FIFO has not entered the full state
1) TALP FIFO has entered the full state

PMC-SIERRA

Register 0x81020   A1SP Status Register (A1SP_STAT_REG)

Bit	Type	Function	Default
15	RO	Unused	X
14	RO	Unused	X
13	RO	Unused	X
12	RO	Unused	X
11	RO	Unused	X
10	RO	Unused	X
9	RO	Unused	X
8	RO	Unused	X
7	RO	Unused	X
6	RO	TALP_FIFO_FULL	0
5	RO	RSTAT_FIFO_FULL	0
4	RO	RSTAT_FIFO_EMPB	0
3	RO	CASCHG_FIFO_FULL	0
2	RO	CASCHG_FIFO_EMPB	0
1	RO	OAM_INTR	0
0	RO	FR_ADV_FIFO_FULL	0

The bits in this register indicate current status and are not cleared on reads. There is one register for each A1SP block.

FR_ADV_FIFO_FULL

When set indicates the Frame Advance FIFO is full. On read:

0) Frame Advance FIFO is not full
1) Frame Advance FIFO is full

OAM_INTR

When set, indicates the A1SPn block has received a new OAM cell. On read:

0) A1SPn has not received a new OAM cell
1) A1SPn has received a new OAM cell

CASCHG_FIFO_EMPB

When clear, indicates the CAS Change FIFO is empty. On read:

0) CAS Change FIFO is empty
1) CAS Change FIFO is not empty

CASCHG_FIFO_FULL

When set, indicates the CAS Change FIFO is full. On read:

0) CAS Change FIFO is not full
1) CAS Change FIFO is full

RSTAT_FIFO_EMPB

When clear, indicates the Receive Status FIFO is empty. On read:

0) Receive Status FIFO is empty
1) Receive Status FIFO is not empty

RSTAT_FIFO_FULL

When set, indicates the Receive Status FIFO is full. On read:

0) Receive Status FIFO is not full
1) Receive Status FIFO is full

TALP_FIFO_FULL

When set, indicates the TALP FIFO is full. On read:

0) TALP FIFO is not full
1) TALP FIFO is full

Register 0x81030   A1SP CAS Change FIFO (A1SP_CASCHG_FIFO)

Bit	Type	Function	Default
15:0	RO	CAS_STATUS	X

This register is the read port of a 64 word FIFO that is used to indicate changes in CAS on a given channel on a first come first serve basis. If the FIFO overflows the CASCHG_FIFO_FULL bit will be set in the A1SP_INTR_REG. When this FIFO goes from an empty to a non-empty condition the CASCHG_ FIFO_EMPB bit in the A1SP_INTR_REG will be set. The presence of data in this FIFO will set the CASCHG_FIFO_EMPB bit in the A1SP_STAT_REG. Read A1SP_STAT_REG to determine when FIFO goes empty again. If CAS change detection is not enabled on a given channel then the channel will not write to this FIFO.

Any changes in the CAS value in either direction will cause an entry to be written to the FIFO if CAS Change detection is enabled for this line. Note that for a CAS change to be valid, it has to be stable for two consecutive samples. Any additional filtering must be done in the external framers.

Bit	Type	Function	Default
15:8	RO	CHAN_NUM	XX_{Hn}
7:4	RO	RX_CAS	X_H
3:0	RO	TX_CAS	X_H

TX_CAS

This field indicates the current value of the transmit CAS ABCD bits.

RX_CAS

This field indicates the current value of the receive CAS ABCD bits.

CHAN_NUM

This field indicates the channel that encountered a change in activity status.

Register 0x81040   A1SP Receive Status FIFO (A1SP_RSTAT_FIFO)

Bit	Type	Function	Default
15:8	RO	QUEUE_NUMBER[7:0]	X
7:6	RO	Unused	X
5	RO	R_LINE_RESYNC	X
4	RO	T_LINE_RESYNC	X
3	RO	Reserved	X
2	RO	EXIT_UNDERRUN	X
1	RO	ENTER_UNDERRUN	X
0	RO	RECEIVE_QUEUE_ERR	X

This register is the read port of a 64 word FIFO that is used to capture receive status events on a first come first serve basis. If the FIFO overflows the RSTAT_FIFO_FULL bit will be set in the A1SP_INTR_REG. The presence of data in this FIFO will set the RSTAT_FIFO_EMPB bit in the A1SP_INTR_REG. The RSTAT_EN_REG controls whether certain errors or status conditions cause an entry to be written into the RSTAT_FIFO.

RECEIVE_QUEUE_ERR

An error or status condition occurred on the receive queue identified in QUEUE_NUMBER. Read sticky bit register for this queue to determine actual event. The RCV_Q_ERR_EN register controls which Receive queue sticky bits will cause an entry into this FIFO.

ENTER_UNDERRUN

The queue identified by QUEUE_NUMBER just entered the underrun state.

EXIT_UNDERRUN

The queue identified by QUEUE_NUMBER just exited the underrun state.

T_LINE_RESYNC

The transmit line identified by QUEUE_NUMBER(7:5) entered a resync state.

R_LINE_RESYNC

The receive line identified by QUEUE_NUMBER(7:5) entered a resync state.

QUEUE_NUMBER[7:0]

Identifies the queue on which the reported event occurred.

Register 0x81100   Master Interrupt Enable Register (MSTR_INTR_EN_REG)

Bit	Type	Function	Default
15	RO	Unused	X
14	RO	Unused	X
13	R/W	Reserved	0
12	R/W	Reserved	0
11	R/W	Reserved	0
10	R/W	R_UTOP_RUNT_CL_EN	0
9	R/W	UTOP_LFIFO_FULL_EN	0
8	R/W	T_UTOP_XFR_ERR_EN	0
7	R/W	T_UTOP_FULL_EN	0
6	R/W	UTOP_PAR_ERR_EN	0
5	R/W	Reserved	0
4	R/W	RAM_PAR_ERR_EN	0
3	R/W	Reserved	0
2	R/W	Reserved	0
1	R/W	Reserved	0
0	R/W	A1SP0_INTR_EN	0

The above enable bits control the corresponding interrupt bits in the MSTR_INTR_REG. When an enable bit is set to a logic 1, the corresponding error event will cause INTB to go active.

Register 0x81110   A1SP Interrupt Enable Register (A1SP_EN_REG)

Bit	Type	Function	Default
15	RO	Unused	X
14	RO	Unused	X
13	RO	Unused	X
12	RO	Unused	X
11	RO	Unused	X
10	RO	Unused	X
9	RO	Unused	X
8	RO	Unused	X
7	RO	Unused	X
6	R/W	TALP_FIFO_FULL	0
5	R/W	RSTAT_FIFO_FULL	0
4	R/W	RSTAT_FIFO_EMPB	0
3	R/W	CASCHG_FIFO_FULL	0
2	R/W	CASCHG_FIFO_EMPB	0
1	R/W	OAM_INTR	0
0	R/W	FR_ADV_FIFO_FULL	0

The above enable bits control the corresponding interrupt bits in the A1SP_INTR_REG. When an enable bit is set to a logic 1, the corresponding error event will cause the A1SP_INTR bit to be set in the MSTR_INTR_REG.

Register 0x81140   Receive Status FIFO Enable Register (RSTAT_EN_REG)

Bit	Type	Function	Default
15:6	RO	Unused	X
5	R/W	R_LINE_RESYNC_EN	0
4	R/W	T_LINE_RESYNC_EN	0
3	R/W	Reserved	0
2	R/W	EXIT_UNDERRUN_EN	0
1	R/W	ENTER_UNDERRUN_EN	0
0	R/W	RECEIVE_QUEUE_ERR_EN	0

The above enable bits control the corresponding status bits in the RCV_STAT_FIFO. When an enable bit is set to a logic 1, the corresponding receive status event will cause an entry to be made into the RCV_STAT_FIFO. This mask applies to all receive queues.

Register 0x81150   Receive Queue Error Enable (RCV_Q_ERR_EN)

Bit	Type	Function	Default
15	R/W	Reserved	0
14	R/W	CELL_RECEIVED	0
13	R/W	Reserved	0
12	R/W	PTR_RULE_ERROR	0
11	R/W	ALLOC_TBL_BLANK	0
10	R/W	POINTER_SEARCH	0
9	R/W	FORCED_UNDERRUN	0
8	R/W	SN_CELL_DROP	0
7	R/W	POINTER_RECEIVED	0
6	R/W	PTR_PARITY_ERR	0
5	R/W	SRTS_RESUME	0
4	R/W	SRTS_UNDERRUN	0
3	R/W	RESUME	0
2	R/W	PTR_MISMATCH	0
1	R/W	OVERRUN	0
0	R/W	UNDERRUN	0

The above enable bits control what is done when R_ERROR_STKY bits are set in the R_QUEUE_TBL. It controls which types of error/status conditions cause the RECEIVE_QUEUE_ERR indication in the RCV_STAT_FIFO to be set. All queues are configured the same way. Only the first enabled condition which occurs for a given queue will cause an entry to be made in the RCV_STAT_FIFO until the sticky bit register is cleared. So usually you should only enable bits that will not occur normally.

11.7   CAS Change Detection Configuration Registers

These registers control how CAS Change detection is configured for the AAL1gator-4/8

Table 24   CAS Change Detection Configuration Registers Memory Map

Address	Register Description	Register Mnemonic
0x82000 – 0x8200F	Reserved	
0x82010 – 0x8201F	Reserved	
0x82100 – 0x821FF	Reserved	
0x82200 – 0x8220F	Reserved	
0x82210 – 0x82217	Reserved	
0x82220	A1SP CAS Change Detection Enable Register	CASCHG_EN_REG
0x82300 –	A1SP CAS Change Configuration Table	CASCHG_CFG_TBL

Address	Register Description	Register Mnemonic
0x823FF		

PMC-SIERRA

Register 0x82220   A1SP CAS Change Detection Enable Table

This table contains the CAS Change detection enables for the A1SP block which contains 8 lines. The register contains the enables for 8 lines.

ADDRESS	LINE Configuration
0x82220	CASCHG_EN [7:0]

Bit	Type	Function	Default
15	R/W	Reserved	0
14	R/W	CASCHG_EN_7	0
13	R/W	Reserved	0
12	R/W	CASCHG_EN_6	0
11	R/W	Reserved	0
10	R/W	CASCHG_EN_5	0
9	R/W	Reserved	0
8	R/W	CASCHG_EN_4	0
7	R/W	Reserved	0
6	R/W	CASCHG_EN_3	0
5	R/W	Reserved	0
4	R/W	CASCHG_EN_2	0
3	R/W	Reserved	0
2	R/W	CASCHG_EN_1	0
1	R/W	Reserved	0
0	R/W	CASCHG_EN_0	0

CASCHG_EN_n

This bit enables CAS change detection on each line

0: CAS ChangeIdle Detection Disabled
1: CAS Change Detection EnabledProcessor

Register 0x82300-0x823FF   A1SP CAS Change Configuration Table

Bit	Type	Function	Default
15:12	R/W	RX_MASK	X
11:8	R/W	TX_MASK	X
7:4	R	Reserved	X
3:0	R	Reserved	X

Reserved

   Reserved for internal use. Initialize to '0'.

TX_MASK

   These bits are used as a mask on the TX_CAS field. Only changes in unmasked CAS bits
   will cause an interrupt to the processor. If a bit is set, it will be masked.

RX_MASK

   These bits are used as a mask on the RX_CAS field. Only changes in unmasked CAS bits
   will cause an interrupt to the processor. If a bit is set, it will be masked.

11.8 DLL Control and Status Registers

These registers allow the DLL to be configured and indicate the current status of the DLL.

Table 25   DLL Control and Status Registers Memory Map

Address	Register Description	Register Mnemonic
0x84000	DLL Configuration Register	DLL_CFG_REG
0x84001	Reserved	
0x84002	DLL SW Reset Register	DLL_SW_RST_REG
0x84003	DLL Control Status Register	DLL_STAT_REG

Register 0x84000H   DLL Configuration Register (DLL_CFG_REG)

Bit	Type	Function	Default
15:6		Unused	X
Bit 5		Reserved	0
Bit 4	R/W	OVERRIDE	0
Bit 3		Unused	X
Bit 2		Reserved	0
Bit 1		Reserved	0
Bit 0		Reserved	0

The DLL Configuration Register controls the basic operation of the DLL.

OVERRIDE

The override control (OVERRIDE) disables the DLL operation. When OVERRIDE is set low, the DLL generates the clock by delaying the SYS_CLK until the rising edge of internal SYS_CLK occurs at the same time as the rising edge of external SYS_CLK. When OVERRIDE is set high, the clock output is a buffered version of the SYS_CLK input.

Note that when clearing the OVERRIDE bit, the DLL should be reset so that it acquires sync cleanly. The RUN bit is only cleared by a DLL reset so it will give a false indication if the DLL is not reset.

Register 0x84002H   DLL SW Reset Register (DLL_SW_RST_REG)

Bit	Type	Function	Default
15:8		Unused	X
7:0	R	TAP[7:0]	X

Writing to this register performs a software reset of the DLL. A software reset requires a maximum of 24*256 SYS_CLK cycles for the DLL to regain lock. During this time the DLLCLK phase is adjusting from its current position to delay tap 0 and back to a lock position. Check the RUN bit to see when LOCK has occurred.

TAP[7:0]

The tap status register bits (TAP[7:0]) specifies the delay line tap the DLL is using to generate the outgoing clock. When TAP[7:0] is logic zero, the DLL is using the delay line tap with minimum phase delay. When TAP[7:0] is equal to 255, the DLL is using the delay line tap with maximum phase delay.

Register 0x84003H   DLL Control Status Register (DLL_STAT_REG)

Bit	Type	Function	Default
Bit 7	R	SYS_CLKI	X
Bit 6	R	INT_SYS_CLKI	X
Bit 5	R	ERRORI	X
Bit 4	R	CHANGEI	X
Bit 3		Unused	X
Bit 2	R	ERROR	X
Bit 1	R	CHANGE	0
Bit 0	R	RUN	0

The DLL Control Status Register provides information of the DLL operation.

RUN

The DLL lock status register bit (RUN) indicates the DLL found a delay line tap in which
the phase difference between the rising edge of Internal SYS_CLK and the rising edge of
external SYS_CLK is zero. After system reset, RUN is logic zero until the phase detector
indicates an initial lock condition. When the phase detector indicates lock, RUN is set to
logic 1. The RUN register bit is cleared only by a hardware reset or a DLL software reset
(writing to DLL_SW_RST_REG). This bit should be polled when taking the chip out of
reset. No other operations should take place until RUN is set.

CHANGE

The delay line tap change register bit (CHANGE) indicates the DLL has moved to a new
delay line tap. CHANGE is set high for eight SYS_CLK cycles when the DLL moves to a
new delay line tap.

ERROR

The delay line error register bit (ERROR) indicates the DLL has run out of dynamic range.
When the DLL attempts to move beyond the end of the delay line, ERROR is set high.
When ERROR is high, the DLL cannot generate a clock phase which causes the rising edge
of internal SYS_CLK to be aligned to the rising edge of external SYS_CLK. ERROR is set
low, when the DLL captures lock again.

CHANGEI

The delay line tap change event register bit (CHANGEI) indicates the CHANGE register bit
has changed value. When the CHANGE register changes from a logic zero to a logic one,
the CHANGEI register bit is set to logic one. The CHANGEI register bit is cleared
immediately after it is read, thus acknowledging the event has been recorded.

ERRORI

The delay line error event register bit (ERRORI) indicates the ERROR register bit has gone high. When the ERROR register changes from a logic zero to a logic one, the ERRORI register bit is set to logic one. The ERRORI register bit is cleared immediately after it is read, thus acknowledging the event has been recorded.

INT_SYS_CLKI

The reference clock event register bit INT_SYS_CLKI provides a method to monitor activity on the reference clock. When the internal SYS_CLK primary input changes from a logic zero to a logic one, the INT_SYS_CLKI register bit is set to logic one. The INT_SYS_CLKI register bit is cleared immediately after it is read, thus acknowledging the event has been recorded.

SYS_CLKI

The system clock event register bit SYSLCKI provides a method to monitor activity on the system clock. When the SYS_CLK primary input changes from a logic zero to a logic one, the SYS_CLKI register bit is set to logic one. The SYS_CLKI register bit is cleared immediately after it is read, thus acknowledging the event has been recorded.

PMC-SIERRA

12   JTAG Test Port

The AAL1gator-4/8 JTAG Test Access Port (TAP) allows access to the TAP controller and the 4 TAP registers: instruction, bypass, device identification and boundary scan. Using the TAP, device input logic levels can be read, device outputs can be forced, the device can be identified and the device scan path can be bypassed. For more details on the JTAG port, please refer to the Operations section.

Table 26   Instruction Register

Length– - 3 bits

Instructions	Selected Register	Instruction Codes, IR[2:0]
EXTEST	Boundary Scan	000
IDCODE	Identification	001
SAMPLE	Boundary Scan	010
BYPASS	Bypass	011
BYPASS	Bypass	100
STCTEST	Boundary Scan	101
BYPASS	Bypass	110
BYPASS	Bypass	111

Table 27   Identification Register

Length	32 bits
Version number	2_H
Part Number	$7313(A8)/7314_H$
Manufacturer's identification code	$0CD_H$
Device identification	173130CD(A8)/173140CD

For BSDL refer to PMC-2000647 AAL1gator-8 Boundary Scan Description Language (BSDL) Source Code. Also refer to AAL1gator-4 BSDL Source Code.

The Boundary Scan String for AAL1gator-8 is as follows:

Table 28   Boundary Scan String for AAL1gator-8

Length = 387 bits

SYSCLK is at end of scan chain (closest to TDO)

Pin/Enable	Cell Type	Register Bit
SYSCLK	IN_CELL	0
RAM_ADDR_0_OEB	ENABLE	1

Pin/Enable	Cell Type	Register Bit
RAM_ADDR[0]	IO_CELL	2
RAM_ADDR_1_OEB	ENABLE	3
RAM_ADDR[1]	IO_CELL	4
RAM_ADDR_2_OEB	ENABLE	5
RAM_ADDR[2]	IO_CELL	6
RAM_ADDR_3_OEB	ENABLE	7
RAM_ADDR[3]	IO_CELL	8
RAM_ADDR_4_OEB	ENABLE	9
RAM_ADDR[4]	IO_CELL	10
RAM_ADDR_5_OEB	ENABLE	11
RAM_ADDR[5]	IO_CELL	12
RAM_ADSCBOEB	ENABLE	13
RAM_ADSCB	OUT_CELL	14
RAM_ADDR_6_OEB	ENABLE	15
RAM_ADDR[6]	IO_CELL	16
RAM_ADDR_7_OEB	ENABLE	17
RAM_ADDR[7]	IO_CELL	18
RAM_ADDR_9_OEB	ENABLE	19
RAM_ADDR[9]	IO_CELL	20
RAM_CSBOEB	ENABLE	21
RAM_CSB	OUT_CELL	22
RAM_ADDR_10_OEB	ENABLE	23
RAM_ADDR[10]	IO_CELL	24
RAM_ADDR_11_OEB	ENABLE	25
RAM_ADDR[11]	IO_CELL	26
RAM_ADDR_12_OEB	ENABLE	27
RAM_ADDR[12]	IO_CELL	28
RAM_ADDR_13_OEB	ENABLE	29
RAM_ADDR[13]	IO_CELL	30
RAM_ADDR_14_OEB	ENABLE	31
RAM_ADDR[14]	OUT_CELL	32
RAM_ADDR_15_OEB	ENABLE	33
RAM_ADDR[15]	OUT_CELL	34
RAM_ADDR_16_OEB	ENABLE	35
RAM_ADDR[16]	OUT_CELL	36
RAM_WEB_0_OEB	ENABLE	37
RAM_WEB[0]	OUT_CELL	38
RAM_PAR_0_OEB	ENABLE	39
RAM_PAR[0]	IO_CELL	40
RAM_PAR_1_OEB	ENABLE	41

Pin/Enable	Cell Type	Register Bit
RAM_PAR[1]	IO_CELL	42
RAM_ADDR_8_OEB	ENABLE	43
RAM_ADDR[8]	IO_CELL	44
RAM_WEB_1_OEB	ENABLE	45
RAM_WEB[1]	OUT_CELL	46
RAM_D_0_OEB	ENABLE	47
RAM_D[0]	IO_CELL	48
RAM_D_1_OEB	ENABLE	49
RAM_D[1]	IO_CELL	50
RAM_D_2_OEB	ENABLE	51
RAM_D[2]	IO_CELL	52
RAM_D_3_OEB	ENABLE	53
RAM_D[3]	IO_CELL	54
RAM_D_4_OEB	ENABLE	55
RAM_D[4]	IO_CELL	56
RAM_D_5_OEB	ENABLE	57
RAM_D[5]	IO_CELL	58
RAM_D_6_OEB	ENABLE	59
RAM_D[6]	IO_CELL	60
RAM_D_7_OEB	ENABLE	61
RAM_D[7]	IO_CELL	62
RAM_D_8_OEB	ENABLE	63
RAM_D[8]	IO_CELL	64
RAM_D_10_OEB	ENABLE	65
RAM_D[10]	IO_CELL	66
RAM_D_11_OEB	ENABLE	67
RAM_D[11]	IO_CELL	68
RAM_OEBOEB	ENABLE	69
RAM_OEB	OUT_CELL	70
RAM_D_12_OEB	ENABLE	71
RAM_D[12]	IO_CELL	72
RAM_D_13_OEB	ENABLE	73
RAM_D[13]	IO_CELL	74
RAM_D_14_OEB	ENABLE	75
RAM_D[14]	IO_CELL	76
RAM_D_15_OEB	ENABLE	77
RAM_D[15]	IO_CELL	78
CTL_CLK	IN_CELL	79
CRL_CLK	IN_CELL	80
RAM_D_9_OEB	ENABLE	81

Pin/Enable	Cell Type	Register Bit
RAM_D[9]	IO_CELL	82
LINE_MODE	IN_CELL	83
TL_SYNC_7_OEB	ENABLE	84
TL_SYNC[7]	IO_CELL	85
TL_CLK_7_OEB	ENABLE	86
TL_CLK[7]	IO_CELL	87
TL_SIG_7_OEB	ENABLE	88
TL_SIG[7]	OUT_CELL	89
TL_DATA_7_OEB	ENABLE	90
TL_DATA[7]	OUT_CELL	91
RL_SIG[7]	IN_CELL	92
RL_CLK[7]	IN_CELL	93
RL_SYNC[7]	IN_CELL	94
RL_DATA[7]	IN_CELL	95
TL_SYNC_6_OEB	ENABLE	96
TL_SYNC[6]	IO_CELL	97
TL_CLK_6_OEB	ENABLE	98
TL_CLK[6]	IO_CELL	99
TL_SIG_6_OEB	ENABLE	100
TL_SIG[6]	IO_CELL	101
TL_DATA_6_OEB	ENABLE	102
TL_DATA[6]	IO_CELL	103
RL_SIG[6]	IN_CELL	104
RL_CLK[6]	IN_CELL	105
RL_SYNC[6]	IN_CELL	106
RL_DATA[6]	IN_CELL	107
TL_SYNC_5_OEB	ENABLE	108
TL_SYNC[5]	IO_CELL	109
TL_CLK_5_OEB	ENABLE	110
TL_CLK[5]	IO_CELL	111
TL_SIG_5_OEB	ENABLE	112
TL_SIG[5]	IO_CELL	113
TL_DATA_5_OEB	ENABLE	114
TL_DATA[5]	IO_CELL	115
RL_SIG[5]	IN_CELL	116
RL_CLK[5]	IN_CELL	117
RL_SYNC_5_OEB	ENABLE	118
RL_SYNC[5]	IO_CELL	119
RL_DATA_5_OEB	ENABLE	120
RL_DATA[5]	IO_CELL	121

Pin/Enable	Cell Type	Register Bit
TL_SYNC_4_OEB	ENABLE	122
TL_SYNC[4]	IO_CELL	123
TL_CLK_4_OEB	ENABLE	124
TL_CLK[4]	IO_CELL	125
TL_SIG_4_OEB	ENABLE	126
TL_SIG[4]	IO_CELL	127
TL_DATA_4_OEB	ENABLE	128
TL_DATA[4]	IO_CELL	129
RL_SIG_4_OEB	ENABLE	130
RL_SIG[4]	IO_CELL	131
RL_CLK[4]	IN_CELL	132
RL_SYNC_4_OEB	ENABLE	133
RL_SYNC[4]	IO_CELL	134
RL_DATA_4_OEB	ENABLE	135
RL_DATA[4]	IO_CELL	136
TL_SYNC_3_OEB	ENABLE	137
TL_SYNC[3]	IO_CELL	138
TL_CLK_3_OEB	ENABLE	139
TL_CLK[3]	IO_CELL	140
TL_SIG_3_OEB	ENABLE	141
TL_SIG[3]	IO_CELL	142
TL_DATA_3_OEB	ENABLE	143
TL_DATA[3]	IO_CELL	144
RL_CLK[3]	IN_CELL	145
RL_SIG_3_OEB	ENABLE	146
RL_SIG[3]	IO_CELL	147
RL_SYNC[3]	IN_CELL	148
RL_DATA_3_OEB	ENABLE	149
RL_DATA[3]	IO_CELL	150
TL_SYNC_2_OEB	ENABLE	151
TL_SYNC[2]	IO_CELL	152
TL_CLK_2_OEB	ENABLE	153
TL_CLK[2]	IO_CELL	154
TL_SIG_2_OEB	ENABLE	155
TL_SIG[2]	IO_CELL	156
TL_DATA_2_OEB	ENABLE	157
TL_DATA[2]	IO_CELL	158
RL_SIG_2_OEB	ENABLE	159
RL_SIG[2]	IO_CELL	160
RL_CLK[2]	IN_CELL	161

Pin/Enable	Cell Type	Register Bit
RL_SYNC[2]	IN_CELL	162
RL_DATA_2_OEB	ENABLE	163
RL_DATA[2]	IO_CELL	164
TL_SYNC_1_OEB	ENABLE	165
TL_SYNC[1]	IO_CELL	166
TL_CLK_1_OEB	ENABLE	167
TL_CLK[1]	IO_CELL	168
TL_SIG_1_OEB	ENABLE	169
TL_SIG[1]	IO_CELL	170
TL_DATA_1_OEB	ENABLE	171
TL_DATA[1]	IO_CELL	172
RL_SIG_1_OEB	ENABLE	173
RL_SIG[1]	IO_CELL	174
RL_CLK[1]	IN_CELL	175
RL_SYNC[1]	IN_CELL	176
RL_DATA_1_OEB	ENABLE	177
RL_DATA[1]	IO_CELL	178
TL_SYNC_0_OEB	ENABLE	179
TL_SYNC[0]	IO_CELL	180
TL_CLK_0_OEB	ENABLE	181
TL_CLK[0]	IO_CELL	182
RESERVED_IN	IN_CELL	183
TL_SIG_0_OEB	ENABLE	184
TL_SIG[0]	IO_CELL	185
TL_DATA_0_OEB	ENABLE	186
TL_DATA[0]	IO_CELL	187
RL_SIG_0_OEB	ENABLE	188
RL_SIG[0]	IO_CELL	189
RL_CLK[0]	IN_CELL	190
RL_SYNC_0_OEB	ENABLE	191
RL_SYNC[0]	IO_CELL	192
RL_DATA_0_OEB	ENABLE	193
RL_DATA[0]	IO_CELL	194
RSTB	IN_CELL	195
SRTS_STBHOEB	ENABLE	196
SRTS_STBH	OUT_CELL	197
ADAP_STBHOEB	ENABLE	198
ADAP_STBH	OUT_CELL	199
RESERVED_OEB	ENABLE	200
RESERVED_OUT	OUT_CELL	201

Pin/Enable	Cell Type	Register Bit
CGC_LINE_3_OEB	ENABLE	202
CGC_LINE[3]	OUT_CELL	203
CGC_LINE_2_OEB	ENABLE	204
CGC_LINE[2]	OUT_CELL	205
CGC_LINE_1_OEB	ENABLE	206
CGC_LINE[1]	OUT_CELL	207
CGC_LINE_0_OEB	ENABLE	208
CGC_LINE[0]	OUT_CELL	209
CGC_SER_DOEB	ENABLE	210
CGC_SER_D	IO_CELL	211
CGC_VALIDOEB	ENABLE	212
CGC_VALID	IO_CELL	213
TL_CLK_OEOEB	ENABLE	214
TL_CLK_OE	IO_CELL	215
NCLK	IN_CELL	216
CGC_DOUT_3_OEB	ENABLE	217
CGC_DOUT[3]	OUT_CELL	218
CGC_DOUT_1_OEB	ENABLE	219
CGC_DOUT[1]	OUT_CELL	220
CGC_DOUT_0_OEB	ENABLE	221
CGC_DOUT[0]	OUT_CELL	222
INTBOEB	ENABLE	223
INTB	OUT_CELL	224
ACKBOEB	ENABLE	225
ACKB	OUT_CELL	226
CGC_DOUT_2_OEB	ENABLE	227
CGC_DOUT[2]	OUT_CELL	228
CSB	IN_CELL	229
RDB	IN_CELL	230
WRB	IN_CELL	231
ALE	IN_CELL	232
A[0]	IN_CELL	233
A[1]	IN_CELL	234
A[2]	IN_CELL	235
A[3]	IN_CELL	236
D_0_OEB	ENABLE	237
D[0]	IO_CELL	238
D_1_OEB	ENABLE	239
D[1]	IO_CELL	240
D_3_OEB	ENABLE	241

Pin/Enable	Cell Type	Register Bit
D[3]	IO_CELL	242
A[4]	IN_CELL	243
A[5]	IN_CELL	244
A[6]	IN_CELL	245
A[7]	IN_CELL	246
D_4_OEB	ENABLE	247
D[4]	IO_CELL	248
D_5_OEB	ENABLE	249
D[5]	IO_CELL	250
D_6_OEB	ENABLE	251
D[6]	IO_CELL	252
D_7_OEB	ENABLE	253
D[7]	IO_CELL	254
A[8]	IN_CELL	255
A[9]	IN_CELL	256
A[10]	IN_CELL	257
D_2_OEB	ENABLE	258
D[2]	IO_CELL	259
A[11]	IN_CELL	260
D_8_OEB	ENABLE	261
D[8]	IO_CELL	262
D_9_OEB	ENABLE	263
D[9]	IO_CELL	264
D_10_OEB	ENABLE	265
D[10]	IO_CELL	266
D_11_OEB	ENABLE	267
D[11]	IO_CELL	268
A[12]	IN_CELL	269
A[13]	IN_CELL	270
A[14]	IN_CELL	271
A[15]	IN_CELL	272
D_12_OEB	ENABLE	273
D[12]	IO_CELL	274
D_13_OEB	ENABLE	275
D[13]	IO_CELL	276
D_14_OEB	ENABLE	277
D[14]	IO_CELL	278
D_15_OEB	ENABLE	279
D[15]	IO_CELL	280
A[16]	IN_CELL	281

Proprietary and Confidential to PMC-Sierra, Inc., and for its customers' internal use.
Document No.: PMC-2000098 Issue 1

225

Pin/Enable	Cell Type	Register Bit
A[17]	IN_CELL	282
A[18]	IN_CELL	283
A[19]	IN_CELL	284
RATM_DATA_15_OEB	ENABLE	285
RATM_DATA[15]	IO_CELL	286
RATM_DATA_14_OEB	ENABLE	287
RATM_DATA[14]	IO_CELL	288
RATM_DATA_13_OEB	ENABLE	289
RATM_DATA[13]	IO_CELL	290
RATM_DATA_12_OEB	ENABLE	291
RATM_DATA[12]	IO_CELL	292
TPHY_ADD_4_OEB	ENABLE	293
TPHY_ADD[4]	IO_CELL	294
RATM_DATA_11_OEB	ENABLE	295
RATM_DATA[11]	IO_CELL	296
RATM_DATA_10_OEB	ENABLE	297
RATM_DATA[10]	IO_CELL	298
RATM_DATA_9_OEB	ENABLE	299
RATM_DATA[9]	IO_CELL	300
RATM_DATA_8_OEB	ENABLE	301
RATM_DATA[8]	IO_CELL	302
TPHY_ADD_3_OEB	ENABLE	303
TPHY_ADD[3]	IO_CELL	304
RATM_PAROEB	ENABLE	305
RATM_PAR	IO_CELL	306
TPHY_ADD_2_OEB	ENABLE	307
TPHY_ADD[2]	IO_CELL	308
RATM_CLK	IN_CELL	309
TPHY_ADD_1_OEB	ENABLE	310
TPHY_ADD[1]	IO_CELL	311
TPHY_ADD_0_OEB	ENABLE	312
TPHY_ADD[0]	IO_CELL	313
RATM_CLAVOEB	ENABLE	314
RATM_CLAV	IO_CELL	315
RATM_SOCOEB	ENABLE	316
RATM_SOC	IO_CELL	317
RATM_DATA_7_OEB	ENABLE	318
RATM_DATA[7]	IO_CELL	319
RATM_DATA_6_OEB	ENABLE	320
RATM_DATA[6]	IO_CELL	321

Pin/Enable	Cell Type	Register Bit
RATM_DATA_5_OEB	ENABLE	322
RATM_DATA[5]	IO_CELL	323
RATM_DATA_4_OEB	ENABLE	324
RATM_DATA[4]	IO_CELL	325
RATM_ENBOEB	ENABLE	326
RATM_ENB	IO_CELL	327
RATM_DATA_3_OEB	ENABLE	328
RATM_DATA[3]	IO_CELL	329
RATM_DATA_2_OEB	ENABLE	330
RATM_DATA[2]	IO_CELL	331
RATM_DATA_1_OEB	ENABLE	332
RATM_DATA[1]	IO_CELL	333
RATM_DATA_0_OEB	ENABLE	334
RATM_DATA[0]	IO_CELL	335
TATM_DATA_3_OEB	ENABLE	336
TATM_DATA[3]	IO_CELL	337
TATM_DATA_0_OEB	ENABLE	338
TATM_DATA[0]	IO_CELL	339
TATM_DATA_1_OEB	ENABLE	340
TATM_DATA[1]	IO_CELL	341
TATM_DATA_2_OEB	ENABLE	342
TATM_DATA[2]	IO_CELL	343
TATM_CLAVOEB	ENABLE	344
TATM_CLAV	IO_CELL	345
TATM_DATA_4_OEB	ENABLE	346
TATM_DATA[4]	IO_CELL	347
TATM_DATA_5_OEB	ENABLE	348
TATM_DATA[5]	IO_CELL	349
TATM_DATA_6_OEB	ENABLE	350
TATM_DATA[6]	IO_CELL	351
TATM_DATA_7_OEB	ENABLE	352
TATM_DATA[7]	IO_CELL	353
TATM_SOCOEB	ENABLE	354
TATM_SOC	IO_CELL	355
TATM_ENBOEB	ENABLE	356
TATM_ENB	IO_CELL	357
RPHY_ADD_0_OEB	ENABLE	358
RPHY_ADD[0]	IO_CELL	359
TATM_CLK	IN_CELL	360
RPHY_ADD_1_OEB	ENABLE	361

PMC-SIERRA

Pin/Enable	Cell Type	Register Bit
RPHY_ADD[1]	IO_CELL	362
RPHY_ADD_3_OEB	ENABLE	363
RPHY_ADD[3]	IO_CELL	364
RPHY_ADD_2_OEB	ENABLE	365
RPHY_ADD[2]	IO_CELL	366
TATM_PAROEB	ENABLE	367
TATM_PAR	IO_CELL	368
TATM_DATA_8_OEB	ENABLE	369
TATM_DATA[8]	IO_CELL	370
TATM_DATA_9_OEB	ENABLE	371
TATM_DATA[9]	IO_CELL	372
TATM_DATA_10_OEB	ENABLE	373
TATM_DATA[10]	IO_CELL	374
TATM_DATA_11_OEB	ENABLE	375
TATM_DATA[11]	IO_CELL	376
RPHY_ADD_RSXOEB	ENABLE	377
RPHY_ADD_RSX	IO_CELL	378
TATM_DATA_12_OEB	ENABLE	379
TATM_DATA[12]	IO_CELL	380
TATM_DATA_13_OEB	ENABLE	381
TATM_DATA[13]	IO_CELL	382
TATM_DATA_14_OEB	ENABLE	383
TATM_DATA[14]	IO_CELL	384
TATM_DATA_15_OEB	ENABLE	385
TATM_DATA[15]	IO_CELL	386

The Boundary Scan String for AAL1gator-4 is as follows:

Table 29   Boundary Scan String for AAL1gator-4

Length = 334 bits

SYSCLK is at end of scan chain (closest to TDO)

Pin/Enable	Cell Type	Register Bit
SYSCLK	IN_CELL	0
RAM1_ADDR_0_OEB	ENABLE	1
RAM1_ADDR[0]	IO_CELL	2
RAM1_ADDR_1_OEB	ENABLE	3
RAM1_ADDR[1]	IO_CELL	4
RAM1_ADDR_2_OEB	ENABLE	5
RAM1_ADDR[2]	IO_CELL	6

Pin/Enable	Cell Type	Register Bit
RAM1_ADDR_3_OEB	ENABLE	7
RAM1_ADDR[3]	IO_CELL	8
RAM1_ADDR_4_OEB	ENABLE	9
RAM1_ADDR[4]	IO_CELL	10
RAM1_ADDR_5_OEB	ENABLE	11
RAM1_ADDR[5]	IO_CELL	12
RAM1_ADSCBOEB	ENABLE	13
RAM1_ADSCB	OUT_CELL	14
RAM1_ADDR_6_OEB	ENABLE	15
RAM1_ADDR[6]	IO_CELL	16
RAM1_ADDR_7_OEB	ENABLE	17
RAM1_ADDR[7]	IO_CELL	18
RAM1_ADDR_9_OEB	ENABLE	19
RAM1_ADDR[9]	IO_CELL	20
RAM1_CSBOEB	ENABLE	21
RAM1_CSB	OUT_CELL	22
RAM1_ADDR_10_OEB	ENABLE	23
RAM1_ADDR[10]	IO_CELL	24
RAM1_ADDR_11_OEB	ENABLE	25
RAM1_ADDR[11]	IO_CELL	26
RAM1_ADDR_12_OEB	ENABLE	27
RAM1_ADDR[12]	IO_CELL	28
RAM1_ADDR_13_OEB	ENABLE	29
RAM1_ADDR[13]	IO_CELL	30
RAM1_ADDR_14_OEB	ENABLE	31
RAM1_ADDR[14]	OUT_CELL	32
RAM1_ADDR_15_OEB	ENABLE	33
RAM1_ADDR[15]	OUT_CELL	34
RAM1_ADDR_16_OEB	ENABLE	35
RAM1_ADDR[16]	OUT_CELL	36
RAM1_WEB_0_OEB	ENABLE	37
RAM1_WEB[0]	OUT_CELL	38
RAM1_PAR_0_OEB	ENABLE	39
RAM1_PAR[0]	IO_CELL	40
RAM1_PAR_1_OEB	ENABLE	41
RAM1_PAR[1]	IO_CELL	42
RAM1_ADDR_8_OEB	ENABLE	43
RAM1_ADDR[8]	IO_CELL	44
RAM1_WEB_1_OEB	ENABLE	45
RAM1_WEB[1]	OUT_CELL	46

Pin/Enable	Cell Type	Register Bit
RAM1_D_0_OEB	ENABLE	47
RAM1_D[0]	IO_CELL	48
RAM1_D_1_OEB	ENABLE	49
RAM1_D[1]	IO_CELL	50
RAM1_D_2_OEB	ENABLE	51
RAM1_D[2]	IO_CELL	52
RAM1_D_3_OEB	ENABLE	53
RAM1_D[3]	IO_CELL	54
RAM1_D_4_OEB	ENABLE	55
RAM1_D[4]	IO_CELL	56
RAM1_D_5_OEB	ENABLE	57
RAM1_D[5]	IO_CELL	58
RAM1_D_6_OEB	ENABLE	59
RAM1_D[6]	IO_CELL	60
RAM1_D_7_OEB	ENABLE	61
RAM1_D[7]	IO_CELL	62
RAM1_D_8_OEB	ENABLE	63
RAM1_D[8]	IO_CELL	64
RAM1_D_10_OEB	ENABLE	65
RAM1_D[10]	IO_CELL	66
RAM1_D_11_OEB	ENABLE	67
RAM1_D[11]	IO_CELL	68
RAM1_OEBOEB	ENABLE	69
RAM1_OEB	OUT_CELL	70
RAM1_D_12_OEB	ENABLE	71
RAM1_D[12]	IO_CELL	72
RAM1_D_13_OEB	ENABLE	73
RAM1_D[13]	IO_CELL	74
RAM1_D_14_OEB	ENABLE	75
RAM1_D[14]	IO_CELL	76
RAM1_D_15_OEB	ENABLE	77
RAM1_D[15]	IO_CELL	78
CTL_CLK	IN_CELL	79
CRL_CLK	IN_CELL	80
RAM1_D_9_OEB	ENABLE	81
RAM1_D[9]	IO_CELL	82
LINE_MODE	IN_CELL	83
TL_SYNC_3_OEB	ENABLE	84
TL_SYNC[3]	IO_CELL	85
TL_CLK_3_OEB	ENABLE	86

Pin/Enable	Cell Type	Register Bit
TL_CLK[3]	IO_CELL	87
TL_SIG_3_OEB	ENABLE	88
TL_SIG[3]	IO_CELL	89
TL_DATA_3_OEB	ENABLE	90
TL_DATA[3]	IO_CELL	91
RL_CLK[3]	IN_CELL	92
RL_SIG_3_OEB	ENABLE	93
RL_SIG[3]	IO_CELL	94
RL_SYNC[3]	IN_CELL	95
RL_DATA_3_OEB	ENABLE	96
RL_DATA[3]	IO_CELL	97
TL_SYNC_2_OEB	ENABLE	98
TL_SYNC[2]	IO_CELL	99
TL_CLK_2_OEB	ENABLE	100
TL_CLK[2]	IO_CELL	101
TL_SIG_2_OEB	ENABLE	102
TL_SIG[2]	IO_CELL	103
TL_DATA_2_OEB	ENABLE	104
TL_DATA[2]	IO_CELL	105
RL_SIG_2_OEB	ENABLE	106
RL_SIG[2]	IO_CELL	107
RL_CLK[2]	IN_CELL	108
RL_SYNC[2]	IN_CELL	109
RL_DATA_2_OEB	ENABLE	110
RL_DATA[2]	IO_CELL	111
TL_SYNC_1_OEB	ENABLE	112
TL_SYNC[1]	IO_CELL	113
TL_CLK_1_OEB	ENABLE	114
TL_CLK[1]	IO_CELL	115
TL_SIG_1_OEB	ENABLE	116
TL_SIG[1]	IO_CELL	117
TL_DATA_1_OEB	ENABLE	118
TL_DATA[1]	IO_CELL	119
RL_SIG_1_OEB	ENABLE	120
RL_SIG[1]	IO_CELL	121
RL_CLK[1]	IN_CELL	122
RL_SYNC[1]	IN_CELL	123
RL_DATA_1_OEB	ENABLE	124
RL_DATA[1]	IO_CELL	125
TL_SYNC_0_OEB	ENABLE	126

Pin/Enable	Cell Type	Register Bit
TL_SYNC[0]	IO_CELL	127
TL_CLK_0_OEB	ENABLE	128
TL_CLK[0]	IO_CELL	129
RESERVED_IN	IN_CELL	130
TL_SIG_0_OEB	ENABLE	131
TL_SIG[0]	IO_CELL	132
TL_DATA_0_OEB	ENABLE	133
TL_DATA[0]	IO_CELL	134
RL_SIG_0_OEB	ENABLE	135
RL_SIG[0]	IO_CELL	136
RL_CLK[0]	IN_CELL	137
RL_SYNC_0_OEB	ENABLE	138
RL_SYNC[0]	IO_CELL	139
RL_DATA_0_OEB	ENABLE	140
RL_DATA[0]	IO_CELL	141
RSTB	IN_CELL	142
SRTS_STBHOEB	ENABLE	143
SRTS_STBH	OUT_CELL	144
ADAP_STBHOEB	ENABLE	145
ADAP_STBH	OUT_CELL	146
RESERVED_OEB	ENABLE	147
RESERVED_OUT	OUT_CELL	148
CGC_LINE_3_OEB	ENABLE	149
CGC_LINE[3]	OUT_CELL	150
CGC_LINE_2_OEB	ENABLE	151
CGC_LINE[2]	OUT_CELL	152
CGC_LINE_1_OEB	ENABLE	153
CGC_LINE[1]	OUT_CELL	154
CGC_LINE_0_OEB	ENABLE	155
CGC_LINE[0]	OUT_CELL	156
CGC_SER_DOEB	ENABLE	157
CGC_SER_D	IO_CELL	158
CGC_VALIDOEB	ENABLE	159
CGC_VALID	IO_CELL	160
TL_CLK_OEOEB	ENABLE	161
TL_CLK_OE	IO_CELL	162
NCLK	IN_CELL	163
CGC_DOUT_3_OEB	ENABLE	164
CGC_DOUT[3]	OUT_CELL	165
CGC_DOUT_1_OEB	ENABLE	166

Pin/Enable	Cell Type	Register Bit
CGC_DOUT[1]	OUT_CELL	167
CGC_DOUT_0_OEB	ENABLE	168
CGC_DOUT[0]	OUT_CELL	169
INTBOEB	ENABLE	170
INTB	OUT_CELL	171
ACKBOEB	ENABLE	172
ACKB	OUT_CELL	173
CGC_DOUT_2_OEB	ENABLE	174
CGC_DOUT[2]	OUT_CELL	175
CSB	IN_CELL	176
RDB	IN_CELL	177
WRB	IN_CELL	178
ALE	IN_CELL	179
A[0]	IN_CELL	180
A[1]	IN_CELL	181
A[2]	IN_CELL	182
A[3]	IN_CELL	183
D_0_OEB	ENABLE	184
D[0]	IO_CELL	185
D_1_OEB	ENABLE	186
D[1]	IO_CELL	187
D_3_OEB	ENABLE	188
D[3]	IO_CELL	189
A[4]	IN_CELL	190
A[5]	IN_CELL	191
A[6]	IN_CELL	192
A[7]	IN_CELL	193
D_4_OEB	ENABLE	194
D[4]	IO_CELL	195
D_5_OEB	ENABLE	196
D[5]	IO_CELL	197
D_6_OEB	ENABLE	198
D[6]	IO_CELL	199
D_7_OEB	ENABLE	200
D[7]	IO_CELL	201
A[8]	IN_CELL	202
A[9]	IN_CELL	203
A[10]	IN_CELL	204
D_2_OEB	ENABLE	205
D[2]	IO_CELL	206

Pin/Enable	Cell Type	Register Bit
A[11]	IN_CELL	207
D_8_OEB	ENABLE	208
D[8]	IO_CELL	209
D_9_OEB	ENABLE	210
D[9]	IO_CELL	211
D_10_OEB	ENABLE	212
D[10]	IO_CELL	213
D_11_OEB	ENABLE	214
D[11]	IO_CELL	215
A[12]	IN_CELL	216
A[13]	IN_CELL	217
A[14]	IN_CELL	218
A[15]	IN_CELL	219
D_12_OEB	ENABLE	220
D[12]	IO_CELL	221
D_13_OEB	ENABLE	222
D[13]	IO_CELL	223
D_14_OEB	ENABLE	224
D[14]	IO_CELL	225
D_15_OEB	ENABLE	226
D[15]	IO_CELL	227
A[16]	IN_CELL	228
A[17]	IN_CELL	229
A[18]	IN_CELL	230
A[19]	IN_CELL	231
RATM_DATA_15_OEB	ENABLE	232
RATM_DATA[15]	IO_CELL	233
RATM_DATA_14_OEB	ENABLE	234
RATM_DATA[14]	IO_CELL	235
RATM_DATA_13_OEB	ENABLE	236
RATM_DATA[13]	IO_CELL	237
RATM_DATA_12_OEB	ENABLE	238
RATM_DATA[12]	IO_CELL	239
TPHY_ADD_4_OEB	ENABLE	240
TPHY_ADD[4]	IO_CELL	241
RATM_DATA_11_OEB	ENABLE	242
RATM_DATA[11]	IO_CELL	243
RATM_DATA_10_OEB	ENABLE	244
RATM_DATA[10]	IO_CELL	245
RATM_DATA_9_OEB	ENABLE	246

Pin/Enable	Cell Type	Register Bit
RATM_DATA[9]	IO_CELL	247
RATM_DATA_8_OEB	ENABLE	248
RATM_DATA[8]	IO_CELL	249
TPHY_ADD_3_OEB	ENABLE	250
TPHY_ADD[3]	IO_CELL	251
RATM_PAROEB	ENABLE	252
RATM_PAR	IO_CELL	253
TPHY_ADD_2_OEB	ENABLE	254
TPHY_ADD[2]	IO_CELL	255
RATM_CLK	IN_CELL	256
TPHY_ADD_1_OEB	ENABLE	257
TPHY_ADD[1]	IO_CELL	258
TPHY_ADD_0_OEB	ENABLE	259
TPHY_ADD[0]	IO_CELL	260
RATM_CLAVOEB	ENABLE	261
RATM_CLAV	IO_CELL	262
RATM_SOCOEB	ENABLE	263
RATM_SOC	IO_CELL	264
RATM_DATA_7_OEB	ENABLE	265
RATM_DATA[7]	IO_CELL	266
RATM_DATA_6_OEB	ENABLE	267
RATM_DATA[6]	IO_CELL	268
RATM_DATA_5_OEB	ENABLE	269
RATM_DATA[5]	IO_CELL	270
RATM_DATA_4_OEB	ENABLE	271
RATM_DATA[4]	IO_CELL	272
RATM_ENBOEB	ENABLE	273
RATM_ENB	IO_CELL	274
RATM_DATA_3_OEB	ENABLE	275
RATM_DATA[3]	IO_CELL	276
RATM_DATA_2_OEB	ENABLE	277
RATM_DATA[2]	IO_CELL	278
RATM_DATA_1_OEB	ENABLE	279
RATM_DATA[1]	IO_CELL	280
RATM_DATA_0_OEB	ENABLE	281
RATM_DATA[0]	IO_CELL	282
TATM_DATA_3_OEB	ENABLE	283
TATM_DATA[3]	IO_CELL	284
TATM_DATA_0_OEB	ENABLE	285
TATM_DATA[0]	IO_CELL	286

Pin/Enable	Cell Type	Register Bit
TATM_DATA_1_OEB	ENABLE	287
TATM_DATA[1]	IO_CELL	288
TATM_DATA_2_OEB	ENABLE	289
TATM_DATA[2]	IO_CELL	290
TATM_CLAVOEB	ENABLE	291
TATM_CLAV	IO_CELL	292
TATM_DATA_4_OEB	ENABLE	293
TATM_DATA[4]	IO_CELL	294
TATM_DATA_5_OEB	ENABLE	295
TATM_DATA[5]	IO_CELL	296
TATM_DATA_6_OEB	ENABLE	297
TATM_DATA[6]	IO_CELL	298
TATM_DATA_7_OEB	ENABLE	299
TATM_DATA[7]	IO_CELL	300
TATM_SOCOEB	ENABLE	301
TATM_SOC	IO_CELL	302
TATM_ENBOEB	ENABLE	303
TATM_ENB	IO_CELL	304
RPHY_ADD_0_OEB	ENABLE	305
RPHY_ADD[0]	IO_CELL	306
TATM_CLK	IN_CELL	307
RPHY_ADD_1_OEB	ENABLE	308
RPHY_ADD[1]	IO_CELL	309
RPHY_ADD_3_OEB	ENABLE	310
RPHY_ADD[3]	IO_CELL	311
RPHY_ADD_2_OEB	ENABLE	312
RPHY_ADD[2]	IO_CELL	313
TATM_PAROEB	ENABLE	314
TATM_PAR	IO_CELL	315
TATM_DATA_8_OEB	ENABLE	316
TATM_DATA[8]	IO_CELL	317
TATM_DATA_9_OEB	ENABLE	318
TATM_DATA[9]	IO_CELL	319
TATM_DATA_10_OEB	ENABLE	320
TATM_DATA[10]	IO_CELL	321
TATM_DATA_11_OEB	ENABLE	322
TATM_DATA[11]	IO_CELL	323
RPHY_ADD_RSXOEB	ENABLE	324
RPHY_ADD_RSX	IO_CELL	325
TATM_DATA_12_OEB	ENABLE	326

Pin/Enable	Cell Type	Register Bit
TATM_DATA[12]	IO_CELL	327
TATM_DATA_13_OEB	ENABLE	328
TATM_DATA[13]	IO_CELL	329
TATM_DATA_14_OEB	ENABLE	330
TATM_DATA[14]	IO_CELL	331
TATM_DATA_15_OEB	ENABLE	332
TATM_DATA[15]	IO_CELL	333

13 Operation

This section discusses procedures for setting up or configuring different functions of the AAL1gator-4/8.

13.1 Hardware Configuration

The AAL1gator-4/8 can be configured in several different modes. The line mode of operation needs to be setup from hardware reset and cannot be changed once the chip is powered up. The line mode is controlled by the LINE_MODE pin. This pin will determine whether the line interface supports 4 low speed lines, or 1 H-MVIP bi-directional lines. See the description of the Line interface for more details.

The UTOPIA interface can also be set up in different modes. Because different modes require different sides of the bus driving the control signals, the UTOPIA will power up with all outputs tri-stated. If it is desired at the system level to pull some of the control signal so that they default to one way or the other upon power-up, this can be done using weak pull-up or pull-down resistors. The UTOPIA interface will remain tri-state until the UI_EN bit in the UI_COMN_CFG register is set.

The AAL1gator-4/8 can either generate the transmit line clocks internally or use clocks supplied to its transmit line clock inputs. Because the device does not know upon power up which mode will be used, there is a TLCLK_OE signal which can be used to tell the chip to generate clocks or not generate clocks. If this pin is tied low the chip will not generate a clock until it is configured to source a clock. If this pin is tied high the chip will use the clock provided on its RL_CLK pin as it TL_CLK and will drive this clock externally. Note the option to drive clocks is only available in Direct Low Speed mode.

13.2 Start-Up

The AAL1gator-4/8 uses an internal DLL on SYS_CLK to maintain low skew on the ram interface. When the chip is taken out of hardware reset, the DLL will go into hunt mode and will adjust the internal SYS_CLK until it aligns with the external SYS_CLK. The microprocessor should poll the RUN bit in DLL_STAT_REG until this bit is set.

At this point the entire chip with the exception of the microprocessor interface and the DLL are in reset. Before any configuration can be done, including accessing the ram, the chip must be taken out of software reset by clearing the SW_RESET bit in the DEV_ID_REG. Once taken out of reset the external RAM should be cleared to all zeros. Then all static chip tables in memory, such as the T_SEQNUM_TBL and R_CRC_SYNDROME table, should be initialized. At this point, the A1SP block is still in reset because the A_SW_RESET bit in the A_CMD_REG register is still set. The UTOPIA interface is disabled and all UTOPIA outputs are tri-stated because the UI_EN bit in the UI_COMN_CFG register is not set. The line interface is configured in the mode indicated by the LINE_MODE pin but all internal registers are in their reset state. The Line Interface is out of reset at this point but will only be driving data as if all lines and/or queues are disabled.

13.2.1 Line Configuration

There are line interface registers for Direct Mode. These registers should be set up before the A1SP is taken out of reset.

While the A1SP is in reset the memory mapped registers which contain the line configuration (LIN_STR_MODE and HS_LIN_REG) can be initialized. Note that the R_CHAN_2_QUE_TBL registers and R_STATE_0 and R_LINE_STATE registers cannot be accessed because they are internal and are being held in reset.

Once the line is initialized and the LIN_STR_MODE and HS_LIN_REG memory registers are initialized the CMD_ATTN bit in the A_CMD_REG bit can be set so that the A1SP can read its configuration. The A_SW_RESET bit should remain set.

NOTE: LOW_CDV is a bit in LIN_STR_MODE register that schedules unstructured cells every 47 bytes instead of on frame boundaries. This will reduce the amount of Cell Delay Variation of unstructured cells. Because this mode uses a different scheduling algorithm it is important that TX_ACTIVE in the queue table is not set when changing the state of LOW_CDV.. Therefore when using LOW_CDV the following guidelines must be followed:

- Always configure line before configuring queue. Queue should always be disabled (TX_ACTIVE=0) when reconfiguring the line.

- Set TX_ACTIVE as the last write of the transmit queue table when configuring the queue.

- These guidelines are recommended all the time, but especially need to be followed if LOW_CDV is used.

Note that line configuration is not complete until the AAL1gator32 reads the LIN_STR_MODE and HS_LIN_REG information from ram after CMD_ATTN bit is set. The CMD_ATTN bit should be polled until CMD_ATTN is clear before proceeding to queue configuration. See Line Configuration Details below for more details.

13.2.2 Queue Configuration

Once this is complete the A_SW_RESET bit in the A_CMD_REG can be cleared which will take the A1SP out of reset. The R_CHAN_2_QUE_TBL will then begin a 640 SYS_CLK cycle initialization, which reset each timeslot to playing out conditioned data. At this point the queues can be initialized as needed.

Set TX_ACTIVE as the last write of the transmit queue table and clear R_CHAN_DISABLE as the last write of the receive queue table.

Queues are disabled by clearing TX_ACTIVE in the transmit queue table and setting R_CHAN_DISABLE in the receive queue table. Queues must be disabled at least 6 ms prior to being reconfigured, to insure that the queue has actually been de-queued from the Transmit Calendar Table. If resetting the device, it is important to remember that a lot of control and context information is stored in external ram and needs to be manually re-initialized. For instance the Transmit Calendar Table should be cleared prior to restarting queues after a software or hardware reset.

If the line needs to be reconfigured, disable all queues on that line prior to changing the line configuration.

13.2.3 Adding Queues

Queues are added by writing to the ADDQ_FIFO with the number of the queue to be added. See Processor Interface section for more details

13.2.4 Line Configuration Details

This section is intended to be a guide for programmers.

Mode Selection

The mode of the Line Interface Block is controlled by the LINE_MODE input pin, which is also readable by software via a read-only LINE_MODE register. This pin should be tied to a certain level through initial hardware reset and not be changed while out of the reset state. This pin also controls the mode of the entire Line Interface block, so the entire block is either in Direct mode or H-MVIP mode.

Direct Mode

The options available in Direct Mode are:

- Per line synchronization: Frame or multiframe basis, and internal control or externally controlled

- Per line format: PMC standard format or MVIP format.

Synchronization

Synchronization can be configured on a per line basis, and is controlled by the MF_SYNC_MODE bit in the Low Speed Configuration Register (LS_Ln_CFG_REG), and the GEN_SYNC bit in the LIN_STR_MODE memory register for each line.

Synchronization can either be done on a multi-frame basis or a frame basis. If multi-frame synchronization is required then MF_SYNC_MODE bit for that line in the LS_Ln_CFG_REG must be set. Otherwise, if frame synchronization or no synchronization is required then leave MF_SYNC_MODE bit clear as default.

These values should be configured before A1SP software reset is released.

In the receive direction, synchronization is always controlled by the external line interface. However, in the transmit direction, synchronization can be controlled from either the local link side or the external line side. Set GEN_SYNC if the local link side is controlling synchronization. Otherwise, if GEN_SYNC is low, then the external lines are controlling synchronization or no synchronization is required.

Low Speed Line Format

There are two choices of line format: 1) PMC standard format, and 2) MVIP format.

The format can be controlled on a per line basis. If MVIP_EN is set in the Low Speed
Configuration Register for that line (LS_Ln_CFG_REG), then the line is in MVIP-90 mode.
Otherwise the line is in PMC standard format.

This value should be configured before A1SP software reset is released.

Note that if a mixture of MVIP-90 lines and non MVIP-90 lines are used then line 0 must be
MVIP-90.

If lines are configured in MVIP-90 mode then TL_SYNC0 becomes the F0B input (125-us
frame sync signal) and CRL_CLK becomes the C4B clock (4.096 MHz).

AAL1gator-4/8 samples F0B on a C4B falling edge, and expects F0B to be exactly one C4B
clock cycle wide, but F0B need not mark every 125 us frame. AAL1gator-4/8 samples
RL_DATA and RL_SIG at the 3/4 point in the MVIP-90 bit-period, which is a C4B rising edge.
AAL1gator drives TL_DATA and TL_SIG at the C4B falling edge at the start of the MVIP-90
bit-period.

Set LIN_STR_MODE_n=0x0001 (no CAS) or LIN_STR_MODE_n=0x0003 (with CAS) for
line 0, and any other MVIP-90 lines.

High Speed Line in Direct Mode

A single high speed line up to 52 MHz can be run on line 0. The HS_LIN_REG controls the
High Speed functionality. The CLK_SOURCE_TX and CLK_SOURCE_RX fields in the
LIN_STR_MODE memory register control the clock mode. In high speed mode only "000"
(clock is an input), or "001" (loop timing) modes are permitted.

The A_SW_RESET bit in the A_CMD_REG memory register functions as a queue reset signal
in high speed mode. If the state of LOOPBACK_ENABLE in TRANSMIT_CONFIG is desired
to be changed, the high speed queue must be reset using the A_SW_RESET bit.

Also when re-activating a highspeed queue, if it is required that the first sequence number of the
new connection has to be 0, then the queue must be reset using the A_SW_RESET bit.

Otherwise, clearing and setting the TX_ACTIVE bit in QUEUE_CONFIG can be used to
deactivate and activate the queue.

H-MVIP Mode

In H-MVIP mode synchronization is always controlled from the external interface and the sync
signal is always considered to be a frame synchronization signal. Therefore
MF_SYNC_MODE and GEN_SYNC should be inactive for all lines when this mode is in use.

When in H-MVIP mode, the line should be configured to be in normal mode, by clearing
MVIP_EN bit in the corresponding LS_Ln_CFG_REG. In the LIN_STR_MODE register, the
following bits should be disabled (value = '0'), LOW_CDV, E1_WITH_T1_SIG, T1_MODE,
GEN_SYNC, CLK_SOURCE_TX, CLK_SOURCE_RX and SRTS_EN.

13.3   UTOPIA Interface Configuration

There is very little setup required to configure the UTOPIA Interface. For typical operation, the UI_COMN_CFG_REG, UI_SRC_CFG_REG, and UI_SNK_CFG_REG need to be written to select the mode of operation and the UI_SRC_ADD_CFG and UI_SNK_ADD_CFG need to be programmed for a pre-defined address of the device. Once the registers are written with the proper configuration information, the enable bit should be set to enable normal operation. The UI_EN bit should be disabled by a chip, which is connected to the AAL1gator via the UTOPIA/Any-PHY bus, prior to its reset or reconfigured. This disabling is required to prevent deactivation or reconfiguration in the middle of a cell transfer.

Aside from the normal configurations, the block can also be placed in loopback where cells received on the UI interface are transmitted back out onto the UI interface. The block can be configured to loop all received cells by setting the U2U_LOOP bit in the UI_COMN_CFG register. Alternately, only cells with a VCI that matches the VCI contained in U2U_LOOP_VCI register can be looped back while other cells proceed through normally. The VCI based UI to UI loopback should be disabled when writing to U2U_LOOP_VCI.

13.4   Special Queue Configuration Modes

13.4.1 AAL0

AAL0 is a null ATM adaptation layer. In this mode, all 48 payload bytes of the ATM cell contain data. In this mode, there is no data structure, signaling or sequence number. However, AAL0 mode can be enabled for a queue, which is mapped to an entire link, or it can be mapped to a single DS0 queue, or it can be mapped to a group of DS0s.

However, it is important to note that because there is no structure, the AAL0 queue will function as unstructured data. If the queue is a nxDS0 queue, the data will maintain byte alignment across the ATM network, but the bytes may not arrive in the same DS0s they were transmitted.

AAL0 mode is enabled by setting AAL0_MODE_ENABLE in the TRANSMIT_CONFIG memory register in the transmit queue table and by setting the R_AAL0_MODE bit in the R_MP_CONFIG memory register in the receive queue table.

13.5   JTAG Support

The AAL1gator-4/8 supports the IEEE Boundary Scan Specification as described in the IEEE 1149.1 standards. The Test Access Port (TAP) consists of the five standard pins, TRSTB, TCK, TMS, TDI and TDO used to control the TAP controller and the boundary scan registers. The TRSTB input is the active-low reset signal used to reset the TAP controller. TCK is the test clock used to sample data on input, TDI and to output data on output, TDO. The TMS input is used to direct the TAP controller through its states. The basic boundary scan architecture is shown below.

Figure 70   Boundary Scan Architecture
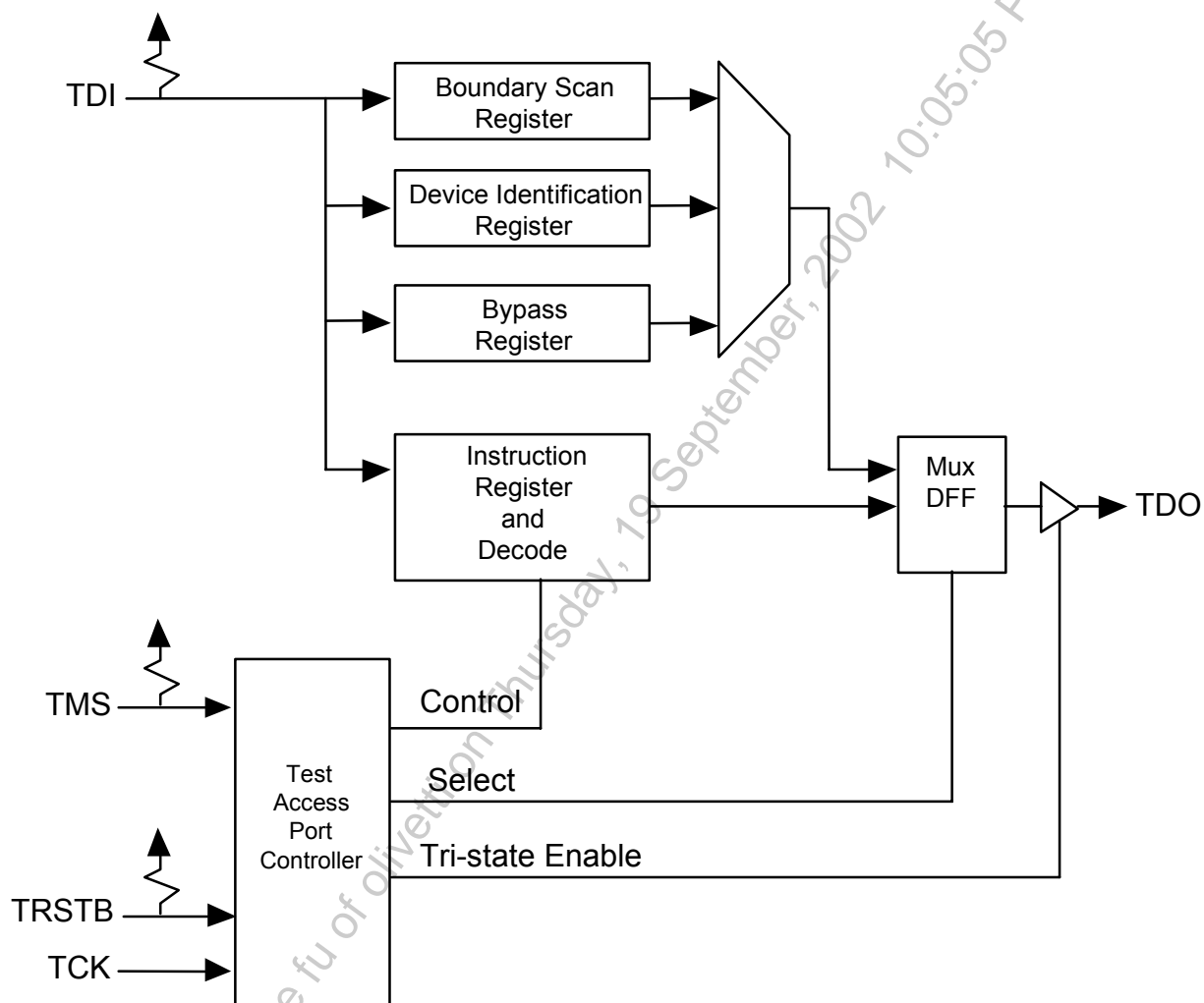
The boundary scan architecture consists of a TAP controller, an instruction register with instruction decode, a bypass register, a device identification register and a boundary scan register. The TAP controller interprets the TMS input and generates control signals to load the instruction and data registers. The instruction register with instruction decode block is used to select the test to be executed and/or the register to be accessed. The bypass register offers a single-bit delay from primary input, TDI to primary output, TDO. The device identification register contains the device identification code.

The boundary scan register allows testing of board inter-connectivity. The boundary scan register consists of a shift register place in series with device inputs and outputs. Using the boundary scan register, all digital inputs can be sampled and shifted out on primary output, TDO. In addition, patterns can be shifted in on primary input, TDI and forced onto all digital outputs.

13.5.1 TAP Controller

The TAP controller is a synchronous finite state machine clocked by the rising edge of primary input, TCK. All state transitions are controlled using primary input, TMS. The finite state machine is described below.

Figure 71   TAP Controller Finite State Machine

All transitions dependent on input TMS
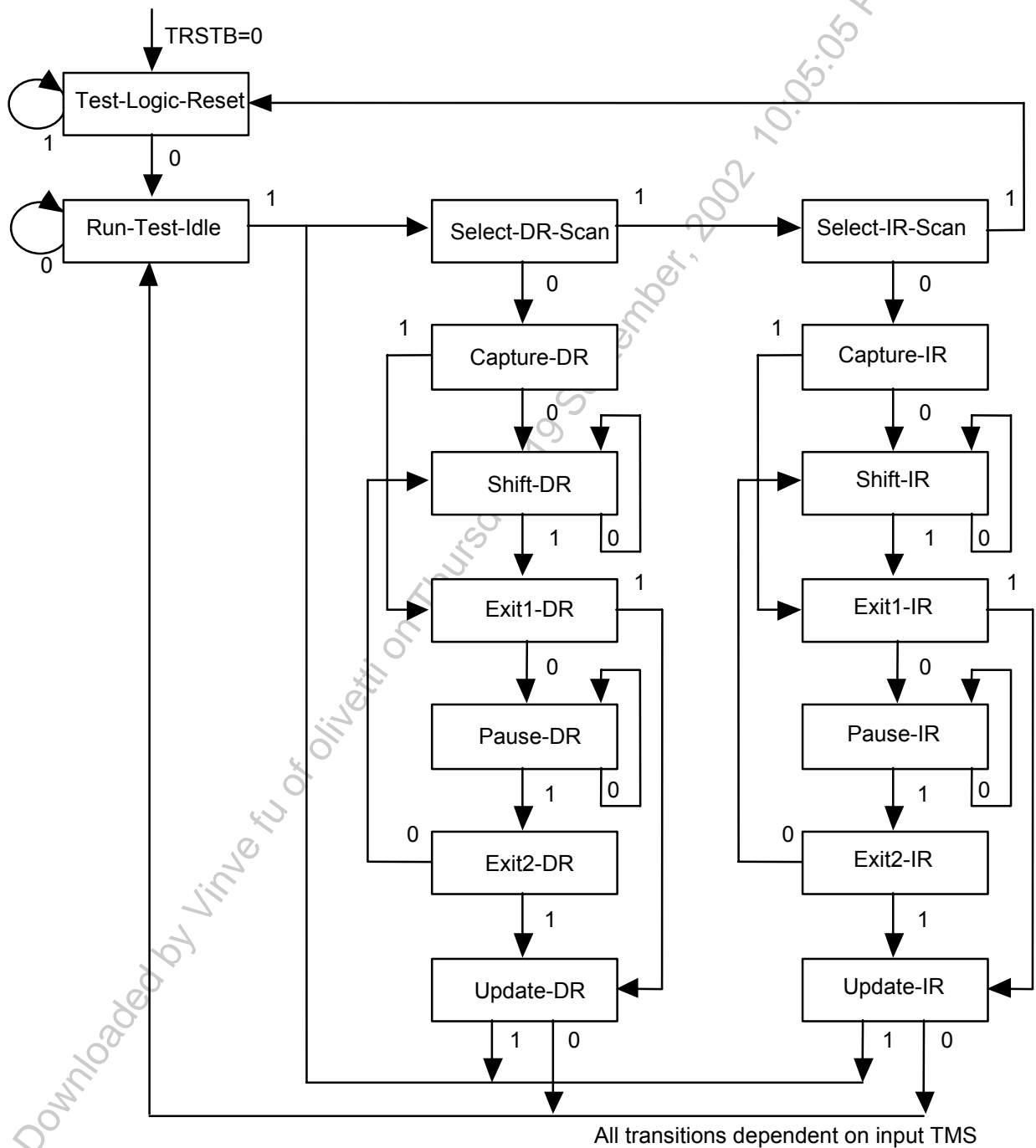
Test-Logic-Reset

The test logic reset state is used to disable the TAP logic when the device is in normal mode operation. The state is entered asynchronously by asserting input, TRSTB. The state is entered synchronously regardless of the current TAP controller state by forcing input, TMS high for 5 TCK clock cycles. While in this state, the instruction register is set to the IDCODE instruction.

Run-Test-Idle

The run test/idle state is used to execute tests.

Capture-DR

The capture data register state is used to load parallel data into the test data registers selected by the current instruction. If the selected register does not allow parallel loads or no loading is required by the current instruction, the test register maintains its value. Loading occurs on the rising edge of TCK.

Shift-DR

The shift data register state is used to shift the selected test data registers by one stage. Shifting is from MSB to LSB and occurs on the rising edge of TCK.

Update-DR

The update data register state is used to load a test register's parallel output latch. In general, the output latches are used to control the device. For example, for the EXTEST instruction, the boundary scan test register's parallel output latches are used to control the device's outputs. The parallel output latches are updated on the falling edge of TCK.

Capture-IR

The capture instruction register state is used to load the instruction register with a fixed instruction. The load occurs on the rising edge of TCK.

Shift-IR

The shift instruction register state is used to shift both the instruction register and the selected test data registers by one stage. Shifting is from MSB to LSB and occurs on the rising edge of TCK.

Update-IR

The update instruction register state is used to load a new instruction into the instruction register. The new instruction must be scanned in using the Shift-IR state. The load occurs on the falling edge of TCK.

The Pause-DR and Pause-IR states are provided to allow shifting through the test data and/or instruction registers to be momentarily paused.

Boundary Scan Instructions

The following is an description of the standard instructions. Each instruction selects an serial test data register path between input, TDI and output, TDO.

BYPASS

The bypass instruction shifts data from input, TDI to output, TDO with one TCK clock period delay. The instruction is used to bypass the device.

EXTEST

The external test instruction allows testing of the interconnection to other devices. When the current instruction is the EXTEST instruction, the boundary scan register is place between input, TDI and output, TDO. Primary device inputs can be sampled by loading the boundary scan register using the Capture-DR state. The sampled values can then be viewed by shifting the boundary scan register using the Shift-DR state. Primary device outputs can be controlled by loading patterns shifted in through input TDI into the boundary scan register using the Update-DR state.

SAMPLE

The sample instruction samples all the device inputs and outputs. For this instruction, the boundary scan register is placed between TDI and TDO. Primary device inputs and outputs can be sampled by loading the boundary scan register using the Capture-DR state. The sampled values can then be viewed by shifting the boundary scan register using the Shift-DR state.

IDCODE

The identification instruction is used to connect the identification register between TDI and TDO. The device's identification code can then be shifted out using the Shift-DR state.

STCTEST

The single transport chain instruction is used to test out the TAP controller and the boundary scan register during production test. When this instruction is the current instruction, the boundary scan register is connected between TDI and TDO. During the Capture-DR state, the device identification code is loaded into the boundary scan register. The code can then be shifted out output, TDO using the Shift-DR state.

Boundary Scan Cells

In the following diagrams, CLOCK-DR is equal to TCK when the current controller state is SHIFT-DR or CAPTURE-DR, and unchanging otherwise. The multiplexer in the center of the diagram selects one of four inputs, depending on the status of select lines G1 and G2. The ID Code bit is as listed in the Boundary Scan Register table in the JTAG Test Port section 11.2.

Figure 72   Input Observation Cell (IN_CELL)
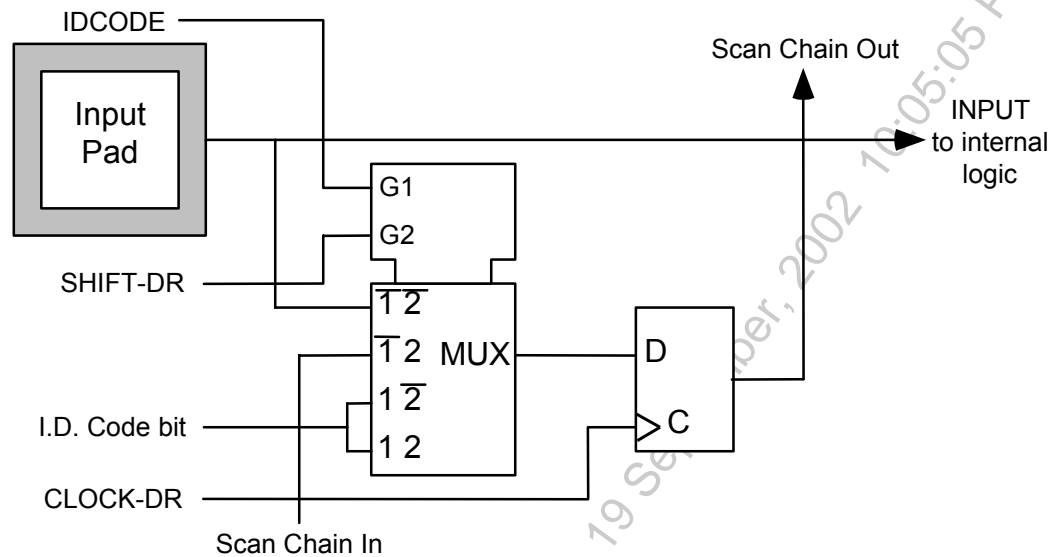
Figure 73   Output Cell (OUT_CELL)
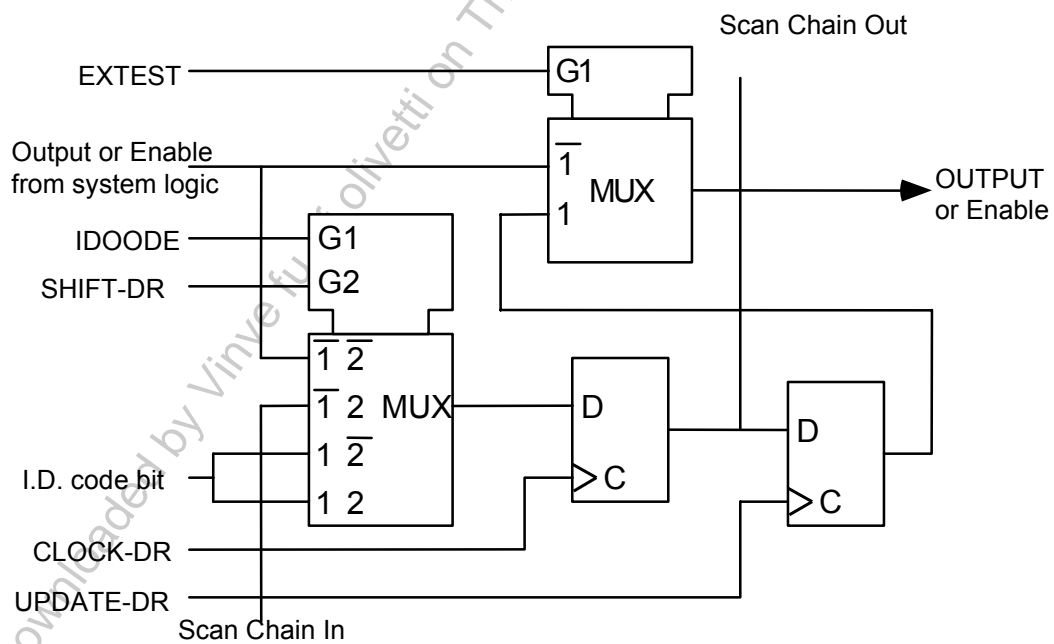
Figure 74   Bidirectional Cell (IO_CELL)
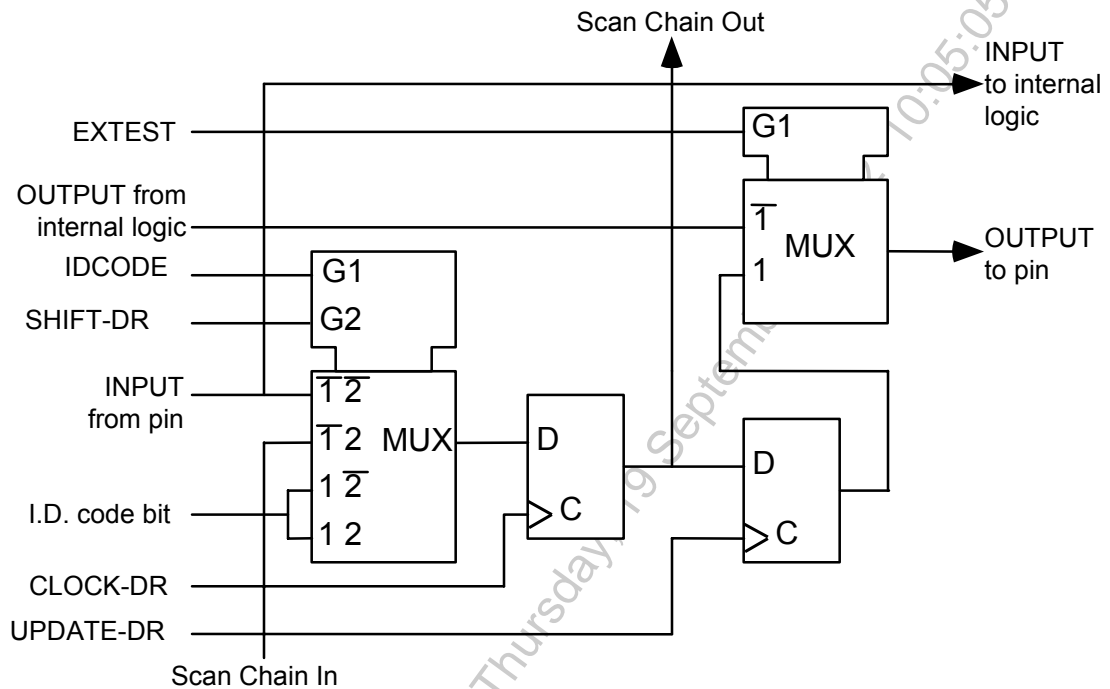
Figure 75   Layout of Output Enable and Bidirectional Cells
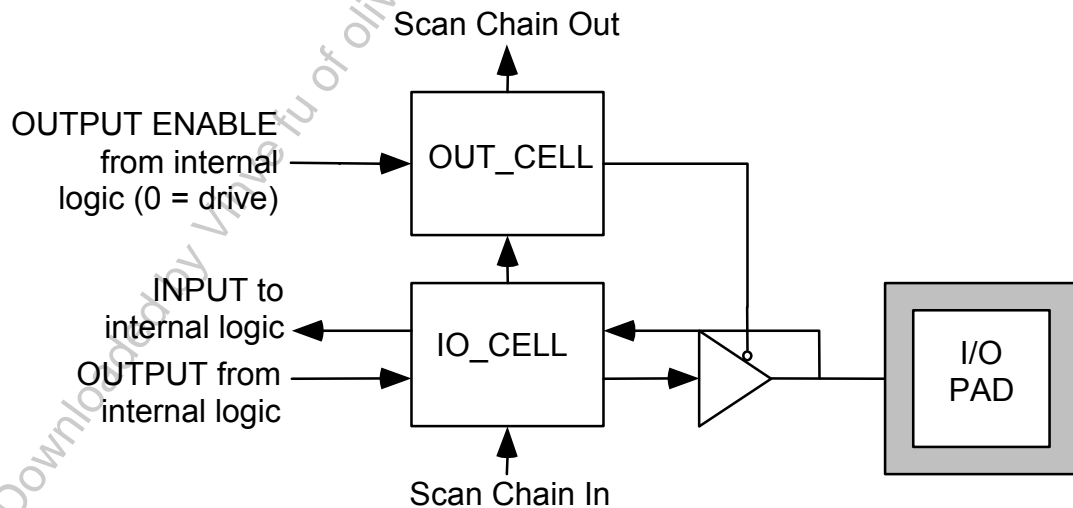
14   Functional Timing

This section shows the functional relationship between inputs and outputs. No propagation delays are shown.

Figure 76   Pipelined Single-Cycle Deselect SSRAM
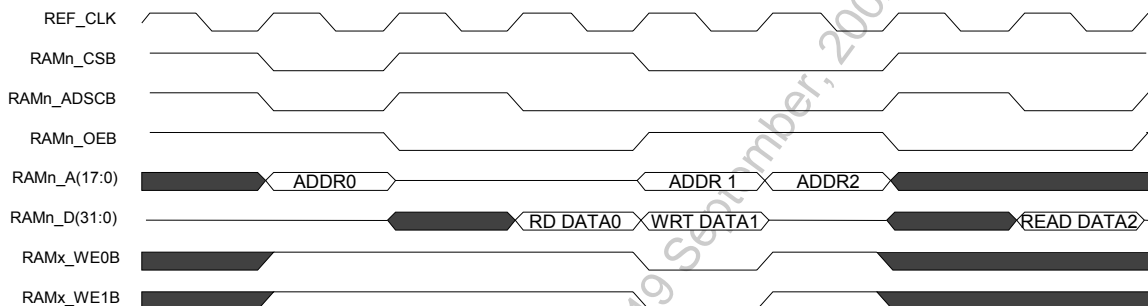
The diagram above illustrates the SSRAM accesses when it is configured to use a pipelined single-cycle deselect SSRAM. In this mode, ADSC is used and bursting is not used. A new address is presented for each access along with the associated control (CSB, ADSCB, OEB, WEB). The RAM_ADSCB signal is pulsed during the last cycle of a read prior to a write to place the SSRAM into a deselect state. The deselect state of the SSRAM causes the SSRAM to tristate it's data bus after the rising edge of the clock. This operation reduces contention on the bus when switching from a read to a write operation. The RAM_OEB is used to prevent bus contention when switching from a write to a read.

PMC-SIERRA

Figure 77   Pipelined ZBT SSRAM
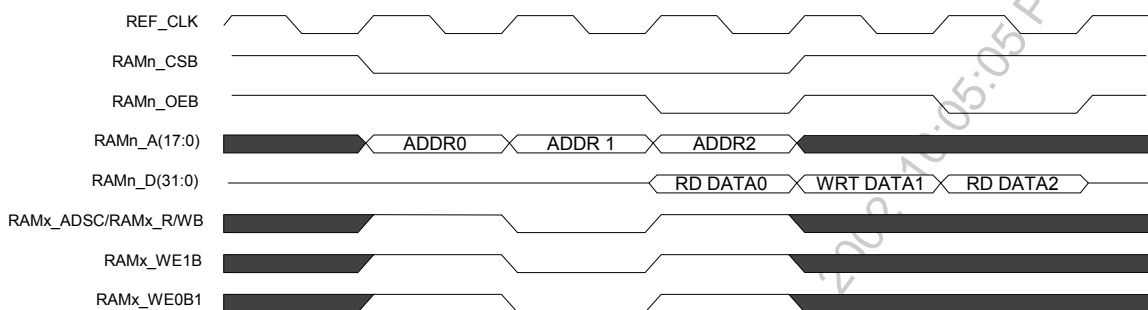
The diagram above illustrates the SRAM accesses when it is configured to use the pipelined ZBT SSRAM for improved throughput. In this mode, RAM_ADSC is redefined as a RAM_R/WB and bursting is not used. A new address is presented for each access along with the associated control (CSB, R/WB, OEB, WEB). The write data is placed on the bus 2 cycles after the write address and command The RAM_OEB is used to prevent bus contention when switching from a write to a read. When switching from a read to a write, some bus contention may be present but is minimal since the ZBTs SSRAMs are quick to disable their buffers and the gator is slower to enable it's buffers.

14.1 Source Utopia

In ATM master mode, the SRC_INTF block sources TATM_DATA, TATM_PAR, TATM_SOC, and TATM_ENB while receiving TATM_CLAV. The Start-Of-Cell (TATM_SOC) indication is generated coincident with the first word (8-bit or 16-bit) of each cell that is transmitted on TATM_DATA. TATM_DATA, TATM_PAR, and TATM_SOC are driven at all times. The write enable signal indicates which clock cycles contain valid data for the Utopia bus. The device will not assert the TATM_ENB signal until it has a full cell to send.

Note that during hardware/software reset, or when UI_EN in the UI_COMN_CFG register is deasserted low, all Utopia interface outputs are tri-stated.

In ATM master mode, the SRC_INTF responds to the TATM_CLAV by beginning a cell transfer as shown in Figure 78 below. If TATM_CLAV is asserted and the UI_SRC_INTF has data to send, it will do so by asserting TATM_TENB while driving data. Octet (byte) level handshaking and data transfer pausing is not supported, thus although TPA_I may go low during the middle of a transfer as shown in the Figure 78 example, the data transfer will still continue and TENB_O will not be deasserted.

PMC-SIERRA

Figure 78   SRC_INTF Start of Transfer Timing (Utopia 1 ATM Mode)
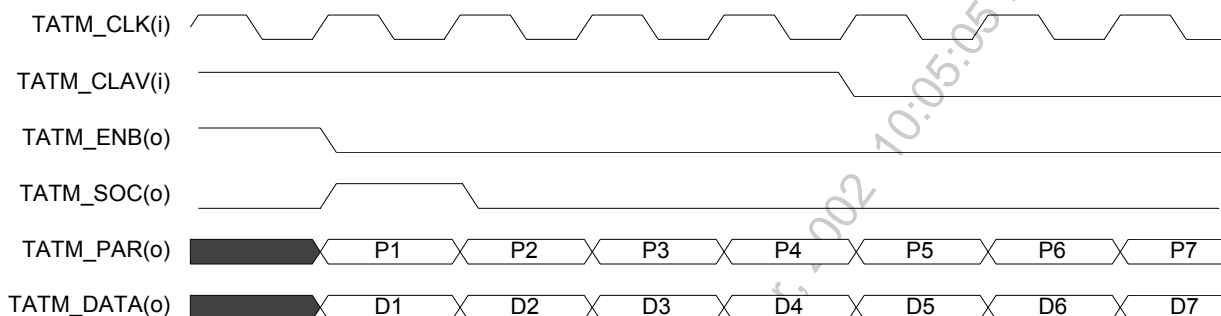
The end of a transfer for the SRC_INTF in Utopia 1 ATM Master mode is shown in Figure 79. Per the Utopia 1 specification, TPA_I must be deasserted low by the PHY slave at least four cycles before the end of the cell if the PHY cannot accept another complete cell. Note that one additional cycle is used to allow the input TPA signal to be clocked into the master device, thus the PHY must assert TPA low along with data byte D49 and no later if it cannot accept another cell.

Figure 79   SRC_INTF End-of-Transfer Timing (Utopia 1 ATM Mode)
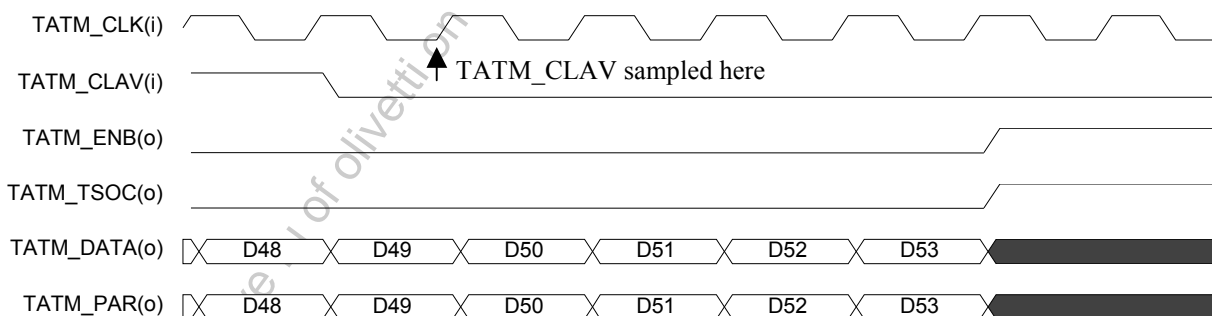
In PHY slave mode, the SRC_INTF block sources RPHY_DATA, RPHY_PAR, RPHY_SOC, and RPHY_CLAV, while receiving RPHY_ENB. The RPHY_ADDR(4:0) inputs are not used in Utopia 1 PHY slave mode but must be set to be the same value as the UI_SRC_ADDR_CFG register value for proper operation. The RPHY_SOC indication is generated coincident with the first word (8-bit or 16-bit) of each cell that is transmitted on RPHY_DATA. In PHY mode, RPHY_DATA, RPHY_SOC, and RPHY_PAR are driven only when valid data is being sent; otherwise they are tristated.

Figure 80   UI_SRC_INTF Start-of-Transfer Timing (Utopia 1 PHY Mode)
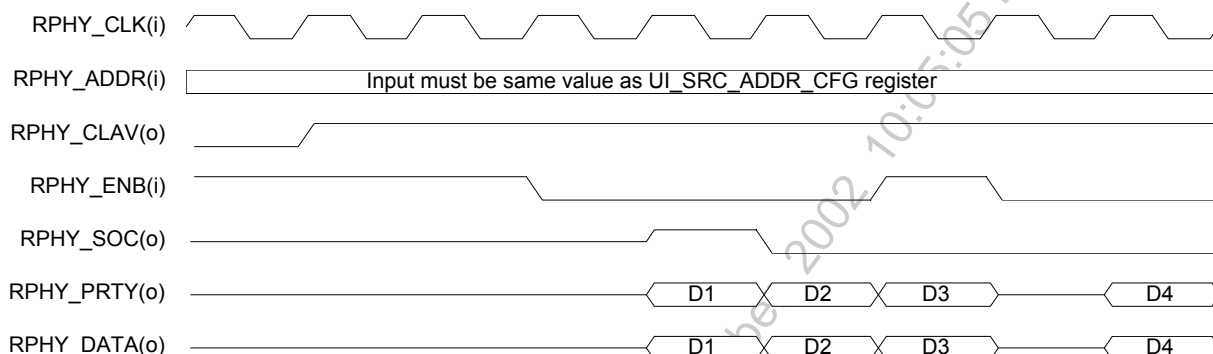
The cell available (RPHY_CLAV) signal indicates when the device has a complete cell to send. In Utopia 1 slave (SPHY) mode, RPHY_CLAV is always driven. Figure 80 above, shows a start-of-transfer for Utopia 1 PHY slave mode. If a cell is being read in Utopia 1 PHY mode, RPHY_CLAV will be asserted until the cell has been read out of the source MCFF FIFO and there are no more cells to send. Data is placed on RPHY_DATA any cycle following one in which RPHY_ENB was asserted. In Utopia 1 PHY mode RPHY_ENB can be deasserted during the cell transfer for data transfer pausing. Figure 81 below shows the end of transfer behavior in PHY mode. Note that in Utopia 1 mode the status of RPHY_CLAV should reflect the current cell transfer, thus RPHY_CLAV remains asserted until the last byte/word of the cell.

Figure 81   UI_SRC_INTF End-of-Transfer (Utopia 1 PHY Mode)
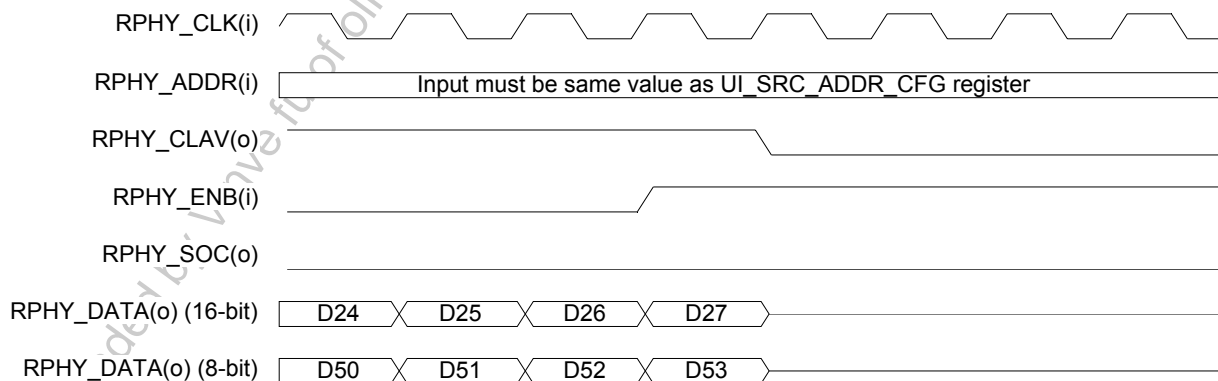
In Utopia 2 PHY slave (MPHY) mode, RPHY_ADDR is used for device polling and selection. RPHY_CLAV is only driven during cycles following ones in which RPHY_ADDR[4:0] matches the CFG_ADDR[4:0] in the UI_SRC_ADD_CFG register. When a channel is being polled in Utopia 2 slave mode (MPHY), the value of RPHY_CLAV will be 1 until the cell has been read out of the FIFO and there are no more cells to send. In Utopia 2 mode, the SRC_INTF block has to be selected for data to be driven. This is done when RPHY_ADDR[4:0] matches source CFG_ADDR[4:0] in the UI_SRC_ADDR_CFG register the cycle before RPHY_ENB goes low. A start-of-transfer sequence for Utopia 2 PHY slave mode is shown below in Figure 82. Transfer pausing is supported in Utopia 2 PHY slave mode, thus RPHY_ENB can be deasserted during the cell transfer and the data driven by the SRC_INTF will stop one cycle later.

Figure 82   UI_SRC_INTF Start-of-Transfer Timing (Utopia 2 PHY Mode)
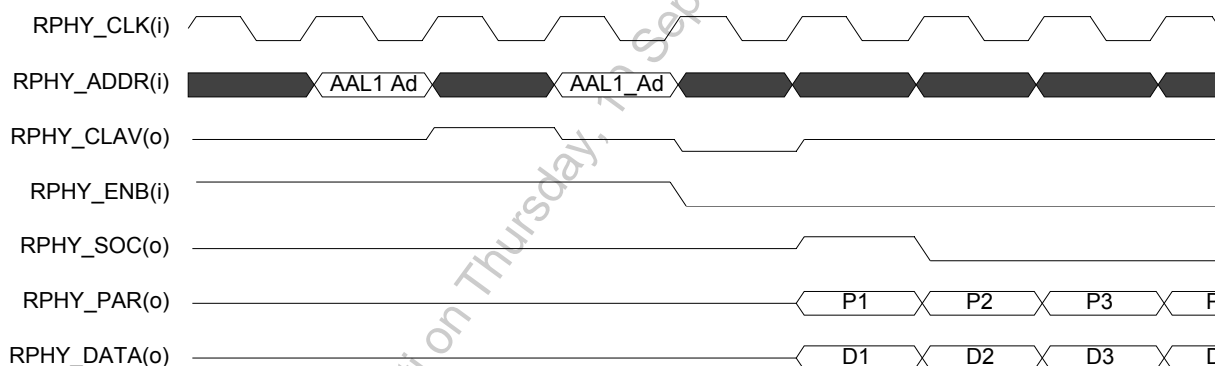
The end-of-transfer behavior is shown in Figure 83. As with Utopia 1 mode, the state of RPHY_CLAV reflects the current cell transfer status and so remains asserted until the last data byte/word. The SRC_INTF in this example does not have another cell to send, so RPHY_CLAV shows a low value after the transfer.

Figure 83   UI_SRC_INTF End-of-Transfer Timing (Utopia 2 PHY Mode)
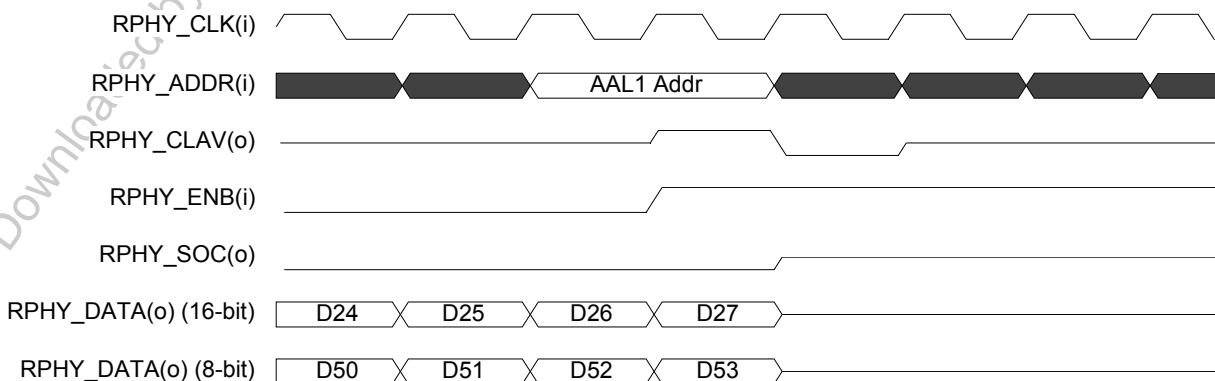
Figure 84 is a functional timing of the SRC_INTF for the start of a cell transfer when configured as an Any-PHY compliant receive slave. During a polling operation, when the SRC_INTF determines that the UI_SRC_ADDR_CFG address is on the bus it responds by driving RPHY_CLAV two cycles later. If CS_MODE_EN is set in the UI_SRC_CFG register then CSB must also be driven low one cycle after the RPHY_ADDR for proper response. If the SRC_INTF has a cell to send it will drive RPHY_CLAV high and, as a result, the master will activate RPHY_ENB to initiate a transfer. As with Utopia 2, the SRC_INTF is selected if its address is on RPHY_ADDR inputs during the cycle before RPHY_ENB goes low and in response transmits an entire cell. RPHY_RSX is driven high during the prepended byte address and RPHY_SOC is driven high during the first header byte of the ATM cell. Since data transfer pausing is not supported, once a transfer is initiated, RPHY_ENB should remain asserted until the last data byte/word. Note that RPHY_CLAV will be deasserted along with the second data byte/word driven on the bus if the SRC_INTF does not have another cell to send, as in Figure 84. This is because, in Any-PHY mode, the RPHY_CLAV signal should always reflect the cell available status for the next possible future transfer rather than the current one as in Utopia mode.

Figure 84   UI_SRC_INTF Start-of-Transfer Timing (Any-PHY PHY Mode)
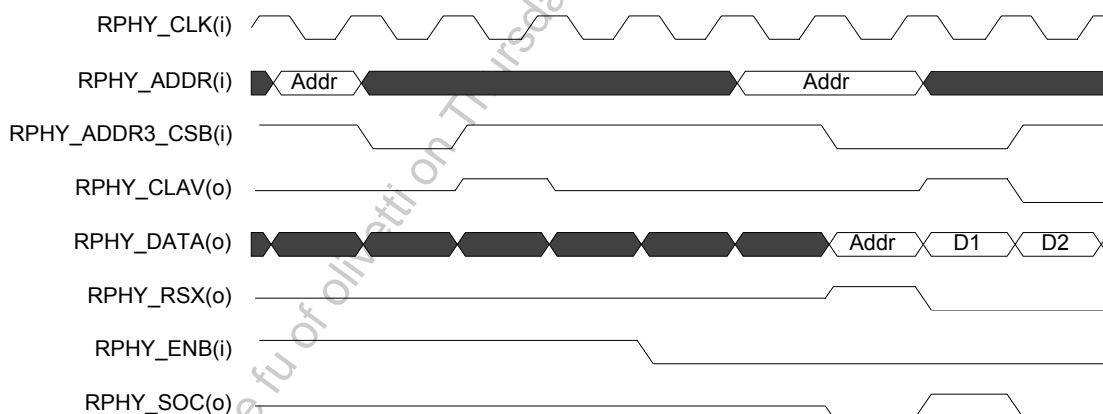
Figure 85 below shows an example of the end of a cell transfer for the SRC_INTF while in Any-PHY PHY slave mode. In this particular instance, the SRC_INTF still does not have another cell to send for this particular address, thus RPHY_CLAV is not asserted when polled.

Figure 85   UI_SRC_INTF End-of-Transfer Timing (Any-PHY PHY mode)
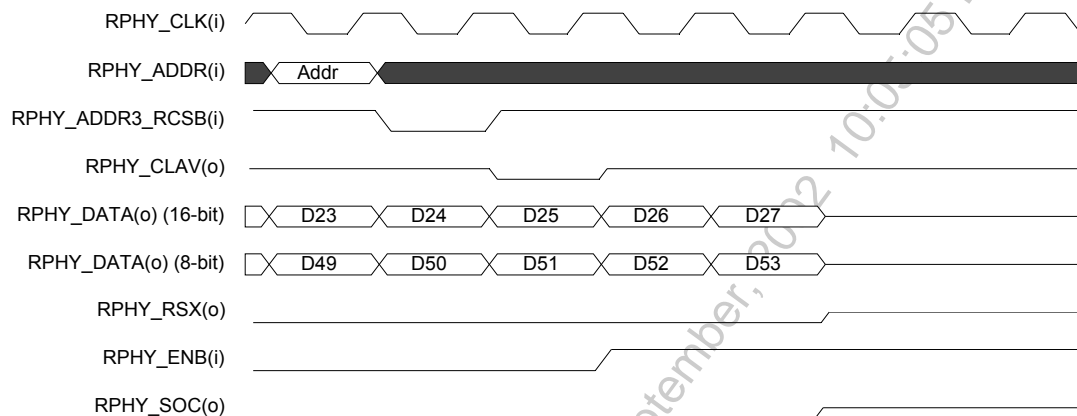
14.2 Sink Utopia

In ATM master mode, the SNK_INTF block receives RATM_DATA, RATM_PAR, RATM_SOC, and RATM_CLAV while driving RATM_ENB. During reset the SNK_INTF tristates the RATM_ENB (and all other SRC_INTF outputs). After the end of reset and after UI_EN in the UI_COMN_CFG register is set, if the RATM_CLAV input signal is not asserted the SNK_INTF waits and RATM_ENB is deasserted high. As shown in Figure 86, once the RATM_CLAV input signal is asserted, the RATM_RENB output signal is asserted low 2 cycles later and it is expected that the PHY slave will deliver the first data byte one cycle after RENB goes low. Typically a start-of-cell signal will be sent by the PHY slave along with the first byte of data, however RATM_SOC is optional and the RATM_SOC input could be tied low. Note however, not using an SOC eliminates the possibility of runt cell detection. Once RATM_ENB goes low, a counter is started, and 53 bytes are expected. If an SOC occurs within a cell, the counter reinitializes and the previous corrupted cell will be dropped and the second good cell will be received. The SNK_INTF block stores the ATM cell in the receive FIFO. The 4 cell MCFF FIFO allows the interface to accept data at the maximum rate and if the FIFO fills, the RATM_ENB signal will not be asserted again until RATM_CLAV is asserted and the device is ready to accept an entire cell. The maximum supported clock rate is 52 MHz.

Figure 86 SNK_INTF Start-of-Transfer Timing (Utopia 1 ATM Mode)
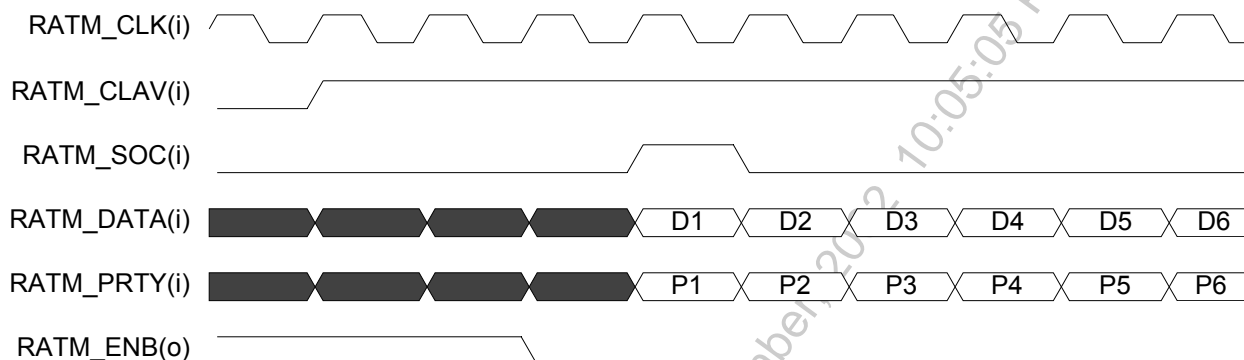
Note that in ATM master mode the SNK_INTF uses cell based handshaking, thus once a transfer has begun, RATM_ENB will be asserted continuously until the end of the cell regardless of the state of RATM_CLAV. Figure 87 shows the end of transfer signal behavior. In this example, since the PHY slave does not have another cell to deliver, the RATM_CLAV signal is deasserted after the last byte is delivered. Also, the AAL1GATOR acting as an ATM master always deasserts the RATM_ENB at the end of a cell transfer. RATM_CLAV is then sampled again on the following clock cycle, and if asserted, RATM_ENB will be asserted again for a new transfer.

Figure 87 SNK_INTF End-of-Transfer Timing (Utopia 1 ATM Mode)

In PHY slave mode, the SNK_INTF block receives TPHY_DATA, TPHY_SOC, and TPHY_ENB while driving TPHY_CLAV. TPHY_CLAV indicates when the device is ready to receive a complete cell. In Utopia 1 mode, TPHY_CLAV is always driven. In Utopia 2 mode, TPHY_CLAV is only driven during cycles following ones in which TPHY_ADDR[4:0] matches the address CFG_ADDR[4:0] in the UI_SNK_ADDR_CFG register.

Figure 88 below shows the start of transfer timing for Utopia 1 Phy slave mode. At reset, and when UI_EN in the UI_COMN_CFG register is set low, the SNK_INTF block tristates TPHY_CLAV (not shown). While not in reset, the SNK_INTF asserts TPHY_CLAV if it can accept a cell and waits for TPHY_ENB to be asserted.

When TPHY_ENB is asserted, a counter is started and 53 bytes are received (27 words in 16-bit mode). If a new TPHY_SOC occurs within a cell, the counter reinitializes. This means that the corrupted cell will be dropped and the second good cell will be received.

In Utopia 1 PHY mode, the SNK_INTF will accept data as long as TPHY_ENB is asserted and the sink MCFF FIFO has room. Data transfer pausing is supported in Utopia 1 PHY mode, thus if TPHY_ENB is deasserted, as in Figure 88, the data transfer is paused. The 4 cell MCFF FIFO allows the interface to accept data at the maximum rate. If the FIFO fills, the TPHY_CLAV signal will be deasserted by data transfer cycle D12 (D10 in 16-bit mode) and not be asserted again until the device is ready to accept an entire cell (This is not shown in a figure).

Figure 88   SNK_INTF Start-of-Transfer Timing (Utopia 1 PHY Mode)
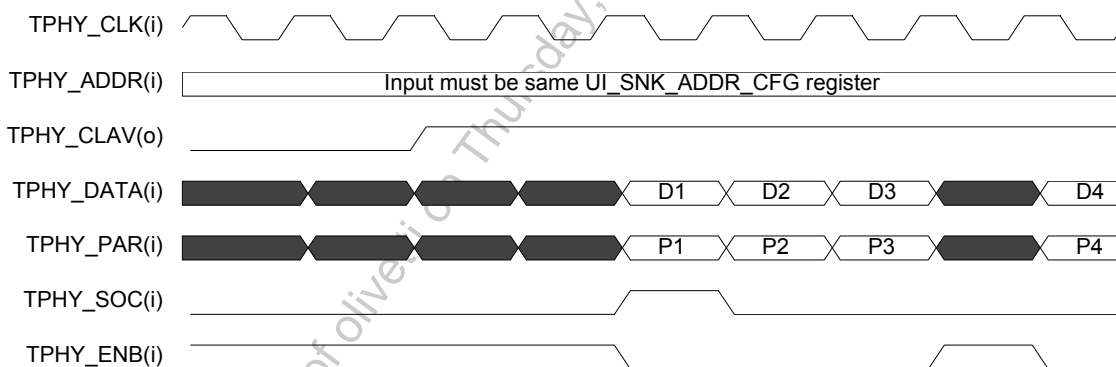
In Utopia 2 mode, a cell transfer is started as a result of an ATM master polling the SNK_INTF with an address matching the value in the UI_SNK_ADDR_CFG register. The SNK_INTF responds by asserting TPHY_CLAV the cycle after UI_SNK_RADR matches the UI_SNK_ADDR_CFG register value. Selection occurs when the value on the TPHY_ADDR matches the configured address during the cycle before UI_SNK_RENB is brought low. This is shown in Figure 89 below. The SNK_INTF will accept data as long as TPHY_ENB is asserted and it has room. Data transfer pausing is supported in Utopia 2 single address PHY mode, thus if TPHY_ENB is deasserted as in Figure 89 the data transfer is paused. Furthermore, in Utopia 2 mode, although a cell transfer has been started, additional polling may show TPHY_CLAV as being asserted if there is still room in the sink FIFO as shown at the end of Figure 89.

PMC-SIERRA

Figure 89   SNK_INTF Start-of-Transfer Utopia 2
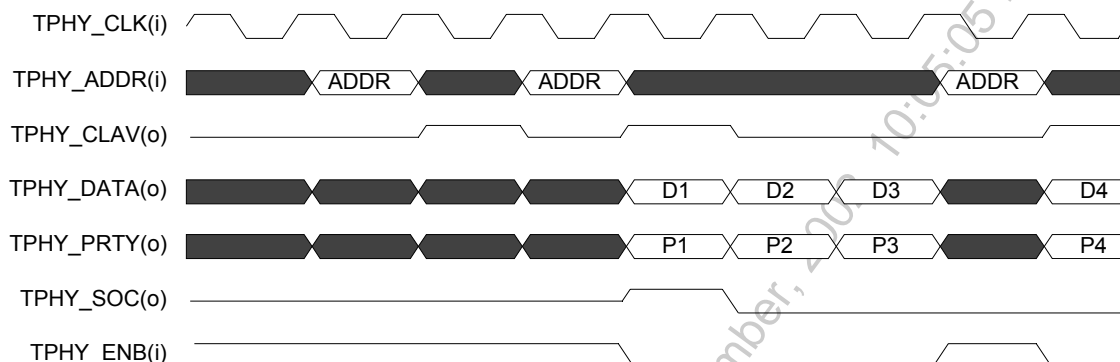
If the current cell being transferred will fill up the last of the four cell slots in the sink MCFF FIFO, then the TPHY_CLAV signal will be deasserted by the D10 or D12 data transfer cycle, in 16-bit and 8-bit modes respectively. This is shown in Figure 90. Note this transition will only be visible on TPHY_CLAV if the address is being polled as in the figure.

Figure 90   SNK_INTF CLAV Disable Utopia 2
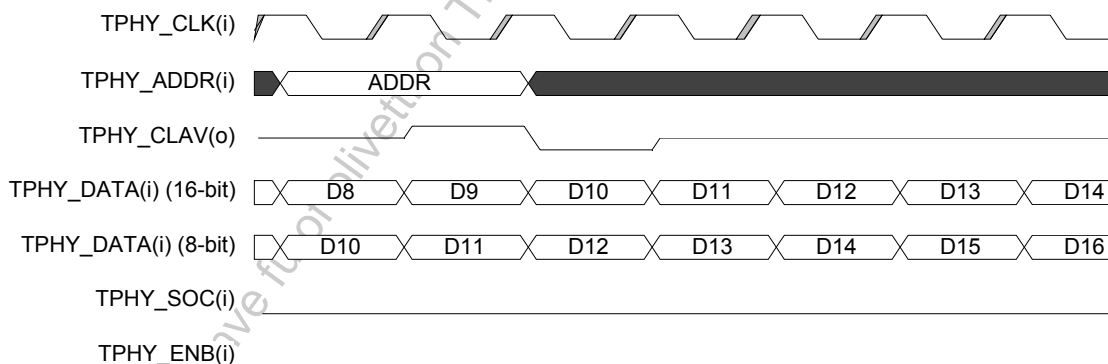
Since the Utopia 2 PHY mode utilizes the sink MCFF as a single four cell deep FIFO, the TPHY_CLAV signal can be asserted any time a cell slot becomes empty in the MCFF as is shown in Figure 91. The end of the transfer is coincident with the TPHY_ENB being deasserted, although it is possible for a master to keep TPHY_ENB asserted if the SNK_INTF has cell space available and the master has another cell to send. Note that once deasserted, TPHY_CLAV can become active at any time during a cell transfer if cell space becomes available in the sink MCFF FIFO.

Figure 91   SNK_INTF End-of-Transfer Utopia 2
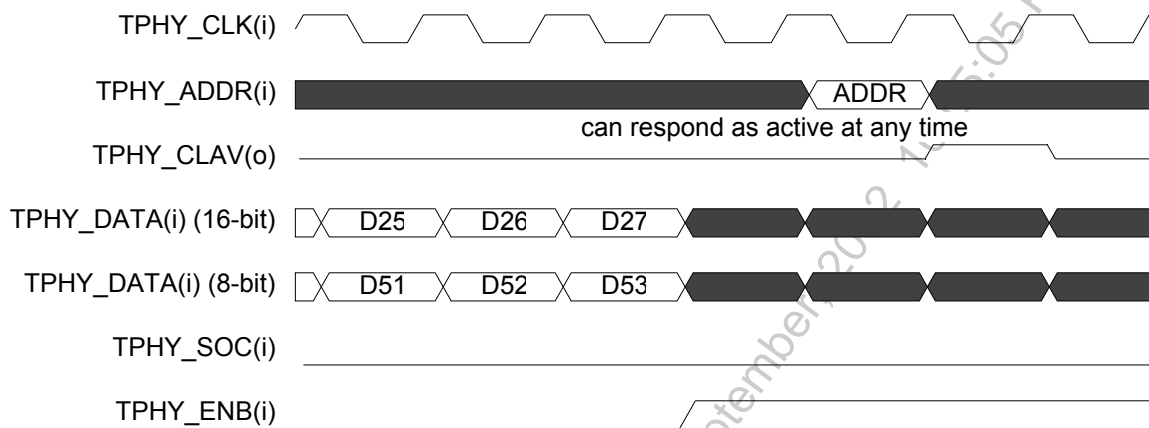
Figure 92 gives an example of the functional timing of the UI_SNK_INTF when configured as a Any-PHY compliant transmit slave. When the SNK_INTF has room for a cell and determines its address is on the bus, it will respond by driving TPHY_CLAV high **two** cycles later. If CS_MODE_EN is set in the UI_SNK_CFG register, then CSB should be driven low one cycle after the TPHY_ADDR. As a result, the master will activate TPHY_ENB to initiate a transfer. When TPHY_TSX is high the SNK_INTF will compare the prepended address with value stored in UI_SNK_ADDR_CFG register and if they match it will accept the data. TPHY_TSX should be driven high during the prepended byte address and TPHY_SOC should be driven high during the first header byte of the ATM cell.

Only cell level handshaking and no data transfer pausing is supported in Any-PHY mode, thus once transfer is initiated TPHY_ENB should remain asserted until the last data is transferred. Note that TPHY_CLAV is masked after the poll (when the address is applied) which is coincident with the assertion of TSX. Also note that, in Any-PHY mode both TPHY_ENB and TPHY_SOP are optional and both could be tied low indefinitely.

Figure 92   SNK_INTF Start-of-Transfer (Any-PHY PHY Mode)
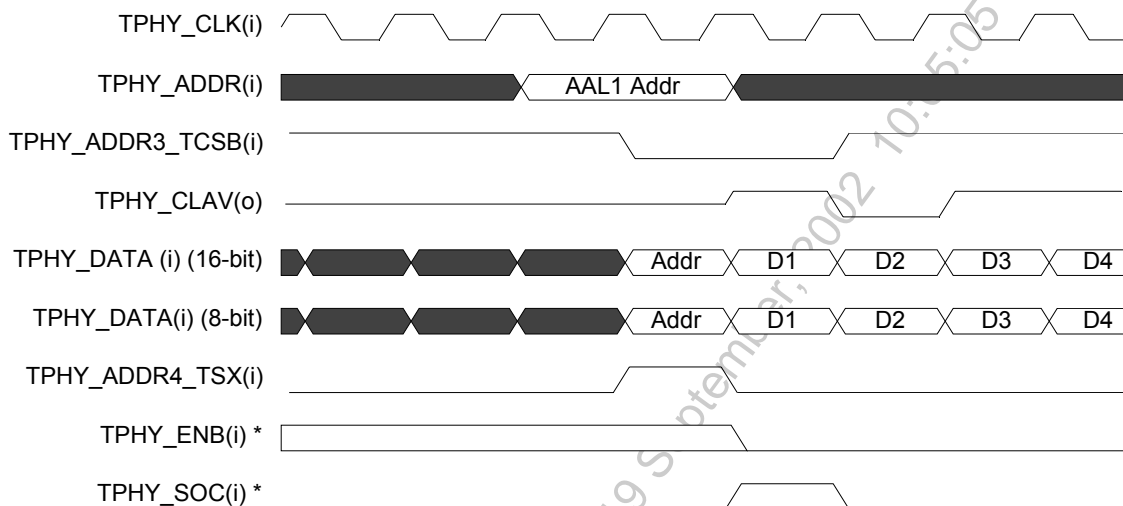
* TPHY_ENB and TPHY_SOC could be tied low for Any-PHY mode

Figure 93 shows the end of transfer for Any-PHY mode.

Figure 93   SNK_INTF End-of-Transfer (Any-PHY PHY Mode)
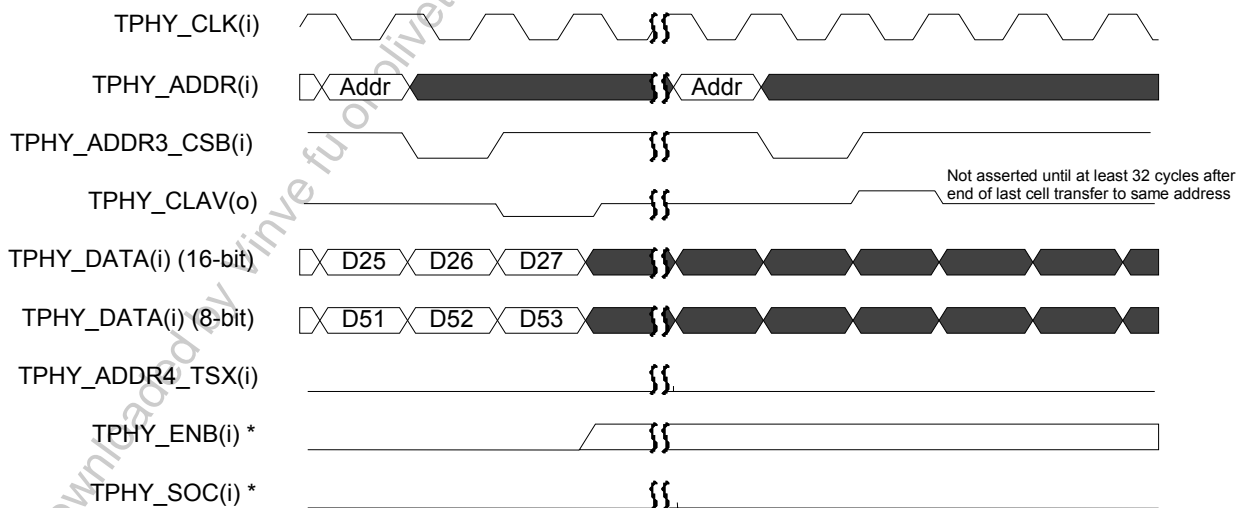
* TPHY_ENB and TPHY_SOC could be tied low for Any-PHY mode

14.3 Processor I/F

The microprocessor accesses the device by means of the CSB, RDB, and WRB lines. To perform a write cycle, the microprocessor asserts the CSB and WRB lines. The AAL1gator-4/8 then passes the address and data to the RAM interface or internal registers and, when complete, signals the microprocessor with the ACKB line. See Figure 94 below. To perform a read, the microprocessor asserts CSB and RDB and waits for ACKB before reading the data. See Figure 95 below.

If CSB, WRB and RDB are all active at the same time, all chip outputs go tri-state. Therefore, WRB and RDB should never be active at the same time. Because CSB, WRB, RDB are asynchronously sampled, the CSB or (RDB and WRB) signals should be deactivated for 2 SYS_CLK cycles in between each access.

The amount of time it takes for ACKB to go active is dependant on what other operations are happening at the same time and where the access is targeted. Register or internal memory operations will be responded to within a few cycles. Accesses to RAM usually take less than 10 clock cycles but sometimes can take much longer if there is a lot of contention for the RAM.

If interfacing to a multiplexed address data bus, an optional ALE signal is provided. If ALE is not required it should be tied high. See Figure 96 and Figure 97.

Figure 94   Microprocessor Write Access
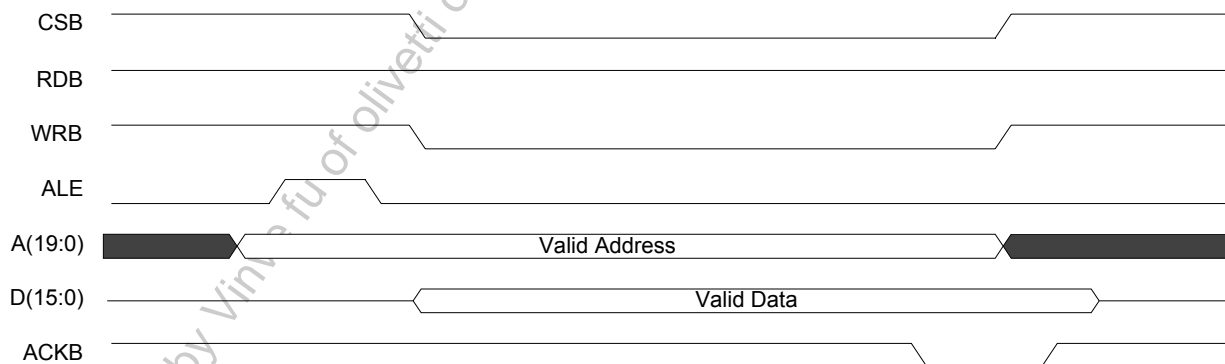
Figure 95   Microprocessor Read Access
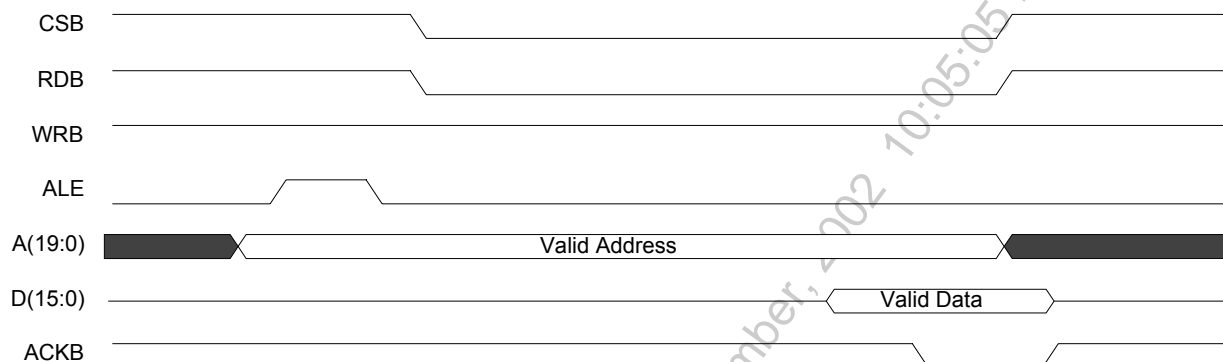
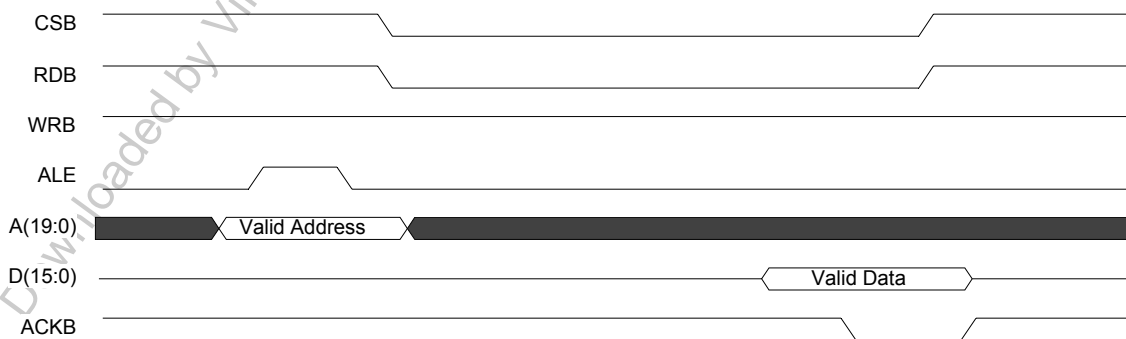**Figure 96   Microprocessor Write Access with ALE**
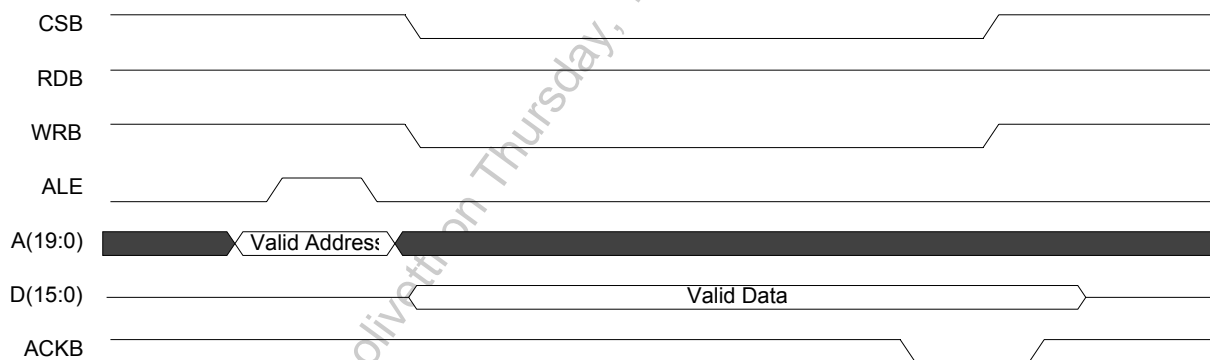
Figure 97   Microprocessor Read Access with ALE

PMC-SIERRA

14.4 External Clock Generation Control I/F (CGC)

Status information is played out on the External Clock Control (CGC) Interface which includes the external output signals: CGC_DOUT(3:0), CGC_LINE(3:0), SRTS_STBH, and ADAP_STBH.

Information is played out for all chip modes (Direct, and H-MVIP).

The interface clocking operates according to the following pseudocode:

- If a channel pair is being serviced, start a state machine to play out the channel status, as shown in Table 30 asserting ADAP_STBH as each state is played out.

- Else, if an SRTS difference value is ready, it is played out with SRTS_STBH asserted and ADAP_STBH deasserted.

- Else if a cell is received and a valid averaged relative buffer depth can be computed, start a state machine to play out the averaged relative buffer depth and queue number, as shown in Table 31, asserting ADAP_STBH as each state is played out.

- Once playout of a certain data type has begun if will not be interrupted.

14.4.1 SRTS Data Output

In SRTS mode the CGC block puts out a 4 bit value which represents the difference between the local SRTS value and the received SRTS value. Figure 98 below shows a typical CGC output in SRTS mode.

Figure 98   SRTS Data
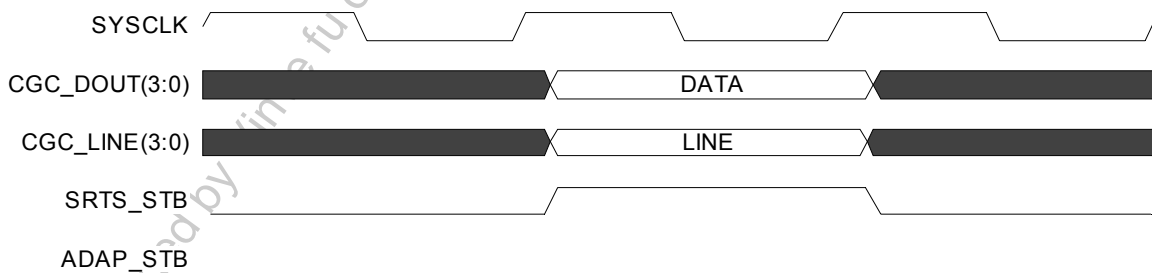
14.4.2 Channel Underrun Status Output

Figure 99 below shows an example of the channel underrun status being reported on the CGC Interface. In the example the line number is 3 the channel pair is 7 (channels 14 and 15), and the high channel (channel 15) is in underrun while the low channel (channel 14) is in normal mode. These values are encoded into the 4 data words following the format shown in Table 30.

Status is only reported when a channel enters or exits underrun.

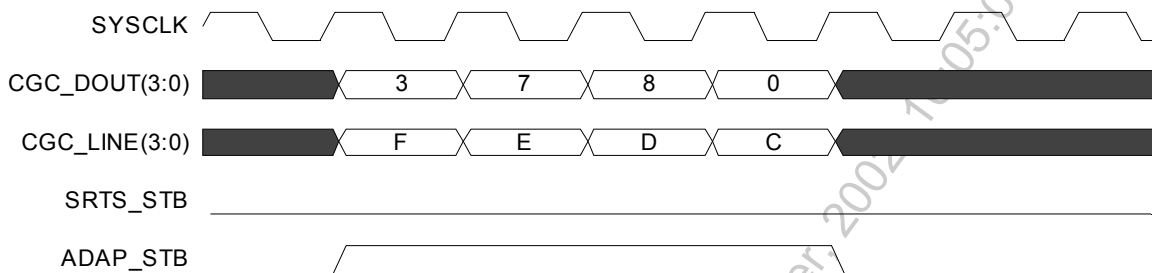**Figure 99   Channel Status Functional Timing**

Table 30   Channel Status

CGC_LINE(3:0) Value	CGC_DOUT(3:0) Value			
	3	**2**	**1**	**0**
15	0	Line(3)	Line(2)	Line(1)
14	Channel(4)	Channel(3)	Channel(2)	Channel(1)
13	underrun_h	0	underrun_l	0
12	0	0	0	0
	0	0	0	0

14.4.3   Adaptive Status Output

The AAL1gator provides the average buffer depth in units of bytes for an external circuit to generate an adaptive clock. (If adap_filt_size is set to zero it will provide the current buffer depth with no averaging). The general mechanism is often termed "buffer centering". A clock delta value is determined externally by subtracting the nominal buffer depth(value of CDVT) from the actual buffer depth. This delta value is then transformed into the frequency selection for an external frequency synthesizer. The closed-loop action of this circuit causes the delta value to find a center point. When the delta is above the center point, there is too much data buffered and the frequency must be increased. When the delta is below the center point, the frequency must be reduced.

The AAL1gator implements a programmable weighted moving average internally. However if an alternative adaptive algorithm is desired then this information can be processed externally. Also in High Speed mode, since an E3 or DS3 clock cannot be internally synthesized, this information can be used for external synthesis of an E3 or DS3 clock.

Figure 100 below shows an example of the CGC Interface for adaptive data. In this example the line number is 5, and the average buffer depth in units of bytes is 43. These values are encoded into the 6 data words following the format shown in Table 31.

Table 31   Frame Difference

CGC_LINE_OUT(3:0) Value	CGC_DOUT(3:0) Value			
	3	2	1	0
5	queue(7)	queue(6)	queue(5)	queue(4)
4	queue(3)	queue(2)	queue(1)	queue(0)
3	0	0	buff_depth(13)	buff_depth(12)
2	buff_depth(11)	buff_depth(10)	buff_depth(9)	buff_depth(8)
1	buff_depth(7)	buff_depth(6)	buff_depth(5)	buff_depth(4)
0	buff_depth(3)	buff_depth(2)	buff_depth(1)	buff_depth(0)

Figure 100   Adaptive Data Functional Timing
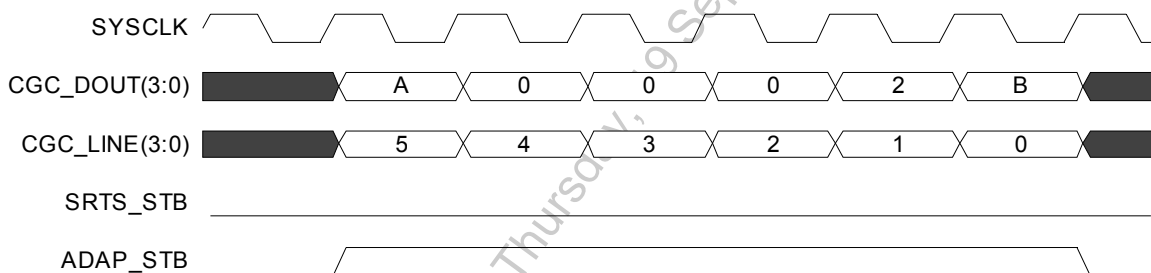
14.5 Ext Freq Select Interface

The Ext Freq Select Interface allows an external source to directly select the line clock frequency from any one of 171 T1 or 240 E1 frequencies centered around the nominal clock rate. For T1 the legal input values are –83 to 88. For E1 the legal input values are –128 to 111. Any values outside of this range will be clamped to these levels. These levels correspond to a +/-200 ppm T1 clock and a +/- 100 ppm E1 clock.

The External Interface block has two input ports that allow an external source to control the frequency synthesizers internal to the CGC. These two ports are CGC_SER_D and CGC_VALID. CGC_SER_D contains the data that selects one of the 171/240 frequencies and CGC_VALID indicates when this data is valid. There should be a rising edge of CGC_VALID at the start of each timing message. And CGC_VALID should go low at the end of each timing message. CGC_VALID only needs to be deactivated for one cycle between timing messages. The CGC block decodes the incoming line number and if the address matches one of its 8 line numbers passes the data on to the appropriate frequency synthesizer.

Figure 101 below shows an example of where Line 19 is being programmed to run with the frequency setting of –79 (Two's complement). See the section on the Frequency Synthesizer block for a discussion of the different frequency settings which range from -83 to 88 in T1 mode and –128 to 111 in E1 mode.

PMC-SIERRA

Figure 101   Ext Freq Select Functional Timing

SYSCLK

LINE NUMBER   FREQUENCY

CGC_SER_IN

CGC_SER_VAL
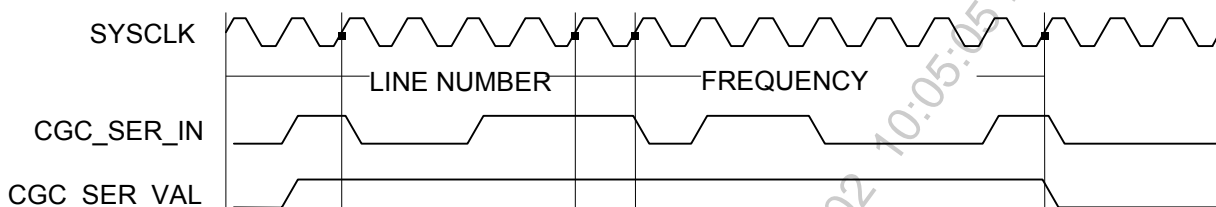
14.6   Line Interface Timing
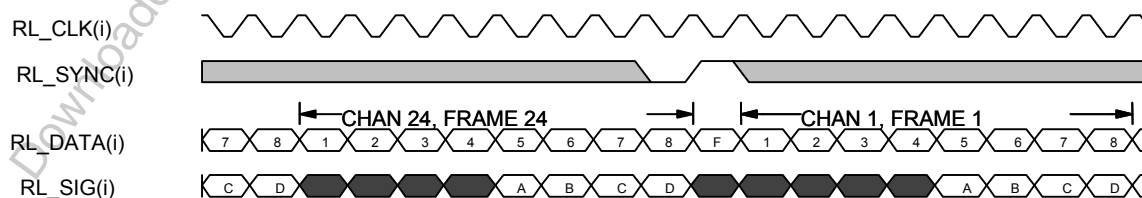
14.6.1   Direct Mode

Receive Line Timing

In T1 mode, the rising edge of RL_SYNC must coincide with the leading edge of the F bit. If RL_SYNC is programmed as an MF pulse, then that edge should also be the leading edge of a multiframe. All input signals are sampled on the falling edge of RL_CLK[i]/CRL_CLK, as shown in the following figure.

Figure 102 and Figure 103 show how the transmitter receives data from the line interface. These lines typically interface with the receive output portion of the corresponding framer. The timing parameters are given in the AC timing section. RL_SYNC is used in structured modes to align the frame and/or multiframe of the incoming data. RL_SYNC can be configured to be a frame sync or a multi-frame sync by the value of MF_SYNC_MODE. This input is ignored in unstructured modes.

In T1 mode the rising edge of RL_SYNC must coincide with leading edge of the F bit. If RL_SYNC is programmed as a MF pulse then that edge should also be the leading edge of a multiframe. All input signals are sampled on the falling edge of RL_CLK[i]/CRL_CLK. as shown in the following figure.

**Figure 102   Receive Line Side T1 Timing(RL_CLK = 1.544 MHz)**

RL_CLK(i)

RL_SYNC(i)

CHAN 24, FRAME 24   CHAN 1, FRAME 1

RL_DATA(i)   7 8 1 2 3 4 5 6 7 8 F 1 2 3 4 5 6 7 8

RL_SIG(i)   C D   A B C D   A B C D

CRL_CLK timing is the same as RL_CLK[i] timing.

In E1 mode the rising edge of RL_SYNC should coincide with the leading edge of the bit 1 of the first channel within a frame. If RL_SYNC is programmed as a MF pulse then that edge should also be the leading edge of a multiframe. All input signals are sampled on the falling edge of RL_CLK[i]/CRL_CLK. as shown in the following figure.

Figure 103   Receive Line Side E1 Timing(RL_CLK = 2.048 MHz)
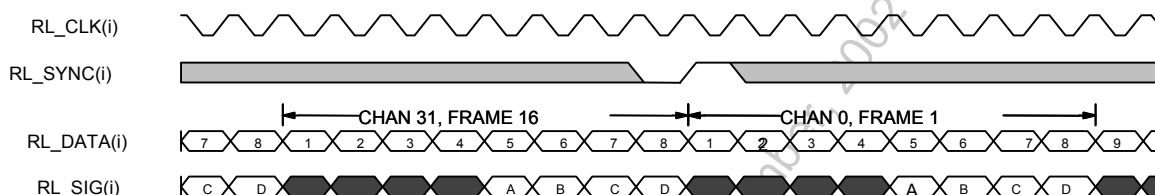
CRL_CLK timing is the same as RL_CLK[i] timing.

When the line is in MVIP-90 mode, as controlled by the MVIP_EN mode bit in the Low Speed Line Configuration Register, a common active low frame pulse is used; F0B. This frame pulse is sampled using the falling edge of the 4 MHz C4Binput clock signal. C4B is also used to clock the data. The AAL1gator-4/8 samples the data provided on RL_SER[n] at the ¾ point of the data bit using the rising edge of C4B. 1 is the most significant bit and 8 is the least significant bit of each octet.

Note that GEN_SYNC is ignored in this mode. F0B must be externally generated and must be only one 4 MHz clock cycle wide.

CAS signaling can be transported by passing it during the last nibble of each time slot.

**Figure 104   MVIP-90 Receive Functional Timing**
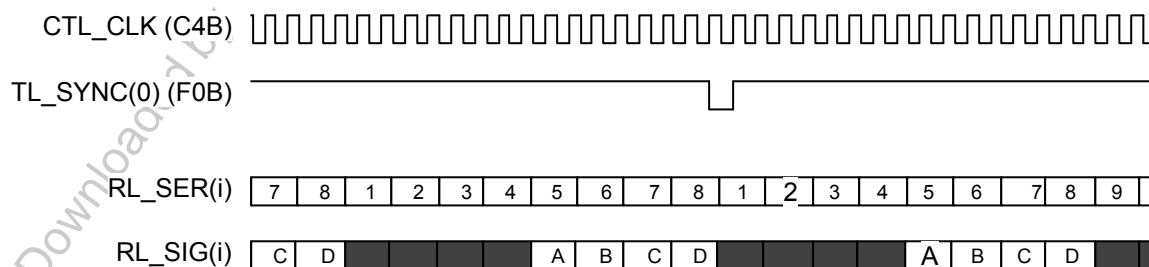
Transmit Line Timing

Figure 105 and

Figure 106, show how the receiver sends data to the line interface. These lines typically interface with the transmit input portion of the corresponding framer. The timing parameters are given in the AC timing section. TL_SYNC is used in structured modes to align the frame and/or multiframe of the incoming data. TL_SYNC can be configured to be a frame sync or a multi-frame sync by the value of MF_SYNC_MODE. This input is ignored in unstructured modes.

In T1 mode the rising edge of TL_SYNC coincides with leading edge of the F bit. If TL_SYNC is programmed as a MF pulse then that edge will also be the leading edge of a multiframe. All output signals are driven on the rising edge of TL_CLK[i]/CTL_CLK. as shown in the following figure.

Figure 105   Transmit Line Side T1 Timing(TL_CLK = 1.544 MHz)
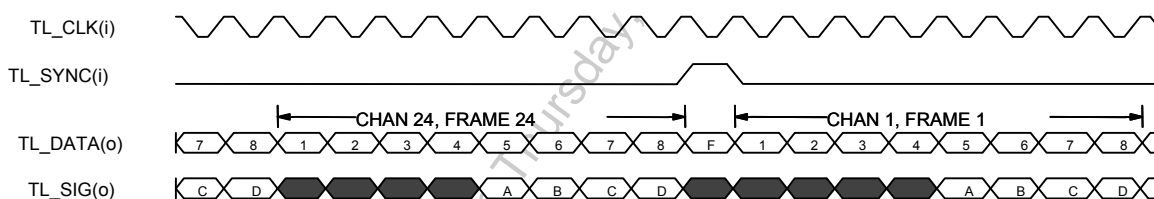
CTL_CLK timing is the same as TL_CLK[i] timing.

In E1 mode the rising edge of TL_SYNC coincides with the leading edge of the bit 1 of the first channel within a frame. If TL_SYNC is programmed as a MF pulse then that edge should also be the leading edge of a multiframe. All input signals are sampled on the falling edge of TL_CLK[i]/CTL_CLK. as shown in the following figure.

Figure 106   Transmit Line Side E1 Timing(TL_CLK = 2.048 MHz)
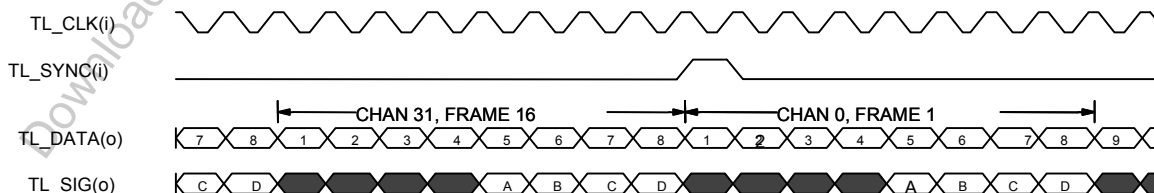
CTL_CLK timing is the same as TL_CLK[i] timing.

When the line is in MVIP-90 mode, as controlled by the MVIP_EN mode bit in the Low Speed Line Configuration Register, a common active low frame pulse is used; F0B. This frame pulse is sampled using the falling edge of the 4 MHz C4B input clock signal.

Note that GEN_SYNC is ignored in this mode. F0B must be externally generated and must be only one 4 MHz clock cycle wide.

The AAL1gator-4/8 updates the data provided on TL_SER[n] on every second falling edge of C4B. The first bit of the next frame is updated on TL_SER on the falling C4B clock edge for which F0B is also sampled low. 1 is the most significant bit and 8 is the least significant bit of each octet.

CAS signaling can be transported by passing it during the last nibble of each time slot.

Figure 107   MVIP-90 Transmit Functional Timing
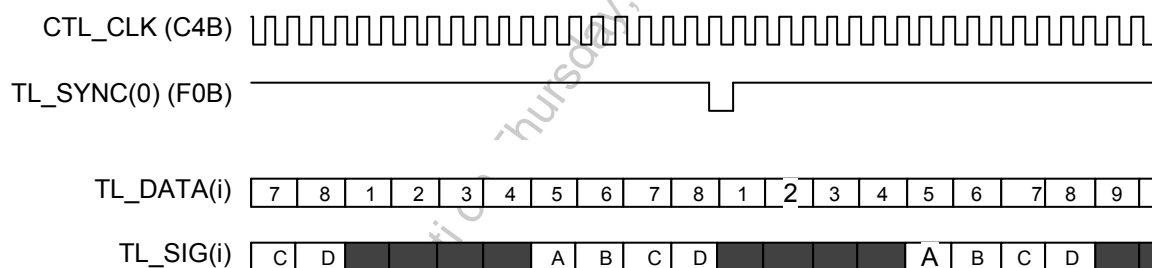
14.6.2 H-MVIP Timing

In H-MVIP mode one 8 Mbps incoming external links are broken into four 2 Mbps internal links. Also four 2 Mbps internal links are combined into one 8 Mbps outgoing external lines.

A common active low frame pulse; F0B; is used to indicate the start of a 128 time slot frame and is shared by all incoming and outgoing lines. The F0B signal is expected to be driven off the rising edge of C4B and is sampled using the falling edge of C4B. The sync signal is used to generate internal RLI_FSYNC and TLI_FSYNC signals for each 2 Mbps link. In addition GEN_SYNC is ignored as the frame pulse is always externally generated from F0B. Due to pipelining delays, the sync signals are offset from the start of frame on the internal links.

PMC-SIERRA

Receive Side Timing.

In H-MVIP mode there is a 16 MHz (C16B) clock that is used to clock in data and a 4 MHz clock (C4B) which is used to clock in the frame pulse. The falling edge of C4B is used to clock in F0B. This falling edge also coincides with the start of reception of the first data bit of the frame. C16B is used by the internal clock mux logic to generate a 2.048 MHz TLI_CLK signal for each internal local link. Data is captured on the second rising edge of C16B that occurs for each data bit.

CAS signaling can be transported by passing it during the last nibble of each time slot.

Figure 108 shows the timing around F0B pulse and the lower two local links within a group of four. Figure 108 shows an expanded view, including all four local links within each group of four. All signals prefixed with "RLI_" are internal signals and are not visible.

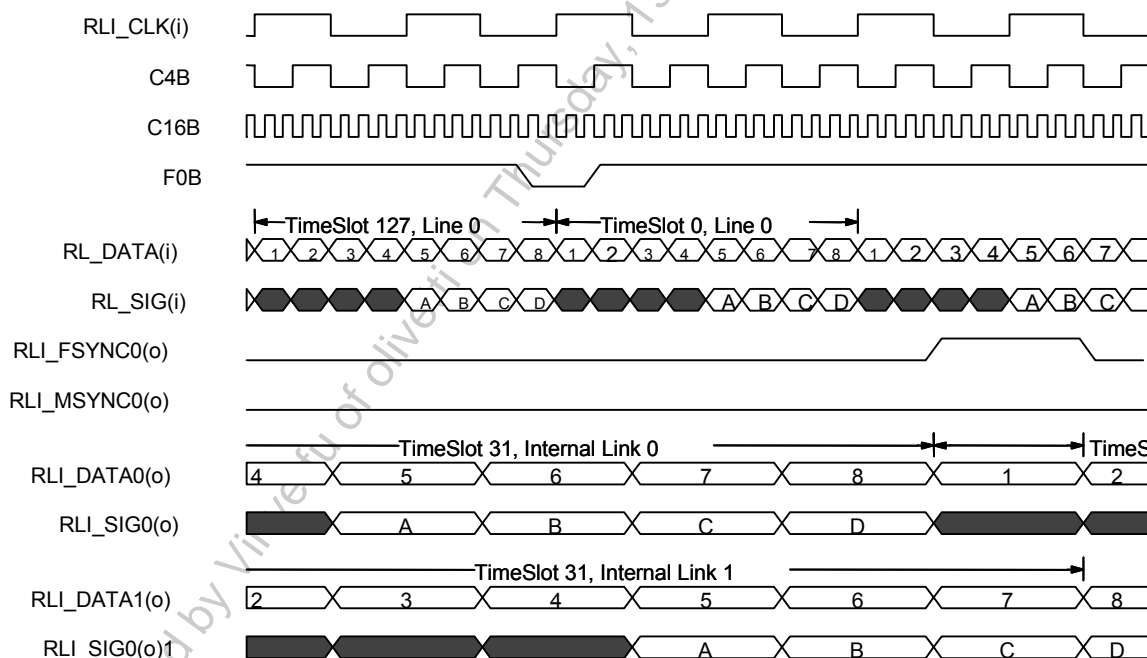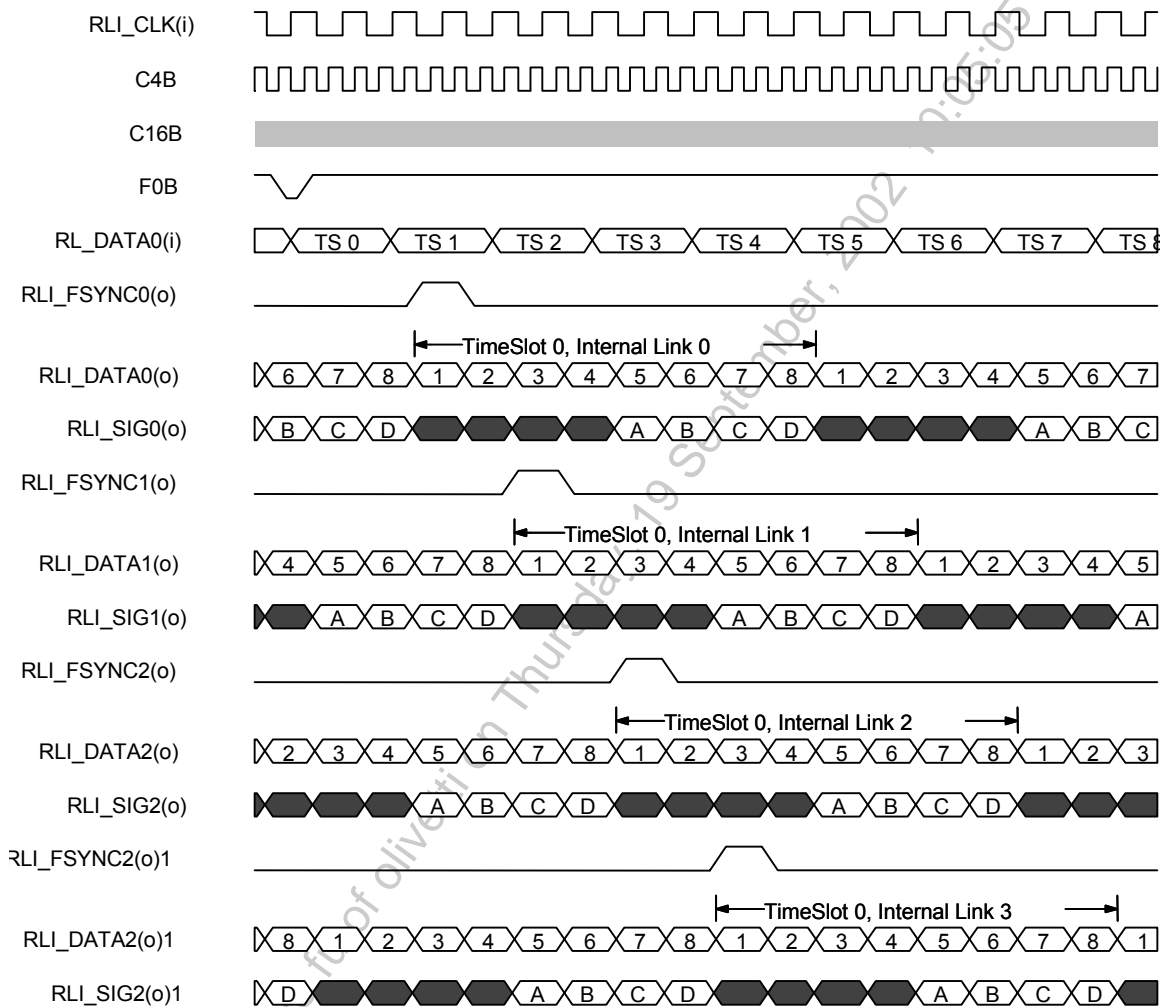**Figure 108   Receive H-MVIP Timing, Close-Up View**

Figure 109   Receive H-MVIP Timing, Expanded View

Transmit Side Timing

In H-MVIP mode, on the transmit side, there is a 16 MHz clock (C16B) that is used to clock out data and a 4 MHz clock (C4B) which is used to clock in the frame pulse. The falling edge of C4B is used to clock in F0B. This falling edge also coincides with the start of transmission of the first data bit of the frame. C16B is used by the internal clock mux logic to generate a 2.048 MHz TLI_CLK signal for each internal local link.

CAS signaling can be transported by passing it during the last nibble of each time slot.

Figure 110 shows the timing around F0B pulse and the lower two local links within a group of four. Figure 110 shows an expanded view, including all four local links within each group of four. All signals prefixed with "TLI_" are internal signals and are not visible.
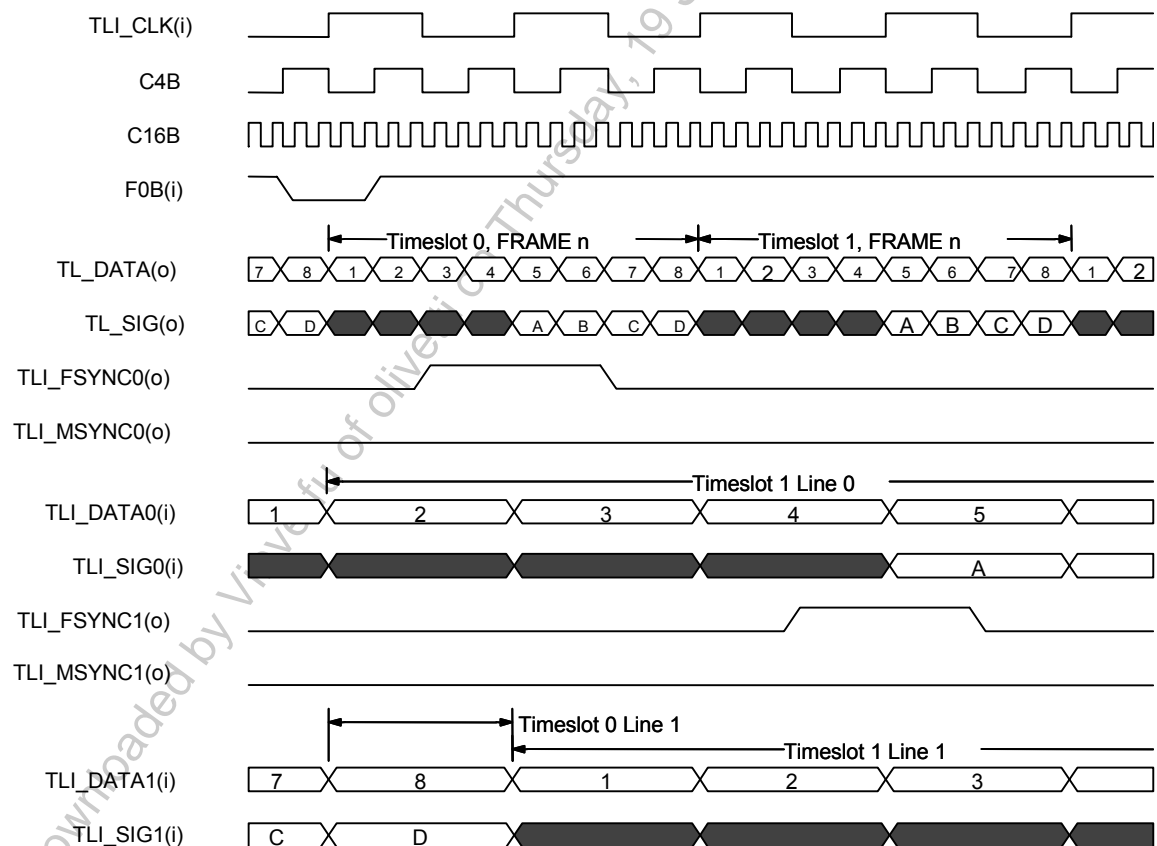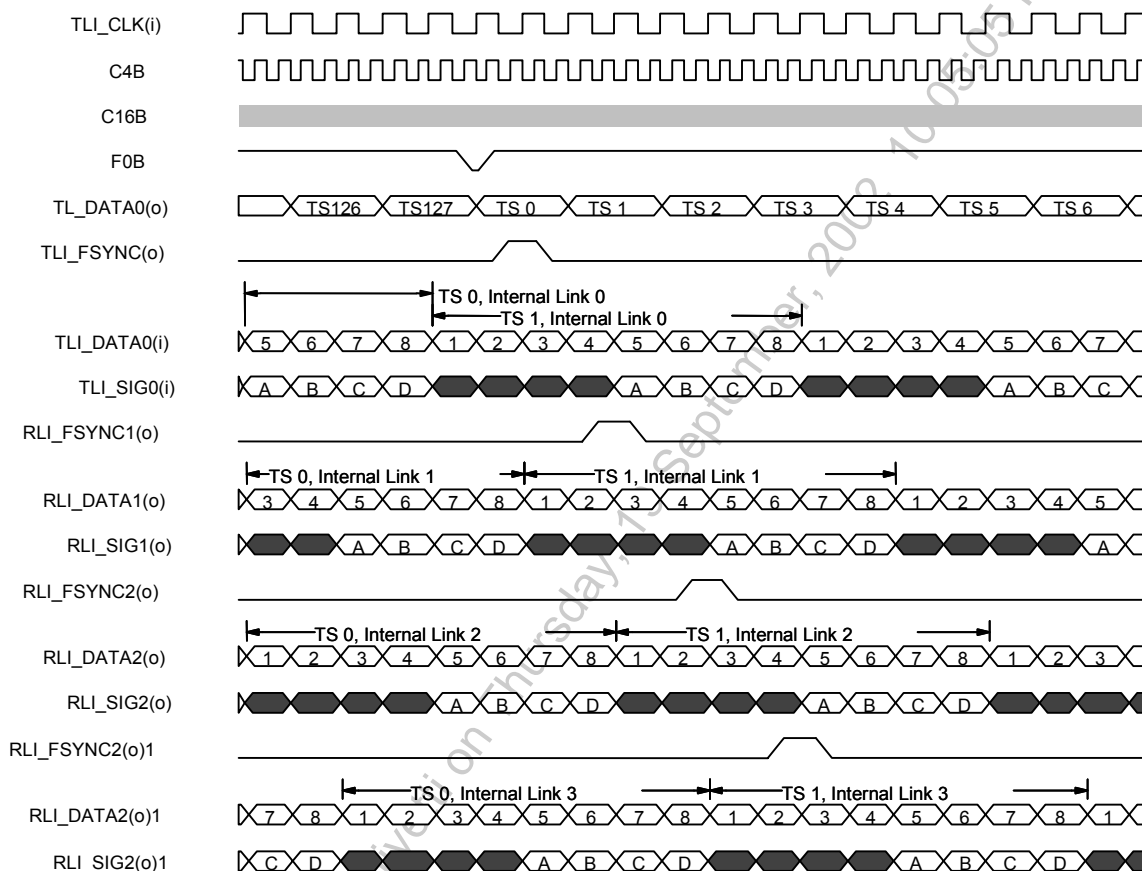
Figure 110 Transmit H-MVIP Timing, Close-up View

Figure 111   Transmit H-MVIP Timing, Expanded View

14.6.3 DS3/E3 Timing

In UDF-HS mode there is no structure and the data is sampled using the falling edge of
RL_CLK as shown in Figure 112. Because of the higher frequency, data should be updated
using the falling edge of RL_CLK instead of the rising edge, which is used in low speed mode.
This allows the full clock cycle to be used instead of half a clock cycle.

Figure 112   Receive High-Speed Functional Timing
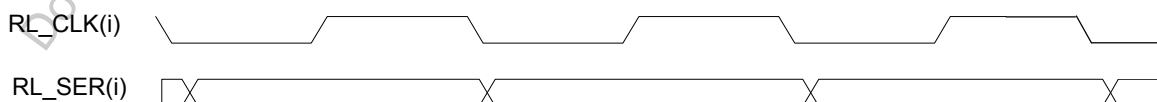
In UDF-HS mode there is no structure and the data is driven using the rising edge of TL_CLK as shown in Figure 113. Data should be latched using the rising edge of TL_CLK so that the full cycle can be used.

Figure 113   Transmit High-Speed Functional Timing
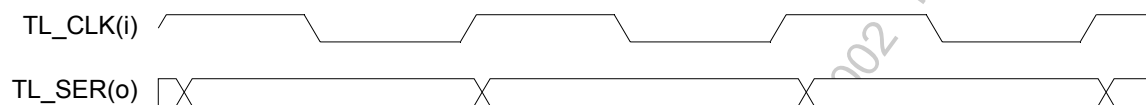
15 Absolute Maximum Ratings

Maximum rating are the worst case limits that the device can withstand without sustaining permanent damage. They are not indicative of normal mode operation conditions.

Table 32   Absolute Maximum Ratings

Ambient Temperature under Bias	-40°C to +85°C
Storage Temperature	-40°C to +125°C
Supply Voltage (+3.3 Volt V$_{DD3.3}$)	-0.3V to +4.6V
Supply Voltage (+2.5 Volt V$_{DD2.5}$)	-0.3V to +3.5V
Voltage on Any Pin	-0.3V to 5.5V
Static Discharge Voltage	±1000 V
Latch-Up Current	±100 mA
DC Input Current	±20 mA
Lead Temperature	+230°C
Absolute Maximum Junction Temperature	+150°C

16 D.C. Characteristics

(T_A = -40°C to +85°C, $V_{DD3.3}$ = 3.0 to 3.6 V, $V_{DD2.5}$ = 2.3 to 2.7 V)

Table 33 AAL1GATOR-4/8 D.C. Characteristics

Symbol	Parameter	Min	Typ	Max	Units	Conditions
VDD3.3	3.3V Power Supply	3.0	3.3	3.6	Volts	Note 5.
VDD2.5	2.5V Power Supply	2.3	2.5	2.7	Volts	Note 5.
VIL	Input Low Voltage	-0.5		0.8	Volts	All non-clock inputs, except TL_CLK. Also not rl_sync[3,2,1] and RSTB and TRSTB
VIH	Input High Voltage	2.0		5.5	Volts	All non-clock inputs, , except TL_CLK. Also not rl_sync[3,2,1] and RSTB and TRSTB
VOL	Output or Bi-directional Low Voltage			0.4	Volts	IOL = -8 mA for the UTOPIA outputs. – -4mA for TDO. –6 mA for everything else. Notes 3, 5.
VOH	Output or Bi-directional High Voltage	2.4			Volts	IOH = 8 mA for the UTOPIA outputs. 4mA for TDO. 6 mA for everything else. Notes 3, 5.
V_{T+}	Schmitt Triggered Input High Voltage	2.0		5.5	Volts	All clock inputs, except TL_CLK. Also rl_sync[3,2,1] and RSTB and TRSTB
V_{T-}	Schmitt Triggered Input Low Voltage	-0.2		0.6	Volts	All clock inputs, except TL_CLK. Also rl_sync[3,2,1] and RSTB and TRSTB
I_{ILPU}	Input Low Current	+10	45	+150	µA	V_{IL} = GND, Notes 1, 3, 5.
I_{IHPU}	Input High Current	-10	0	+10	µA	V_{IH} = V_{DD}, Notes 1, 3
I_{IL}	Input Low Current	-10	0	+10	µA	V_{IL} = GND, Notes 2, 3
I_{IH}	Input High Current	-10	0	+10	µA	V_{IH} = V_{DD}, Notes 2, 3

Symbol	Parameter	Min	Typ	Max	Units	Conditions
C_{IN}	Input Capacitance		5		pF	Excludes package. Package typically 2 pF. Note 5.
C_{OUT}	Output Capacitance		5		pF	All pins. Excludes package. Package typically 2 pF. Note 5.
C_{IO}	Bi-directional Capacitance		5		pF	All pins. Excludes package. Package typically 2 pF. Note 5.
L_{PIN}	Pin Inductance		2		nH	All pins. Note 5.
I_{DDOP} (LS2.5v)	Core Operating Current (Low Speed 2.5v).		72		mA	Direct Mode, Outputs typically loaded. All lines in E1 mode.
IDDOP (LS3.3v)	I/O Operating Current (Low Speed mode 3.3v)		42		mA	Direct Mode, Outputs typically loaded. All lines in E1 mode.
IDDOP (HS2.5v)	Core Operating Current (HS mode 2.5v).		92		mA	HS Mode, Outputs typically loaded. One HS line at 52 MHz.
IDDOP (HS3.3v)	I/O Operating Current (HS mode 3.3v).		47		mA	HS Mode, Outputs typically loaded. One HS line at 52 MHz.

Notes on D.C. Characteristics:

1. Input pin or bi-directional pin with internal pull-up resistor.

2. Input pin or bi-directional pin without internal pull-up resistor.

3. Negative currents flow into the device (sinking), positive currents flow out of the device (sourcing).

4. Input pin or bi-directional pin with internal pull-down resistor.

5. Typical values are given as a design aid. The product is not tested to the typical values given in the data sheet.

17 A.C. Timing Characteristics

(T_A = -40°C to +85°C, $V_{DD3.3}$ = 3.0 to 3.6 V, $V_{DD2.5}$ = 2.3 to 2.7 V)

Notes on Input Timing:

1. When a set-up time is specified between an input and a clock, the set-up time is the time in nanoseconds from the 1.4 Volt point of the input to the 1.4 Volt point of the clock.

2. When a hold time is specified between an input and a clock, the hold time is the time in nanoseconds from the 1.4 Volt point of the clock to the 1.4 Volt point of the input.

3. It is recommended that the transition time on all clock inputs is less than 15 ns.

Notes on Output Timing:

1. Output propagation delay time is the time in nanoseconds from the 1.4 Volt point of the reference signal to the 1.4 Volt point of the output.

2. Maximum and minimum output propagation delays are measured with a 100 pF load on all the outputs for the UTOPIA interface, and 50 pF load on microprocessor outputs, and 25 pF on every other output.

3. Output tristate delay is the time in nanoseconds from the 1.4 Volt point of the reference signal to the point where the total current delivered through the output is less than or equal to the leakage current.
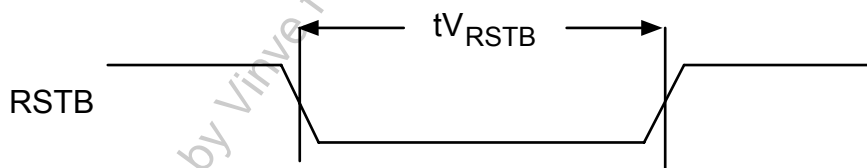
17.1 Reset Timing

Table 34   RTSB Timing

Symbol	Description	Min	Max	Units
tVRSTB	RSTB Pulse Width	64		SYS_CLK cycles*
*SYS_CLK must cycle through at least 64 rising edges while RSTB=0.				

Figure 114   RSTB Timing

17.2 SYS_CLK Timing

Table 35   SYS_CLK Timing

Symbol	Description	Min	Max	Units
fSYS	SYS_CLK Frequency (See notes below)	25	45	MHz
dSYS	SYS_CLK Duty Cycle	40	60	

Notes on SYS_CLK Timing:

1. Inputs are latched on rising edge of SYS_CLK. Outputs are driven off rising edge of SYS_CLK.

2. If any internal TL_CLK synthesizer is used, SYS_CLK must be 38.88 MHz +/- 50 ppm.

3. If no internal TL_CLK synthesizer is used, the SYS_CLK tolerance can be relaxed to +/- 200 ppm.

4. If it is desired that the internally synthesized TL_CLK is locked to a network clock, then SYS_CLK needs to be locked to that network clock.

5. To maintain sufficient bandwidth on all lines, SYS_CLK must be at least 38.88 MHz. For each line that is not used on the most loaded A1SP, the minimum frequency can be decreased by 4.5 MHz, if the internal clock synthesizers are not used.

6. The SYS_CLK minimum frequency is due to the internal DLL.

Figure 115   SYS_CLK Timing

SYSCLK

17.3   NCLK Timing

Table 36   NCLK Timing

Symbol	Description	Min	Max	Units
fNCLK	NCLK Frequency (See notes below)		77.76 +50ppm	MHz
dNCLK	NCLK Duty Cycle	40	60	

Notes on NCLK Timing:

1. If DS3 is used NCLK should be 77.76 +/- 50 ppm.

2. An internal divider is available per A1SP, which can divide down the NCLK frequency. The internal NCLK frequency needs to be 2.43 MHz +/- 50 ppm for E1 and T1, 38.88 MHz +/- 50 ppm for E3, and 77.76 MHz +/-50 ppm for DS3.

Figure 116   NCLK Timing

NCLK

17.4   Microprocessor Interface Timing Characteristics

Table 37   Microprocessor Interface Read Access

Symbol	Parameter	Min	Max	Units
tSAR	Address to Valid Read Set-up Time	10		ns
tHAR	Address to Valid Read Hold Time	5		ns
tSALR	Address to Latch Set-up Time	10		ns
tHALR	Address to Latch Hold Time	10		ns
tVL	Valid Latch Pulse Width	5		ns
tSLR	Latch to Read Set-up	0		ns
tHLR	Latch to Read Hold	5		ns
tPACKL	Valid Read to Valid ACKB Propagation Delay	3	400*	SYS CLK cycles
tZRD	Valid Read Negated to Output Tri-state		20	ns
tZACKH	Valid Read Negated to ACK Tri-state		20	ns
tSRD	Data to Valid ACKB Setup time	5		ns
* Microprocessor may momentarily experience excessive delays when accessing the external SSRAM, but will average 10-15 SYSCLK cycles. Microprocessor accesses to internal registers will not experience these excessive delays.				

Figure 117   Microprocessor Interface Read Timing
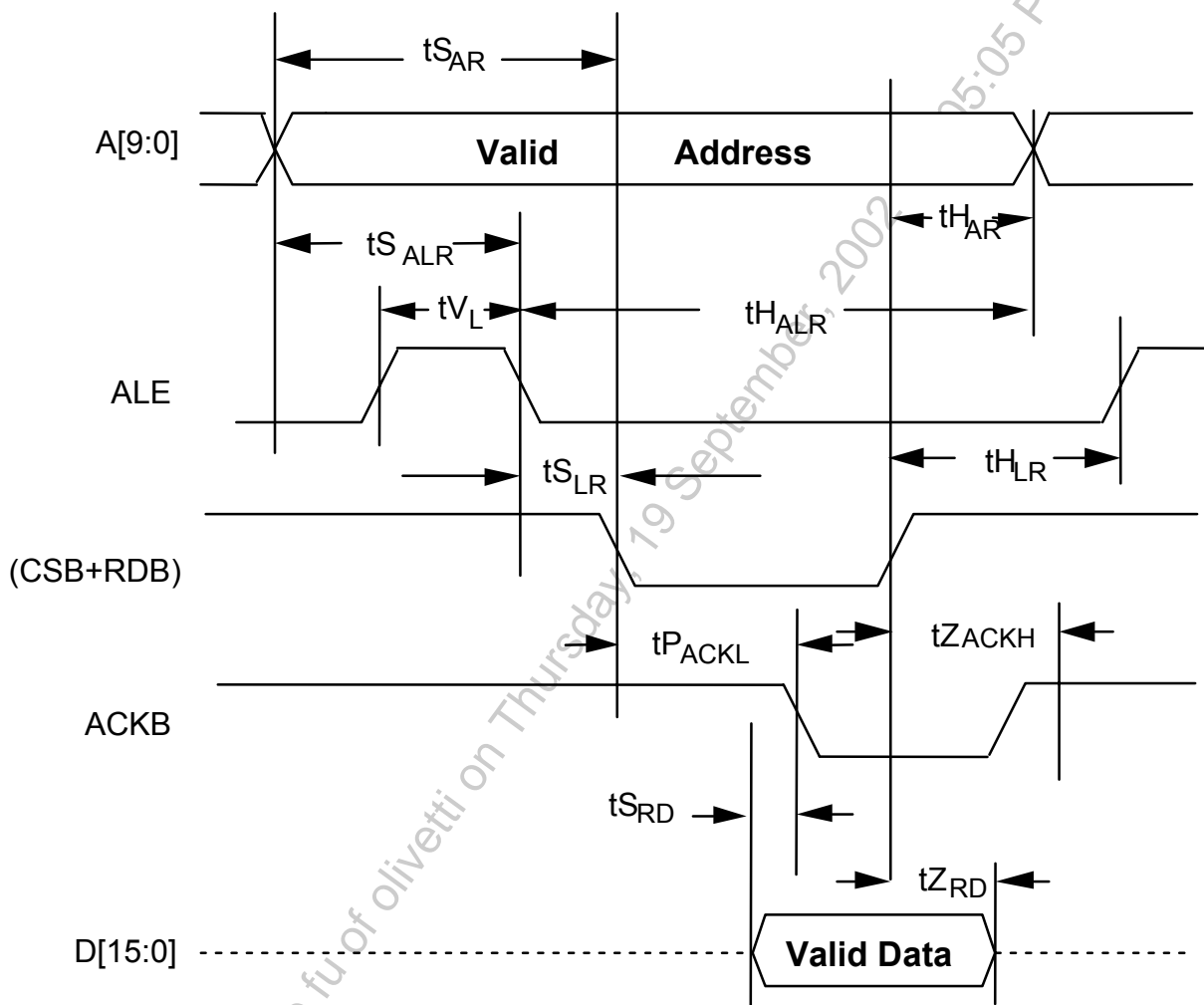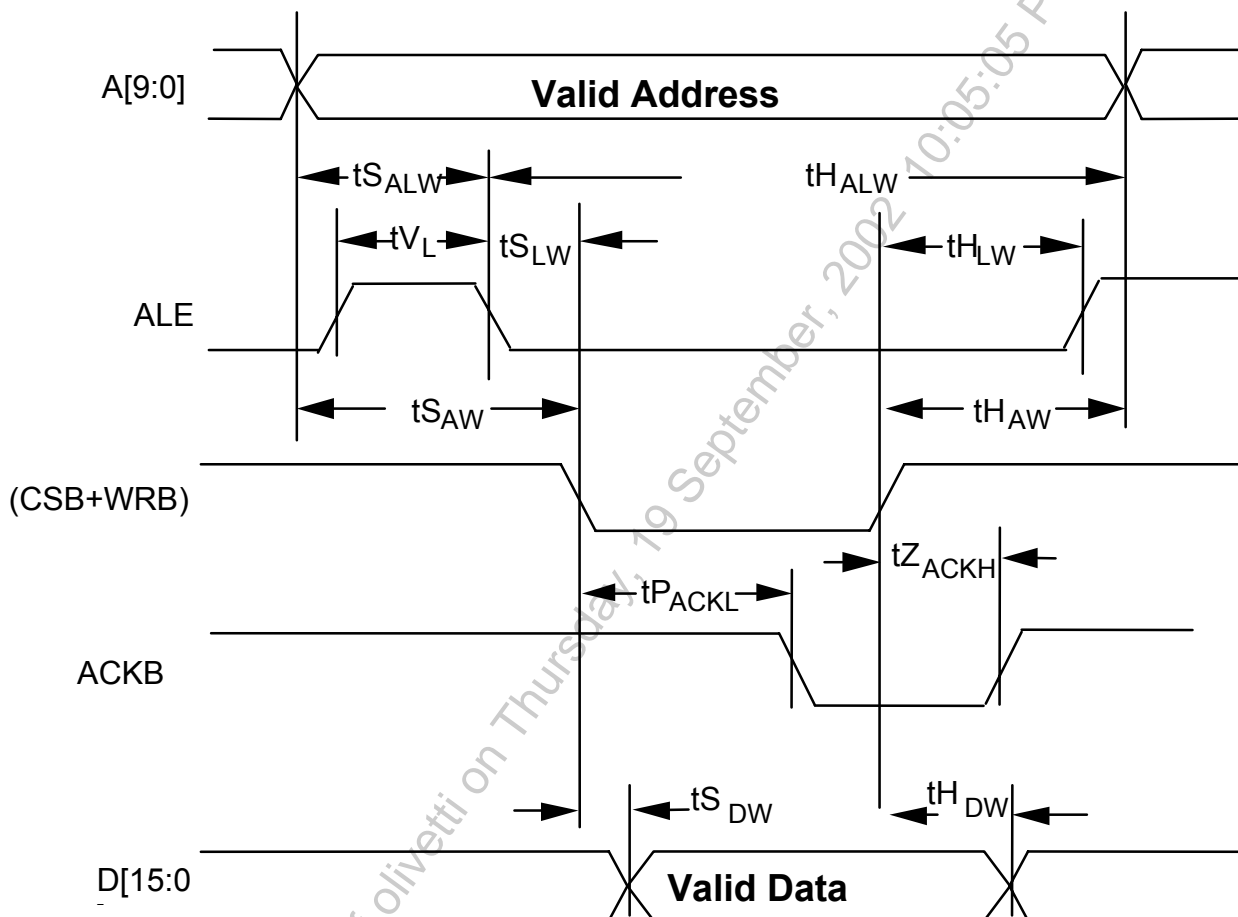

Notes on Microprocessor Interface Read Timing:

1. Output propagation delay time is the time in nanoseconds from the 1.4 Volt point of the reference signal to the 1.4 Volt point of the output.

2. Maximum output propagation delays are measured with a 50 pF load on the Microprocessor Interface data bus, (D[15:0]).

3. A valid read cycle is defined as a logical OR of the CSB and the RDB signals.

4. In non-multiplexed address/data bus architectures, ALE should be held high so parameters tSALR, tHALR, tVL, and tSLR are not applicable.

5. Parameter tHAR is not applicable if address latching is used.

6. Read Timing is dependent upon the region read and the SYS_CLK frequency. With a 40 MHZ SYS_CLK, Registers reads typically complete within 70 ns. Internal memory table reads typically complete within 110 ns, and external memory reads typically complete within 180ns to 200 ns. However memory accesses can take up to 10 us if there is a lot of contention.

7. WRB must be high during reads.

8. Because CSB, WRB, RDB are asynchronously sampled, the CSB or (RDB and WRB) signals should be deactivated for 2 SYS_CLK cycles in between each access.

Table 38   Microprocessor Interface Write Access

Symbol	Parameter	Min	Max	Units
TSAW	Address to Valid Write Set-up Time	10		ns
TSDW	Write active to Data Valid Set-up Time		10	ns
TSALW	Address to Latch Set-up Time	10		ns
THALW	Address to Latch Hold Time	10		ns
TVL	Valid Latch Pulse Width	5		ns
TSLW	Latch to Write Set-up	0		ns
THLW	Latch to Write Hold	5		ns
THDW	Data to Valid Write Hold Time	5		ns
THAW	Address to Valid Write Hold Time	5		ns
TPACKL	Valid Write to Valid ACKB Propagation Delay	3	400	SYS_CLK cycles
TZACKH	Valid Write Negated to Output Tri-state		20	ns
* Microprocessor may momentarily experience excessive delays when accessing the external SSRAM, but will average 10-15 SYSCLK cycles. Microprocessor accesses to internal registers will not experience these excessive delays.				

Figure 118 Microprocessor Interface Write Timing


Notes on Microprocessor Interface Write Timing:

1. A valid write cycle is defined as a logical OR of the CSB and the WRB signals. These signals must be held active until ACKB goes low.

2. In non-multiplexed address/data bus architectures, ALE should be held high so parameters tSALW, tHALW, tVL, tSLW and tHLW are not applicable.

3. Parameter tHAW is not applicable if address latching is used.

4. Write Timing is dependent upon the region read and the SYS_CLK frequency. With a 40 MHZ SYS_CLK, Registers writes typically complete within 70 ns. Internal memory table writes typically complete within 110 ns, and external memory writes typically complete within 180ns to 200 ns. However memory accesses can take up to 10 us if there is a lot of contention.

5. Data is sampled internally one SYS_CLK cycle after CSB and WRB are detected low.

6. RDB must be high during writes.

7. Because CSB, WRB, RDB are asynchronously sampled, the CSB or (RDB and WRB) signals should be deactivated for 2 SYS_CLK cycles in between each access.

17.5   External Clock Generation Control Interface

Table 39   External Clock Generation Control Interface

Symbol	Description	Min	Max	Units
fSCLK	SYS_CLK Frequency (See notes below)	25	45	MHz
DSCLK	SYS_CLK Duty Cycle	40	60	%
tS	Input Set-up time to SYS_CLK	4		ns
tH	Input Hold time to SYS_CLK	1		ns
tP	SYS_CLK High to Output Valid	1	12	ns

**Figure 119   External Clock Generation Control Interface Timing**
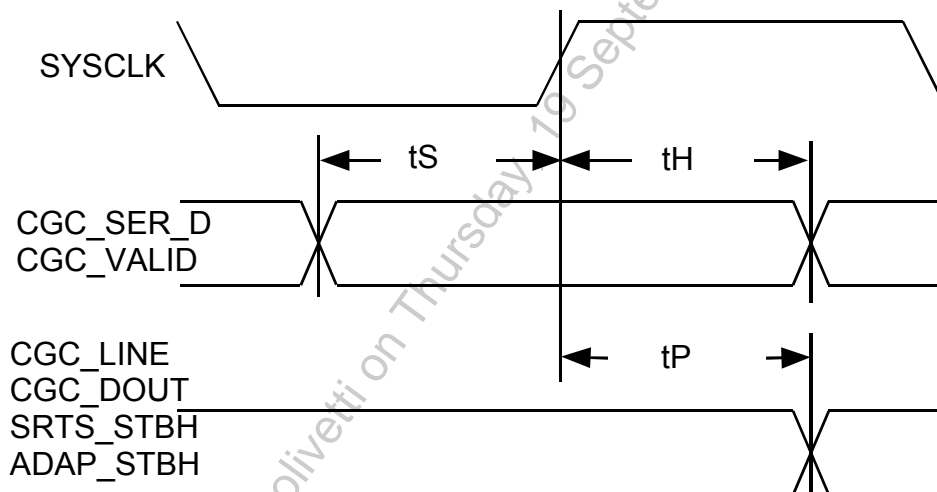
17.6   RAM Interface

Table 40   RAM Interface

Symbol	Description	Min	Max	Units
FSCLK	SYS_CLK, Frequency	25	45	MHz
DSCLK	SYS_CLK Duty Cycle	40	60	%
tS	Input Set-up time to SYS_CLK	4		ns
tH	Input Hold time to SYS_CLK	1		ns
tP	SYS_CLK High to Output Valid	1.5	12	ns
tZ	SYS_CLK High to Output High-Impedance	1.5	12	ns
tZB	SYS_CLK High to Output Driven	1.5	12	ns

Figure 120   RAM Interface Timing
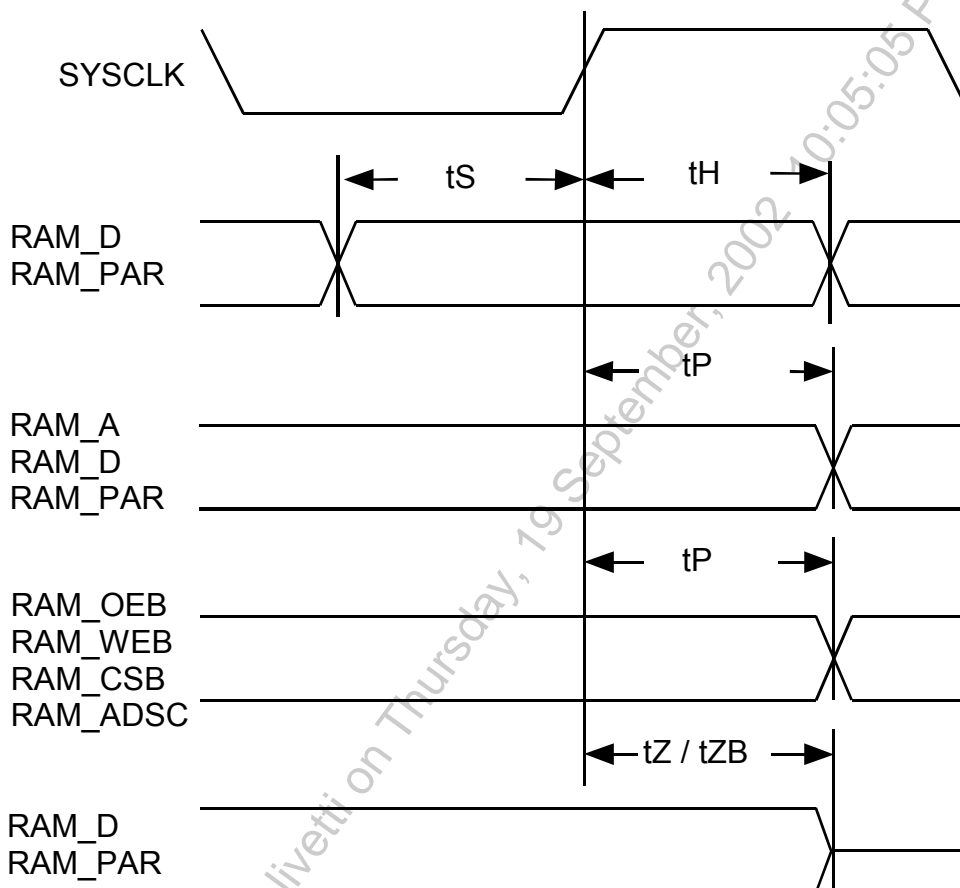
17.7 Utopia Interface

Table 41   UTOPIA Source and Sink Interface

Symbol	Description	Min	Max	Units
F	TATM_CLK/RATM_CLK Frequency	10	52	MHz
tD	TATM_CLK/RATM_CLK Duty Cycle	40	60	%
tS	Input Set-up time to TATM_CLK/RATM_CLK	3.5		ns
tH	Input Hold time to TATM_CLK/RATM_CLK	0.5		ns
tP	TATM_CLK/RATM_CLK High to Output Valid	2	14 / 11.5*	ns
tZ	TATM_CLK/RATM_CLK High to Output High-Impedance	2	14 / 11.5*	ns
tZB	TATM_CLK/RATM_CLK High to Output Driven	2	14 / 11.5*	ns

* First number is with 100 pF load, Second number is with 50 pF load.
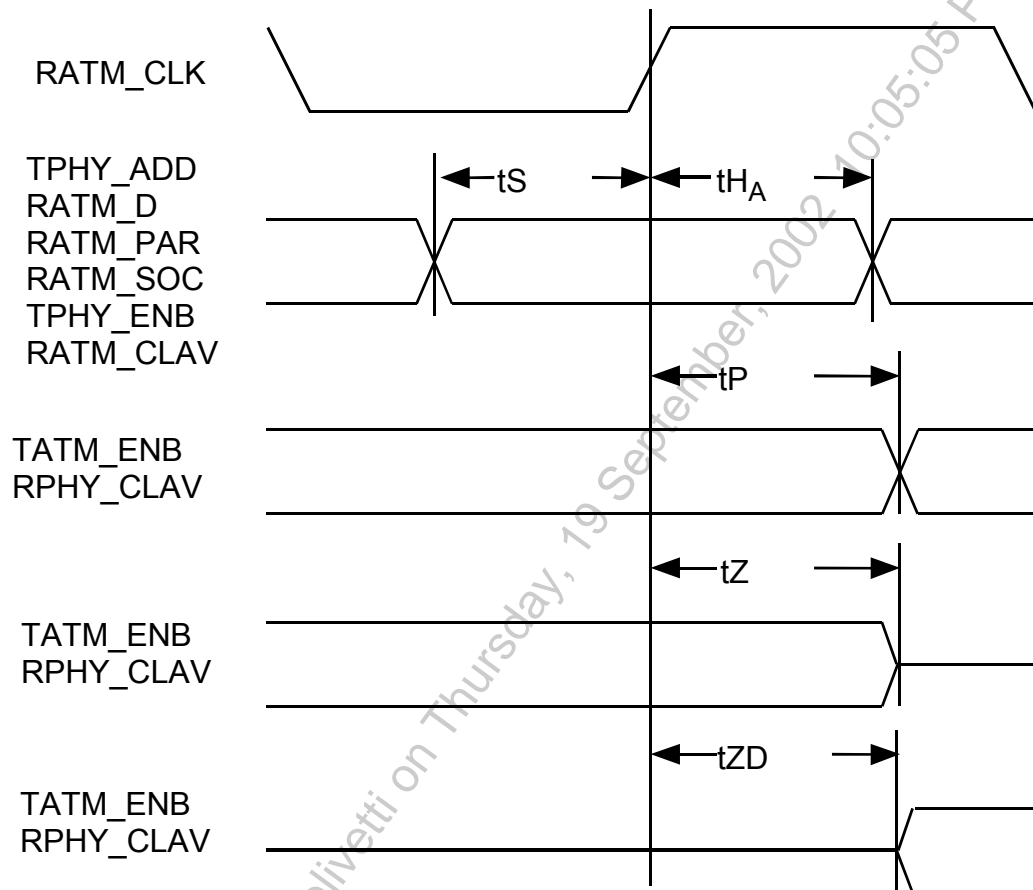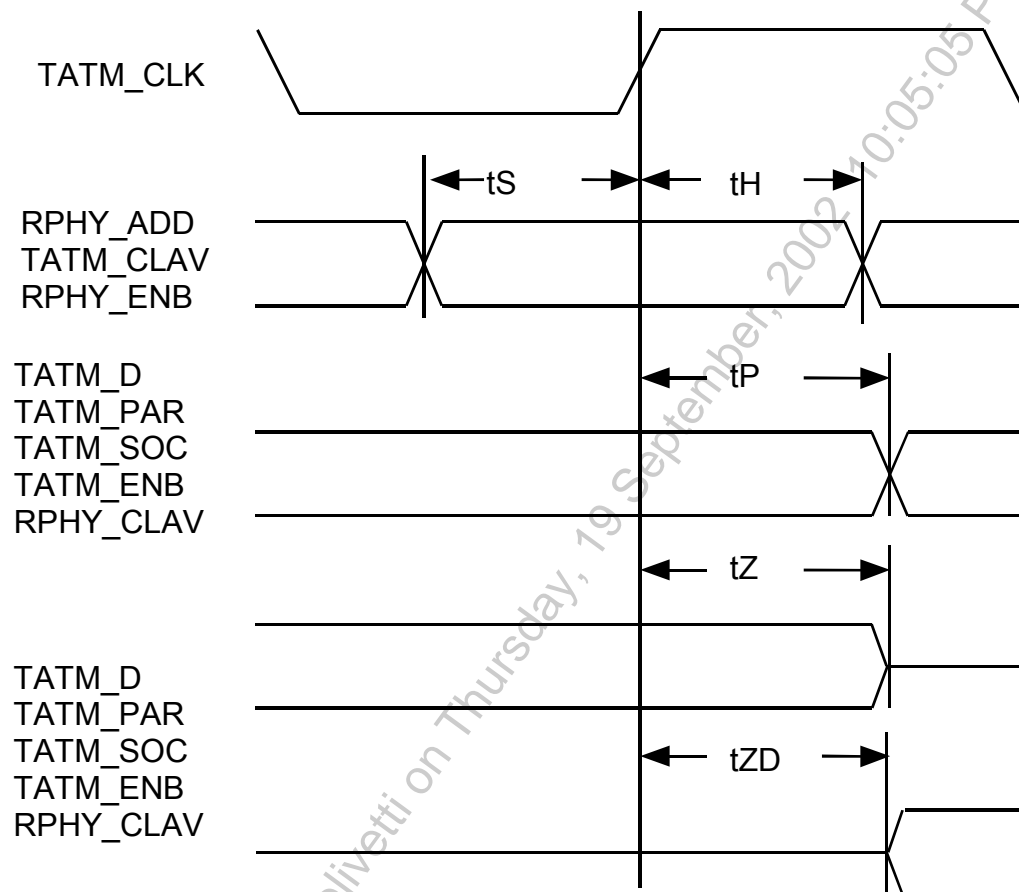
Figure 121   Sink UTOPIA Interface Timing

PMC-SIERRA

**Figure 122   Source UTOPIA Interface Timing**

17.8   LINE I/F Timing

17.8.1 Direct Timing

Transmit

Table 42   Transmit Low Speed Interface Timing

Symbol	Description	Min	Max	Units
f_T	TL_CLK Frequency	0	15	MHz
d_T	TL_CLK Duty Cycle	40	60	%
tS	Input Set-up time to falling edge of TL_CLK	10		ns
tH	Input Hold time from falling edge TL_CLK	10		ns
tP	TL_CLK High to Output Valid	3	15	ns

Figure 123   Transmit Low Speed Interface Timing
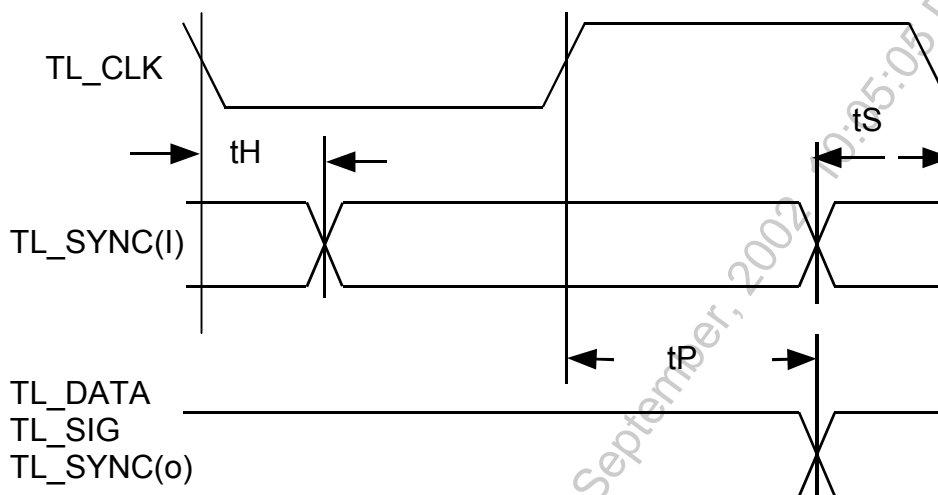
Notes on Transmit Low Speed Timing:

1. Outputs are driven using the rising edge of TL_CLK and inputs are expected to be driven using the rising edge of TL_CLK.

2. Inputs are latched using the falling edge of TL_CLK.

3. The maximum frequency per line in low speed mode is 15 MHz. However aggregate speed for all lines within an A1SP must be 20 MHz or less with a SYS_CLK of 38.88 MHz.

Receive

Table 43   Receive Low Speed Interface Timing

Symbol	Description	Min	Max	Units
f_R	RL_CLK Frequency (See note 3)	0	15	MHz
d_R	RL_CLK Duty Cycle	40	60	%
tS	Input Set-up time to falling edge of RL_CLK	10		ns
tH	Input Hold time from falling edge RL_CLK	10		ns

Notes on Receive Low Speed Timing:

1. Inputs are expected to be driven using the rising edge of RL_CLK.

2. Inputs are latched using the falling edge of RL_CLK.

3. The maximum frequency per line in low speed mode is 15 MHz. However aggregate speed for all lines within an A1SP must be 20 MHz or less with a SYS_CLK of 38.88 MHz.
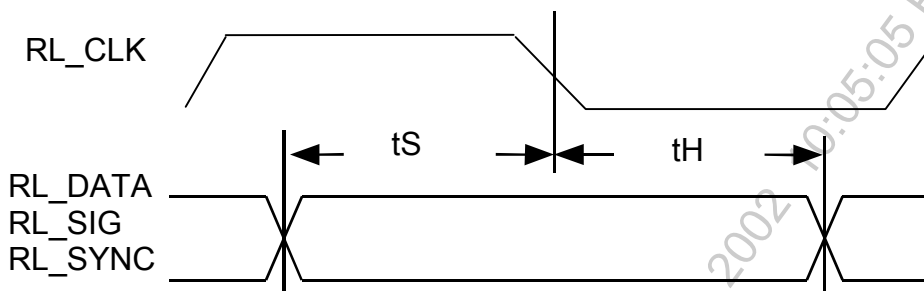
4. For applications which use SRTS clock recovery, the RL_CLK must not be a gapped clock and jitter should be less than 0.3 UI.

Figure 124   Receive Low Speed Interface Timing

17.8.2 H-MVIP Timing

Table 44   H-MVIP Sink Timing

Symbol	Description	Min	Max	Units
	C16B Frequency (See Note 1)	16.368	16.400	MHz
	C16B Duty Cycle	40	60	%
	C4B Frequency (See Note 2)	4.092	4.100	MHz
	C4B Duty Cycle	40	60	%
t_{PC16C4}	C16B to C4B skew	-10	10	ns
t_{SC16B}	RL_DATA, RL_SIG Set-Up Time to rising edge of C16B	5		ns
t_{HC16B}	RL_DATA, RL_SIG Hold Time from rising edge of C16B	5		ns
t_{SC4B}	F0B Set-Up Time to falling edge of C4B	5		ns
t_{HC4B}	F0B Hold Time from falling edge of C4B	5		ns

Notes on H-MVIP Sink Timing:

1. Measured between any two C16B falling edges.

2. Measured between any two C4B falling edges.
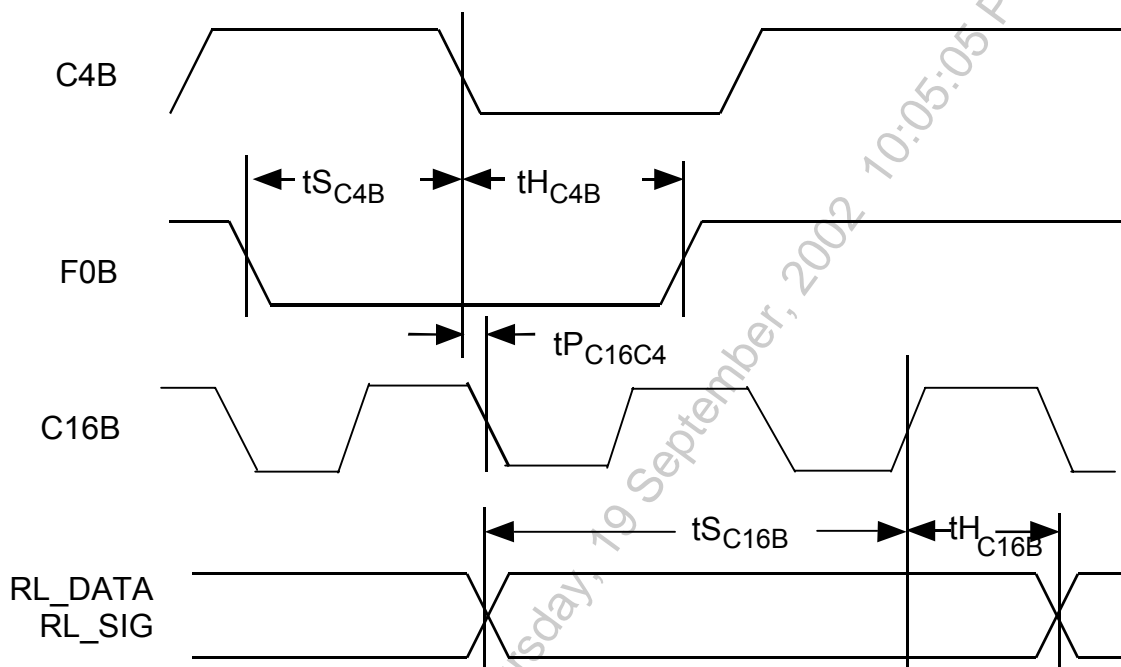
Figure 125   H-MVIP Sink Data & Frame Pulse Timing

Table 45   H-MVIP Source Timing

Symbol	Description	Min	Max	Units
$^t P_{C16B}$	C16B Low to TL_DATA, TL_SIG valid	3	15	ns

Notes on H-MVIP Source Timing:

1. Outputs are driven off the falling edge of C16B and are held for two C16B clock cycles

Figure 126   H-MVIP Ingress Data Timing
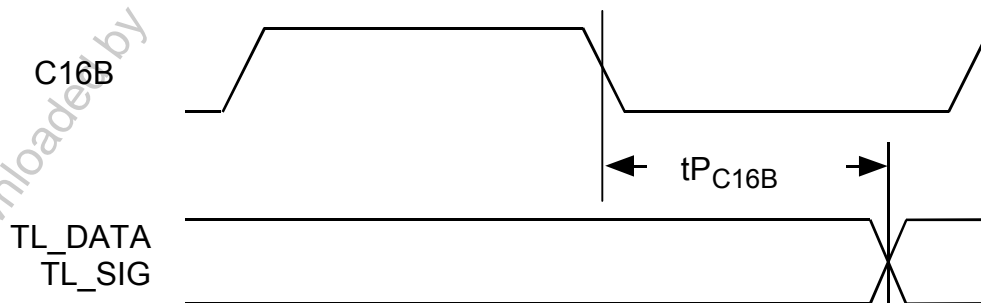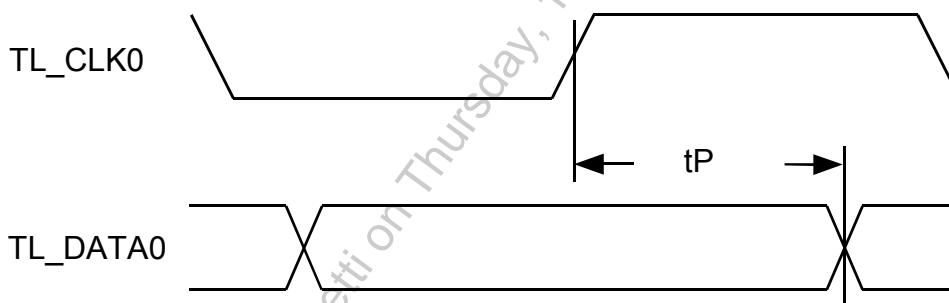
17.8.3 High Speed Timing

Transmit

Table 46   Transmit High Speed Interface Timing

Symbol	Description	Min	Max	Units
f_T	TL_CLK0 Frequency (see note 2)	0	45	MHz
d_T	TL_CLK0 Duty Cycle	40	60	%
tP	TL_CLK0 High to Output Valid	3	13	ns

Notes on Transmit High Speed Timing:

1. Outputs are driven using the rising edge of TL_CLK.

2. The maximum frequency per line in High Speed mode is 45 MHz when SYS_CLK is 38.88 MHz. However if SYS_CLK is 45 MHz then the maximum frequency for TL_CLK0 is 52 MHz.

Figure 127   Transmit High Speed Timing

Receive

Table 47   Receive High Speed Interface Timing

Symbol	Description	Min	Max	Units
f_R	RL_CLK0 Frequency	0	45	MHz
d_R	RL_CLK0 Duty Cycle	40	60	%
tS	Input Set-up time to falling edge of RL_CLK0	5		ns
tH	Input Hold time from falling edge RL_CLK0	1		ns

Notes on Receive High Speed Timing:

1. Inputs are latched using the falling edge of RL_CLK.

2. The maximum frequency per line in High Speed mode is 45 MHz when SYS_CLK is 38.88 MHz. However if SYS_CLK is 45 MHz then the maximum frequency for RL_CLK0 is 52 MHz.

3. For applications which use SRTS clock recovery, the RL_CLK must not be a gapped clock and jitter should be less than 0.3 UI.

PMC-SIERRA

Figure 128   Receive High Speed Interface Timing
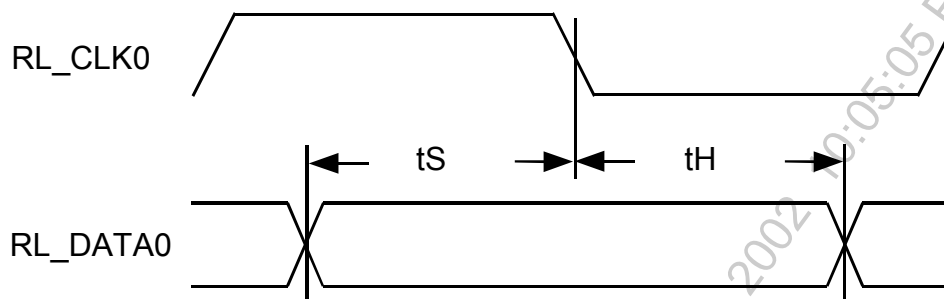
17.9   JTAG Timing

Table 48   JTAG Port Interface

Symbol	Description	Min	Max	Units
	TCK Frequency		1	MHz
	TCK Duty Cycle	40	60	%
t_STMS	TMS Set-up time to TCLK	50		ns
t_HTMS	TMS Hold time to TCLK	50		ns
t_STDI	TDI Set-up time to TCLK	50		ns
t_HTDI	TDI Hold time to TCLK	50		ns
t_PTDO	TCLK Low to TDO Valid	2	50	ns
t_VTRSTB	TRSTB Pulse Width	100		ns

Figure 129   JTAG Port Interface Timing

Proprietary and Confidential to PMC-Sierra, Inc., and for its customers' internal use.
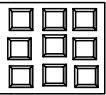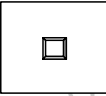Document No.: PMC-2000098 Issue 1

294
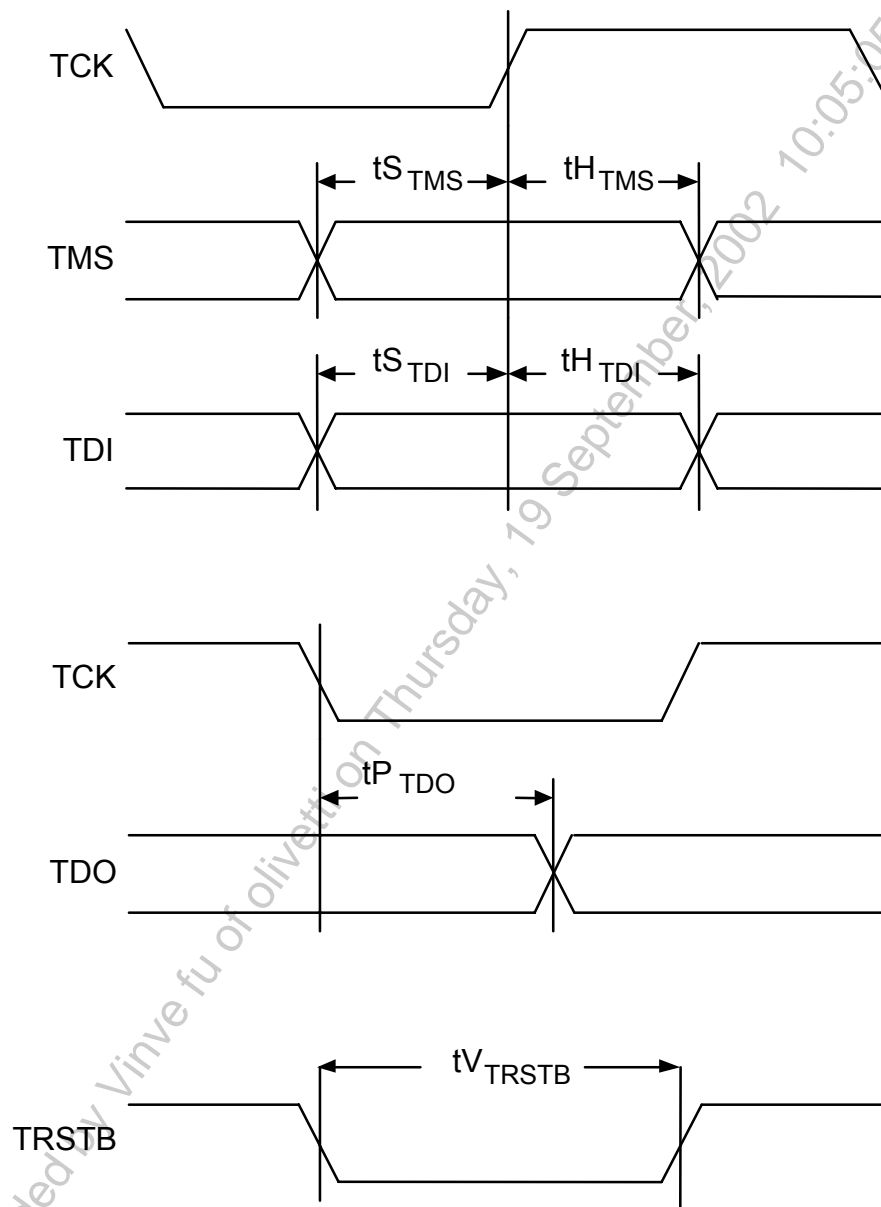
18 Ordering and Thermal Information

Table 49 AAL1GATOR-4/8 (PM73123/4) Ordering Information

Part No.	Description
PM73123-PI (AAL1gator-8)	324 Pin Plastic Ball Grid Array (PBGA)
PM73124-PI (AAL1gator-4)	324 Pin Plastic Ball Grid Array (PBGA)

Table 50 AAL1GATOR-4/8 (PM73123/4) Thermal Information

PART NO.	CASE TEMPERATURE	Theta Ja	Theta Jc
PM73123-PI, PM73124-PI	-40°C to +85°C	42°C/W	4.67

			Forced Air (Linear Feet per Minute)				
Theta JA @ 0.5 watts		Conv	100	200	300	400	500
Dense Board		41.6	37.3	34.2	32.2	30.8	29.9
JEDEC Board		23.2	21.5	20.3	19.5	18.9	18.4

1. DENSE Board is defined as a 3S3P board and consists of a 3x3 array of device PM73123-PI located as close to each other as board design rules allow. All PM73123-PI devices are assumed to be dissipating 0.5 Watts. Θ_{JA} listed is for the device in the middle of the array.

2. JEDEC Board Θ_{JA} is the measured value for a single thermal device in the same package on a 2S2P board following EIA/JESD 51-3

![PMC-SIERRA logo]

19   Mechanical Information

324 PIN PBGA -23x23 MM BODY - (P SUFFIX)

NOTE: Only 4 Layer Package is used.

TOP VIEW

BOTTOM VIEW

SIDE VIEW
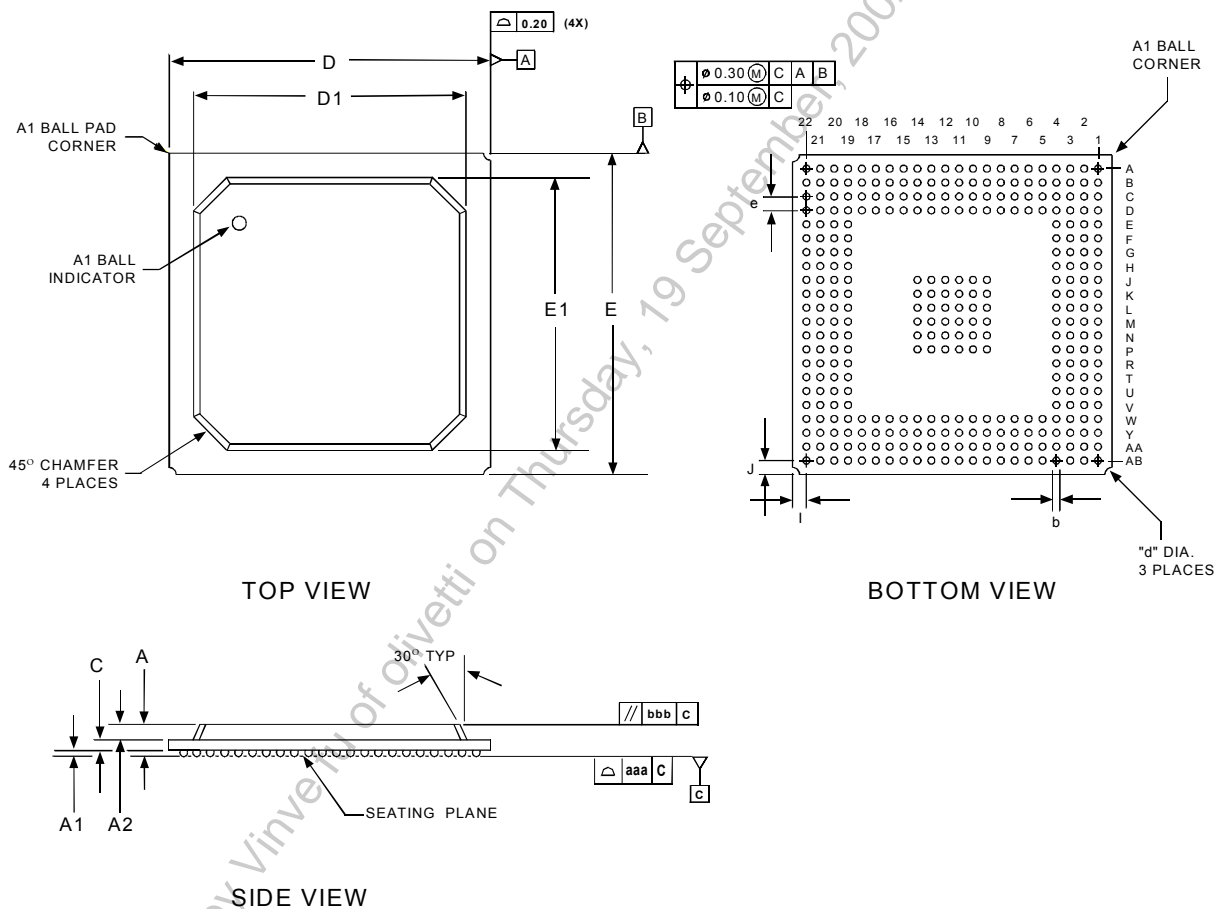
NOTES: 1) ALL DIMENSIONS IN MILLIMETER.
2) DIMENSION aaa DENOTES COPLANARITY.
3) DIMENSION bbb DENOTES PARALLEL.

PACKAGE TYPE : 324 PLASTIC BALL GRID ARRAY - PBGA																
BODY SIZE : 23 x 23 x 2.28 MM (4 layer)																
Dim.	A (2 layer)	A (4 layer)	A1	A2	D	D1	C (2 layer)	C (4 layer)	E	E1	I	J	b	d	e	aaa	bbb
Min	1.82	2.07	0.40	1.12	-	19.00	0.30	0.55	-	19.00	-	-	0.50	-	-	-	-
Nom.	2.03	2.28	0.50	1.17	23.00	19.50	0.36	0.61	23.00	19.50	1.00	1.00	0.63	1.00	1.00	-	-
Max.	2.22	2.49	0.60	1.22	-	20.20	0.40	0.67	-	20.20	-	-	0.70	-	-	0.15	0.35

20 Definitions

Transmit Signals	All signals related to transmitting ATM or TDM data from the chip.
Receive Signals	All signals related to receiving ATM or TDM data into the chip.
Transmit Blocks	All blocks which process data in the ingress direction; towards the ATM network.
Receive Blocks	All blocks which process data in the egress direction; away from the ATM network.
AAL1	ATM Adaptation Layer 1
ATM	Asynchronous Transfer Mode
CAS	Channel Associated Signaling
CBR	Constant Bit Rate
CCS	Common Channel Signaling
CDV	Cell Delay Variation
CDVT	Cell Delay Variation Tolerance
CES	Circuit Emulation Services
CBR	Constant Bit Rate
CLP	Cell Loss Priority
CRC	Cyclic Redundancy Check
CRC-10	10-bit Cyclic Redundancy Check
CSD	Cell Service Decision
CSI	Convergence Sublayer Indication
DAC	Digital-to-Analog Converter
DACS	Digital Access Cross-connect System
DDS	Digital Data Service
DS0	Digital Signal Level 0
DS1	Digital Signal Level 1
DS3	Digital Signal Level 3
E1	European Digital Signal Level 1
E3	European Digital Signal Level 3
ESF	Extended Super Frame
FIFO	First-In, First-Out
FPGA	Field Programmable Gate Array
HEC	Header Error Check
LIU	Line Interface Unit
LSB	Least Significant Bit
M13	Multiplexer Level 1 to Level 3
MIAC	Memory Interface and Arbitration Controller
MIB	Management Information Base
Mod	modulo

MPHY	Multi-PHY
MSB	Most Significant Bit
OAM	Operations, Administration, and Maintenance
PBX	Private Branch Exchange
PCR	Peak Cell Rate
PDH	Pleisochronous Digital Hierarchy
PHY	Physical Layer
PLL	Phase-Locked Loop
PCM	Pulse Code Modulation
PRBS	Pseudorandom Bit Sequence
PTI	Payload Type Indicator
RALP	Receive Adaptation Layer Processing
RATM	Receive UTOPIA ATM Layer
RFTC	Receive Frame Transfer Controller
RUTOPIA	Receive UTOPIA
SAR	Segmentation and Reassembly
SDF-FR	Structured Data Format, Frame-based
SDF-MF	Structured Data Format, Multiframe-based
SDU	Service Data Unit
SF	Super Frame
SN	Sequence Number
SNP	Sequence Number Protection
SOC	Start-Of-Cell
SP	Supervisory Processor
SPHY	Single PHY
SRAM	Static Random Access Memory
SRTS	Synchronous Residual Time Stamp
SSRAM	Synchronous Static Random Access Memory
TALP	Transmit Adaptation Layer Processor
TATM	Transmit UTOPIA ATM Layer
TDM	Time Division Multiplexing
TFTC	Transmit Frame Transfer Controller
TLIP	Transmit Line Interface Processor
TTL	Transistor-to-Transistor Logic
TUTOPIA	Transmit UTOPIA
UDF	Unstructured Data Format
UDF-HS	Unstructured Data Format, High Speed
UDF-ML	Unstructured Data Format, Multiple Line
UI	Unit Interval
UTOPIA	Universal Test and Operations Physical Interface for ATM

VC	Virtual Circuit
VCI	Virtual Circuit Identifier
VCO	Voltage Controlled Oscillator
VCXO	Voltage Controlled Crystal Oscillator
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit
VP	Virtual Path
VPI	Virtual Path Identifier
ZBT	Zero Bus Turnaround
HEC	Header Error Check

Notes