PM73121

AAL1gator II

AAL1 Segmentation And Reassembly Processor

DATA SHEET

Issue 3: January 1999

NOTE: The AAL1gator II device contains SRTS logic that Bellcore holds the patent on.
Please refer to the NOTE on page 172 for more information regarding Bellcore's SRTS patent.

WHAT'S NEW IN THIS DATA SHEET?

This revision history documents the changes that occur from one data sheet version to the next version.

From Version	To Version	Major Changes

Issue 2	Issue 3	• Corrected the Function description in section 7.8.12 "R_OAM_QUEUE", on page 163.
		• Removed "Preliminary" from the headers.
		• Removed the "P" suffix from the part number in Appendix 29, "Ordering Information", on page 205.
		• Changed the following timing parameters:
		  • Interrupt Timing: PROC_INTR Tq(max) from 16.5 ns to 17 ns.
		  • Microprocessor RAM Read Cycle: /MEM_CS Tq(max) from 17.7 ns to 18 ns.
		  • Microprocessor RAM Read Cycle: Tqmoe(max) from 24.7 ns to 25 ns.
		  • Microprocessor RAM Write Cycle: /PROC_ACK Tq(max) from 17.5 ns to 18 ns.
		  • Microprocessor RAM Write Cycle: /MEM_CS Tq(max) from 17.7 ns to 18 ns.
		  • Microprocessor Write Command Register: /PROC_ACK Tq(max) from 17.5 ns to 18 ns.
		  • Transmit Side Interface: RL_SER Th(min) from 1.2 ns to 2 ns.
		  • Transmit Side High-Speed Interface: RL_SER Th(min) from 1.2 ns to 2 ns.
		  • Transmit UTOPIA ATM Timing: TATM_DATA Tq(max) from 12.7 ns to 13 ns.
		  • TUTOPIA SPHY Timing: RPHY_DATA Tq(max) from 12.7 ns to 13 ns.
		  • TUTOPIA MPHY Timing: RPHY_DATA Tq(max) from 12.7 ns to 13 ns.

From Version	To Version	Major Changes

Issue 1	Issue 2	• Removed Pin 237, P_OUT, from Pinout Table.
		• In T_QUEUE_TBL, added clarifications to QUEUE_CREDITS and AVG_SUB_VALU fields for single DS0 no pointer mode.
		• Changed ItypE3 to ItypDS3 in DC Operating Conditions Table.
		• Changed the following timing parameters:
		  • Interrupt Timing: PROC_INTR Tq(max) from 16 ns to 16.5 ns.
		  • Microprocessor RAM Read Cycle: /MEM_CS Tq(max) from 15 ns to 17.7 ns.
		  • Microprocessor RAM Read Cycle: Tqmoe(max) from 22 ns to 24.7 ns.
		  • Tzsu, Tded, and Tzen are now specified as typical, instead of minimum and maximum.
		  • Microprocessor RAM Write Cycle: /PROC_ACK Tq(max) from 15 ns to 17.5 ns.
		  • Microprocessor RAM Write Cycle: /MEM_CS Tq(max) from 15 ns to 17.7 ns.
		  • Microprocessor Write Command Register: /PROC_ACK Tq(max) from 15 to 17.5 ns.
		  • RAM Write Cycle: /MEM_WE Twp(min) from Tch - 1 to Tch - 1.3, and Twp(max) from Tch to Tch +0.3.
		  • Receive Side Low Speed Interface: TL_SER, TL_SIG Tq(max) from 12 ns to 14 ns.
		  • Transmit Side Interface: RL_SER Th(min) from 1.0 to 1.2 ns.
		  • Transmit Side High-Speed Interface: RL_SER Th(min) from 1.0 to 1.2 ns.
		  • Transmit UTOPIA ATM Timing: TATM_DATA Tq(max) from 12 ns to 12.7 ns.
		  • TUTOPIA SPHY Timing: RPHY_DATA Tq(max) from 12 ns to 12.7 ns.
		  • TUTOPIA MPHY Timing: RPHY_DATA Tq(max) from 12 ns to 12.7 ns.
		• Added DC Operating Conditions: I_{TYPE1}(max)=420mA and I_{TYPDS3}(max)=482mA.
		• In Absolute Maximum Ratings section, removed undershoot/ overshoot specification, and replaced with absolute maximum voltage range for TTL inputs.
		• Moved all timing requirements on external logic for RAM and Microprocessor interface from section 6.5 to section 8.11.

From Version	To Version	Major Changes
04/17/98	Issue 1	• Changed from WAC-121-A to PM73121. • Changed from User's Manual to Long Form Data Sheet. • Deleted references to the BT_Mode and default mode. • Added PMC part numbers to Figure 5 on page 11, Figure 6 on page 12, Figure 7 on page 13, and Figure 86 on page 168. • Under the "R_LINE_STATE Word Format" section on page 156, added "Not used in UDF-HS mode." for the R_UNDERRUN and R_RESUME field descriptions. • Under section 7.11 "Activating a New Queue on an Active Line", on page 167, changed from "CMD_REG_ATTN" to "CSD_REG_ATTN bit".
01/21/98	04/17/98	• Changed references from SRTS_PORT to SRTS_LINE throughout the manual. • Added the sixth bullet on page 6. • Under "Potential System Impacts" on page 7, added "Hardware Considerations". • Deleted the first paragraph on page 65. • Replaced section 3.7.1 "SRTS for the Receive Side" starting on page 66 with section 3.7.1 "Generation of TL_CLK" starting on page 68. • Added section 3.7.1.1 "Recovered Mode" starting on page 69, section 3.7.1.2 "Synthesize a Nominal E1 or T1 Clock" starting on page 69, and section 3.7.1.3 "Synthesize an E1 or T1 Clock based on SRTS" starting on page 70. • In Table 9 on page 88, changed the last sentence of the "SYS_CLK" description to read "The maximum frequency is 40 MHz. • In Table 10 on page 89, added the note to the description of "/SCAN_TRST" on page 89. • Under section 6.5 "RAM and Microprocessor Timing" starting on page 104, changed the first sentence of the third paragraph from "running at 38.88 MHz" to "running near maximum speed". • The first sentence of the fourth paragraph changed from "(38.88 MHz)" to "(40.00 MHz)". • In Figure 69 on page 105, changed SYS_CLK from "38.88 MHz" to "40.00 MHz". • In Table 21 on page 105, changed the maximum value of Number 12 from "8" to "7".

From Version	To Version	Major Changes
01/21/98	04/17/98	• Under section 6.5.1 “RAM Timing” starting on page 103, changed from “12 ns or 15 ns SRAMs” to “12 ns SRAMs”, changed from “data setup time of 8 ns” to “data setup time of 7 ns”, and changed from “SYS_CLK is 38.88 MHz” to “SYS_CLK is 40 MHz”. • In the table after Figure 80 on page 119, changed the Fc maximum value from “38.89” to “40.00”. • Under “QUEUE_CONFIG Word Format” section starting on page 135, added the second and third bullet to the note for the “FRAMES_PER_CELL” description and the note for the “BYTES_PER_CELL” description. • Under section 7.8.6 “R_CH_TO_QUEUE_TBL”, on page 148, added the sixth sentence to the Function description. • Under section 8.6 “Board Requirements for the SRAM Interface” starting on page 174, changed the third paragraph from “SRAMs must be 15 ns or faster” to “SRAMS must be 12 ns or faster”, changed from “data setup of 8 ns” to “data setup of 7 ns” and deleted the following four sentences. • Added section 8.7 “UDF-HS Mode SRTS-Based Clock Recovery Application for DS3” starting on page 180, section 8.8 “Interfacing with the Mitel MT8980 Digital Switch”, on page 182, section 8.9 “Interfacing with the ATM Cell Multiplexer (WAC-185-B-X)”, on page 183, and section 8.10 “Jitter Characteristics of Clock Synthesis Logic” on page 185.
10/17/97 (first version of the WAC-121-A User’s Manual)	01/21/98	• In the first sentence under “Description” on page 1, added E3 to the list of lines supported. • Under section 2.2 “Circuit Interface Features” starting on page 16, added the Dallas Semiconductor part DS2152 to the second bullet, added the Dallas Semiconductor part DS2154 to the third bullet, deleted the IgT DS3 Framer (TAC-030-A), and deleted the Dallas part number DS2180A. • Under section 2.4 “Receive Interface Features” on page 18, changed the fourth bullet text from “0 to 48 ms” to “0 to 24 ms”. • Under section 2.5 “Statistics” starting on page 19, deleted text from the fourth bullet and added the fourth sentence in the first paragraph on page 20. • Under section 2.7 “SRTS and Transmit Line Interface Clock Configurations” on page 21, and section 3.1. “SRTS for the Transmit Side” added the NOTE regarding Bellcore’s SRTS patent.

From Version	To Version	Major Changes
10/17/97 (first version of the WAC-121-A User's Manual)	01/21/98	• Under section 2.8.3 "Peak Cell Rates and Partial Cells", on page 22, changed the formula for $TCGT_{OAM}$. • Under section 3.1.2 "Transmit Signaling Freezing", on page 32, added the Dallas Semiconductor part numbers DS2152 and DS2154. • Changed the signal name text in the parentheses in Figure 23 on page 35. • Added step 8 on page 25, step 9 on page 25, and step 10 on page 25. • Deleted the NOTE after step 10 on page 36 and added ten more paragraphs. • Added section 3.2.1 "Transmit CDV" starting on page 37. • Under section 3.3.2.1 "Header Construction", on page 39, revised the fourth sentence. • Under section 3.4 "Transmit UTOPIA Interface Block (TUTOPIA)" starting on page 41, added the last two sentences to the first paragraph on page 43. • Under section 3.6 "Receive Adaptation Layer Processor (RALP)" starting on page 46, changed the last sentence of the third paragraph on page 47, added text to the second, sixth, and eight bullets on page 50. • Under section 3.6.2 "Underrun" starting on page 60, added text to the third paragraph on page 60. • Under section 3.6.3 "Pointer Processing", on page 61, added to text to the third paragraph. • Under section 3.6.4 "Overrun" starting on page 62, added text to the first and third paragraphs. • Added step 6 to the NOTES in Figure 47 on page 64. • Under section 3.6.5 "Counters and Sticky Bits", on page 65, revised the first paragraph. • Under section 3.7 "Receive Frame Transfer Controller (RFTC)" starting on page 65. • Under section 3.7.1 "SRTS for the Receive Side" starting on page 66, revised the first and second paragraphs on page 69. • In the headings of the first columns in Table 3 on page 72 and in Table 4 on page 72, changed "SRTS_PORT(3:0)" to "SRTS_LINE(3:0)". • Under section 3.8 "Memory Interface and Arbitration Controller (MIAC)" starting on page 74, added the last sentence to the third paragraph. • Added section 3.9 "Configuration", on page 75.

From Version	To Version	Major Changes
10/17/97 (first version of the WAC-121-A User's Manual)	01/21/98	• In Table 6, starting on page 81, deleted the last two sentences from the description of "RPHY_CLAV" on page 82 and the description of "TPHY_CLAV" on page 83. • Under Figure 62 on page 96, changed the Maximum value of Tq from "12" to "13". • Under section 6.4.1 "RUTOPIA as the ATM Layer Device", on page 100, added the second paragraph. • In Table 21 on page 105, changed the Maximum value of number 16 from "2" to "1" and number 18 from "7" to "6". • Under Figure 69 on page 104, changed the Minimum value of Twc from "Tp-1" to "Tp-2", Twsu from "2" to "1", and added text to the second bullet under NOTES. • Revised Figure 70 on page 105 and the corresponding table. • Under Figure 71 on page 107, changed the Maximum value of Tded from "24" to "25" and Taed from "19" to "20". • Under Figure 72 on page 109, changed the Maximum value of Taed from "19" to "20". • Under Figure 73 on page 112, changed the Maximum value of Tded from "24" to "25". • Under Figure 76 on page 117, changed the Maximum value of Tq from "12" to "16". • Deleted section 7 "Boundary Scan" starting on page 119 and renumbered the remaining sections. • Under "QUE_CREDITS Word Format" section on page 137, changed from "Reserved" to "FRAME_REMAINDER" and added description. • Under the "CMDREG Word Format" section on page 165, revised the description for "CSD_ATTN". • Added the second bullet to NOTES on page 166. • Under section 7.11 "Activating a New Queue on an Active Line", on page 167, added the third bullet to NOTES. • Added section 8.4 "External FIFO Application" starting on page 171. • Under section 8.5 "External SRTS-Based Clock Recovery Application", on page 172, added the last two paragraphs.

Table of Contents

List of Figures

List of Tables

Description

The AAL1 Segmentation And Reassembly (SAR) Processor (AAL1gator II™) 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 II device.

FEATURES

Circuit Interface Features

- Provides AAL1 segmentation and reassembly of eight 2 Mbit/s data streams or one 45 Mbit/s or less data stream.
- Supports 256 Virtual Channels (VCs) (32 per line).
- Supports n × 64 structured data format.
- Supports arbitrary timeslot-to-VC mappings, including alternating timeslots.
- Provides Common Channel Signaling (CCS) and Channel Associated Signaling (CAS) configuration options.
- Provides per-VC data and signaling conditioning in both the transmit and the receive directions.
- Arbitrates a 16-bit microprocessor interface to a 128K × 16 (12 ns) SRAM.
- Supports multicast connections, ATM Monitoring (AMON), Remote Monitoring (RMON), and ATM Circuit Steering (ACS).
- Supports adaptive clocking in Structured Data Format, Frame-based (SDF-FR), Structured Data Format, Multiframe-based (SDF-MF), and Unstructured Data Format, Multiple Line (UDF-ML) modes.

Transmit Cell Interface Features

- Provides an ATM-layer or PHY-layer 33 MHz UTOPIA interface. Both Single PHY (SPHY) and Multi-PHY (MPHY) modes are supported.
- Provides per-VC transmit queueing.
- Provides a calendar queue service algorithm that produces minimal Cell Delay Variation (CDV).
- Provides a supervisory transmit buffer for Operations, Administration, and Maintenance (OAM), and for ATM signaling.
- Generates pointers for structured data transmission.
- Provides sequence number and sequence number protection generation.
- Provides partially filled cell generation with the length configurable on a per-VC basis.
- Generates and transmits Synchronous Residual Time Stamp (SRTS) values for unstructured modes.
- Built-in transmit line clock generation based on received SRTS values, receive line clock, or a nominal frequency.

Receive Cell Interface Features

- Provides an ATM-layer or PHY-layer 33 MHz UTOPIA interface. Both SPHY and MPHY modes are supported.
- Provides per-VC queues.
- Provides per-VC CDV tolerance settings.
- Provides per-VC partially filled cell length settings.
- Provides a supervisory receive queue for OAM cells.
- Verifies and corrects sequence numbers in accordance with ITU-T Recommendation I.363.1.
- Processes sequence numbers in accordance with the “Fast SN Algorithm”, as specified in the ITU-T Recommendation I.363.1.
- Maintains bit integrity through individual errored cells or up to six lost cells. Takes into account pointer bytes.
- During underruns, can output fixed, pseudorandom, or old data.
- Provides processor interrupts for OAM cell receptions.
- Provides a multiplexed interface to external receive Phase-Locked Loops (PLLs) for SRTS clock recovery for unstructured modes or adaptive clock recovery.

Statistics Features

- Counts invalid Cyclic Redundancy Check (CRC) values for sequence numbers.
- Counts OAM cells and dropped OAM cells.
- Counts data cells transmitted per VC.
- Counts conditioned data cells transmitted per VC.
- Counts cells not transmitted due to line resynchronization per VC.
- Counts cells received, dropped, lost, or misinserted per VC.
- Counts cells with incorrect Sequence Number (SN) or incorrect Sequence Number Protection (SNP).
- Counts underrun occurrences per VC.
- Counts overrun occurrences per VC.
- Counts pointer reframes and pointer parity errors per VC.

Figure 1 shows a simple block diagram of the AAL1gator II.

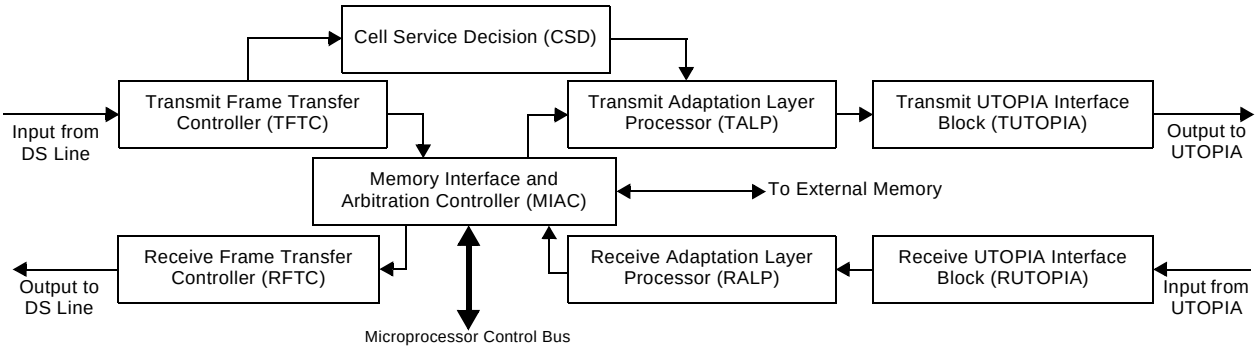


Figure 1. AAL1gator II Block Diagram

Figure 2 shows a system block diagram of the AAL1gator II connected to eight 2 Mbit/s data streams.

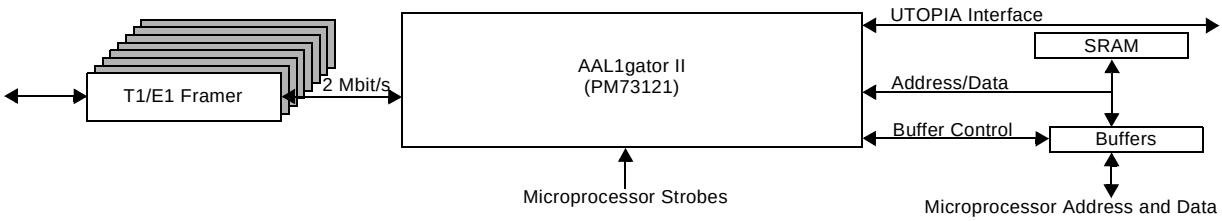


Figure 2. Block Diagram of the AAL1gator II Connected to Eight 2 Mbit/s Data Streams

Figure 3 shows a system block diagram of the AAL1gator II connected to one 45 Mbit/s data stream.

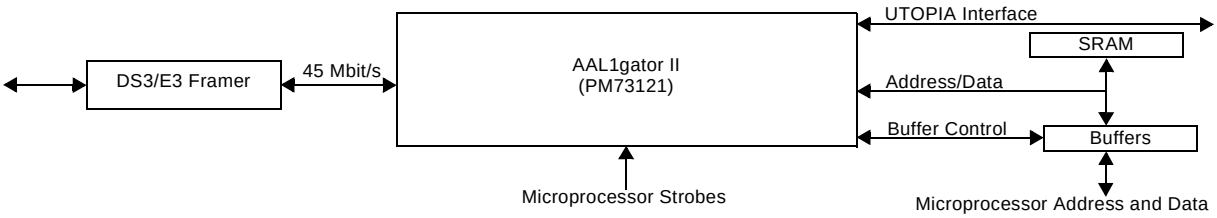


Figure 3. Block Diagram of the AAL1gator II Connected to One 45 Mbit/s Data Stream

NEW FEATURES

Changes in the AAL1gator II (PM73121) version from the WAC-021-C-X version are as follows:

- After power-up, setting the CMD_REG_ATTN register bit (refer to page 166) causes the device revision register to be loaded with code $121A_h$.
- Supports the "Fast Algorithm" method for processing SNs, as specified in the ITU-T Recommendation I.363.1.
  - This method maintains bit integrity through any single corrupted cell, single misinserted cell, or up to six missing cells as determined by the value of the SN and SNP field. However, bit integrity cannot be maintained if an underrun occurs.
  - If structured cells are used, adjustments for missing or errored pointer cells are considered. Pointer cells will be predicted to be in the cells that should contain pointers as determined by the ITU-T Recommendation I.363.1.
  - The value of the inserted cells can be FF_h, old data, one byte of conditioned data, or pseudorandom data that uses the conditioned data and replaces the MSB with a pseudorandom pattern $x^{18} + x^7 + 1$. The pseudorandom option is not available for UDF-HS mode. The pattern used is selectable on a per-VC basis.
  - The maximum number of cells, (MAX_INSERT) in the R_SN_CONFIG word (up to six) inserted per queue is programmable on a per-queue basis (refer to "R_SN_CONFIG Word Format" on page 160).
  - There is a bit, DISABLE_SN in the R_SN_CONFIG register (refer to "R_SN_CONFIG Word Format" on page 160), to optionally disable SN processing for a queue.
  - A bit has been added to disable the automatic dropping of the first cell received, as would be normally required by ITU-T Recommendation I.363.1 (refer to "NODROP_IN_START" on page 161).
- The UTOPIA interface can operate as an SPHY or MPHY device, in addition to its current ATM mode.
- When a queue underruns, old data, one byte of conditioned data, or pseudorandom data that uses the conditioned data and replaces the MSB with a pseudorandom pattern ($x^{18} + x^7 + 1$) will be automatically inserted. The choice is selectable per DS0. Pseudorandom options are not available for UDF-HS mode.
- Added the following seven new counters per queue:
  - number of dropped cells
  - number of lost cells
  - number of pointer reframes
  - number of underruns

    - number of overruns
    - number of pointer parity errors
    - number of misinserted cells
- A SUPPRESS_TRANSMISSION bit has been added (refer to section "IDLE_CONFIG Word Format" on page 139) to temporarily disable transmission while continuing to schedule cells.
- Two new control bits have been added to allow queues on an SDF-MF line to be configured in SDF-FR mode. One bit (refer to "T_CHAN_NO_SIG" on page 136) is for the transmit direction, and one bit (refer to "R_CHAN_NO_SIG" on page 157) is for the receive direction. This is the same as setting up individual queues in SDF-FR mode.
- A new E1 mode has been added to allow a 24-frame multiframe instead of a 16-frame multiframe. In this mode, signaling is updated every 24 frames. When processing cells, the signaling data will appear after N × 24 bytes of data, where N is the number of DS0s in the queue.
- The SRTS code from the Field Programmable Gate Array (FPGA) has been incorporated into the chip for each line. There is also a clock multiplexer that enables the AAL1gator II to generate the TL_CLK based on either the RL_CLK, a nominal E1 or T1 clock, or a synthesized clock based on the received SRTS.
- The R_INCORRECT_SNP counter (refer to "R_INCORRECT_SNP Word Format" on page 158) has been changed to conform to the Circuit Emulation Service (CES) MIB. That is, all cells with an incorrect SNP will be counted, regardless of whether or not the SN was corrected.
- The R_SEQUENCE_ERR counter (refer to "R_SEQUENCE_ERR Word Format" on page 157) has been changed to conform to the CES MIB. That is, only transitions from the SYNC state to the OUT_OF_SEQUENCE state are counted, as specified in ITU-T Recommendation I.363.1.
- Both E1 and T1 lines can be supported at the same time.
- The scan string for the current scan order has changed.
- The following four pins have been added:
    - PHY_ENABLE (refer to page 81)
    - TPHY_ADDR (refer to page 83)
    - RPHY_ADDR (refer to page 83)
    - TLCLK_OUTPUT_EN (refer to page 87)
- The following software fields have been added or expanded:
    - In the COMP_LIN_REG (refer to section 7.4.2 "COMP_LIN_REG" starting on page 125), added the following fields:

    - MIXED_MODE_EN
    - SPHY_EN
  - In the LIN_STR_MODE word (refer to section 7.4.3 “LIN_STR_MODE” starting on page 126), added the following fields:
    - TL_MODE
    - R1_WITH_T1_SIG
    - CLK_SOURCE
  - In the QUEUE_CONFIG word (refer to “QUEUE_CONFIG Word Format” starting on page 135), added the following field:
    - T_CHAN_NO_SIG
  - In the CH_TO_QUEUE word (refer to “CH_TO_QUEUE Word Format” starting on page 148), expanded the following fields:
    - RX_COND_H
    - RX_COND_L
  - In the R_MAX_BUF word (refer to “R_MAX_BUF Word Format” on page 157), added the following field:
    - R_CHAN_NO_SIG
- Added the following nine registers per queue:
  - IDLE_CONFIG (refer to “IDLE_CONFIG Word Format” on page 139)
  - R_SN_CONFIG (refer to “R_SN_CONFIG Word Format” on page 160)
  - R_DROPPED_CELLS (refer to “R_DROPPED_CELLS Word Format” on page 161)
  - R_UNDERRUNS (refer to “R_UNDERRUNS Word Format” on page 162)
  - R_LOST_CELLS (refer to “R_LOST_CELLS Word Format” on page 162)
  - R_OVERRUNS (refer to “R_OVERRUNS Word Format” on page 162)
  - R_POINTER_REFRAMES (refer to “R_POINTER_REFRAMES Word Format” on page 162)
  - R_PTR_PAR_ERR (refer to “R_PTR_PAR_ERR Word Format” on page 162)
  - R_MISINSERTED (refer to “R_MISINSERTED Word Format” on page 163)
- The maximum system clock frequency is now 40 MHz.

POTENTIAL SYSTEM IMPACTS

Hardware Considerations

The AAL1gator II device is pin-for-pin compatible with the WAC-021-CX. However, the following hardware issues are items to consider.

- The DS3 mode requires a 40 MHz system clock to maintain throughput.
- Power consumption for the AAL1gator II is about 300 mW higher than for the WAC-021-CX device.

Software Considerations

The first two issues are addressed by PMC-Sierra's AAL1gator II software Device Control Package.

- The math table was modified to support E1 and T1 lines at the same time. This was required since the WAC-021-CX has a different math table for E1 lines than for T1 lines. The PM73121 can configure some lines in E1 mode and some lines in T1 mode. As a result, only one math table is required and anyone using E1 SDF-MF mode needs to use the new math table.
- New counters have to been added to support the CES MIB. These count values will automatically be written into memory at previously unused locations.
- The PM73121 calculates buffer overflow as data is written into memory. The WAC-021-CX calculates buffer overflow only at the start of a cell being written into memory. Because of this difference in calculating buffer overflow, it is possible that a queue configured with a certain MAX_BUF size with the WAC-021-CX may show an overflow with the PM73121.
- The SNP algorithm has been changed to conform with the "Fast Algorithm", as specified in the ITU-T Recommendation I.363.1. The new algorithm is more tolerant to single errors and lost cells and has a slightly different behavior. Typically this is transparent to users, unless they are injecting errors. However, one side effect of the new algorithm is that the first cell received for any queue is always dropped. This is not an issue for most applications since there is always a start-up period before any real data is contained in the cells. However, if an application exists that is sensitive to the loss of the first cell, a new control bit has been added to disable dropping the first cell (refer to "NODROP_IN_START" on page 161). This bit defaults to off.
- The INCORRECT_SNP counter (refer to "R_INCORRECT_SNP Word Format" on page 158) has been changed to conform to the CES Management Information Base (MIB). That is, all cells with an incorrect SNP field will be counted, regardless of whether or not the SN was corrected. (In the WAC-021-CX only non-correctable SNPs were counted.)

- The R_SEQUENCE_ERR counter (refer to "R_SEQUENCE_ERR Word Format" on page 157) has been changed to conform to the CES MIB. That is, only transitions from SYNC to OUT_OF_SEQUENCE are counted, as specified in ITU-T Recommendation I.363.1.
- The revision code of the chip has been changed to "121A".

When using the AAL1gator II, you will have to make software updates to take advantage of the new features (refer to "New Features" starting on page 4). These new features require setting control bits or writing fields that are not used in the WAC-021-CX. The default for these bits is off.

PM73121 Required Board Modifications

The PM73121 is pin-for-pin backward compatible with the WAC-021-CX. To allow the AAL1gator II to drop into a board developed for the WAC-021-CX, the following hooks need to be implemented for you to take advantage of internal clock synthesis or UTOPIA PHY mode.

To Use Internal Clock Synthesis

- Provide a means to tristate TL_CLK to the PM73121.
- Provide pads to terminate TL_CLK correctly when sourced by the PM73121. You may need to remove the termination used when the AAL1gator II does not source TL_CLK.
- Be aware of any skew issues that may arise when the AAL1gator II sources the TL_CLK instead of being externally generated, such as an external clock multiplexer.

NOTE: With respect to TL_CLK, the AAL1gator II timing remains the same whether it sources the clock or not.

- Pin 79 is the TLCLK_OUTPUT_EN pin. The AAL1gator II can synthesize a nominal clock, loop the RL_CLK, or use the SRTS to generate the TL_CLK. To select the clock type, configure the LIN_STR_MODE register (refer to "LIN_STR_MODE" starting on page 126). Since there is a period of time between reset and when this register is read, there may not be a TL_CLK. To account for this, tie the TLCLK_OUTPUT_EN high. This will cause the RL_CLK to be looped to the TL_CLK pins until the value has been read. Then each line TL_CLK will switch to the proper value. (Each line can be different). If you do not want a clock to drive between reset and when this register is read, tie TLCLK_OUTPUT_EN low. In summary, provide a means to pull up or pull down pin 79. It will default to pull down via an internal pull-down resistor.

To Use UTOPIA PHY Mode

- Provide a means to pull pin 56 (PHY_EN) high. (It will default to low via an internal pull-down resistor).
- If using MPHY mode, pin 18 will be /TPHY_ADDR and pin 14 will be /RPHY_ADDR. These address pins work like an active low chip select but follow the MPHY timing.
- In PHY mode, all TATM signals will become RPHY signals (/TATM_EN becomes RPHY_CLAV and /TATM_FULL becomes /RPHY_EN). All RATM signals will become TPHY signals (/RATM_EN becomes TPHY_CLAV and /RATM_EMPTY becomes / TPHY_EN).

1 SYSTEM APPLICATIONS

An essential function for ATM switches 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 II is a direct implementation of that service specification in silicon, including the complex $N \times 64$ channelized service and support of CAS.

1.1 Replacing a TDM Digital Access Cross-connect System (DACS) with an ATM System

In one public network application, a TDM DACS is replaced with an ATM system containing an AAL1gator II. When the backbone trunk of a network is changed from TDM to ATM, the TDM DACSs will probably be converted from TDM to ATM. The existing leased line services provided by the DACSs must then be provisioned with the ATM DACS. In this manner, TDM, frame relay, and ATM circuits can be provided in a single network, as shown in Figure 4.

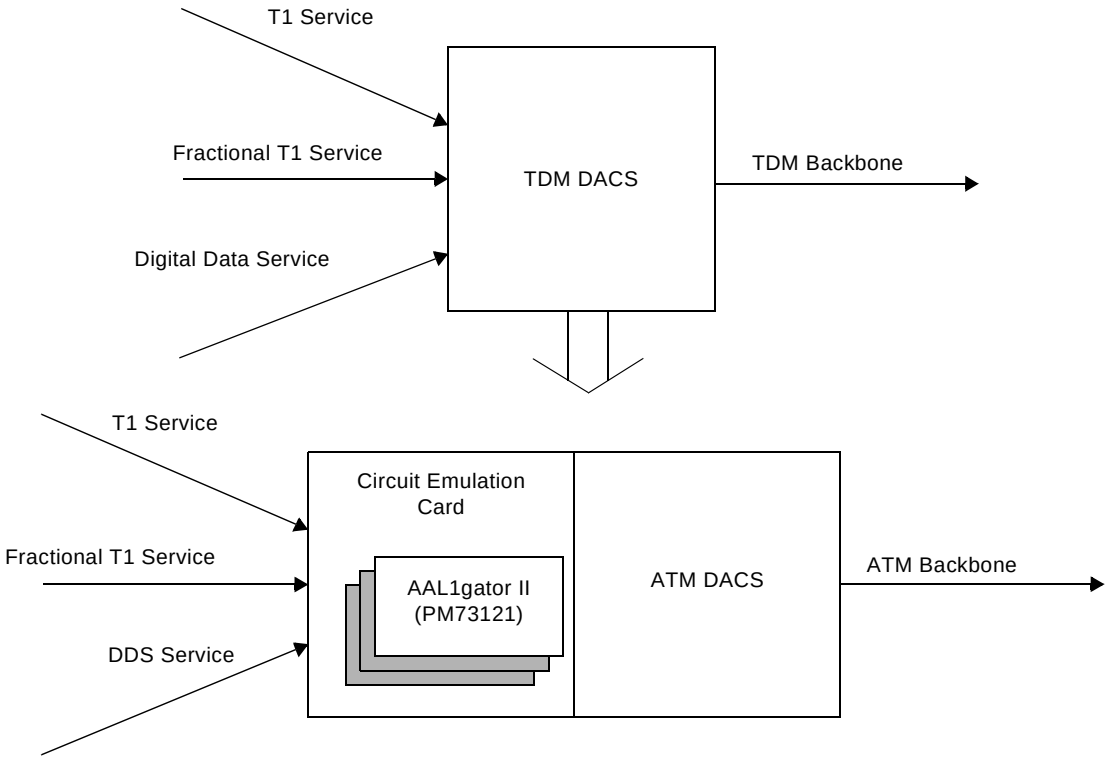


Figure 4. Using the AAL1gator II as Part of a TDM DACS Replacement

1.2 Replacing a Multiplexer Level 1 to Level 3 (M13) with an ATM System

In a second public network application, an M13 is replaced with an ATM system containing an AAL1gator II. M13s are often used to provide DS1 services. As Figure 5 shows, these DS1 circuits must be emulated by an ATM DACS or an ATM M13 when the backbone is converted to ATM.

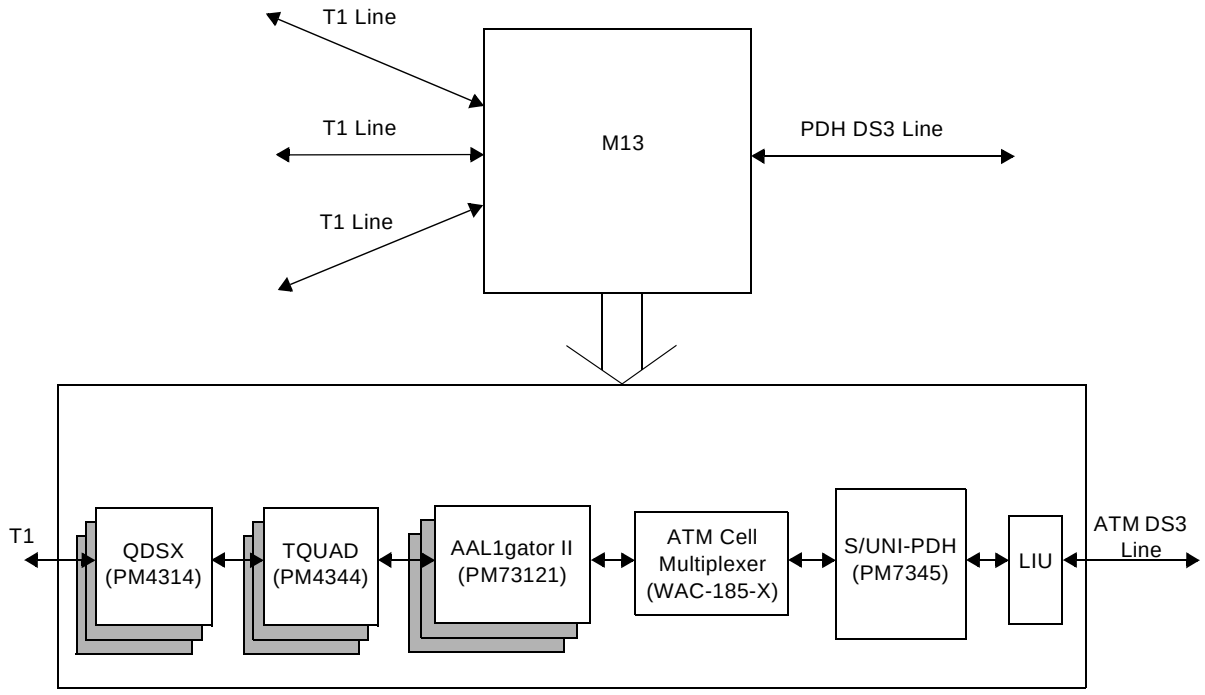


Figure 5. Using the AAL1gator II as Part of an M13 Replacement

1.3 Access Multiplexer Application

An access multiplexer must allow voice circuits to be made across a single ATM service interface. A circuit emulation interface card supports circuit connection to a Private Branch Exchange (PBX). As Figure 6 shows, CAS signaling is needed to support a wide variety of legacy PBXs.

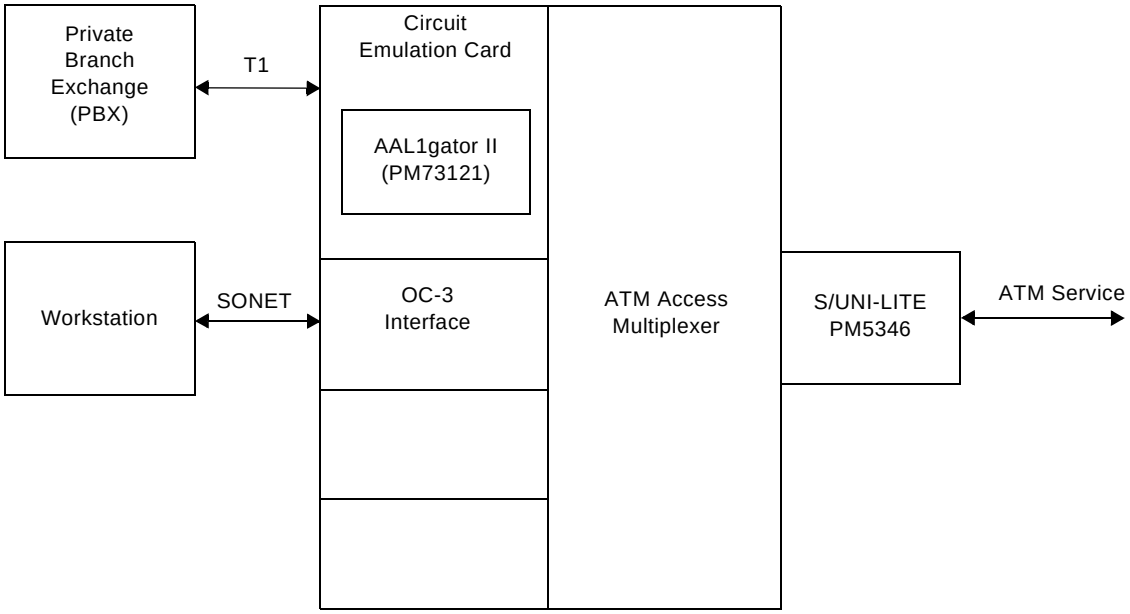


Figure 6. Using the AAL1gator II in an ATM Access Multiplexer Application

1.4 Using the AAL1gator II in an Enterprise ATM Switch Application

Circuit emulation has become a requirement of enterprise switches. Figure 7 shows a complete solution for ATM switches.

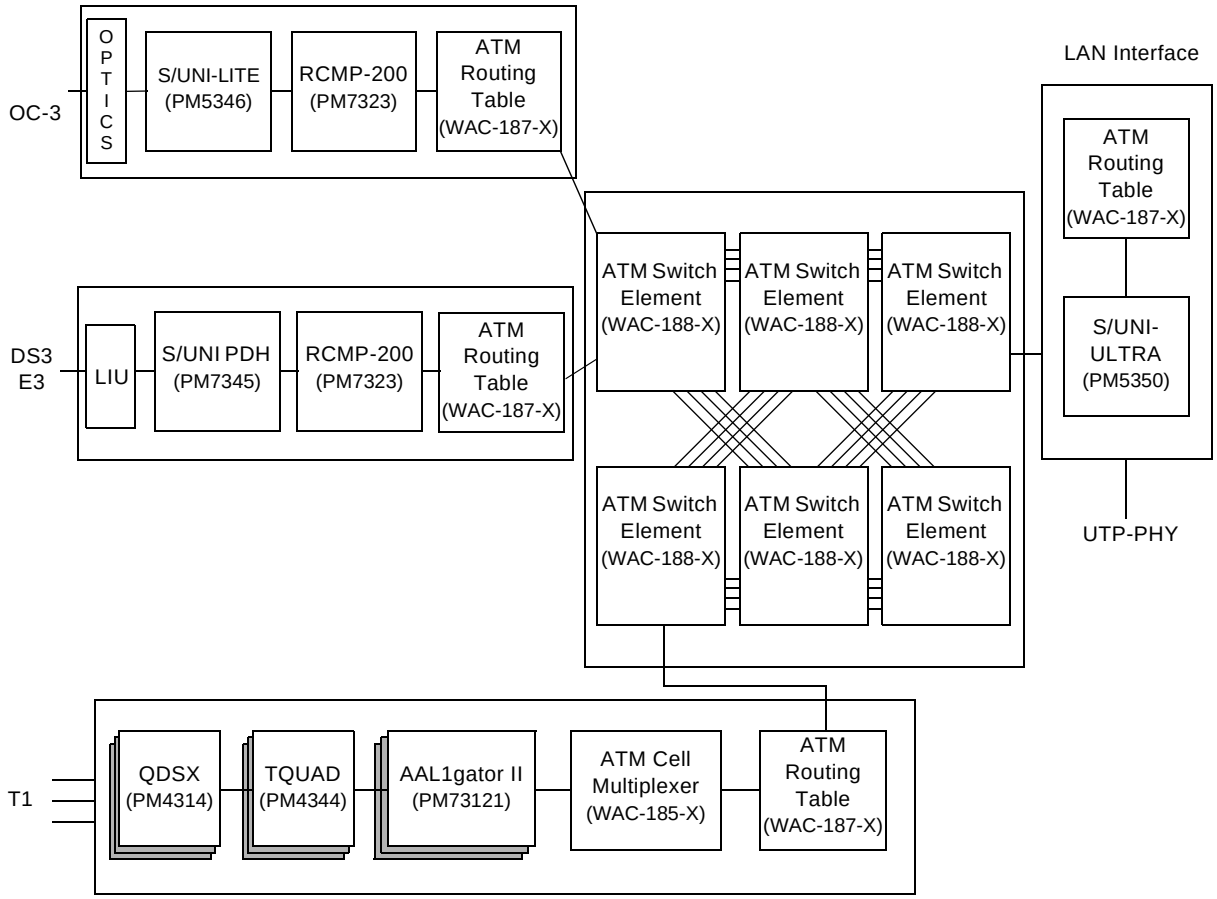


Figure 7. Using the AAL1gator II in an Enterprise ATM Switch Application

2 SYSTEM FEATURES

The AAL1gator II provides DS1, E1, DS3, E3, or other high-speed line interface access to an AAL1 CBR ATM network. It uses an external SRAM for temporary storage of data and statistical data. The device provides a microprocessor interface for configuration, management, and statistics gathering.

ATM technology can handle different types of traffic, including voice and video, within the same network structure. Voice traffic and some kinds of video traffic are classified as CBR traffic. AAL1 is defined for carrying CBR traffic.

2.1 Data Formats

The AAL1gator II supports the data formats described in Table 1.

Table 1. Data Formats and AAL1gator II Modes to Support Them

Format	Corresponding AAL1gator II Mode	Channelization	CAS Signaling	Number of Lines
Unstructured Data Format, Multiple Line	UDF-ML	No	No	8
Unstructured Data Format, High-Speed	UDF-HS	No	No	1
Structured Data Format, Multiframe-based	SDF-MF	Yes	Yes	8
Structured Data Format, Frame-based	SDF-FR	Yes	No	8

These modes are selectable on a per-line basis. In structured mode, each line can support either T1 or E1 interfaces. Line 0 can also support a high-speed UDF that is compatible with the specified T3 or E3 framer.

UDFs are more appropriate for services that require no framing structure to transport data across the network. Unframed point-to-point E1 and T1 and clear channel T1 and E1 are examples of such services. The AAL1gator II uses simplified algorithms for unstructured data, placing data into the appropriate line buffers without regard to frames. All byte locations are used. The Cell Service Decision (CSD) credit assigning algorithm (refer to section 3.2 “Cell Service Decision (CSD) Circuit” starting on page 33) is also simplified. Since there can be only eight queues, the algorithm needs to identify only which line requires service. When UDF-HS is used for line 0, only line 0 is monitored for activity.

Many installed networks, however, look for a framing structure and octet alignment to transport data streams. Carriers have large installed bases of fractional T1 and Digital Data Service (DDS) leased line services that must still be supported when the backbone is cut over to ATM. Many private networks have the same support issue; the backbone is either ATM or circuit-switched, but not both. If it is ATM, then all existing switched circuits must be migrated onto the ATM backbone through circuit emulation. Because of its extensive support of Digital Signal Level 0s

(DS0s), which are 64 Kbit/s channels, the AAL1gator II is an ideal vehicle for constructing ATM-based DACS and other circuit emulation devices. Structured data format is suited for these types of networks and underlying services. The SDF-MF mode is primarily used for applications that need to pass CAS, which carries the ABCD signaling bits. CAS is still common in many TDM networks. The SDF-FR mode, which transfers no signaling, is also supported.

In SDF-MF mode, individual queues can be configured to transfer no signaling. This is the equivalent to SDF-FR mode for a given queue.

2.1.1 Structured Cell Formats

Figure 8 shows examples of four structured cell formats. In the partially filled cell examples, the partially filled level has been set to 42 payload bytes.

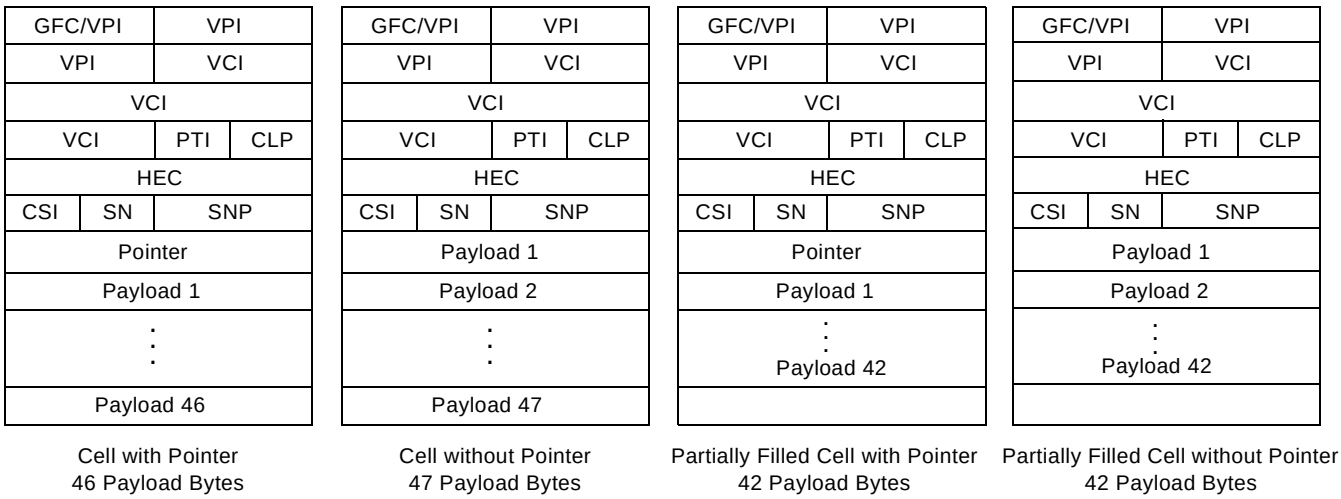


Figure 8. Examples of Structured Cell Formats

2.1.2 Unstructured Cell Formats

Figure 9 shows two unstructured cell formats. In the partially filled cell, the partially filled level has been set to 42 payload bytes.

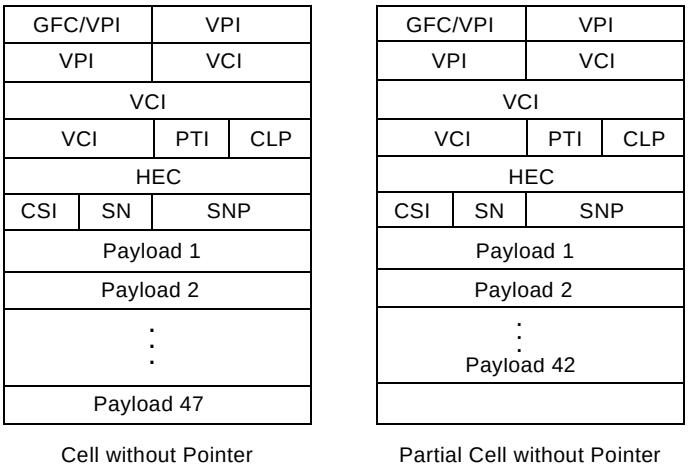


Figure 9. Examples of Unstructured Cell Formats

2.2 Circuit Interface Features

- Provides a convenient interface to all of the following line circuit types:
  - Up to eight T1 framers (PMC-Sierra part number PM4341A T1XC or Base-2™ part number UGA-360).
  - Up to eight E1 framers (PMC part number PM6341 E1XC or Base-2 part number UGA-510).
  - Up to two T1 PMC quad framers (PMC part number PM4344 TQUAD).
  - Up to two E1 PMC quad framers (PMC part number PM6344 EQUAD).
  - One E3 Framer (EAC-030-X)-compatible device (clear channel).
- Supports the following VCs:
  - Up to 256 VCs for channelized T1/E1 (Up to 32 VCs per line for E1, up to 24 VCs per line for T1).
  - Up to eight VCs for multiple-line unstructured data format (UDF-ML mode).
  - One VC for DS3.
  - One VC for E3.
- Compatible with the ATM Forum Circuit Emulation Service (CES) standard (refer to Appendix B, “References”, on page 203).

- Allows the outgoing Virtual Path Identifier (VPI)/Virtual Channel Identifier (VCI) to be set to any value for each VC.
- Maps a selectable 8-bit field of the VCI into 256 possible receive queues.
- Provides bit count integrity by replacing lost AAL Service Data Units (SDUs).
- Allows any combination of timeslots within one T1/E1 line to be mapped to a VCI.
- Supports UDF at arbitrary bit rates up to 20 Mbit/s aggregate throughput.
- Provides a per-line CCS/CAS configuration option.
- Individual lines can be configured as E1 or T1.
- E1 lines can be configured to use T1 signaling rates.
- Provides transmit data and signaling conditioning per VC.
- Provides receive signaling freezing on underrun, overrun, and pointer mismatch and errored cells.
- Provides receive data and signaling conditioning per VC.
- Provides SRTS bit generation and collection for an internal clock synthesizer to drive external receive PLLs for unstructured data formats.
- Provides a 16-bit microprocessor interface to a 128K × 16 SRAM external to the device.
- Provides statistics and interrupts for the microprocessor.

2.3 Transmit Interface Features

The AAL1gator II accepts deframed data as a serial bit stream from multiple external deframer devices. The AAL1gator II then stores the data in an external SRAM, and creates AAL1 ATM cells from the data. The AAL1gator II allows configurations of up to 256 VCs (32 per line) that can transmit from 1 to 32 DS0s (64 Kbit/s channels) within any one T1 or E1 line with arbitrary sequential mapping (including alternating timeslots). The VC queues are serviced with a calendar scheduling mechanism. The transmit side transmit queue controller also supports the transmission of cells generated by the microprocessor. In addition, a variety of statistics are maintained in 16-bit counters. Other transmit interface features include:

- Provides per-VC transmit queuing.
- Provides a calendar queue service algorithm that produces minimal CDV.
- Provides a 33 MHz ATM or PHY layer UTOPIA interface. The PHY side can be either SPHY or MPHY.
- Provides a supervisory transmit buffer for OAM/signaling with Cyclic Redundancy Check-10 (CRC-10) generation.
- Generates sequence numbers and sequence number protection bits.
- Provides partially filled cells with lengths configured on a per-VC basis.

- Supports Synchronous Residual Time Stamp (SRTS) for unstructured data formats. The transmit interface generates the 4-bit SRTS code that reflects the difference between the network clock and the transmitting service clock. If enabled, this SRTS code is inserted into the Convergence Sublayer Indication (CSI) bit of cells with odd sequence numbers.
- Provides data and signaling conditioning on a per-VC basis.
- Provides a 2-cell UTOPIA First-In, First-Out (FIFO).
- Transmit delay is defined by the following three components: the time to schedule a cell, the time to build a cell, and the time to transmit a cell. The time to schedule a cell is from 0 to 125 µs for structured data and from 0 to 256 bit times for unstructured data. The time (in seconds) to build a cell with a full payload is approximately 376 divided by the CBR (in bps). The time to send a cell depends on the number of cells queued for transmission, the amount of OAM traffic being inserted by the microprocessor, and the speed at which cells are accepted by the device connected to the UTOPIA transmitter. Once the payload is available, each cell takes about 8.5 µs to prepare.
- Provides optionally generated transmit line clock based on received SRTS values, looped RL_CLK, or synthesized nominal clock.

2.4 Receive Interface Features

When cells arrive, the AAL1gator II places them in a multiframe buffer based on the channel allocation. The cells are then transmitted to the line interface at the proper time. Other receive interface features include:

- Provides a 33 MHz ATM or PHY layer UTOPIA interface. The PHY side can be either SPHY or MPHY.
- Provides per-VC queues.
- Provides a 0 to 64 ms receive buffer for E1 (0 to 48 ms if using T1 signaling), a 0 to 48 ms receive buffer for T1, a 0 to 4 ms receive buffer for E3, and a 0 to 2.9 ms receive buffer for DS3.
- Provides a 0 to 32 ms per-VC Cell Delay Variation Tolerance (CDVT) setting for E1 (0 to 24 ms if using T1 signaling); provides a 0 to 24 ms per-VC CDVT setting for T1 in increments of 125 µs.
- Provides a 0 to 2 ms CDVT setting for E3 and a 0 to 1.5 ms CDVT setting for DS3.
- Provides overrun and underrun protection.
- Provides pointer misalignment protection.
- Receives, verifies, and corrects sequence numbers in accordance with ITU-T Recommendation I.363.1. (refer to Appendix B, “References”, on page 203).
- Processes sequence numbers in accordance with I.363.1 “Fast SN Algorithm” (refer to Appendix B, “References”, on page 203). Sequence number processing can optionally be disabled on a per queue basis.

- Provides a 256-cell OAM receive queue.
- Provides processor interrupt for OAM cell reception.
- Provides a 0 to 31 sample SRTS CDVT setting.
- Supports SRTS by collecting and queueing SRTS values, calculating a local SRTS value derived from the local transmit clock and the local network reference clock, and driving the difference between the incoming value and the local value out of a multiplexed interface. This difference can then be used to drive eight external digital or analog PLLs. Application support is available for an 8-line SRTS/adaptive digital frequency synthesizer, which can be added externally. Clock synthesis logic, internal to the chip, can optionally be configured to synthesize the TL_CLK based on the received SRTS values.
- In all modes except UDF-HS, supports adaptive clocking by calculating the receive frame difference and providing it to the external adaptive digital frequency synthesizer.
- Provides a 2-cell UTOPIA FIFO.
- Defines receive delay primarily by the following two components: buffering to handle CDV and the internal cell processing time. The internal cell processing time is less than 20 µs.

2.5 Statistics

The AAL1gator II gathers statistical information on all transmitted and received cells. These statistics are maintained in 16-bit counters (which are implemented as rollover counters and are never cleared) in the external memory. Table 2 lists the provided statistics counters.

Table 2. Statistics Counters

Counter Name	Number of Counters	Counter Description
R_OAM_CELL_CNT	1	Count of OAM cells received.
R_DROPPED_OAM_CELL_CNT	1	Count of OAM cells received and dropped due to lack of space in the receive OAM FIFO.
R_INCORRECT_SNP	256	Count of number of occurrences when the SN CRC and/or parity was not correct. Both correctable and noncorrectable errors are reported.
R_SEQUENCE_ERR	256	Count of SN errors. Counts transitions from SYNC to OUT_OF_ SEQUENCE.
R_CELL_CNT	256	Count of billable data cells received from the UTOPIA interface.
R_DROPPED_CELLS	256	Count of cells that were received but dropped.
R_UNDERRUNS	256	Count of the number of underruns on this queue to account for lost cells.
R_LOST_CELLS	256	Count of cells detected lost for this queue.
R_OVERRUNS	256	Count of overruns on this queue.

Table 2. Statistics Counters (Continued)

Counter Name	Number of Counters	Counter Description
R_POINTER_REFRAMES	256	Count of pointer reframes.
R_PTR_PAR_ERR	256	Count of pointer parity errors.
R_MISINSERTED	256	Count of misinserted cells.
T_CELL_CNT	256	Count of billable data cells transmitted to the UTOPIA interface.
T_COND_CELL_CNT	256	Counts data conditioned cells transmitted to the UTOPIA interface.
T_SUPPRESSED_CELL_CNT	256	Counts cells not sent because of a line resynchronization, or, if the device was in UDF-HS mode, because TX_ACTIVE is set. Also if SUPPRESS_TRANSMISSION is set, any cells not sent will increment this counter.

Sticky bits for the following events are also maintained on a per-queue basis in memory (refer to "R_ERROR_STKY Word Format" on page 158):

- A cell was received while the queue was in underrun.
- Receive overrun: a cell was received, causing the maximum buffer depth to be exceeded. This event causes a forced underrun condition.
- A resume function has occurred: a cell was received and placed in the receive buffer causing an end to the underrun condition.
- A cell was dropped in accordance with Recommendation I.363.1 "Fast SN Algorithm". This event causes a forced underrun condition.
- A cell was dropped because of a forced underrun.
- A cell was dropped because of a blank allocation table or disabled receiver.
- A cell was received.
- A cell was dropped because a valid pointer was not found while the queue was in the underrun condition.
- A valid pointer was received.
- An errored pointer was received (parity check failed).
- An SRTS resume occurred.
- A cell was received while the SRTS queue was in underrun.
- A cell was dropped because a pointer mismatch occurred causing a forced underrun condition.

The AAL1gator II sets the status bits every time it detects one of the events listed previously. The microprocessor clears the status bits after it recognizes the event. Because these bits will be polled at a low rate, relative to their occurrence, it is not possible to determine the exact number of

occurrences of each type of event. If a count is required, the counters should be used instead. The status bits for the receive error conditions on each queue are maintained in a single-data word. When a set bit is detected, this single-data word should be cleared immediately.

NOTE: As new events can occur between the read of the status bits and the write to clear the status bits, the status bits should be used for statistics gathering only.

2.6 Interrupts

The AAL1gator II asserts the interrupt line whenever an OAM cell arrives and the interrupt mask is deasserted. The interrupt can be cleared by asserting OAM_INT_MASK or by asserting CLR_RX_OAM_LATCH. Refer to section 7.9 “CMDREG (Command Register)” on page 165.

2.7 SRTS and Transmit Line Interface Clock Configurations

The AAL1gator II provides the following transmit line interface clock configurations on a per-line basis:

- Drives TL_CLK(i) with the looped-back RL_CLK(i).
- Drives TL_CLK(i) with the SRTS-derived clock.
- Drives TL_CLK(i) with a nominal T1/E1 clock of 1.544 MHz or 2.048 MHz.
- Accepts TL_CLK(i) as an input.

NOTE: The AAL1gator II uses Bellcore’s patented SRTS clock recovery technique. Refer to the NOTE on page 172 for additional information regarding Bellcore’s SRTS patent.

The AAL1gator II also provides the following external TL_CLK generation options through the multiplexed SRTS output port. These options can be used when the AAL1gator II accepts TL_CLK(i) as an input:

- The AAL1gator II drives SRTS information out of the SRTS port. An external circuit can use this information to synthesize an SRTS-based TL_CLK. (Provided for backward compatibility with the WAC-021-C-X.)
- The AAL1gator II drives adaptive clocking information out of the SRTS port. An external circuit can use this information to synthesize an adaptive (based on receive-buffer-centering) TL_CLK.

2.8 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.

2.8.1 Peak Cell Rates (PCRs) for Structured Cell Formats

- PCR $\leq 176 \times n$ cells per second where $1 \leq n \leq 32$ (assuming completely filled cells).

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

2.8.2 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 lines are at the same rate, the aggregate 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 PCR ≤ 46,542.
- PCR ≤ 1,000 cells per second 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.

2.8.3 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 equations can be used to determine the PCR when partial cells are used.

First, the maximum average cell generation time (MACGT) is calculated. The MACGT value is equal to the maximum average number of cycles to generate a cell, times the length of the clock cycle. Second, the maximum average cell generation time is multiplied by the number of cells required per second to give the total cell generation time (TCGT).

The following equations are given for different modes of operation. Each equation is calculated on a per-queue basis. If all the queues are configured the same, then the total for a queue can be multiplied by the number of queues. Otherwise, the sum of the calculations for each queue needs to be totaled. The sum must be less than one. The following variables are used:

K = # of data bytes per cell
N = number of timeslots assigned to queue
L = length of clock cycle

NOTE: If OAM cells are used, add the following equation for the OAM cells:

$TCGT_{OAM} = 212 \times L \times$ (maximum number of OAM cells per second)

2.8.3.1 SDF-MF Mode

$$MACGT_{SDFMF} = [208 + (8 \times K)] \times L$$

2.8.3.1.1 For E1 Mode

$$TCGT_{SDFMFE1} = MACGT_{SDFMF} \times (8000 \times N \times 33 \div 32) \div K$$

If all queues are E1 SDF-MF and the same size, all timeslots are used, and SYS_CLK= 38.88 MHz, then the minimum number of bytes per cell is 20.

2.8.3.1.2 For T1 Mode

$$TCGT_{SDFMFT1} = MACGT_{SDFMF} \times (8000 \times N \times 49 \div 48) \div K$$

If all queues are T1 SDF_MF the same size, all timeslots are used, and SYS_CLK= 38.88 MHz, then the minimum number of bytes per cell is 13.

2.8.3.2 SDF-FR Mode

$$MACGT_{SDFFR} = [202 + (8 \times K)] \times L$$

$$TCGT_{SDFFR} = MACGT_{SDFFR} \times (8000 \times N) \div K$$

For E1 mode, if all queues are the same size, all timeslots are used, and SYS_CLK= 38.88 MHz, then the minimum number of bytes per cell is 19.

For T1 mode, if all queues are the same size and all timeslots are used and SYS_CLK= 38.88 MHz, then the minimum number of bytes per cell is 12.

2.8.3.3 UDF-ML Mode

$$MACGT_{UDFML} = [202 + (4 \times K)] \times L$$

$$TCGT_{UDFML} = MACGT_{UDFML} \times (8000 \times 32) \div K$$

If all queues are unstructured and all lines are used and SYS_CLK= 38.88 MHz, then the minimum number of bytes per cell is 14.

The sum of $TCGT_{SDFMFE1}$ + $TCGT_{SDFMFT1}$ + $TCGT_{SDFFR}$ + $TCGT_{UDFML}$ for all queues + $TCGT_{OAM}$ must be less than 1 second.

3 THEORY OF OPERATIONS

The AAL1gator II is divided into the following major blocks, all of which are explained in this section:

- Transmit Frame Transfer Controller (TFTC) block
- Cell Service Decision (CSD) block
- Transmit Adaptation Layer Processor (TALP) block
- Transmit UTOPIA Interface (TUTOPIA) block
- Memory Interface and Arbitration Controller (MIAC) block
- Receive Frame Transfer Controller (RFTC) block
- Receive Adaptation Layer Processor (RALP) block
- Receive UTOPIA Interface (RUTOPIA) block

Figure 10 shows a block diagram of the AAL1gator II and the sequence of events used to segment and reassemble the CBR data.

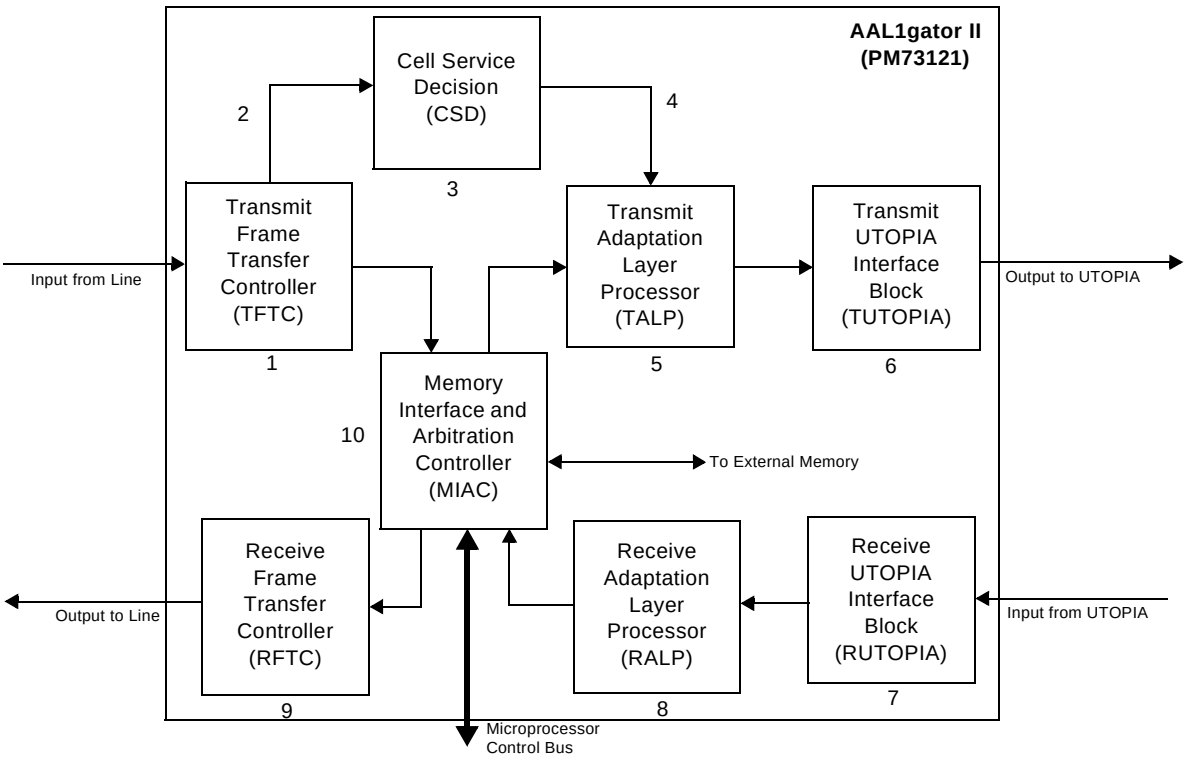


Figure 10. AAL1gator II Block Diagram

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 UTOPIA FIFO.
6. The TUTOPIA interface transmits the cells out of the UTOPIA port.
7. The RUTOPIA interface receives the cell from the UTOPIA port.
8. 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.
9. The RFTC plays the receiver buffer data onto the lines.
10. The MIAC provides arbitration for all memory transaction requests by the various processes.

3.1 Transmit Frame Transfer Controller (TFTC)

The TFTC accepts deframed data from the external T1 or E1 framer device. For structured data, the TFTC uses the synchronization signals (RL_FSYNC and RL_MSYNC) supplied by the framer 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.

A rising edge on RL_FSYNC indicates the beginning of a frame, and a rising edge on RL_MSYNC indicates the beginning of a multiframe. The TFTC will realign when an edge is seen on these signals. It is not necessary to provide an edge at the beginning of every frame or multiframe. The AAL1gator II reads signaling during the last frame of every multiframe. For T1 mode, the AAL1gator II reads signaling on the 24th frame of the multiframe. For E1 mode, the AAL1gator II reads signaling on the 16th frame of 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 in 32 timeslots. In this mode, the AAL1gator II reads signaling on the 24th frame of the multiframe.

The AAL1gator II reads the signaling nibble for each channel when it reads the last nibble of each channel's data. See Figure 11 on page 26 for an example of a T1 frame. See Figure 12 on page 26 for an example of an E1 frame.

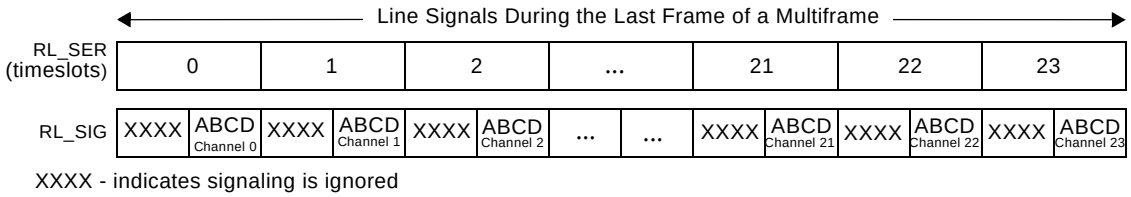


Figure 11. Capture of T1 Signaling Bits

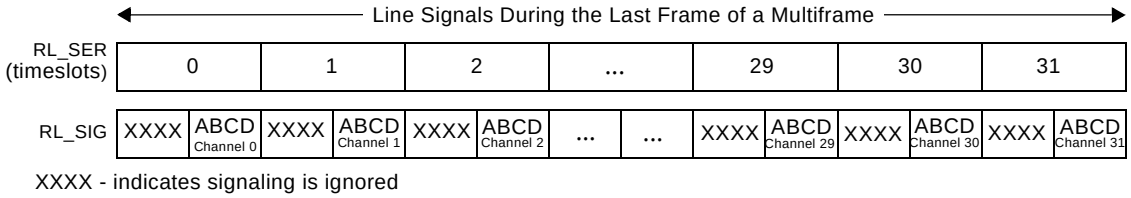


Figure 12. Capture of E1 Signaling Bits

NOTE: The AAL1gator II 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 13 shows the basic components of the TFTC.

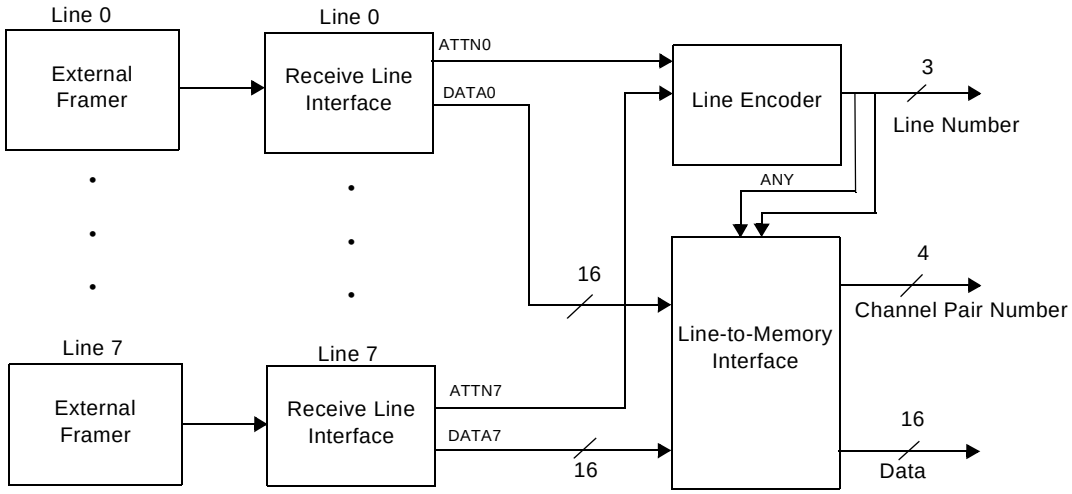


Figure 13. Transmit Frame Transfer Controller

The receive line interface is primarily a serial-to-parallel converter. Serial data, which is derived from the RL_SER pin, 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 132 for more information on the transmit signaling buffer.

Figure 14 shows the format of the transmit data buffer for ESF-formatted T1 data for lines that are in the SDF-MF mode.

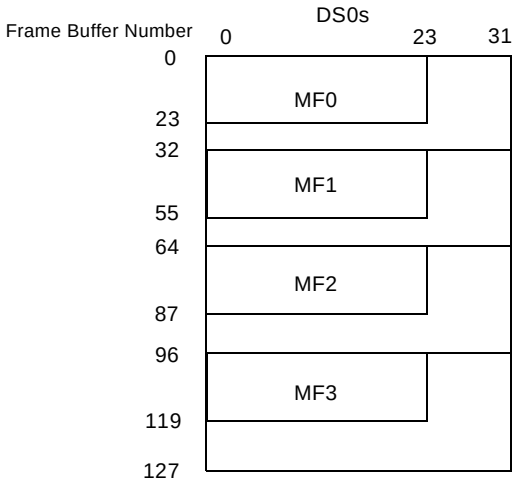


Figure 14. T1 ESF SDF-MF Format of the T_DATA_BUFFER

Figure 15 shows the format of the transmit data buffer for SF-formatted T1 data for lines that are in the SDF-MF mode.

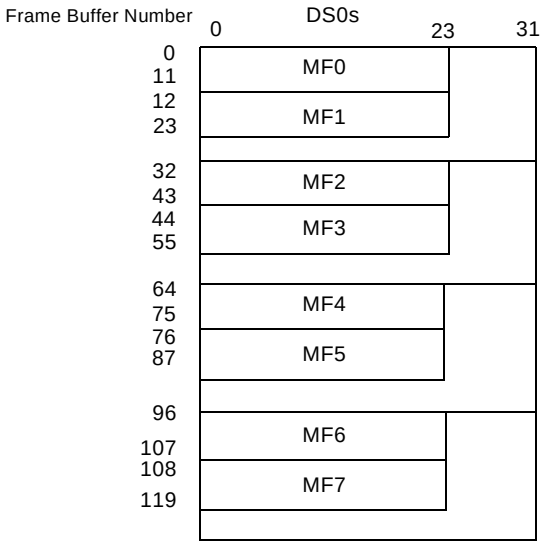


Figure 15. T1 SF SDF-MF Format of the T_DATA_BUFFER

Figure 16 shows the format of the transmit data buffer for T1 data for lines that are in the SDF-FR mode.

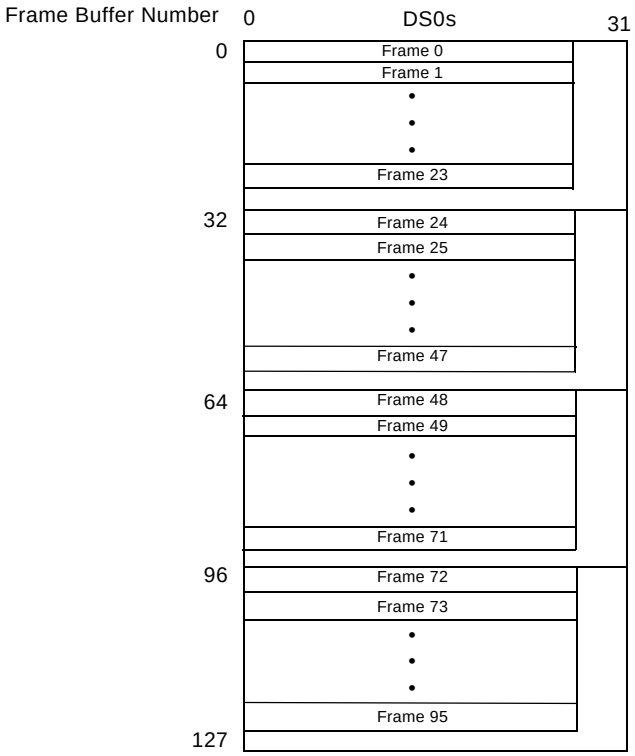


Figure 16. T1 SDF-FR Format of the T_DATA_BUFFER

Figure 17 shows the format of the transmit data buffer for E1 data for lines that are in the SDF-MF mode.

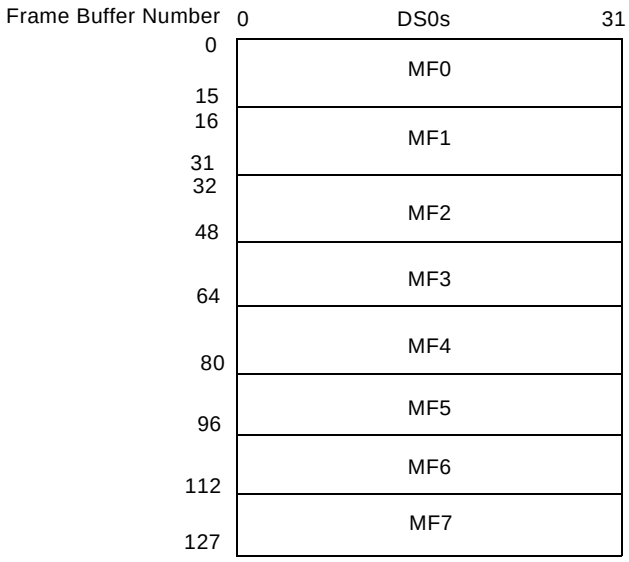


Figure 17. E1 SDF-MF Format of the T_DATA_BUFFER

Figure 18 shows the format of the transmit data buffer for E1 data using T1 signaling, for lines that are in SDF-MF mode.

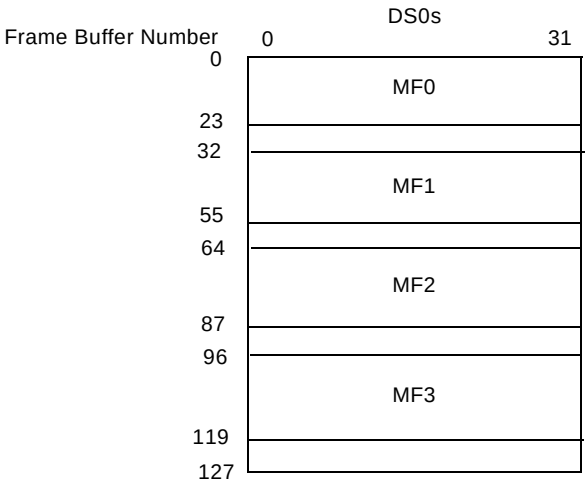


Figure 18. E1 SDF-MF with T1 Signaling Format of T_DATA_BUFFER

Figure 19 shows the format of the transmit data buffer for E1 data for lines that are in the SDF-FR mode.

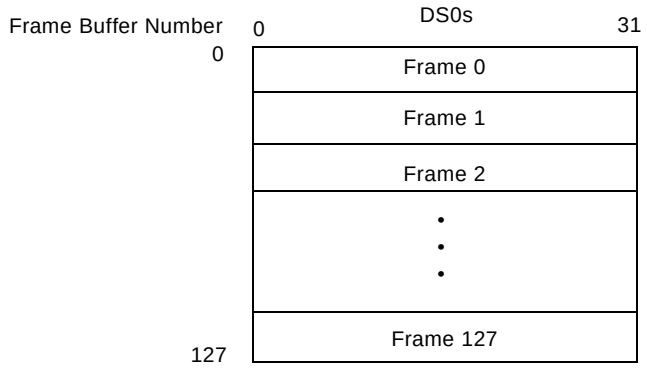


Figure 19. E1 SDF-FR Format of the T_DATA_BUFFER

Figure 20 shows the format of the transmit data buffer for lines that are in UDF-ML mode.

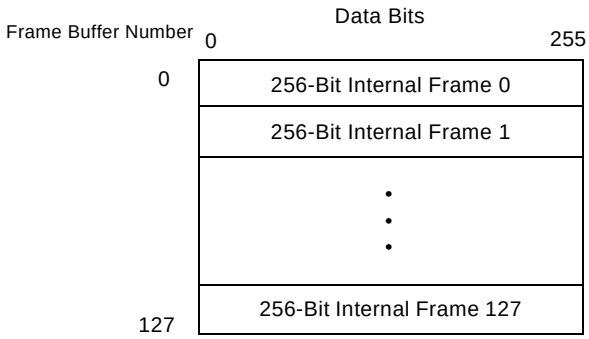


Figure 20. Unstructured Format of the T_DATA_BUFFER

and Figure 21 shows the contents of the transmit signaling buffer for the different signaling modes.

	15 Bit 0	
Word 0	CH1	CH0
1	3	2
2	5	4
3	7	6
4	9	8
5	11	10
6	13	12
7	15	14
8	17	16
9	19	18
10	21	20
11	23	22
12	25	24
13	27	26
14	29	28
15	CH31	CH30

NOTE: The upper nibble of each byte is 0.
T1 format uses word addresses 0 to 11.

Figure 21. SDF-MF Format of the T_SIGNALING_BUFFER (T1 and E1 Modes)

3.1.1 Transmit Conditioning

The T_COND_DATA structure allows conditional data to be defined on a per-DS0 basis. The structure T_COND_SIG 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 3.3 “Transmit Adaptation Layer Processor (TALP)” on page 38) to be directed to build cells from the conditioned data and signaling. The TX_COND bit can be set on a per-queue basis.

3.1.2 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.

3.1.3 SRTS for the Transmit Side

NOTE: The AAL1gator II uses Bellcore’s patented SRTS clock recovery technique. Refer to the NOTE on page 172 for additional information regarding Bellcore’s SRTS patent.

The transmit side supports SRTS only for unstructured data formats on a per-line basis. SRTS support requires an input reference clock, N_CLK. 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 × RL_CLK > N_CLK > 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 78 MHz, which is required for T3 applications. Figure 22 on page 33 shows the process implemented for each UDF line enabled for SRTS, regardless of the reference frequency. The resulting 4-bit SRTS code is then inserted into the CSI bit of the odd numbered cells for that line. 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 × 8 × 47). The divider is aligned on the first cell generation after a reset or a resynchronization to the cell generation process.

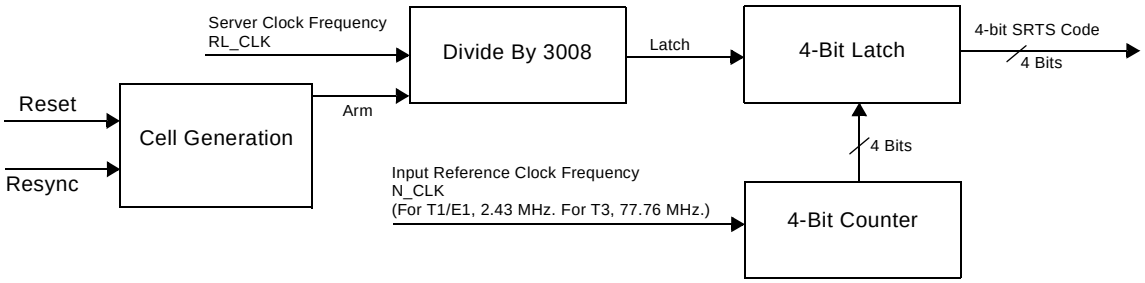


Figure 22. Transmit Side SRTS Support

3.2 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 IDLE_CONFIG word is set, then the cell is scheduled, however, the cell is not transmitted.

The following steps (as well as Figure 23 on page 35) 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 result is a frame differential.

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

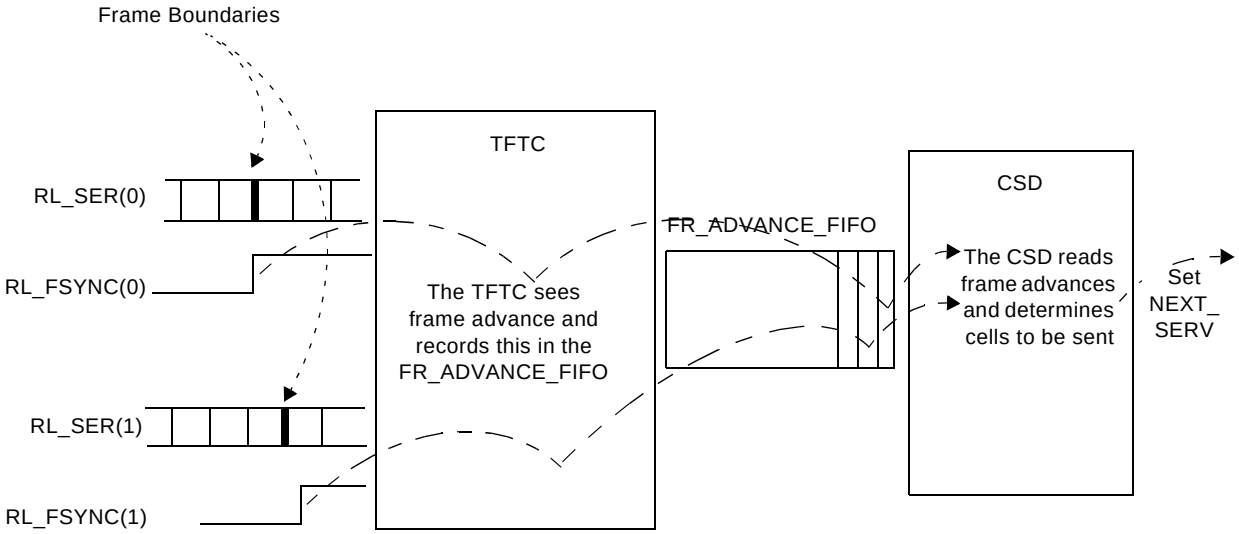


Figure 23. Frame Advance FIFO Operation

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 = $9 \times 4 = 36$
   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 $\times$ 0.5 bytes per channel = 2 bytes per multiframe
   Signaling adjustment = three eighths $\times$ 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 the same procedure. In the case of unstructured lines, the number of channels allocated to the queue is 32. Because there is never a pointer, the average number of credits spent per cell is always 47.

3.2.1 Transmit CDV

The following items affect transmit CDV:

- Cell scheduling,
- Contention with other cells scheduled at same time,
- Actual cell build time, and
- UTOPIA contention.

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.

   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 125 µs of CDV due to the scheduler.

2. Only one cell can be built at a time. Thus if multiple queues are scheduled to send cells during the same frame, additional CDV will be incurred. Since it takes approximately 8 µs to build a cell, each cell that is waited for adds 8 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. Theoretically, it is possible to have all 256 queues scheduled at the same time, however, this will not occur if adequate steps are taken.

   Minimize CDV due to clumping by the following actions:

   Add queues one at a time (PMC-Sierra's driver does this automatically). When queues are added at the same time with the same configuration, they are clumped together. By adding queues one at a time, clumping can be avoided and cell scheduling points can be evenly distributed. Also, queues are only added in frame 0. Since the transmit buffer is 128 frames (for E1) or 96 frames (for T1), this equates to 16 ms (for E1) or 12 ms (for T1). Queues should be added more than 16 ms apart (for E1) or 12 ms (for T1) to prevent them from being clumped together.

   Avoid configurations that will require sending a cell every *n* frames where *n* is an integer divi-

sor of 128 (for E1) or 96 (for T1). For these configurations the queues will always be scheduled in the same frame. Therefore, even if these queues are added at different times they will still be scheduled at the same time.

Staggering clocks of the different lines can also help. Since cells are scheduled when a frame completes, staggering lines can offset the frame completion point for the different lines with respect to each other.

3. The actual build time of a cell depends on microprocessor activity and contention with other internal state machines for the AAL1gator II 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.
4. If there is backpressure on the UTOPIA bus, cells will not be able to be sent which also causes CDV.

3.3 Transmit Adaptation Layer Processor (TALP)

3.3.1 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 II to maintain its maximum CSD data rate.

To send an OAM cell, the Supervisory Processor (SP) writes supervisory cells into one of two dedicated cell buffers located in external memory. When the cell is assembled in the buffer, the SP must set the appropriate bit in the command register (refer to section 7.9 “CMDREG (Command Register)” on page 165). 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 II, OAM cell 0 will always be sent because it is assigned a higher priority. Therefore, to control the order of OAM cell transmission, the SP 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.

3.3.2 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 134). 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.

3.3.2.1 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 IDLE_CONFIG register. Emissions are still scheduled, just the transmissions are suppressed. For any cells that are suppressed, the T_SUPPRESSED_CELL_CNT is incremented.

3.3.2.2 Payload Construction

Payload construction is the most complex task the TALP circuit performs. The signaling 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.
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 (refer to section 7.6.8 “T_QUEUE_TBL” on page 134) 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. 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 24 shows an example of the payload generation process.

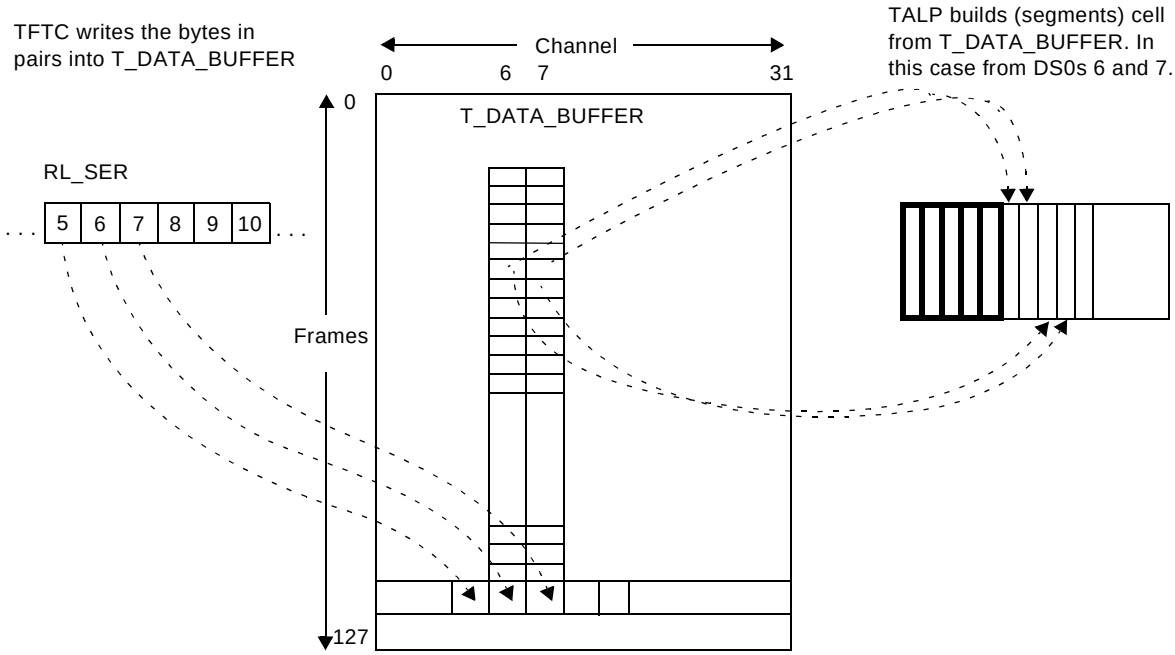


Figure 24. Payload Generation

3.4 Transmit UTOPIA Interface Block (TUTOPIA)

The TUTOPIA block (shown in Figure 10 on page 24) conveys the cells emitted by the TALP to the UTOPIA interface. Depending on the value of PHY_ENABLE input pin (refer to "PHY_ENABLE" on page 81), the UTOPIA interface will either act as an ATM side (controls the write enable signal) or as a PHY side (controls the cell available signal). As a PHY-side device, the TUTOPIA block can also act as an SPHY device or a MPHY device (refer to Appendix B, "References", on page 203 for the UTOPIA Level 2 specification version), depending on the value of the SPHY_EN bit in COMP_LIN_REG. The SPHY_EN bit will default to off, so if the UTOPIA interface is being used in an MPHY environment, there will be no contention while the device is in software reset.

In ATM mode, the TUTOPIA block sources TATM_DATA(7:0), TATM_SOC, and /TATM_EN while receiving /TATM_FULL. The Start-Of-Cell (SOC) indication is generated coincident with the first byte of each cell that is transmitted on TATM_DATA. TATM_DATA 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_EN signal until it has a full cell to send.

The TUTOPIA responds to the /TATM_FULL signal. If the signal is not asserted, and the TUTOPIA has data to send, it will do so by asserting /TATM_EN. If /TATM_FULL is asserted, then the data flow will be paused in exactly two clock cycles, until the /TATM_FULL signal allows the cell to be completed. See the timing diagram in Figure 25 on page 42.

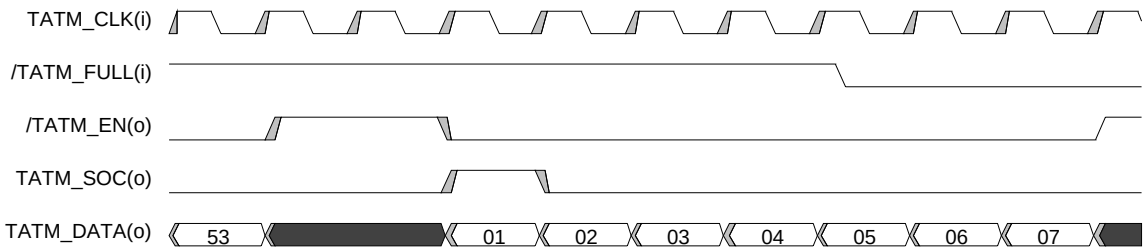


Figure 25. Transmit UTOPIA Timing (ATM Mode)

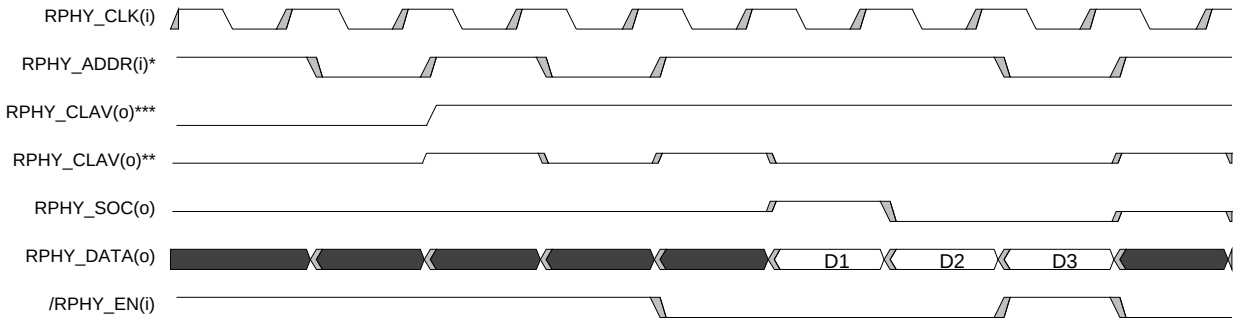
In PHY mode, the TUTOPIA block sources RPHY_DATA(7:0), RPHY_SOC, and RPHY_CLAV, while receiving /RPHY_EN. The SOC indication is generated coincident with the first byte of each cell that is transmitted on RPHY_DATA. In PHY mode, the RATM_DATA and RATM_SOC signals are driven only when valid data is being sent; otherwise they are tristated. /RPHY_ADDR is an input and is used only in MPHY mode.

The cell available (RPHY_CLAV) signal indicates when the device has a complete cell to send. In SPHY mode, RPHY_CLAV is always driven. If in MPHY mode, the output enable for this signal is the /RPHY_ADDR input delayed by one cycle. In MPHY mode, /RPHY_ADDR is tied to one of the address signals so RPHY_CLAV is driven only when polled.

The UTOPIA standard defines a 5-bit address. Since the AAL1gator II has only a single active low address bit, multiple AAL1gator II devices can be connected in parallel to the same MPHY interface by connecting each one to a separate address bit. In this manner, five AAL1gator IIs can be connected to an MPHY interface using the following addresses: "$0F_h$", "17_h", "$1B_h$", "$1D_h$", and "$1E_h$" with no additional logic. If other addresses are needed or additional devices are to be connected to the same interface, additional logic may be required.

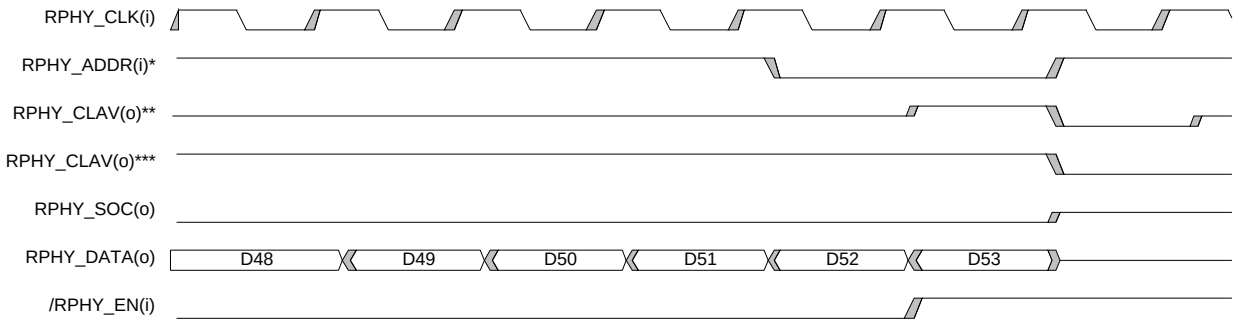
If a cell is being read in SPHY mode, or is being read from a channel that is being polled in MPHY mode, the value of RPHY_CLAV will be 1 until the cell has been read out of the FIFO and there are no more cells to send. In PHY mode, the TUTOPIA block is dependent on the ATM device to read the data by asserting the /RPHY_EN input. In SPHY mode, data is placed on RPHY_DATA any cycle following one in which /RPHY_EN was asserted. In MPHY mode, in

order to output data, the TUTOPIA block also has to be selected, which is done when /RPHY_ADDR is low on the falling edge of /RPHY_EN. See Figure 26 on page 43 and Figure 27 on page 43 for the TUTOPIA transfer timing diagrams.



* Used only in MPHY mode
** For MPHY mode
*** For SPHY mode

Figure 26. TUTOPIA Start-of-Transfer Timing (PHY Mode)



* Used only in MPHY mode
** For MPHY mode
*** For SPHY mode

Figure 27. TUTOPIA End-of-Transfer Timing (PHY Mode)

The AAL1gator II can tolerate temporary deassertions of /TATM_FULL(/RPHY_EN), but it is assumed that enough UTOPIA bandwidth is present to accept the cells that the AAL1gator II can produce in a timely manner. The AAL1gator II can tolerate a deassertion of /TATM_FULL(/RPHY_EN) for up to 128 line interface frames, but this would cause excessive CDV. The TALP circuit writes cells one byte at a time into the FIFO. The SOC is also placed into the FIFO.

Clocking for the read side is derived from the system clock. If the FIFO is full, the cell building process stalls until space becomes available. If the FIFO remains full, additional queues will not be able to be added. This will be indicated to the processor by the CSD_ATTN bit not being cleared after setting that bit.

The TUTOPIA circuit controls when a cell is transmitted from the 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 TUTOPIA 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_FULL (/RPHY_EN) input. The maximum supported clock rate is 33 MHz.

3.5 Receive UTOPIA Interface Block (RUTOPIA)

The RUTOPIA block receives cells from the UTOPIA interface and sends them to the RALP interface. Depending on the value of the PHY_ENABLE input pin, the UTOPIA interface acts either as an ATM side (controls the read enable signal) or as a PHY side (controls the cell available signal). As a PHY-side device, the RUTOPIA block can also act as an SPHY device or as an MPHY device (refer to the UTOPIA Level 2 specification in Appendix B, “References”, on page 203) depending on the value of SPHY_EN in COMP_LIN_REG. The SPHY_EN bit defaults to MPHY mode, so if the device is in an MPHY environment, there will not be any contention while the chip is in software reset.

In ATM mode, the RUTOPIA block receives RATM_DATA(7:0), RATM_SOC, and /RATM_EMPTY while driving /RATM_EN. At reset the RUTOPIA block activates /RATM_EN to prevent excessive queueing in the system. After the end of reset, if the /RATM_EMPTY input signal is not asserted, the RUTOPIA 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. A small intermediate FIFO allows the interface to accept data at the maximum rate. If the FIFO fills, the /RATM_EN signal will not be asserted again until the device is ready to accept an entire cell. The /RATM_EN signal depends only on the cell space and is independent of the state of the /RATM_EMPTY signal. See Figure 28 for the RUTOPIA timing diagram.

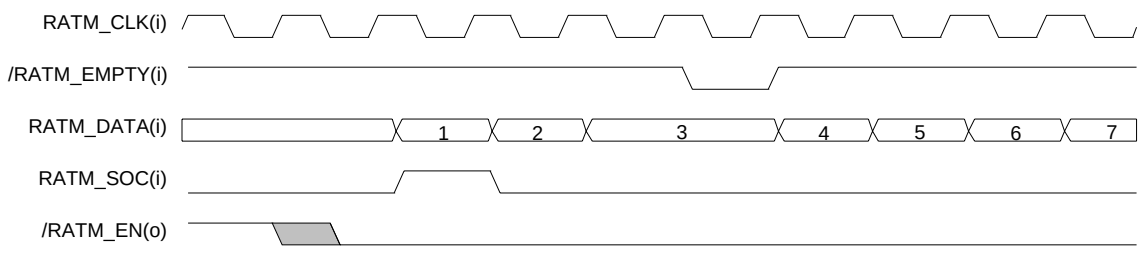


Figure 28. Receive UTOPIA Timing (ATM Mode)

The RUTOPIA block waits for an SOC. When an RATM_SOC signal arrives, a counter is started, and 53 bytes are received. If a new RATM_SOC occurs within a cell, the counter reinitializes. This means that the corrupted cell will be dropped and the good cell will be received. The RUTOPIA block stores the ATM cell in the receive FIFO. If the receive FIFO becomes full, it stops receiving ATM cells from the PHY layer. The /RATM_EMPTY signal can also turn on or off any time, even if the /RATM_EN signal is on. As a result, an incoming byte is valid only when /RATM_EN is active and /RATM_EMPTY is inactive. The bytes are written to the FIFO with RATM_CLK. RATM_CLK is an input to the AAL1gator II. The maximum supported clock rate is 33 MHz.

In PHY mode, the RUTOPIA block receives TPHY_DATA(7:0), TPHY_SOC, and /TPHY_EN while driving TPHY_CLAV. The cell available (TPHY_CLAV) signal indicates when the device is ready to receive a complete cell. In SPHY mode, TPHY_CLAV is always driven. In MPHY mode, the output enable for this signal is the /TPHY_ADDR input delayed by one cycle. In MPHY mode, /TPHY_ADDR is tied to one of the address signals so that TPHY_CLAV is driven only when polled.

The UTOPIA standard defines a 5-bit address. Since the AAL1gator II has only a single active low address bit, multiple AAL1gator II devices can be connected in parallel to the same MPHY interface by connecting each one to a separate address bit. In this manner, five AAL1gator IIs can be connected to an MPHY interface using the following addresses: "$0F_h$", "17_h", "$1B_h$", "$1D_h$", and "$1E_h$" with no additional logic. If other addresses are needed or additional devices are to be connected to the same interface, additional logic may be required.

At reset, the RUTOPIA block tristates TPHY_CLAV. After the end of reset, the RUTOPIA block waits for /TPHY_EN to be asserted, and in SPHY mode, will accept data as long as /TPHY_EN is asserted. In MPHY mode, /TPHY_ADDR must be low on the falling edge of /TPHY_EN in order for the RUTOPIA block to accept the data. When SOC is detected, a counter is started and 53 bytes are received. If a new TPHY_SOC occurs within a cell, the counter reinitializes. This means that the corrupted cell will be dropped and the good cell will be received. A small intermediate FIFO allows the interface to accept data at the maximum rate. If the FIFO fills, the TPHY_

CLAV signal will not be asserted again until the device is ready to accept an entire cell. If the FIFO is filling, TPHY_CLAV will be deasserted when there is room for only four more bytes of data. See Figure 29 and Figure 30 for the RUTOPIA transfer timing diagrams.

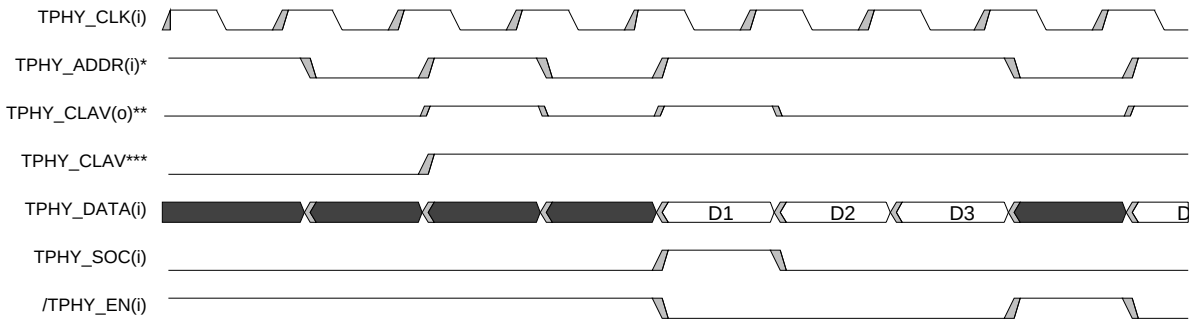


Figure 29. RUTOPIA Start-of-Transfer Timing

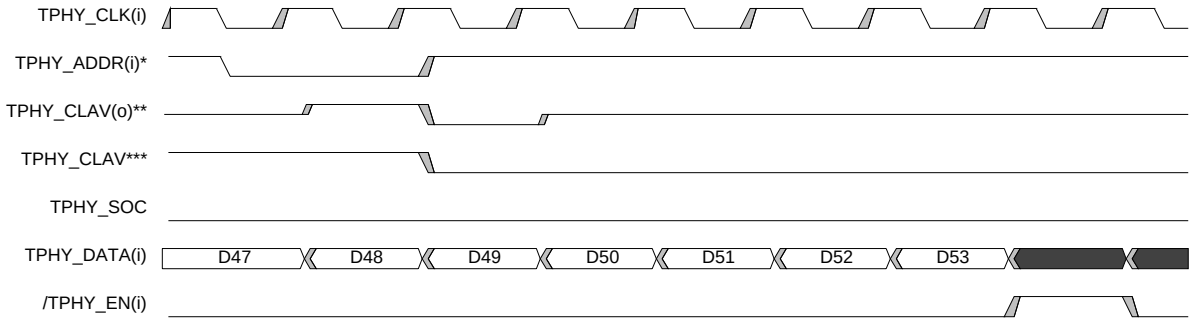


Figure 30. RUTOPIA End-of-Transfer Timing

3.6 Receive Adaptation Layer Processor (RALP)

The RALP moves data from the receive FIFO ATM cells in the RUTOPIA interface block to the external memory. 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 II.

If SHIFT_VCI = 0, cells received with VCI(8) = 0 or Payload Type Indicator (PTI) = 4 to 7 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. Figure 31 shows the interpretation of the incoming cell header.

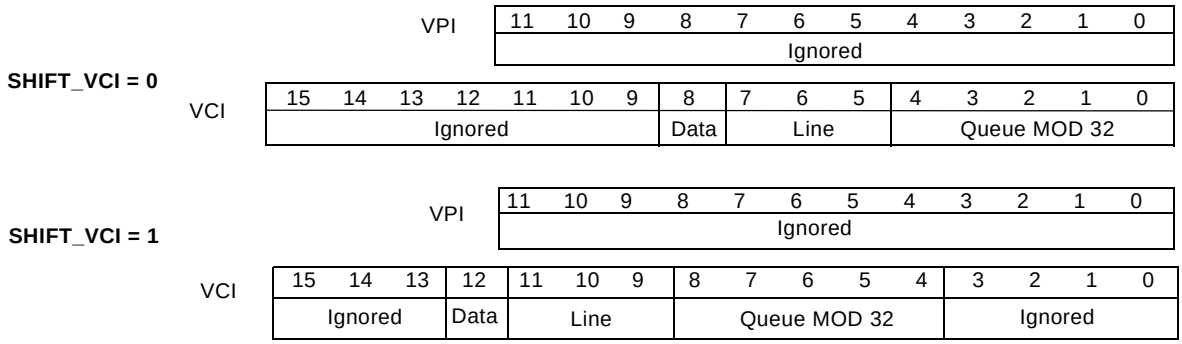


Figure 31. Cell Header Interpretation

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

When the cell is a data cell, the RALP verifies the SN CRC. If the SN CRC does not verify, the RALP attempts to correct it as specified in ITU-T Recommendation I.363.1 and increments the R_INCORRECT_SNP counter (refer to "R_INCORRECT_SNP Word Format" on page 158). If the RALP cannot correct the cell, it identifies the cell as invalid.

If the DISABLE_SN bit (refer to "DISABLE_SN" on page 160) has not been set for this queue, then sequence number processing is done in accordance with the "Fast SN Algorithm" described in ITU-T Recommendation I.363.1. 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. 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 in R_MP_CONFIG (refer to "R_MP_CONFIG Word Format" on page 155).
- 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:
  - 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 no signaling information is written to the signaling buffer. Therefore, the old signaling information will be played out.

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_MAX_BUF word (refer to "R_MAX_BUF Word Format" on page 157).

- The value of the payload data depends on the value of INSERT_DATA in R_SN_CONFIG (refer to "R_SN_CONFIG Word Format" on page 160). 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:
- 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.
- 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 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.
- 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 cells, or if there are up to six lost cells and the queue does not underrun. To maximize the tolerance of the RALP to errored cells, the R_CDVT and R_MAX_BUF values should be increased a little to

guarantee that the queue does not underrun in situations that could be handled by the sequence number processing algorithm. If bit integrity is lost, then the queue will be forced into an underrun condition and realign with the structure if one exists.

NOTE: SN processing takes place independent of whether or not a queue is in underrun. If cells are detected lost while transitioning through an underrun state, or into a forced underrun state, no cells will be inserted.

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. 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 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:

1. 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.
2. No signaling information will be updated until a valid and correct pointer is received.

If DISABLE_SN (refer to “DISABLE_SN” on page 160) is set, no SN or SNP processing occurs. That is, no cells will be inserted or dropped but statistics will be kept.

The RALP 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 (refer to “NODROP_IN_START” on page 161) 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.

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

- If the cell received has an invalid SN, the START state is re-entered and the cell is dropped.
- 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.
- 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.
- 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.
- 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 32 on page 51 for the flow of the “Fast SN Algorithm”.

Anytime a cell is dropped, the R_DROPPED_CELLS counter (refer to “R_DROPPED_CELLS Word Format” on page 161) is incremented and the SN_CELL_DROP sticky bit (refer to “SN_CELL_DROP” on page 158) is set. Anytime a cell is detected lost, the R_LOST_CELLS counter (refer to “R_LOST_CELLS Word Format” on page 162) 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 (refer to "R_SEQUENCE_ERR Word Format" on page 157) is incremented. Anytime a misinserted cell is detected the R_MISINSERTED counter (refer to "R_SEQUENCE_ERR Word Format" on page 157) is incremented.

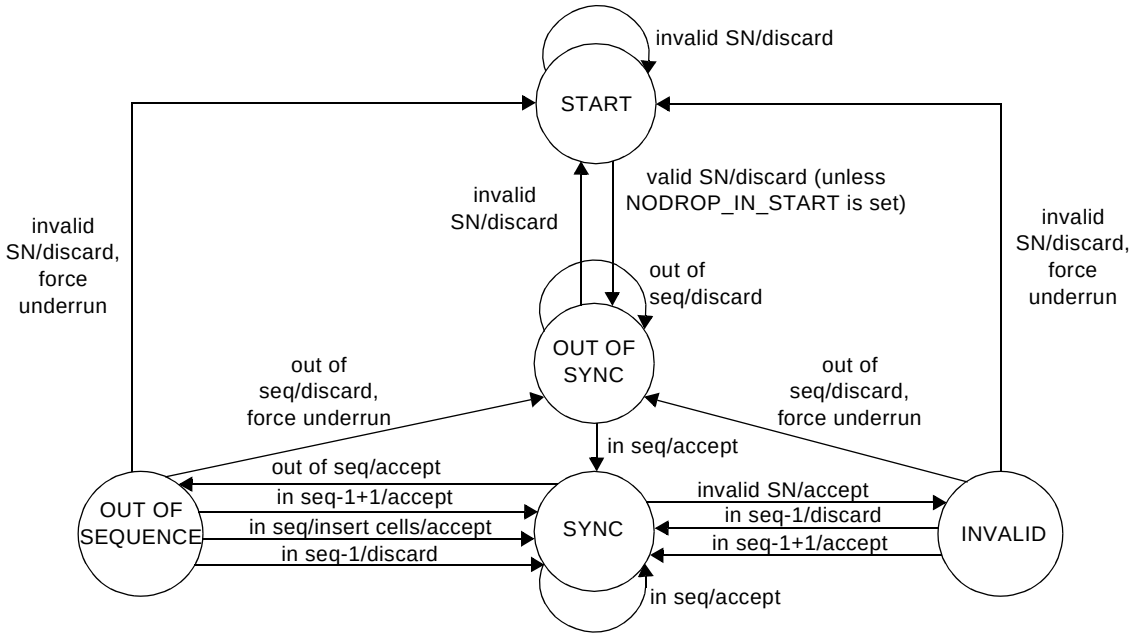


Figure 32. Fast SN Algorithm

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, 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 32 shows this receive cell process. If a valid cell is not received in time, the queue may enter an underrun condition.

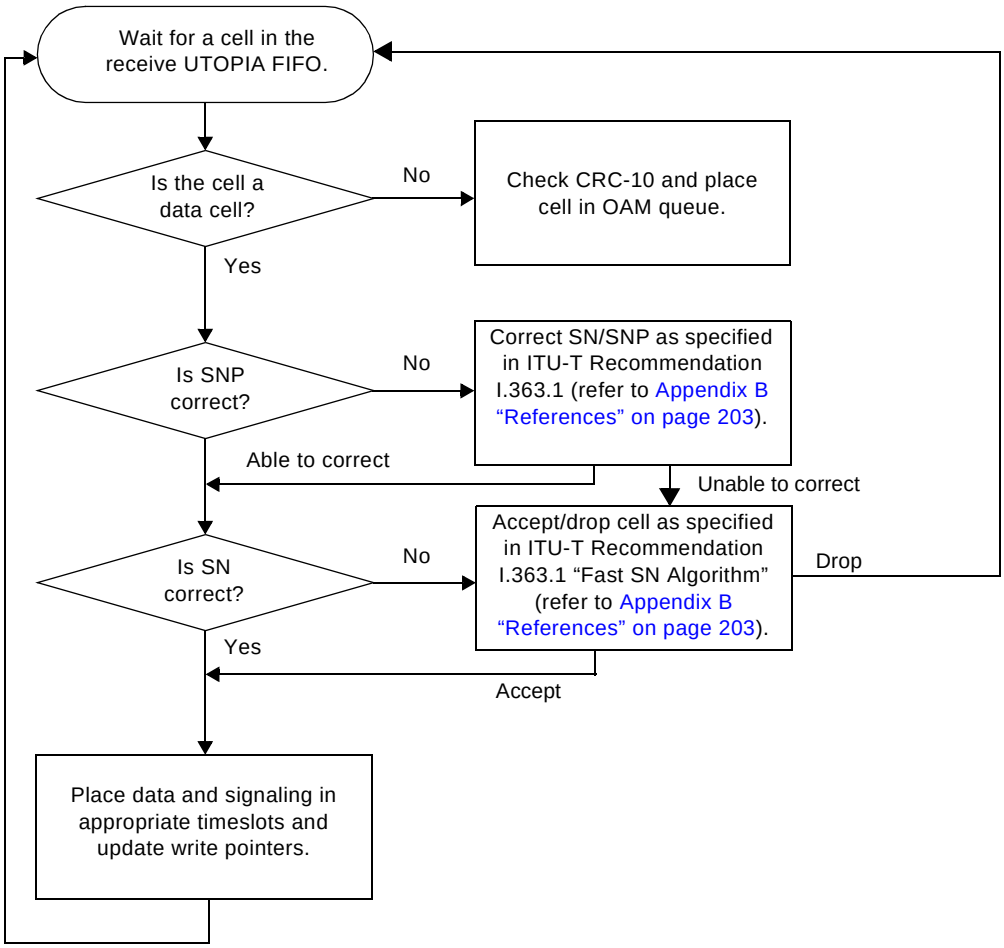


Figure 33. Receive Cell Processing

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 II also provides protection from buffer overrun and pointer misalignment.

The RALP reads the ATM cell from the RUTOPIA 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. Since there are eight lines, 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 data. 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 34 on page 53 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.

Figure 34 and the figures that follow illustrate these points and identify how different data formats are stored in the data and signaling buffers.

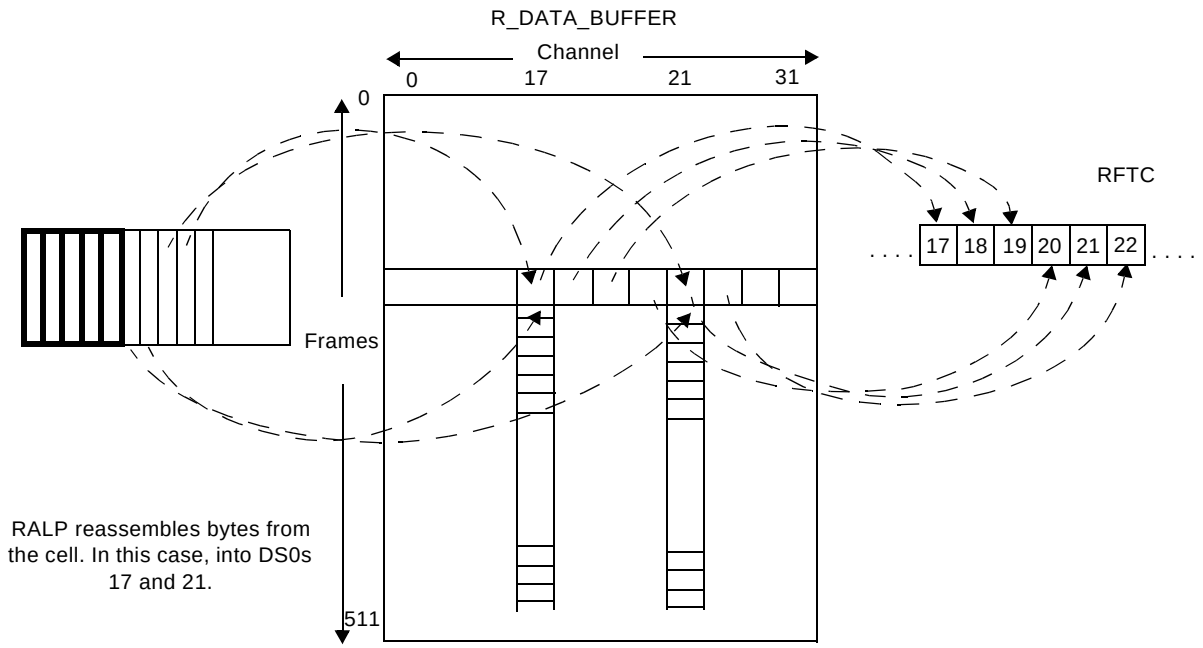


Figure 34. Cell Reception

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 35 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.

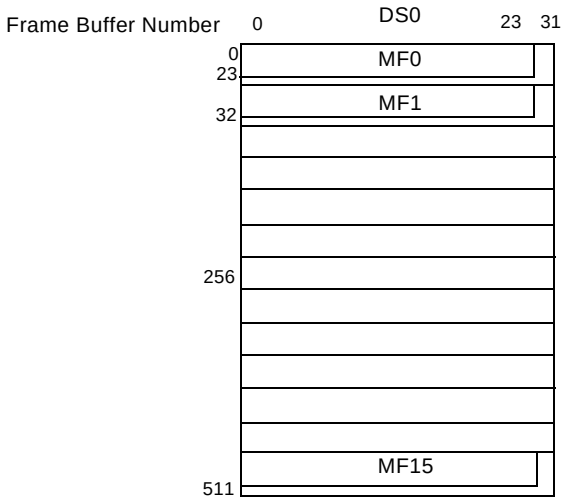


Figure 35. T1 ESF SDF-MF Format of the R_DATA_BUFFER

Figure 36 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.

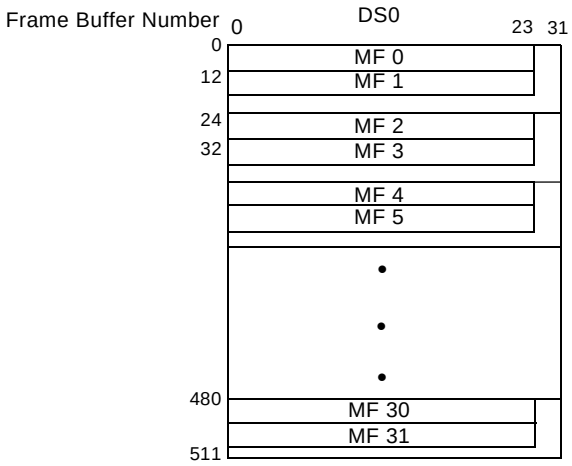


Figure 36. T1 SF SDF-MF Format of R_DATA_BUFFER

Figure 37 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.

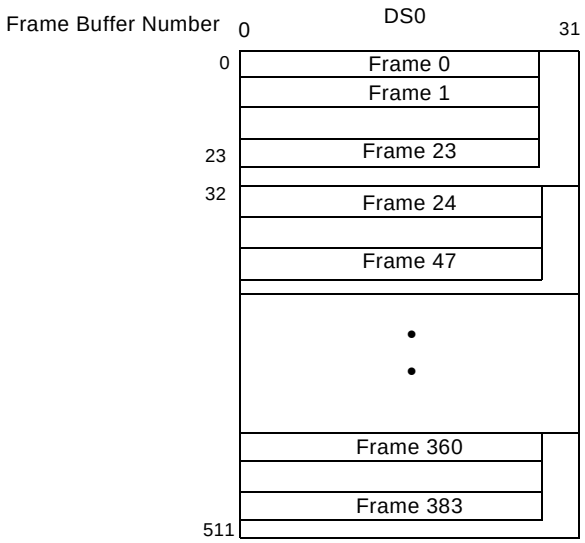


Figure 37. T1 SDF-FR Format of the R_DATA_BUFFER

Figure 38 shows the contents of the receive buffer with E1 data for lines in the SDF-MF mode.

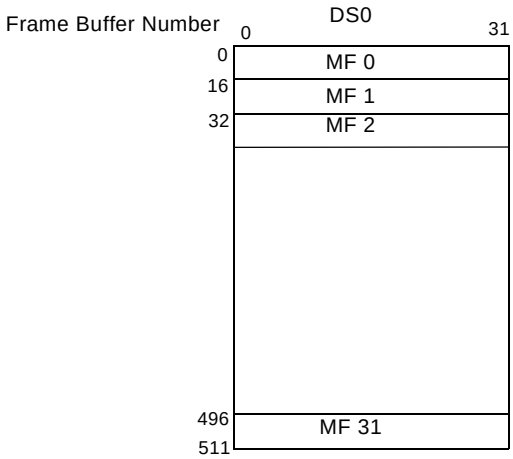


Figure 38. E1 SDF-MF Format of the R_DATA_BUFFER

Figure 39 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.

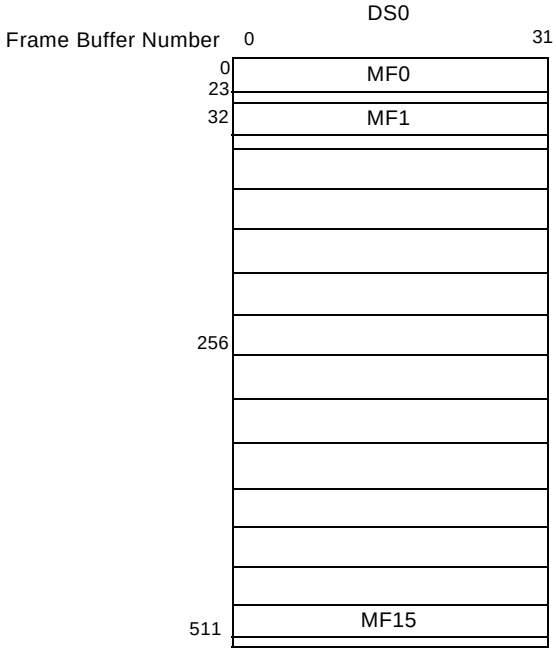


Figure 39. E1 SDF-MF with T1 Signaling Format of the R_DATA_BUFFER

Figure 40 shows the contents of the receive data buffers with E1 data for lines in the SDF-FR mode.

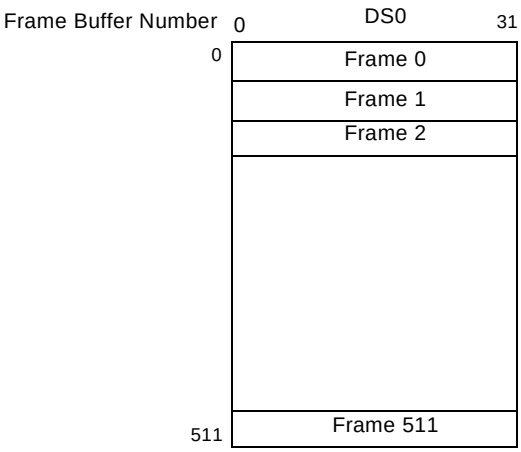


Figure 40. E1 SDF-FR Format of the R_DATA_BUFFER

Figure 41 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 consists of 192 bits. Therefore, each frame buffer contains 1.33 frames of T1 data. This must be considered when determining CDVT numbers.

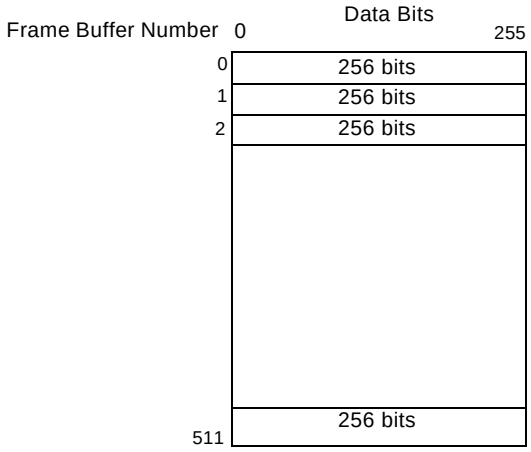


Figure 41. Unstructured Format of the R_DATA_BUFFER

Figure 42 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.

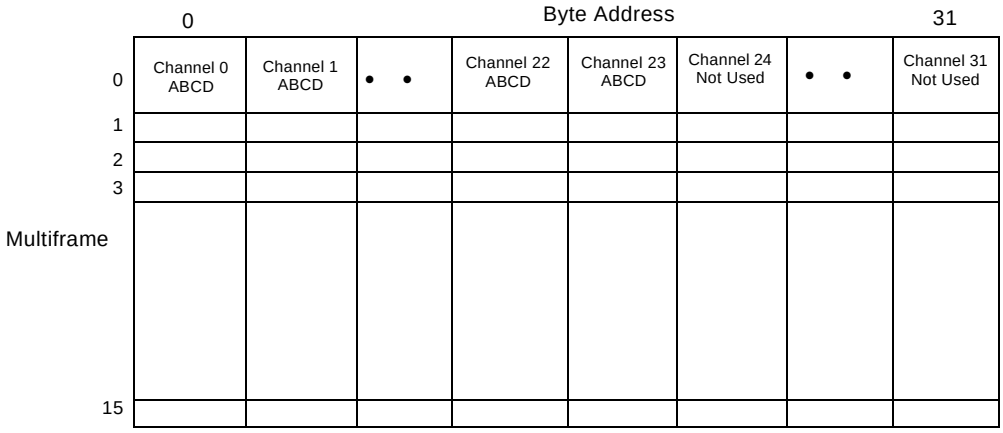


Figure 42. T1 ESF SDF-MF Format of the R_SIG_BUFFER

Figure 43 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.

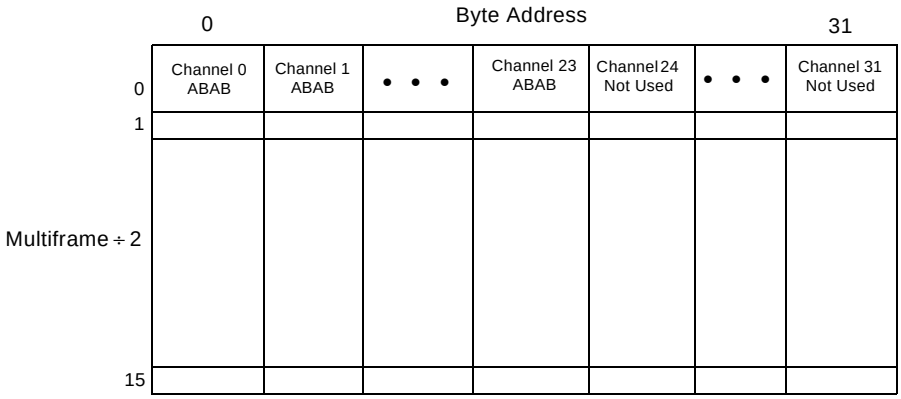


Figure 43. T1 SF SDF-MF Format of the R_SIG_BUFFER

Figure 44 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.

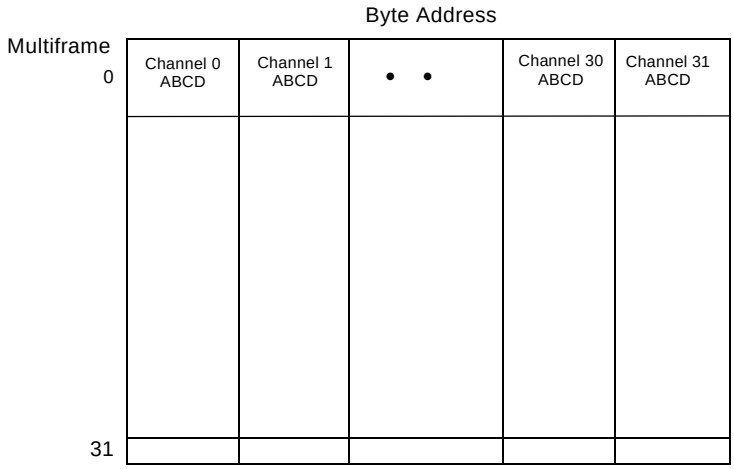


Figure 44. E1 SDF-MF Format of the R_SIG_BUFFER

Figure 45 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.

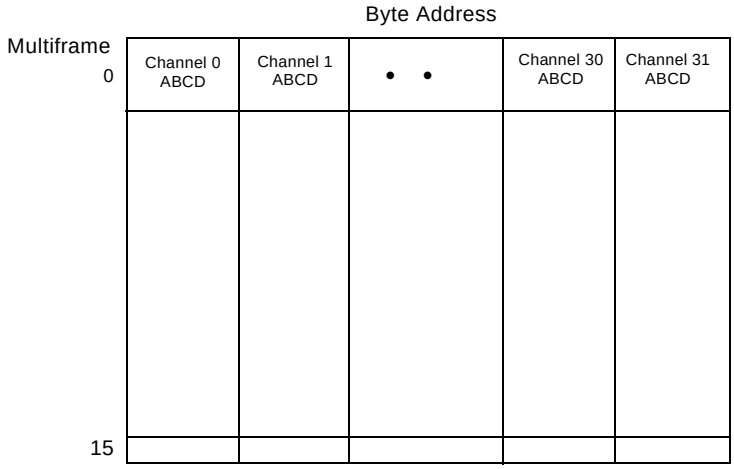


Figure 45. E1 SDF-MF Mode with T1 Signaling Format of the R_SIG_BUFFER

3.6.1 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 (refer to "E1_WITH_T1_SIG" on page 126). 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_MAX_BUF word (refer to "R_MAX_BUF Word Format" on page 157), 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.

3.6.2 Underrun

The AAL1gator II declares an underrun condition for a VC when no data is present in the VC receive buffer. When this situation occurs, the AAL1gator II plays out conditioned data and old signaling from multiframe 0 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. 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 level of the R_UNDERRUN bit in the R_LINE_STATE register. When not in UDF-HS mode, the choice of conditioning data and signaling while in underrun depends on the value of RX_COND in the R_CH_QUEUE_TBL. Three choices exist:

- Play out the data from R_COND_DATA and the signaling from multiframe 0 (Default).
- Play out pseudorandom data and the signaling from multiframe 0. (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 $x^{18} + x^7 + 1$.
- Play out old data and the signaling from multiframe 0. (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.

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

If the data is structured, the RALP searches for a new pointer, and finds the start of the structure. In all cases 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. 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.

Refer to line items 5 and 6 in Figure 47 on page 64 that show how a start up after an underrun occurs.

3.6.3 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 signaling from multiframe 0 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 3.6.2 “Underrun” on page 60.

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.

Figure 46 shows the state machine that checks for valid pointers and structures.

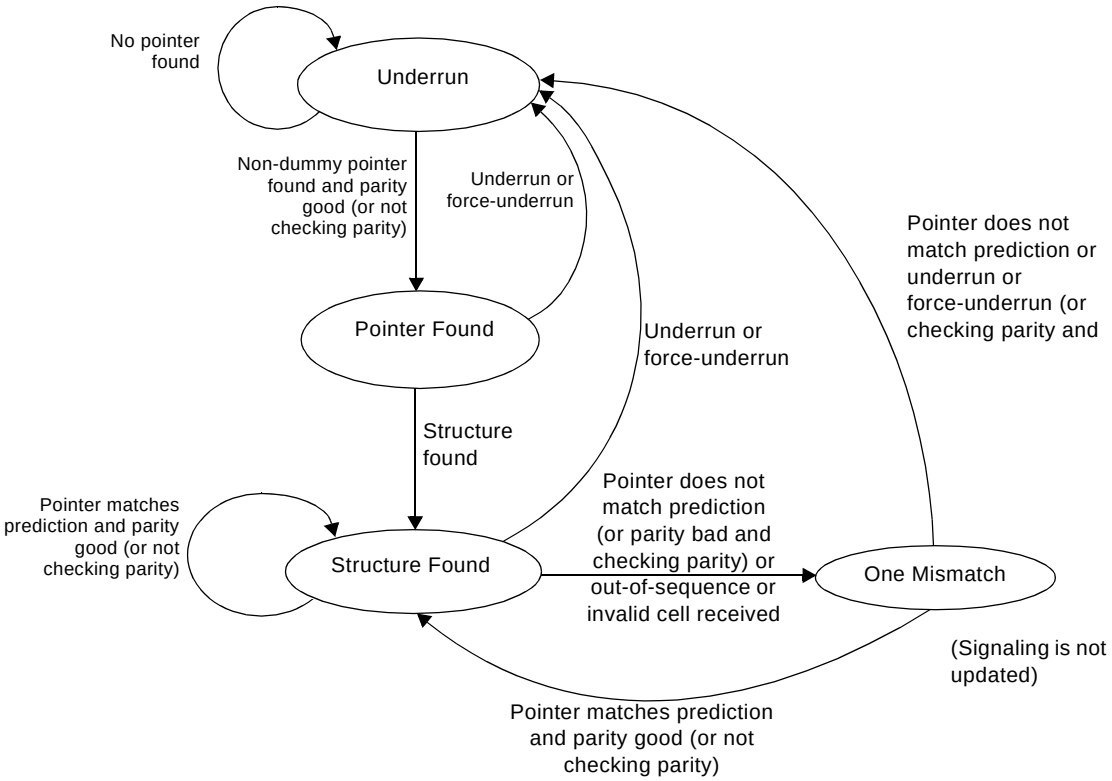


Figure 46. Pointer/Structure State Machine

3.6.4 Overrun

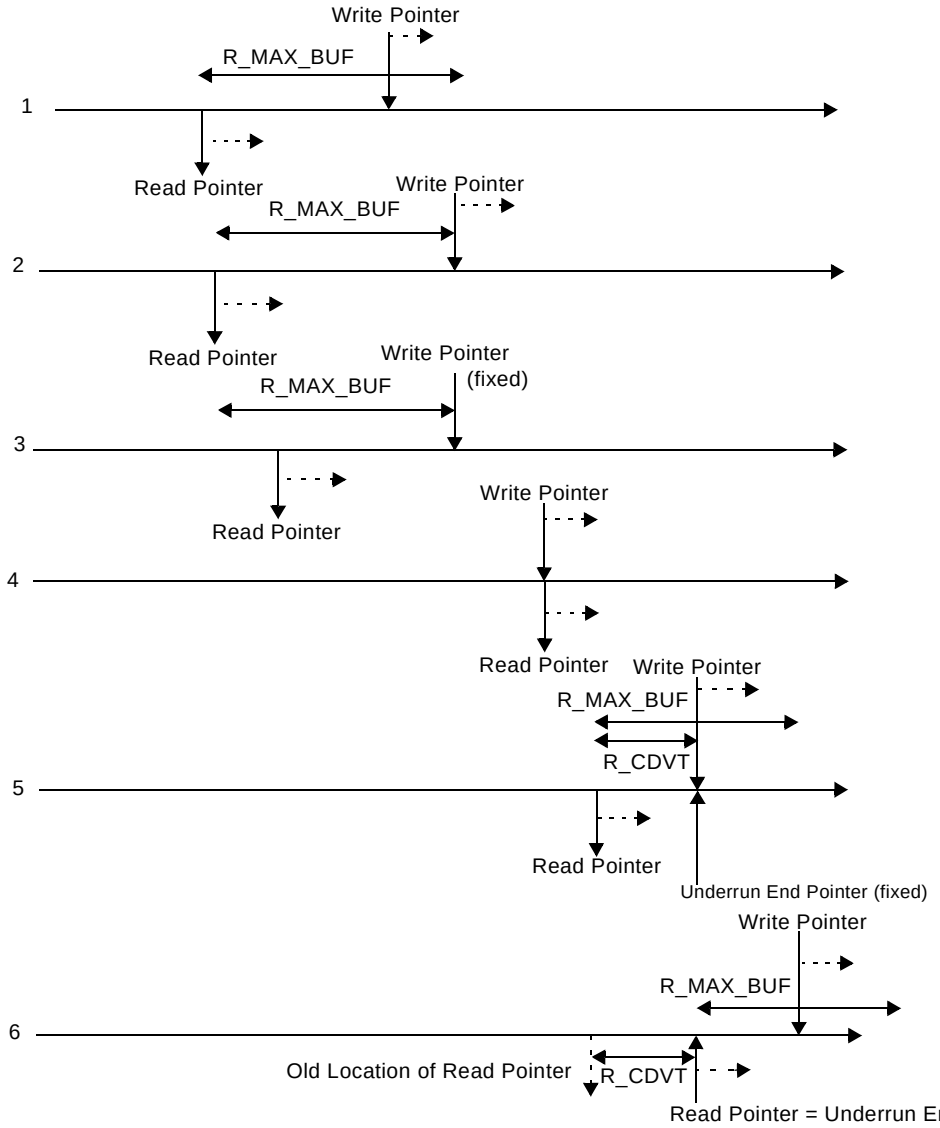
Overrun occurs when the data in the buffer is removed at a slower rate than it is filled. However, because the AAL1gator II buffers are quite large, 16 kBytes per line, by the time this happens, all data in the buffer can be quite old. Therefore, the buffer size is adjustable, which regulates how much data can be stored in the buffer before an overrun occurs. The R_MAX_BUF field in the R_MAX_BUF 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 two times the number of frames required per cell, whichever is greater.

The overrun condition is declared when the data in the receive buffer exceeds the maximum specified buffer size (refer to "R_MAX_BUF Word Format" on page 157). When a cell is received that causes the maximum buffer depth to be exceeded, the OVERRUN sticky bit is set and the AAL1gator II 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 47 on page 64 describes overrun detection, the underrun and recovery process.

Anytime an overflow occurs, the R_OVERRUNS counter (refer to "R_UNDERRUNS Word Format" on page 162) is incremented.

NOTE: 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.



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.

Figure 47. Overrun Detection

3.6.5 Counters and Sticky Bits

The RALP sets sticky bits for overrun, underrun, pointer mismatch, resume, SRTS underrun, SRTS resume, and other conditions. As with all registers, the sticky bits are located in the external RAM. They are set by the AAL1gator II 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 II 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" on page 158 for a description of sticky bits.

3.6.6 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 generating an interrupt. The interrupt is cleared when the CLR_RX_OAM_LATCH (refer to "CMDREG Word Format" on page 165) is set. 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 II also checks the CRC-10 of the cell and records the results in the receive buffer in the CRC_10_PASS parameter (refer to "R_OAM_CELL_n Format" on page 163).

3.6.7 Interrupt Handling

The AAL1gator II uses the following algorithm to handle interrupts:

1. An internal register called the command register (refer to section 7.9 "CMDREG (Command Register)" on page 165) handles interrupts. This command register has an OAM_INT_MASK bit. The microprocessor can access the command register by asserting the ADDR17 pin, the /PROC_CS signal, and either the /PROC_RD or the /PROC_WR signal.
2. At the end of an OAM cell, the RALP generates an interrupt to the microprocessor. If the mask bit is set, the RALP does not present the interrupt to the interrupt pin.
3. At the end of any cell (either an OAM or data cell), the head and tail pointers of the OAM queue table are compared. If they do not match (that is, if there are cells in the OAM FIFO), the RALP sets the interrupt latch.

3.7 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 (refer to section 7.8.6 "R_CH_TO_QUEUE_TBL" on page 148). The RFTC outputs data to the external T1 or E1 framer device. For structured data, the RFTC

uses the synchronization signals (TL_FSYNC and TL_MSYNC) supplied by the framer 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 TL_FSYNC indicates the beginning of a frame, and a rising edge on TL_ MSYNC indicates the beginning of a multiframe. The RFTC realigns when an 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. For T1 mode, signaling data may change every 24th frame. For E1 mode, signaling may change every 16th frame.

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 in 32 timeslots. In this mode, the AAL1gator II reads signaling on the 24th frame of the multiframe.

The signaling nibble is valid for each channel when the last nibble of each channel's data is being driven. See Figure 48 for an example of signaling bits in a T1 frame. See Figure 49 for an example of signaling bits in the E1 mode.

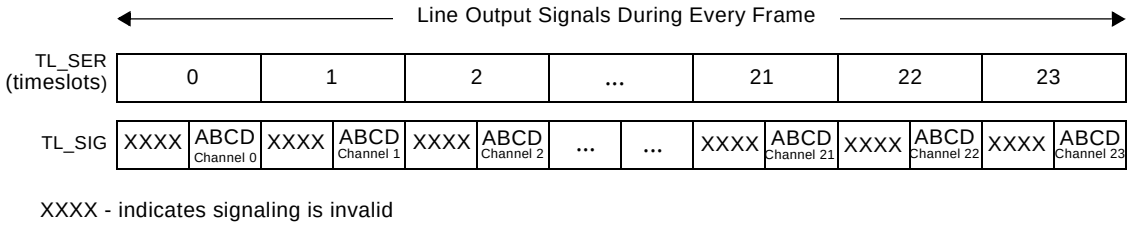


Figure 48. Output of T1 Signaling Bits

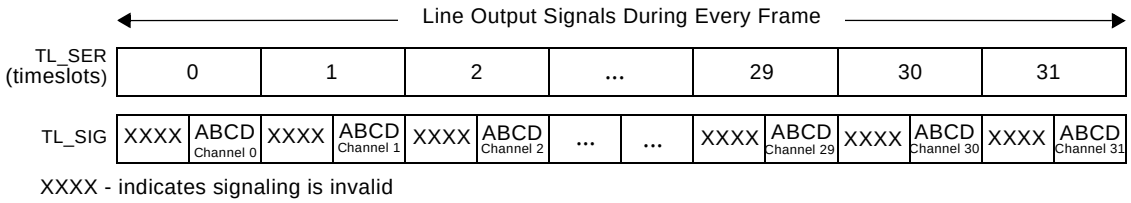


Figure 49. Output of E1 Signaling Bits

NOTE: The AAL1gator II 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 psuedocode (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 the signaling from the R_COND_SIG area. This is the conditioned state.

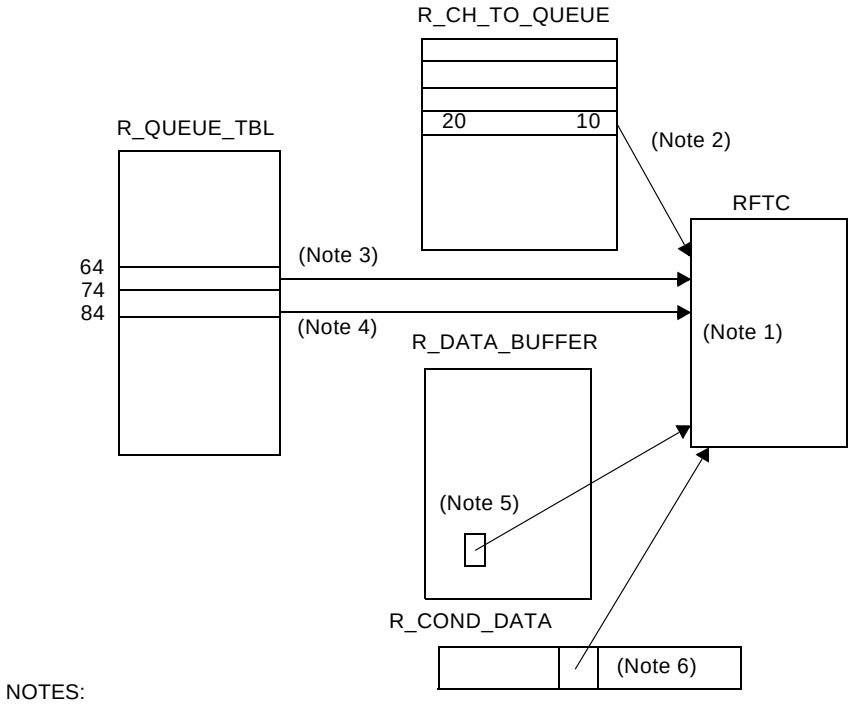
Else if either the R_UNDERRUN bit or the R_RESUME bit is set in R_LINE_STATE, then play out the data and signaling as determined by the value of RX_COND_H (or RX_COND_L). This is the frozen signaling state. The data played out can either be constant, pseudorandom, or old data.

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

When there is no data in the frame buffer, the RFTC sets an underrun bit R_UNDERRUN bit is cleared, and R_RD_FR_PTR = R_WR_FR_PTR. 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 50 shows the channel-to-queue table operation.



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.

Figure 50. Channel-to-Queue Table Operation

3.7.1 Generation of TL_CLK

The Transmit Line Clock (TL_CLK) can be generated in four different ways. It can be sourced externally when TL_CLK is an input. Or the AAL1gator II can generate it either by looping RL_CLK, synthesizing a nominal T1 or E1 clock, or synthesizing an E1 or T1 clock based on received SRTS values.

The AAL1gator II will generate the TL_CLK based on the value of CLK_SOURCE bits in LIN_STR_MODE (refer to section 7.4.3 “LIN_STR_MODE” on page 126). Once the LIN_STR_MODE and line mode information is read, the TL_CLK switches to the desired clock. The four clock choices are: looped RL_CLK, synthesized nominal E1 or T1 clock, a synthesized clock based on the SRTS received values, or an externally generated clock.

For the period of time after the AAL1gator II receives a hardware reset and before the LIN_STR_MODE is read, the TL_CLK operation is dependent on TLCLK_OUTPUT_EN. If the TLCLK_OUTPUT_EN input is tied low, then the TL_CLK pins are inputs and the clock is externally generated. And if TLCK_OUTPUT_EN is high, then TL_CLK defaults to the looped RL_CLK.

NOTES:
- The TLCLK_OUTPUT_EN input controls all lines. Therefore TL_CLK will be generated for all lines or for no lines upon power up. Once the LIN_STR_MODE information is read, each individual TL_CLK may be generated differently according to the CLK_SOURCE bits for that queue. There will be a discontinuity in the clock when switching from one source to another.

- If a synthesized clock is selected but SW_RESET (refer to “SW_RESET” on page 165) is set, the output clock will be “0”. To maintain a continuous clock, define the CLK_SOURCE bits equal to the default value the first time the CMD_REG_ATTN bit is written (SW_RESET = “1”). Then when clearing SW_RESET, set the CLK_SOURCE bits to the desired value and set the CMD_REG_ATTN bit again. This will read the old configuration values, but with the new CLK_SOURCE values.

3.7.1.1 Recovered Mode

Set the CLK_SOURCE bits in the LIN_STR_MODE register for that line to “01”, to generate a clock based on the received RL_CLK. This will put the line into recovered mode. This change will not take affect until the CMD_REG_ATTN bit is set.

3.7.1.2 Synthesize a Nominal E1 or T1 Clock

To generate a nominal E1 or T1 clock, set the CLK_SOURCE bits in the LIN_STR_MODE register for that line to “10”. This change will not take affect until the CMD_REG_ATTN bit is set.

The AAL1gator II can synthesize a nominal E1 clock or T1 clock that is derived from SYS_CLK. For this mode, SYS_CLK 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.

Refer to section 8.10.1 “Nominal T1 Clock” on page 185 and section 8.10.2 “Nominal E1 Clock” on page 187 for jitter characteristics of a synthesized clock. It is recommended that a Line Interface Unit (LIU) with a built in jitter attenuator, such as the Level One LXT305A, be used to obtain improved jitter characteristics.

3.7.1.3 Synthesize an E1 or T1 Clock based on SRTS

NOTE: The AAL1gator II uses Bellcore’s patented SRTS clock recovery technique. Refer to the NOTE on page 172 for additional information regarding Bellcore’s SRTS patent.

Set the CLK_SOURCE bits in the LIN_STR_MODE register for that line to “11” to generate an E1 clock or T1 clock based on received SRTS values. This change will not take affect until the CMD_REG_ATTN bit is set.

For this mode, SYS_CLK must be 38.88 MHz. For SRTS, it is recommended that SYS_CLK has an accuracy of ± 100 ppm. Since a network derived N_CLK is used, SYS_CLK does not have to be derived from the network clock.

The AAL1gator II supports SRTS for unstructured data formats on a per-line basis. SRTS support requires an input reference clock (N_CLK). 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 × TL_CLK > N_CLK > 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 51 on page 71 shows the process implemented for each UDF line enabled for SRTS. The RFTC also generates a local SRTS value from the network clock (N_CLK) and the local TL_CLK. It queues the incoming SRTS values, 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 RFTC uses this value internally to synthesize the TL_CLK.

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 it before it is needed. If the SRTS queue overruns or underruns, a value of 0000 is used.

The RFTC also outputs the SRTS difference through a multiplexed interface. This value can be used by external logic to synthesize the TL_CLK. Refer to section 6.7 “SRTS Timing” on page 118 for timing of this interface. This interface is needed for high speed mode (E3 or DS3) SRTS

which requires an external Voltage Controlled Crystal Oscillator (VCXO) to generate the clock. Refer to section 8.7 “UDF-HS Mode SRTS-Based Clock Recovery Application for DS3” on page 180 for an application note on high speed SRTS.

The E1 SRTS synthesizer can generate an E1 clock over a ± 90 ppm range, while the T1 SRTS synthesizer can generate a T1 clock over a ± 200 ppm range to accommodate older T1 equipment. Due to the T1 synthesizer’s wider range, it has a more jitter than the E1 synthesizer. While the E1 synthesizer falls well within the G.823 jitter mask, the T1 synthesizer has a spike at 8 kHz that exceeds the mask. However, in both cases using an LIU with a built-in jitter attenuator, such as the Level One® LXT305A, eliminates most of the jitter and is recommended when using SRTS. Refer to section 8.10.3 “SRTS T1 Clock” on page 188 and section 8.10.4 “SRTS E1 Clock” on page 191 for the jitter characteristics of the SRTS synthesizer both with and without the external jitter attenuator.

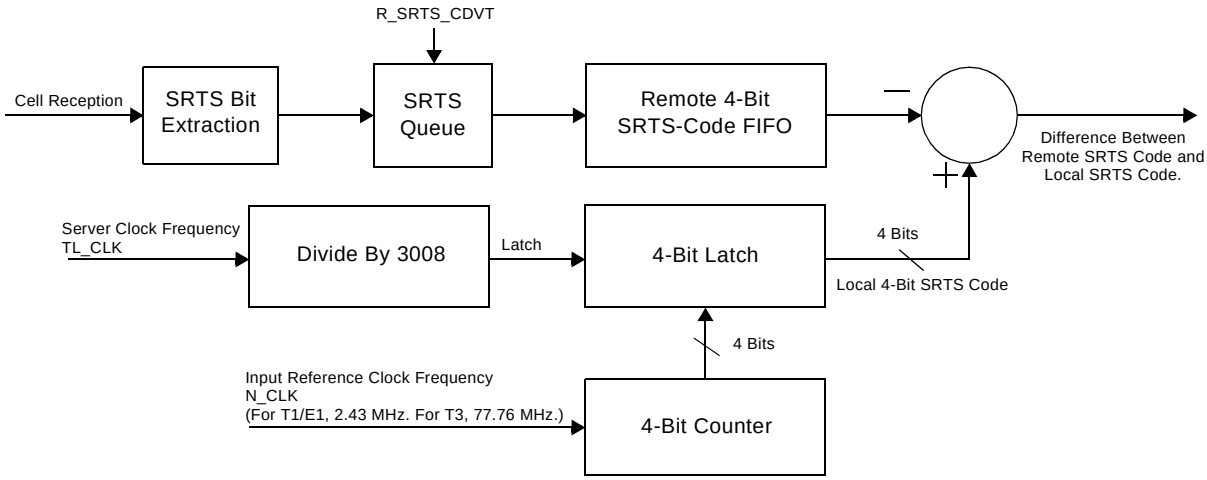


Figure 51. Receive Side SRTS Support

3.7.2 Adaptive Clock Operation

The adaptive clock provides a queue depth difference for control of an external clock. If the depth is low, then the clock frequency should be reduced. If the depth is high, then the clock frequency should be increased. If no data is received, the channel status specifies to either freeze the external clock or to set it to a nominal rate.

The AAL1gator II supports adaptive clocking in SDF-FR, SDF-MF, and UDF-ML modes. It supports adaptive clocking for all connection types within these three modes, including those with partially filled cells or only a single DS0. The device does not support adaptive clocking in UDF-HS mode.

The adaptive clocking mechanism shares the SRTS port with the SRTS mechanism. It operates according to the following psuedocode:

If an SRTS difference value is ready, it is played out with SRTS_STRB asserted and ADAP_STRB deasserted. The SRTS values have higher priority than the adaptive clock values.

Else if a channel pair is being serviced, start a state machine to play out the channel status, as shown in Table 3 on page 72, asserting ADAP_STRB as each state is played out.

Else if a cell is received and a valid frame difference can be computed, start a state machine to play out the frame difference and queue number, as shown in Table 4 on page 72, asserting ADAP_STRB as each state is played out.

The playout of the channel status may interrupt the playout of the frame difference at any point.

Table 3. Channel Status

SRTS_LINE(3:0) Value	SRTS_DOUT(3:0) Value			
	3	2	1	0
15	0	line(2)	line(1)	line(0)
14	channel(4)	channel(3)	channel(2)	channel(1)
13	underrun_h	resume_h	underrun_1	resume_1

NOTES:

- The underrun bit goes high when underrun occurs and will stay high until the underrun condition is cleared.
- Resume goes high when the underrun condition is cleared and goes low when the read pointer equals the underrun_end pointer.
- The status is played out once for every 16 receive line clocks.
- The channel values (4:1) identify 16 channel pairs. The odd and even numbered channels of each pair are carried by the high and low bits, respectively.
- For UDF-ML mode, the values in channel (4:1) and underrun_l can be ignored. Line (2:0) indicate the line number and underrun_h represents the queue status.

Table 4. Frame Difference

SRTS_LINE(3:0) Value	SRTS_DOUT(3:0) Value			
	3	2	1	0
5	cell_vci(7)	cell_vci(6)	cell_vci(5)	cell_vci(4)
4	cell_vci(3)	cell_vci(2)	cell_vci(1)	cell_vci(0)

Table 4. Frame Difference (Continued)

SRTS_LINE(3:0) Value	SRTS_DOUT(3:0) Value			
	3	2	1	0
3	0	0	frame_diff(8)	frame_diff(7)
2	frame_diff(6)	frame_diff(5)	frame_diff(4)	frame_diff(3)
1	frame_diff(2)	frame_diff(1)	frame_diff(0)	0
0	0	0	0	0

NOTES:

- The 9-bit frame_diff value gives the number of frames currently stored in the rx frame buffer. For UDF-ML mode, frame_diff is expressed in 256-bit increments.
- The cell_vci(7:5) identifies the eight line numbers.
- The cell_vci(4:0) bits can be ignored in UDF-ML mode.
- The frame difference values for each queue are played out in sequence 5 to 0 every time a cell is received and a valid frame difference can be calculated.

The AAL1gator II provides the receive buffer frame difference for an external circuit to generate an adaptive TL_CLK signal. The general mechanism is often termed “buffer centering”. A clock delta value is determined externally by subtracting the nominal frame difference (value of R_CDVT) from the actual receive buffer frame difference. This delta value is then transformed into the frequency selection for an external TL_CLK 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 TL_CLK frequency must be increased. When the delta is below the center point, there is too little data buffered and the TL_CLK frequency must be decreased.

As mentioned in the ATM Forum CES Standard Specification (refer to Appendix B, “References”, on page 203), the adaptive clock recovery algorithm does not meet the T1/E1 clock wander requirements. See Figure 52, which shows a direct adaptive clocking implementation.

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. The mechanism shown in Figure 52 can be enhanced for voice applications by adding a low pass filter with a time constant greater than the CDV, along with a fine-granularity frequency synthesizer.

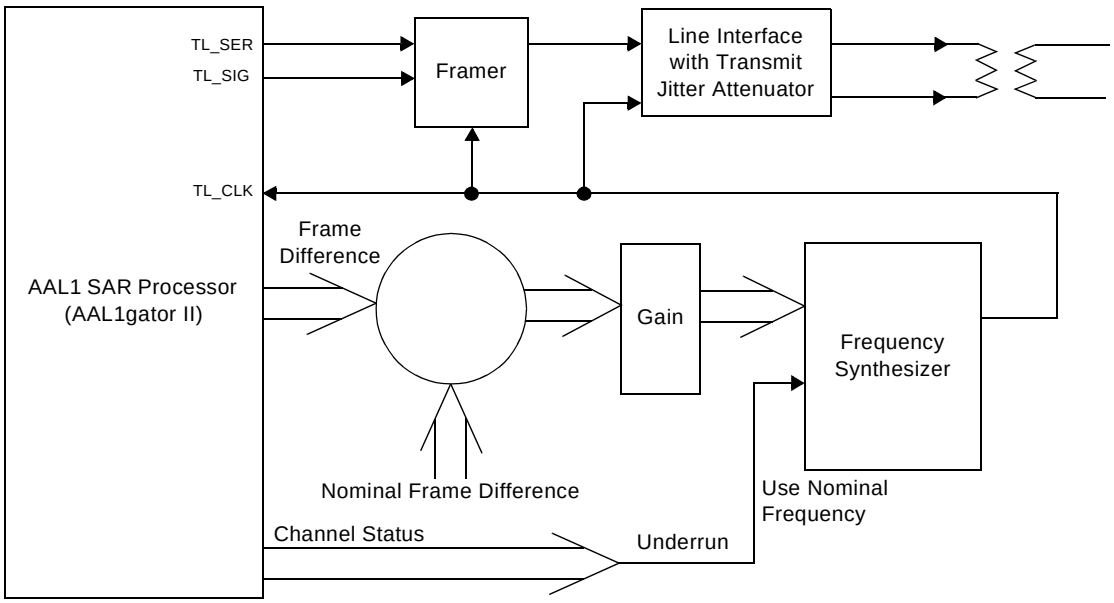


Figure 52. Direct Adaptive Clock Operation

3.8 Memory Interface and Arbitration Controller (MIAC)

The MIAC 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 MIAC. When the MIAC 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 II requires external memory address buffers and data transceivers. The MIAC provides control signals and transceiver enables when a microprocessor cycle is to be performed. The device can directly control external 646-type registered transceivers on the data bus between the memory data interface and the microprocessor's data bus. The SP_DATA_CLK signal controls the writing of data from the memory into the 646 transceiver. The SP_DATA_DIR signal and SP_DATA_EN signal should be used to direct the data through the 646s devices properly. The /SP_ADD_EN signal should be used as a tristate enable for the external address buffers. The AAL1gator II controls the tristating of its own address and data bus drivers so no conflicts occur with the microprocessor external buffers.

To operate at the specified maximum SYS_CLK rate of 38.88 MHz, the external buffers must be of the Fast CMOS TTL (FCT) or Fast (F) family, or of comparable speed. These outputs must be supplied directly to the external devices and no buffering of the signals is allowed. Refer to Figure 108 on page 195 for an example of how to interface memory to the AAL1gator II. Also refer to Table 26 on page 195 for recommended timing parameters.

The microprocessor accesses the device by means of the /PROC_CS, /PROC_RD, and /PROC_WR lines. To perform a write cycle, the /PROC_CS and /PROC_WR lines are asserted. The AAL1gator II then enables the address and data lines at the proper moments and, when complete, the device signals the microprocessor with the /PROC_ACK line. To perform a read, the microprocessor asserts /PROC_CS and /PROC_RD and waits for /PROC_ACK before reading the data.

To prevent the microprocessor from obtaining too many memory cycles and interfering with normal AAL1gator II activity, a holdoff circuit is used. This circuit denies the microprocessor an additional access until 20 SYS_CLK cycles have expired since the last microprocessor access. The HOLDOFF output is asserted whenever this denial period is in effect. The microprocessor can poll this output to determine when it will be allowed an access that is not subjected to HOLDOFF. See timing diagrams starting with Figure 55 on page 92. Refer to section 8 “Application Notes” starting on page 168 for a block diagram of the interface.

3.9 Configuration

To transfer TDM data into cells or to transfer cells into TDM data, the PM73121 and each line needs to be configured, and the queues need to be configured and added to determine how the data should be mapped.

To configure the AAL1gator II and each line, initialize the COMP_LIN_REG and the LIN_STR_MODE register for each line. To have these values take affect, the CMD_ATTN bit must be set and the SW_RESET bit in the CMDREG must be cleared. Since no queues are defined at this time, all timeslots in the R_CH_TD_QUEUE_TBL should be initialized to play out conditioned data and the R_COND_DATA field for each timeslot should be initialized to the desired play out value. If signaling is used, then R_COND_SIG should also be initialized.

Once the line is configured, queues can be added as described in section 7.11 “Activating a New Queue on an Active Line” on page 167.

4 PIN DESCRIPTIONS

4.1 Package Diagram

Figure 53 (parts 1 and 2) shows the physical dimensions for the 240-pin quad flat pack (with square/gullwing leads) used for the AAL1gator II. The package measurements are in millimeters.

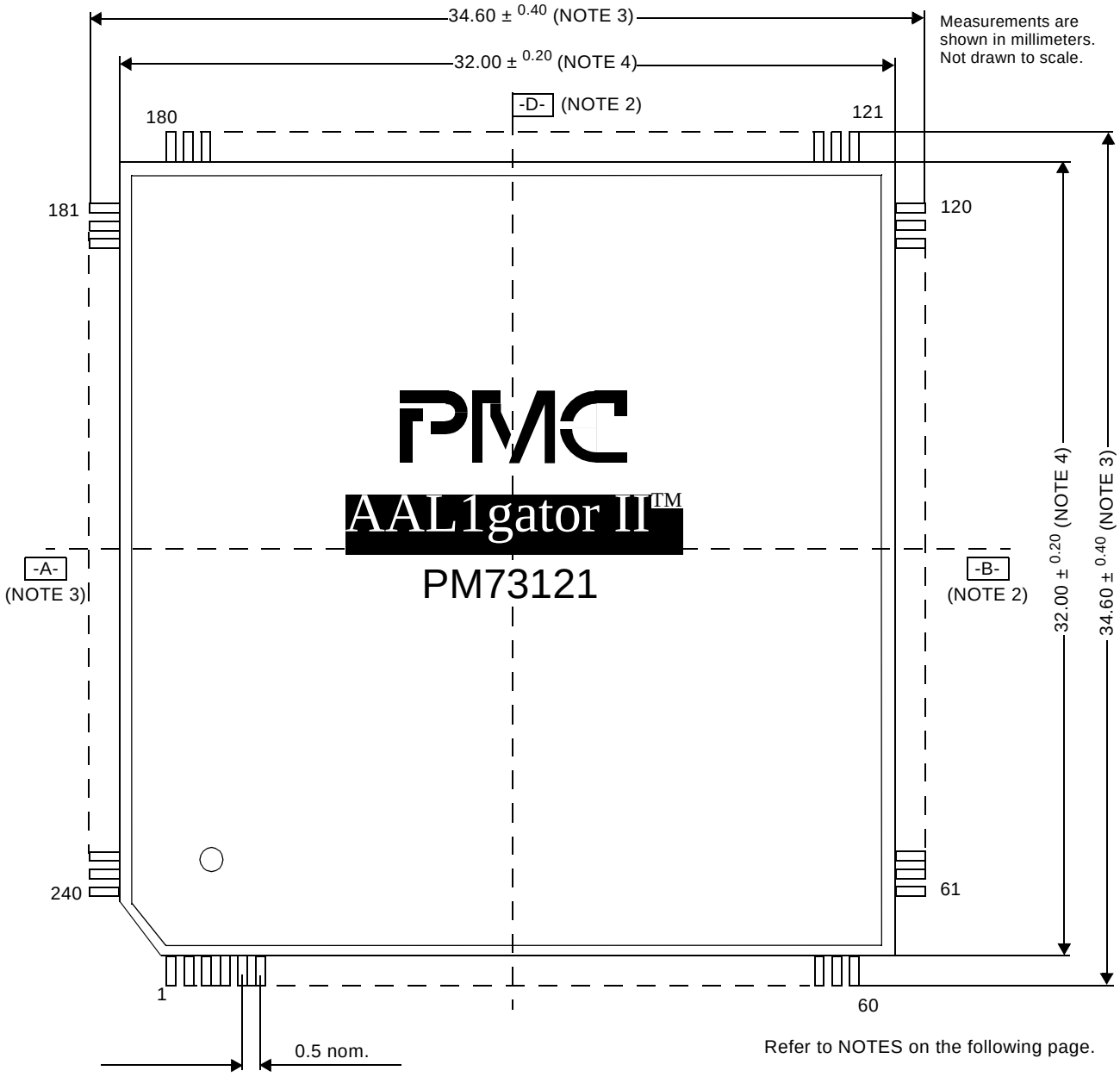
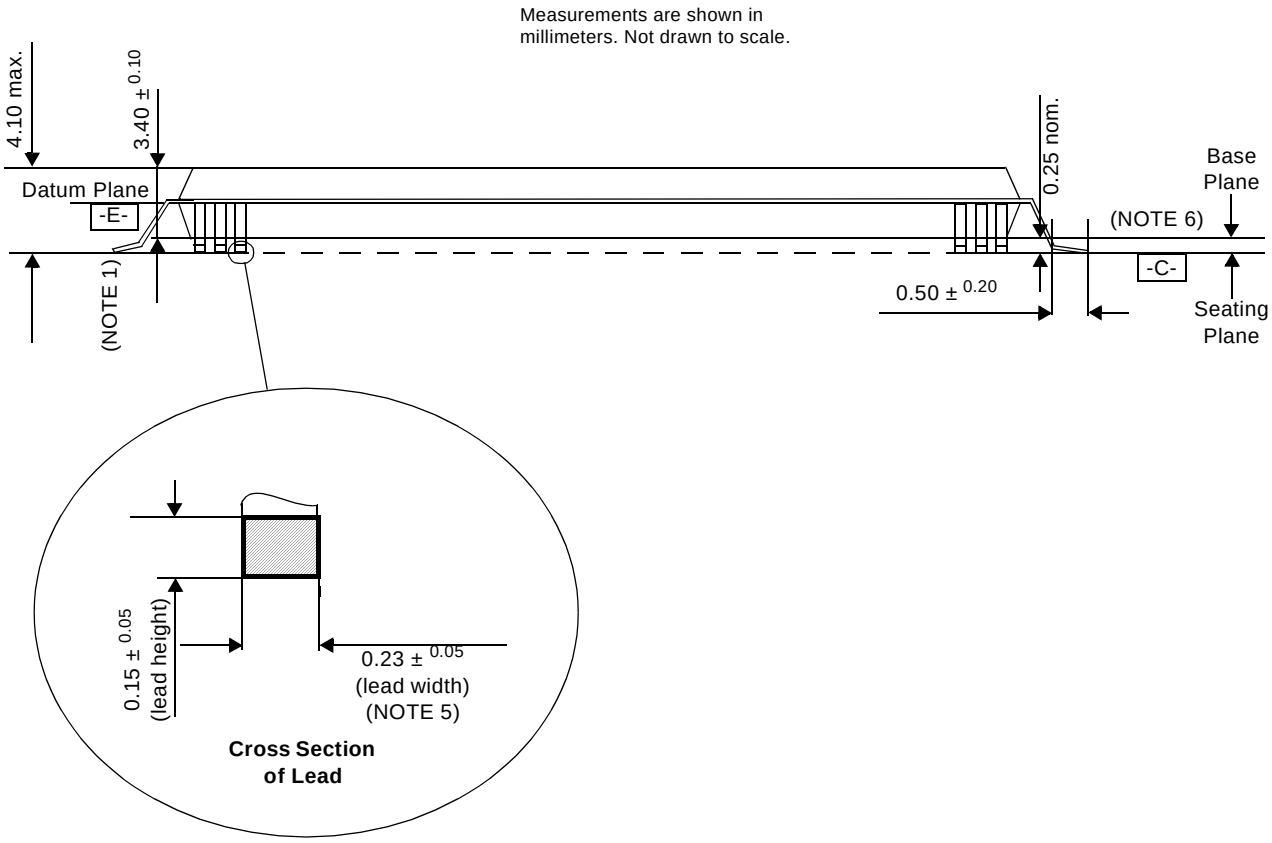


Figure 53. 240-Pin Physical Dimensions Diagram (Part 1 of 2)



NOTES:

1. Datum plane -E- is located at the mold parting line and is coincident with the bottom of the leads where the leads exit the plastic body.
2. Datums -A-, -B-, and -D- to be determined at datum plane -E-.
3. To be determined at seating plane -C-.
4. These dimensions do not include mold protrusion. Allowable protrusion is 0.25 mm per side. These dimensions do not include mold mismatch and are determined at datum plane -E-.
5. This dimension does not include Dambar protrusion. Allowable Dambar protrusion shall be 0.08 mm total in excess of this dimension at maximum material condition. Dambar cannot be located on the lower radius or the foot. Minimum spacing between adjacent leads to be 0.07 mm.
6. Coplanarity = 0.08 mm and is the difference between the highest lead and the seating plane, -C-.
7. Controlling dimension = millimeter.
8. If you need a measurement not shown in this figure, please contact PMC-Sierra.

Figure 53. 240-Pin Physical Dimensions Diagram (Part 2 of 2)

4.2 Pinout

4.2.1 Pinout Diagram

Figure 54 shows a pinout diagram of the AAL1gator II.

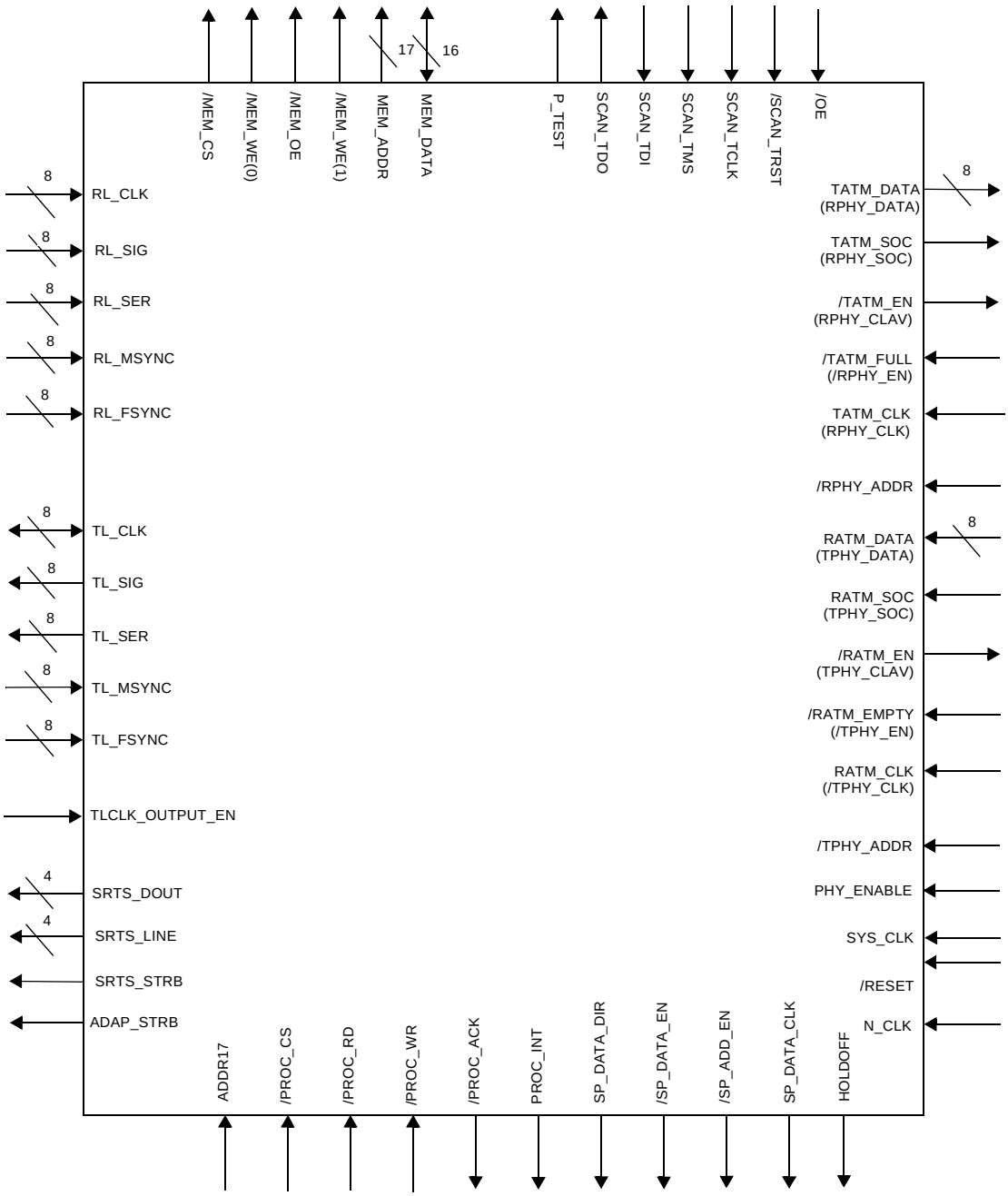


Figure 54. AAL1gator II Pinout Diagram

4.2.2 Pinout Table

Table 5 shows the pinout for the AAL1gator II.

Table 5. AAL1gator II Pinout

Pin	Name	Pin	Name	Pin	Name	Pin	Name
1	VDD	39	SRTS_LINE(2)	77	RL_SER(1)	115	TL_CLK(4)
2	VDD	40	SRTS_LINE(1)	78	RL_SIG(1)	116	GND
3	RATM_DATA(7)*	41	SRTS_LINE(0)	79	TLCLK_OUTPUT_EN	117	RL_CLK(4)
4	RATM_DATA(6)*	42	SRTS_STRB	80	TL_SIG(2)	118	RL_FSYNC(4)
5	RATM_DATA(5)*	43	ADAP_STRB	81	TL_SER(2)	119	GND
6	RATM_DATA(4)*	44	GND	82	TL_MSYNC(2)	120	GND
7	RATM_DATA(3)*	45	N_CLK	83	TL_FSYNC(2)	121	VDD
8	RATM_DATA(2)*	46	GND	84	TL_CLK(2)	122	VDD
9	RATM_DATA(1)*	47	SRTS_LINE(3)	85	GND	123	RL_MSYNC(4)
10	RATM_DATA(0)*	48	VDD	86	RL_CLK(2)	124	RL_SER(4)
11	RATM_SOC*	49	TL_SIG(0)	87	GND	125	RL_SIG(4)
12	/RATM_EMPTY*	50	TL_SER(0)	88	GND	126	NC
13	/RATM_EN*	51	TL_MSYNC(0)	89	SYS_CLK	127	TL_SIG(5)
14	/RPHY_ADDR	52	TL_FSYNC(0)	90	GND	128	TL_SER(5)
15	GND	53	GND	91	NC	129	TL_MSYNC(5)
16	RATM_CLK*	54	TL_CLK(0)	92	VDD	130	TL_FSYNC(5)
17	GND	55	GND	93	VDD	131	TL_CLK(5)
18	/TPHY_ADDR	56	PHY_ENABLE	94	RL_FSYNC(2)	132	GND
19	VDD	57	VDD	95	RL_MSYNC(2)	133	RL_CLK(5)
20	TATM_CLK*	58	RL_CLK(0)	96	RL_SER(2)	134	RL_FSYNC(5)
21	GND	59	GND	97	RL_SIG(2)	135	RL_MSYNC(5)
22	/TATM_EN*	60	GND	98	NC	136	RL_SER(5)
23	/TATM_FULL*	61	VDD	99	TL_SIG(3)	137	RL_SIG(5)
24	TATM_SOC*	62	VDD	100	TL_SER(3)	138	NC (Must be left disconnected)
25	TATM_DATA(7)*	63	RL_FSYNC(0)	101	TL_MSYNC(3)	139	TL_SIG(6)
26	TATM_DATA(6)*	64	RL_MSYNC(0)	102	TL_FSYNC(3)	140	TL_SER(6)
27	TATM_DATA(5)*	65	RL_SER(0)	103	TL_CLK(3)	141	TL_MSYNC(6)
28	TATM_DATA(4)*	66	RL_SIG(0)	104	GND	142	TL_FSYNC(6)
29	GND	67	NC	105	RL_CLK(3)	143	TL_CLK(6)
30	TATM_DATA(3)*	68	TL_SIG(1)	106	RL_FSYNC(3)	144	GND
31	VDD	69	TL_SER(1)	107	RL_MSYNC(3)	145	RL_CLK(6)
32	TATM_DATA(2)*	70	TL_MSYNC(1)	108	RL_SER(3)	146	RL_FSYNC(6)
33	TATM_DATA(1)*	71	TL_FSYNC(1)	109	RL_SIG(3)	147	RL_MSYNC(6)
34	TATM_DATA(0)*	72	TL_CLK(1)	110	NC	148	RL_SER(6)
35	SRTS_DOUT(3)	73	GND	111	TL_SIG(4)	149	RL_SIG(6)

Table 5. AAL1gator II Pinout (Continued)

Pin	Name	Pin	Name	Pin	Name	Pin	Name
36	SRTS_DOUT(2)	74	RL_CLK(1)	112	TL_SER(4)	150	NC (Must be left disconnected)
37	SRTS_DOUT(1)	75	RL_FSYNC(1)	113	TL_MSYNC(4)	151	VDD
38	SRTS_DOUT(0)	76	RL_MSYNC(1)	114	TL_FSYNC(4)	152	TL_SIG(7)
153	TL_SER(7)	175	/PROC_CS	197	MEM_ADDR(3)	219	MEM_DATA(1)
154	TL_MSYNC(7)	176	/PROC_WR	198	MEM_ADDR(4)	220	MEM_DATA(2)
155	TL_FSYNC(7)	177	/PROC_RD	199	MEM_ADDR(5)	221	MEM_DATA(3)
156	TL_CLK(7)	178	SP_DATA_CLK	200	MEM_ADDR(6)	222	GND
157	GND	179	GND	201	MEM_ADDR(7)	223	MEM_DATA(4)
158	RL_CLK(7)	180	GND	202	GND	224	MEM_DATA(5)
159	RL_FSYNC(7)	181	VDD	203	MEM_ADDR(8)	225	MEM_DATA(6)
160	RL_MSYNC(7)	182	VDD	204	MEM_ADDR(9)	226	MEM_DATA(7)
161	RL_SER(7)	183	/RESET	205	MEM_ADDR(10)	227	MEM_DATA(8)
162	RL_SIG(7)	184	/OE	206	MEM_ADDR(11)	228	MEM_DATA(9)
163	NC	185	SP_DATA_DIR	207	MEM_ADDR(12)	229	GND
164	GND	186	/SP_ADD_EN	208	MEM_ADDR(13)	230	MEM_DATA(10)
165	SCAN_TDI	187	/SP_DATA_EN	209	GND	231	MEM_DATA(11)
166	SCAN_TMS	188	/MEM_WE(0)	210	GND	232	MEM_DATA(12)
167	/SCAN_TRST	189	GND	211	MEM_ADDR(14)	233	MEM_DATA(13)
168	SCAN_TDO	190	/MEM_WE(1)	212	VDD	234	MEM_DATA(14)
169	SCAN_TCLK	191	/MEM_OE	213	VDD	235	MEM_DATA(15)
170	P_TEST	192	/MEM_CS	214	MEM_ADDR(15)	236	NC
171	GND	193	GND	215	MEM_ADDR(16)	237	NC
172	HOLDOFF	194	MEM_ADDR(0)	216	ADDR17	238	PULLUP_DISABLE
173	PROC_INT	195	MEM_ADDR(1)	217	NC	239	GND
174	/PROC_ACK	196	MEM_ADDR(2)	218	MEM_DATA(0)	240	GND

* Refer to Table 6 on page 81 for alternative signal names when PHY_ENABLE is in PHY mode.

4.3 Pin Descriptions

4.3.1 UTOPIA Interface Signals

NOTE: Unless stated otherwise, the maximum output current (I_{MAX}) is 4 mA.

Table 6. UTOPIA Interface Signals

ATM Mode Signal	PHY Mode Signal	Pin #	Type	Reset Value*	Description
PHY_ENABLE	PHY_ENABLE	56	In	NA	PHY_ENABLE determines which UTOPIA mode the UTOPIA interface is configured. When 0, the chip is configured with an ATM layer UTOPIA interface and when 1, the chip is configured with a PHY layer UTOPIA interface. An internal pulldown resistor will default the chip to the ATM layer mode if left unconnected.
TATM_CLK	RPHY_CLK	20	In	NA	**ATM:** *Transmit UTOPIA ATM Layer Clock* is the synchronization clock input for synchronizing data output on TATM_DATA. **PHY:** *Receive UTOPIA PHY Layer Clock* is the synchronization clock input for synchronizing data output on RPHY_DATA. Maximum frequency is 33 MHz.
TATM_SOC	RPHY_SOC	24	Out	0(ATM) Z(PHY)	**ATM:** *Transmit UTOPIA ATM Layer Start-Of-Cell* is an active high signal asserted by the AAL1gator II when TATM_DATA contains the first valid byte of the cell. **PHY:** *Receive UTOPIA PHY Layer Start-Of-Cell* is an active high signal asserted by the AAL1gator II when RPHY_DATA contains the first valid byte of the cell. In SPHY mode, the AAL1gator II drives this signal only in cycles following those with /RPHY_EN asserted. In MPHY mode, the AAL1gator II drives this signal only when the ATM layer has selected it for a cell transfer. Maximum output current (I_{MAX}) = 8 mA.
TATM_DATA(7:4) TATM_DATA(3) TATM_DATA(2:0)	RPHY_DATA(7:4) RPHY_DATA(3) RPHY_DATA(2:0)	25-28 30 32-34	Out	0(ATM) Z(PHY)	**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 should be transmitted first. **PHY:** *Receive UTOPIA PHY Layer Data Bits 7 to 0* form the byte-wide data driven to the ATM layer. In SPHY mode, the AAL1gator II drives this bus only in cycles following those with /RPHY_EN asserted. In MPHY mode, the AAL1gator II drives this bus only when the ATM layer has selected it for a cell transfer. Bit 0 is the LSB. Bit 7 is the MSB and should be transmitted first. Maximum output current (I_{MAX}) = 8 mA.

Table 6. UTOPIA Interface Signals (Continued)

ATM Mode Signal	PHY Mode Signal	Pin #	Type	Reset Value*	Description
/TATM_EN	RPHY_CLAV	22	Out	1(ATM) Z(PHY)	**ATM:** *Transmit UTOPIA ATM Layer Enable* is an active low signal asserted by the AAL1gator II during cycles when TATM_DATA contains valid data. It is not asserted until the AAL1gator II is ready to send a full cell. **PHY:** *Receive UTOPIA PHY Layer Cell Available* is an active high signal asserted by the AAL1gator II to indicate it is ready to deliver a complete cell. In MPHY mode, this signal is driven only when /RPHY_ADDR is low in the previous cycle. Maximum output current (I_{MAX}) = 8 mA.
/TATM_FULL	/RPHY_EN	23	In	NA	**ATM:** *Transmit UTOPIA ATM Layer Full* is an active low signal from the PHY layer device to indicate that a maximum of four additional transmit data writes will be accepted. **PHY:** *Receive UTOPIA PHY Layer Enable* is an active low signal asserted by the ATM layer to indicate RPHY_DATA and RPHY_SOC will be sampled at the end of the next cycle. In MPHY mode, the AAL1gator II will drive data only if /RPHY_ADDR was low on the falling edge of /RPHY_EN.
RATM_CLK	TPHY_CLK	16	In	NA	**ATM:** *Receive UTOPIA ATM Layer Clock* is the synchronization clock input for synchronizing data input on RATM_DATA. **PHY:** *Transmit UTOPIA PHY Layer Clock* is the synchronization clock input for synchronizing data input on TPHY_DATA. Maximum frequency is 33 MHz.
RATM_SOC	TPHY_SOC	11	In	NA	**ATM:** *Receive UTOPIA ATM Layer Start-Of-Cell* is an active high signal asserted by the PHY layer when RATM_DATA contains the first valid byte of a cell. **PHY:** *Transmit UTOPIA PHY Layer Start-Of-Cell* is an active high signal asserted by the ATM layer when TPHY_DATA contains the first valid byte of a cell.
RATM_DATA(7:0)	TPHY_DATA(7:0)	3-10	In	NA	**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 and should be received first. **PHY:** *Transmit UTOPIA PHY Layer Data Bits 7 to 0* form the byte-wide data from the ATM layer device. Bit 0 is the LSB. Bit 7 is the MSB and should be received first.

Table 6. UTOPIA Interface Signals (Continued)

ATM Mode Signal	PHY Mode Signal	Pin #	Type	Reset Value*	Description
/RATM_EN	TPHY_CLAV	13	Out	0(ATM) Z(PHY) **	**ATM:** *Receive UTOPIA ATM Layer Enable* is an active low signal asserted by the AAL1gator II to indicate RATM_DATA and RATM_SOC will be sampled at the end of the next cycle. It will not be asserted until the AAL1gator II is ready to receive a full cell. **PHY:** *Receive UTOPIA PHY Layer Cell Available* is an active high signal asserted by the AAL1gator II to indicate there is a cell-space available. In MPHY mode, the AAL1gator II drives this signal only when /TPHY_ADDR is low in the previous cycle. Maximum output current (I_{MAX}) = 8 mA.
/RATM_EMPTY	/TPHY_EN	12	In	NA	**ATM:** *Receive UTOPIA ATM Layer Empty* is an active low signal asserted by the PHY layer to indicate that there is no valid data in the current cycle. **PHY:** *Transmit UTOPIA PHY Layer Enable l* is an active low signal asserted by the ATM layer device during cycles when TPHY_DATA contains valid data. In MPHY mode, the AAL1gator II will accept data only if /TPHY_ADDR was low on the falling edge of /TPHY_EN.
	/TPHY_ADDR	18	In	NA	**ATM:** This signal is not used in ATM mode. **PHY:** *Transmit UTOPIA PHY Layer Address* is an active low address input that is tied to one of the five possible ATM layer MPHY address signals. This input is used as an output enable for TPHY_CLAV and to validate the activation of /TPHY_EN. In SPHY mode, this input is not used. There is an internal pull-down resistor.
	/RPHY_ADDR	14	In	NA	**ATM:** This signal is not used in ATM mode. **PHY:** *Receive UTOPIA PHY Layer Address* is an active low address input that is tied to one of the five possible ATM layer MPHY address signals. This input is used as an output enable for RPHY_CLAV and to validate the activation of /RPHY_EN. In SPHY mode this input is not used. There is an internal pull-down resistor.

* Present when /RESET is asserted and both TATM_CLK and RATM_CLK and SYS_CLK are being clocked.

** /RATM_EN is asserted during reset to prevent excessive queueing in systems that use the AAL1gator II. This value is asserted when /RESET is asserted and both RATM_CLK and SYS_CLK are being clocked. In PHY mode this signal can be pulled to desired value as it is tristated.

4.3.2 Memory Interface Signals

Table 7. Memory Interface Signals

Signal	Pin #	Type*	Reset Value*	Description
MEM_DATA(15:10) MEM_DATA(9:4) MEM_DATA(3:0)	235-230 228-223 221-218	Bi	Z	*Memory Data Bits 15 to 0* form the 16-bit wide data bus to external memory.
MEM_ADDR(16:15) MEM_ADDR(14) MEM_ADDR(13:8) MEM_ADDR(7:0)	215-214 211 208-203 201-194	Out	Z	*Memory Address Data Bits 16 to 0* form the 17-bit wide address bus to external memory. Pulled up with an internal resistor.
/MEM_OE	191	Out	1	*Memory Output Enable* is an active low signal that enables the SRAM to drive data. Maximum output current (I_{MAX}) = 8 mA.
/MEM_WE(0)	188	Out	1	*Memory Write Enable Zero* is an active low signal for the low-byte write. Maximum output current (I_{MAX}) = 8 mA.
/MEM_WE(1)	190	Out	1	*Memory Write Enable One* is an active low signal for the high-byte write. Maximum output current (I_{MAX}) = 8 mA.
/MEM_CS	192	Out	1	*Memory Chip Select* is an active low chip-select signal for external memory. Maximum output current (I_{MAX}) = 8 mA.
*Present when /RESET is asserted and SYS_CLK is being clocked.				

4.3.3 T1/E1 Interface Signals

Table 8. T1/E1 Interface Signals

Signals	Pin #	Type	Reset Value*	Description
TL_FSYNC(7) TL_FSYNC(6) TL_FSYNC(5) TL_FSYNC(4) TL_FSYNC(3) TL_FSYNC(2) TL_FSYNC(1) TL_FSYNC(0)	155 142 130 114 102 83 71 52	In	NA	*Transmit Line Frame Synchronization Bits 7 to 0* are the transmit frame synchronization indications from the framer devices in SDF-MF and SDF-FR modes. The lines originate from the corresponding framer devices 0 to 7.
TL_CLK(7) TL_CLK(6) TL_CLK(5) TL_CLK(4) TL_CLK(3) TL_CLK(2) TL_CLK(1) TL_CLK(0)	156 143 131 115 103 84 72 54	Bi	NA*	*Transmit Line Channel Clock Bits 7 to 0* are the clock lines for the eight T1/E1 lines. The bits clock the data from the AAL1gator II to the corresponding framer devices. In the UDF-HS mode, only line 0 is active. Depending on the value of TLCLK_OUTPUT_EN and the CLK_SOURCE bits, 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 bits. Maximum output current (I_{MAX}) = 8 mA.
* If TLCLK_OUTPUT_EN is high, the pins output the RL_CLK during reset.				

Table 8. T1/E1 Interface Signals (Continued)

Signals	Pin #	Type	Reset Value*	Description
TL_SIG(7) TL_SIG(6) TL_SIG(5) TL_SIG(4) TL_SIG(3) TL_SIG(2) TL_SIG(1) TL_SIG(0)	152 139 127 111 99 80 68 49	Out	0	*Transmit Line Signal Bits 7 to 0* are the CAS signaling outputs to the corresponding framer devices in the SDF-MF mode. Maximum output current (I_{MAX}) = 8 mA.
TL_MSYNC(7) TL_MSYNC(6) TL_MSYNC(5) TL_MSYNC(4) TL_MSYNC(3) TL_MSYNC(2) TL_MSYNC(1) TL_MSYNC(0)	154 141 129 113 101 82 70 51	In	NA	*Transmit Line Multiframe Synchronization Bits 7 to 0* carry multiframe timing information from the corresponding framer devices. These signals do not need to indicate the start of each and every multiframe. They can occur infrequently. Each time one of these signals has an edge to indicate the start of a multiframe, the AAL1gator II re-aligns the multiframe according to where the edge occurred. Tie to ground if not used.
TL_SER(7) TL_SER(6) TL_SER(5) TL_SER(4) TL_SER(3) TL_SER(2) TL_SER(1) TL_SER(0)	153 140 128 112 100 81 69 50	Out	0	*Transmit Line Serial Data Bits 7 to 0* carry the received data to the corresponding framer devices. In the UDF-HS mode, only line 0 is active. Maximum output current (I_{MAX}) = 8 mA.
RL_SIG(7) RL_SIG(6) RL_SIG(5) RL_SIG(4) RL_SIG(3) RL_SIG(2) RL_SIG(1) RL_SIG(0)	162 149 137 125 109 97 78 66	In	NA	*Receive Line Signal Bits 7 to 0* carry the CAS signaling information from the corresponding framer devices in SDF-MF mode. In UDF-HS mode, only line 0 is active.
RL_SER(7) RL_SER(6) RL_SER(5) RL_SER(4) RL_SER(3) RL_SER(2) RL_SER(1) RL_SER(0)	161 148 136 124 108 96 77 65	In	NA	*Receive Line Serial Data Bits 7 to 0* carry the receive data from the corresponding framer devices.

Table 8. T1/E1 Interface Signals (Continued)

Signals	Pin #	Type	Reset Value*	Description
RL_MSYNC(7) RL_MSYNC(6) RL_MSYNC(5) RL_MSYNC(4) RL_MSYNC(3) RL_MSYNC(2) RL_MSYNC(1) RL_MSYNC(0)	160 147 135 123 107 95 76 64	In	NA	*Receive Line Multiframe Synchronization Bits 7 to 0* carry the multiframe timing information from the corresponding framer device. These signals do not need to indicate the start of each and every multiframe - they can occur infrequently. Each time one of these signals has an edge to indicate the start of a multiframe, the AAL1gator II re-aligns the multiframe according to where the edge occurred. Tie to ground if not used.
RL_FSYNC(7) RL_FSYNC(6) RL_FSYNC(5) RL_FSYNC(4) RL_FSYNC(3) RL_FSYNC(2) RL_FSYNC(1) RL_FSYNC(0)	159 146 134 118 106 94 75 63	In	NA	*Receive Line Frame Synchronization Bits 7 to 0* carry the receive frame information from the corresponding framer devices in the SDF-MF and SDF-FR modes. The bits originate from the corresponding framer bits.
RL_CLK(7) RL_CLK(6) RL_CLK(5) RL_CLK(4) RL_CLK(3) RL_CLK(2) RL_CLK(1) RL_CLK(0)	158 145 133 117 105 86 74 58	In	NA	*Receive Clock Bits 7 to 0* form an 8-bit bus. The bits originate from the corresponding framer bits 0 to 7. In the UDF-HS mode, only line 0 is active.
SRTS_DOUT(3:0)	35-38	Out	0	*SRTS Data Out Bits 3 to 0* form the SRTS correction code when SRTS_STRB is asserted; otherwise SRTS_DOUT bits form the channel status and frame difference when ADAP_STRB is asserted.
SRTS_LINE(3) SRTS_LINE(2:0)	47 39-41	Out	0	*SRTS Line Bits 3 to 0* indicate the T1/E1 line that SRTS_DOUT corresponds to when SRTS_STRB is asserted; otherwise SRTS_LINE bits form the adaptive state machine index when ADAP_STRB is asserted.
SRTS_STRB	42	Out	0	*SRTS Strobe* indicates that an SRTS value is present on SRTS_DOUT(3:0) and SRTS_LINE(2:0). This transfer is made synchronous to SYS_CLK.
ADAP_STRB	43	Out	0	*Adaptive Strobe* indicates that the channel status and frame difference are being played out on the SRTS_DOUT. The nibbles are identified by the values on SRTS_LINE.
N_CLK	45	In	NA	*Network Clock* is the network-derived clock for SRTS.

Table 8. T1/E1 Interface Signals (Continued)

Signals	Pin #	Type	Reset Value*	Description
TLCLK_OUTPUT_EN	79	In	NA	*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 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 bits in the LIN_STR_MODE register (refer to section 7.4.3 "LIN_STR_MODE" starting on page 126).
*Present when /RESET is asserted and SYS_CLK is being clocked.				

4.3.4 Microprocessor Interface Signals

Table 9. Microprocessor Interface Signals

Signal	Pin #	Type**	Reset Value*	Description
ADDR17	216	In	NA	*Address Bit 17* is bit 17 of the address bus directly connected to the AAL1gator II. This bit is set to access the CMD_REG word.
HOLDOFF	172	Out	0	When *Holdoff* is asserted, the microprocessor cycle is delayed. Implementations that have fast microprocessors may want to poll this signal before accessing the device.
/PROC_RD	177	In	NA	*Processor Read* is an active low read signal from the microprocessor.
/PROC_WR	176	In	NA	*Processor Write* is an active low write signal from the microprocessor.
/PROC_CS	175	In	NA	*Processor Chip Select* is a memory request from the microprocessor.
/PROC_ACK	174	Out	1	*Processor Acknowledge* is an active low signal acknowledgment to the microprocessor.
PROC_INT	173	Out	0	*Processor Interrupt* is an active high interrupt to the microprocessor.
SP_DATA_CLK	178	Out	1	*Supervisory Processor (microprocessor) Read Clock* is a signal with a rising edge that causes memory or register read data to be written into a 646-type external data buffer from which it is read by the microprocessor. Maximum output current (I_{MAX}) = 8 mA.
SP_DATA_DIR	185	Out	1	*Supervisory Processor (microprocessor) Data Direction* controls the direction of data in a 646-type external data buffer. This signal is driven low when data is directed toward the device and high when data is directed toward the microprocessor. Maximum output current (I_{MAX}) = 8 mA.
/SP_ADD_EN	186	Out	1	*Supervisory Processor (microprocessor) Address Enable* is an active low buffer enable strobe to enable a 244-type address external buffer. Maximum output current (I_{MAX}) = 8 mA.
/SP_DATA_EN	187	Out	1	*Supervisory Processor (microprocessor) Data Enable* is an active low buffer enable signal to enable a 646-type external data buffer. Maximum output current (I_{MAX}) = 8 mA.
SYS_CLK	89	In	NA	*System Clock* is the local clock used by the state machines within the device. For SRTS T1/E1 clock recovery applications using the digital synthesizer (refer to section 8 “Application Notes” on page 168), this clock should be 38.88 MHz (155.52 MHz ÷ 4), but need not necessarily be derived from the network. The maximum frequency is 40 MHz.
/RESET	183	In	NA	*Reset* is an active low hardware reset.
/OE	184	In	NA	*Output Enable* is an active low signal that enables outputs of the device. It allows outputs to be disabled for in-circuit testing. Tie this signal to ground for normal operation.

*Present when /RESET is asserted and SYS_CLK is being clocked.
**All input (In) and bidirectional (Bi) signals are pulled up through a large, internal pull-up resistor.

4.3.5 JTAG and Process Test Signals

Table 10. JTAG and Process Test Signals

Signal	Pin #	Type**	Reset Value*	Description
/SCAN_TRST	167	In	NA	*Scan Test Reset* is an active low signal for boundary scan logic. NOTE: When not using boundary scan, connect this pin to ground. /SCAN_TRST has an internal pull-up resistor.
SCAN_TMS	166	In	NA	*Scan Test Mode Select* is the mode select signal for boundary scan logic.
SCAN_TDI	165	In	NA	*Scan Test Data In* is the serial data input signal for boundary scan.
SCAN_TDO	168	Out	Z	*Scan Test Data Out* is the serial data output signal for boundary scan.
SCAN_TCLK	169	In	NA	*Scan Test Clock* is the clock for boundary scan logic.
P_TEST	170	Out	1	*Process Test* is an output to measure process test parameters only during device manufacture. Leave unconnected during normal operation.
PULLUP_DISABLE	238	In	NA	*Pullup Disable* is used to disable internal pullup resistors during manufacturing tests. Connect directly to ground.

*Present when /RESET is asserted and SYS_CLK is being clocked.
**All input (In) and bidirectional (Bi) signals are pulled up through a large, internal pull-up resistor.

5 PHYSICAL CHARACTERISTICS

Table 11. Absolute Maximum Ratings

Symbol	Parameter	Conditions	Min	Max	Unit
V_{DD}	Supply voltage	With respect to GND (see NOTE below).	-0.3	6.5	V
I_{OUT}	DC output current, per pin		-12	12	mA
T_{STG}	Storage temperature		-65	150	°C
NOTE: Minimum DC input is -0.3 V. The absolute maximum voltage range for TTL inputs is -1.0 V to V_{DD}+0.3 V.					

Table 12. Recommended Operating Conditions

Symbol	Parameter	Conditions	Min	Max	Unit
V_{DD}	Supply voltage		4.75	5.25	V
V_I	Input voltage		0	V_{DD}	V
V_O	Output voltage		0	V_{DD}	V
T_A	Operating temperature		-40	85	°C
t_R	Input rise time			10	ns
t_F	Input fall time			10	ns

Table 13. DC Operating Conditions

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IH}	High-level input voltage	All TTL inputs	2.0			V
V_{IL}	Low-level input voltage	All TTL inputs			.8	V
V_{OH}	High-level output voltage	$I_{OH} = -I_{MAX}$*	2.4			V
V_{OL}	Low-level output voltage	$I_{OL} = I_{MAX}$*			0.4	V
I_{TYPE1}	Typical operating current E1	all lines active, generating TL_CLK		350	420	mA
I_{TYPDS3}	Typical operating current DS3	includes SRTS with 77.76 MHz N_CLK		430	482	mA
I_L	Input leakage current	All inputs	-35	-115	-214	µA

NOTES: •Typical values are T_A = 25°C.
•V_{DD} = 5 V ± 5%.
•*Refer to Table 6 on page 81 through Table 10 on page 89 for the different I_{MAX} values.

Table 14. Capacitance

Symbol	Parameter	Conditions	Min	Max	Unit
C_{IN}	Input capacitance			10	pF
C_{OUT}	Output capacitance			6	pF
C_{EXT}	External loading	To meet timing		50*	pF/pin

NOTES: •Capacitance measured at 25°C.
•Sample tested only.
•*RAM interface has different loading. Refer to section 6.5 “RAM and Microprocessor Timing” on page 103.

6 TIMING DIAGRAMS

6.1 Transmit Side Line Interface Timing

Figure 55, Figure 56 on page 93, Figure 57 on page 94, and Figure 58 on page 94 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 explained in the table following Figure 55.

RL_FSYNC and RL_MSYNC are used in structured modes to align the frame and multiframe of the incoming data. These inputs are ignored in unstructured modes.

The AAL1gator II expects all signals to be asserted from the rising edge of RL_CLK, and samples all signals on the falling edge of RL_CLK, as shown in the following figures.

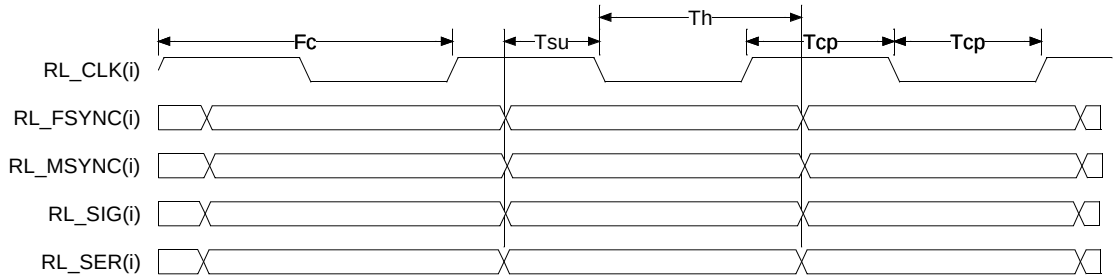


Figure 55. Transmit Side Interface Bit Timing

Symbol	Parameter	Signals	Min	Max	Unit
Fc	Clock frequency	RL_CLK		15	MHz
Tcp	Clock pulse width	RL_CLK	10		ns
Tsu	Clock setup	RL_MSYNC, RL_FSYNC, RL_SIG, RL_SER	5		ns
Th	Clock hold	RL_MSYNC, RL_FSYNC, RL_SIG	1		ns
Th	Clock hold	RL_SER	2		ns

When operating in T1 mode, the AAL1gator II expects signaling only on the lower four bits of each timeslot as shown in Table 15 and Figure 56. Signaling data is accepted from the RL_SIG pin only during the last frame of each multiframe.

The rising edge of RL_FSYNC should only occur during the frame (F) bit of the T1 data stream. The rising edge of RL_MSYNC should only occur during the F bit which starts each 12-frame (SF) or 24-frame (ESF) multiframe. If a sync input occurs when it is not expected, the AAL1gator II will resync to the new framing. The sync pulses do not have to be driven every frame or multiframe.

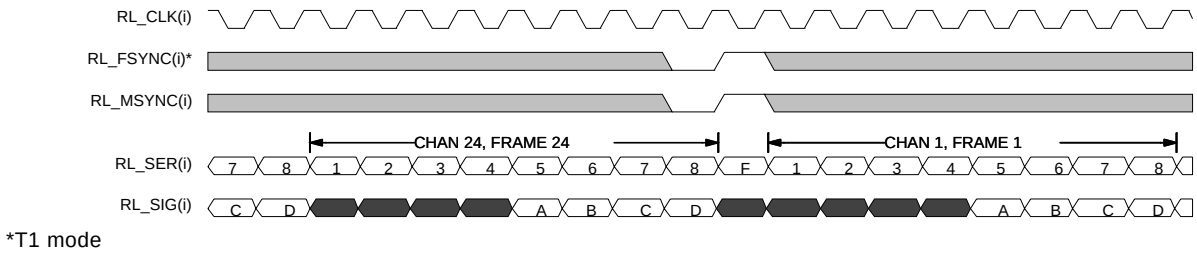


Figure 56. Transmit Side T1 Interface Frame Timing

Table 15. Signaling Format for T1 Mode

PCM Channel Number	1234	5678
	XXXX	ABCD
1	X	1
2	X	2
...	X	...
23	X	23
24	X	24

In E1 mode, signaling data is expected only on the lower four bits of each timeslot as shown in Table 16 on page 94. Signaling data is only accepted from the RL_SIG pin in the last frame of each multiframe. The AAL1gator II treats all 32 timeslots the same. Although E1 data streams contain 30 timeslots of channel data and 2 timeslots of control, data and signaling for all 32 timeslots are stored in memory.

The rising edge of RL_FSYNC should only occur during the first bit of each frame of the E1 data stream. The rising edge of RL_MSYNC should only occur during the first bit of each 16 frame multiframe. If E1_WITH_T1_SIG is set, then the rising edge of RL_MSYNC should only occur during the first bit of each 24 frame multiframe. If a sync input occurs when it is not expected, the AAL1gator II will resync to the new structure. The sync pulses do not have to be driven every frame or multiframe.

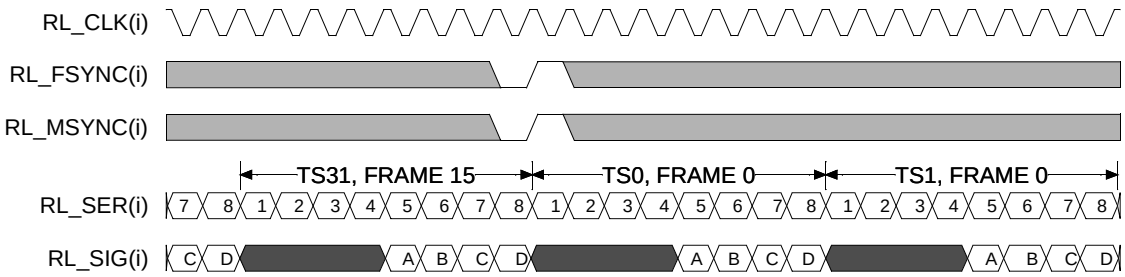


Figure 57. Transmit Side E1 Interface Frame Timing

Table 16. Signaling Format for E1 Mode

PCM Channel Number	1234	5678
	XXXX	ABCD
0	X	0
1	X	1
...	X	...
30	X	30
31	X	31

In UDF-HS mode there is no structure and the data is sampled using RL_CLK(0) as shown in Figure 58.

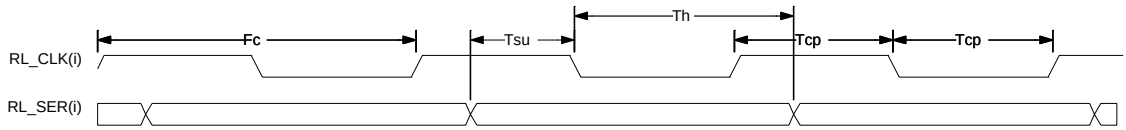


Figure 58. Transmit Side High-Speed Interface Timing

Symbol	Parameter	Signals	Min	Max	Unit
Fc	Clock frequency	RL_CLK(0)		45	MHz
Tcp	Clock pulse width	RL_CLK(0)	10		ns
Tsu	Clock setup	RL_SER(0)	5		ns
Th	Clock hold	RL_SER(0)	2		ns

6.2 Receive Side Line Interface Timing

Figure 59, Figure 60, Figure 61 on page 96, and Figure 62 on page 96 show the receiver transmits data to the lines for low-speed applications. These lines typically interface with the transmit input portion of the corresponding framer. The AAL1gator II drives the same signaling data onto TL_SIG during each frame of a multiframe. Data is output off the rising edge of TL_CLK, and TL_FSYNC and TL_MSYNC are sampled using the falling edge of TL_CLK. The timing parameters are explained in the tables following the figures.

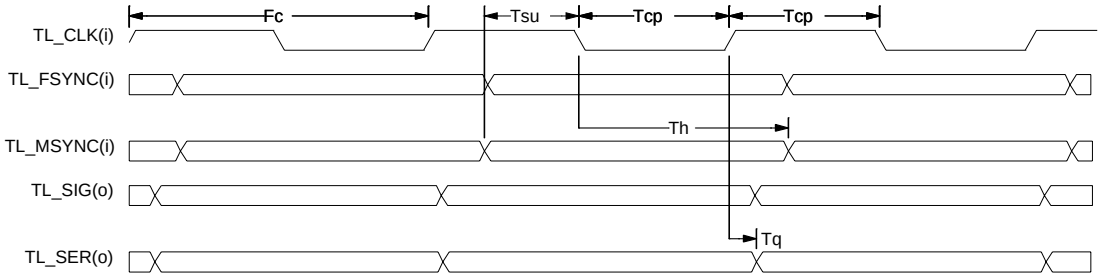


Figure 59. Receive Side Low-Speed Interface Bit Timing

Symbol	Parameter	Signals	Min	Max	Unit
Fc	Clock frequency	TL_CLK		15	MHz
Tcp	Clock pulse width	TL_CLK	10		ns
Tsu	Clock setup	TL_MSYNC, TL_FSYNC	5		ns
Th	Clock hold	TL_MSYNC, TL_FSYNC	1		ns
Tq	Clock-to-output delay	TL_SIG, TL_SER	2	14	ns

The format of signaling data on the TL_SIG output is dependent on the framer operating mode. When operating in T1 mode, the AAL1gator II drives signaling only on the lower four bits of each timeslot as shown in Table 15 on page 93 and Figure 60. In all cases, signaling data is driven on the TL_SIG pin for all frames of each multiframe.

The rising edge of TL_FSYNC should occur only during the frame (F) bit of the T1 data stream. The rising edge of TL_MSYNC should occur only during the F bit that starts each 12-frame (SF) or 24-frame (ESF) multiframe. If a sync input occurs when it is not expected, the AAL1gator II will resync to the new structure. The sync pulses do not have to be driven every frame or multi-frame.

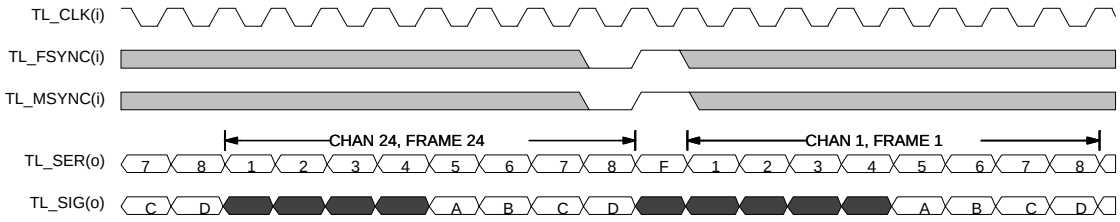


Figure 60. Receive Side T1 Interface Frame Timing

In E1 mode, data is driven on the lower four bits of each timeslot as shown in Table 16 on page 94. Signaling data is driven on the TL_SIG pin for all frames of each multiframe. The AAL1gator II treats all 32 timeslots the same. Although E1 data streams contain 30 timeslots of channel data and 2 timeslots of control, data and signaling for all 32 timeslots are stored in memory.

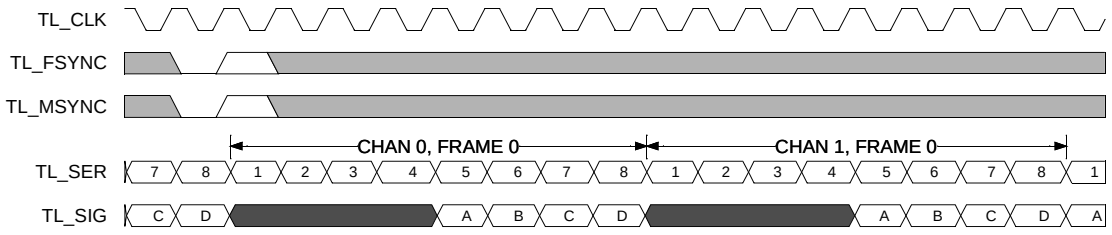


Figure 61. Receive Side E1 Interface Timing

In UDF-HS mode there is no structure and the data is driven using TL_CLK(0) as shown in Figure 62.

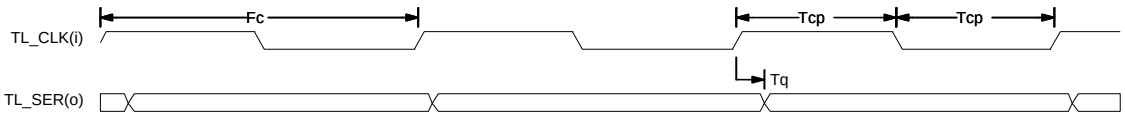


Figure 62. Receive Side High-Speed Interface Timing

Symbol	Parameter	Signals	Min	Max	Unit
Fc	Clock frequency	TL_CLK(0)		45	MHz
Tcp	Clock pulse width	TL_CLK(0)	45	55	%
Tq	Clock-to-output delay	TL_SER(0)	2	13	ns

6.3 Transmit UTOPIA Timing

6.3.1 TUTOPIA as the ATM Layer Device

The transmit ATM UTOPIA timing signals are compatible with the ATM Forum UTOPIA Level 1 byte-by-byte specification (refer to Appendix B, "References", on page 203). Table 17 indicates the transmit UTOPIA signal names and their corresponding UTOPIA designations. The interface will not assert /TATM_EN at the beginning of a cell unless it has a full cell to send. After the AAL1gator II detects /TATM_FULL asserted, it deasserts /TATM_EN two TATM_CLK cycles later. Refer to section 3.4 "Transmit UTOPIA Interface Block (TUTOPIA)" on page 41 for additional information.

Table 17. Transmit Signal Names and Corresponding UTOPIA Designations

Signal Name	UTOPIA Name
TATM_DATA	TxDATA
TATM_SOC	TxSOC
/TATM_EN	TxEnb*
/TATM_FULL	TxFull*
TATM_CLK	TxClk

Figure 63 illustrates the transmit ATM UTOPIA timing.

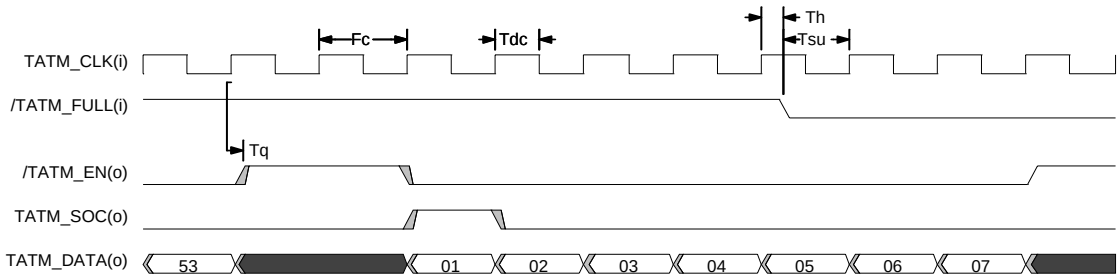


Figure 63. Transmit UTOPIA ATM Timing

Symbol	Parameter	Signals	Min	Max	Unit
Fc	Clock frequency	TATM_CLK		33	MHz
Tdc	Clock duty cycle	TATM_CLK	45	55	%
Tsu	Clock setup	/TATM_FULL	5		ns
Th	Clock hold	/TATM_FULL	1		ns
Tq	Clock-to-output delay	TATM_SOC, /TATM_EN	2	12	ns
Tq	Clock-to-output delay	TATM_DATA	2	13	ns

6.3.2 TUTOPIA as the PHY Layer Device in Single PHY (SPHY) Mode

The TUTOPIA block functions as a receive UTOPIA block when in SPHY mode. The receive SPHY UTOPIA timing signals are compatible with the ATM Forum UTOPIA Level 1 cell-level hand-shaking specification (refer to Appendix B, "References", on page 203). Table 18 indicates the AAL1gator II receive SPHY UTOPIA signal names and their corresponding UTOPIA designations. The AAL1gator II will not assert RPHY_CLAV unless it has a complete cell to send. RPHY_DATA and RPHY_SOC are tristate, except for the cycles following ones in which /RPHY_EN is active. RPHY_CLAV goes inactive one cycle after the last data byte has been output. Figure 64 shows the data output timing for the TUTOPIA block in SPHY mode. The timing parameters used in Figure 64 are defined in the table following the figure. All output timing delays assume a loading of 50 pF. Refer to section 3.4 "Transmit UTOPIA Interface Block (TUTOPIA)" on page 41 for additional information.

Table 18. Receive Signal Names and Corresponding UTOPIA Designations

Signal Name	UTOPIA Name
/RPHY_ADDR	RxAddr
RPHY_DATA	RxData
RPHY_SOC	RxSOC
RPHY_CLAV	RxClav
/RPHY_EN	RxEnb*
RPHY_CLK	RxClk

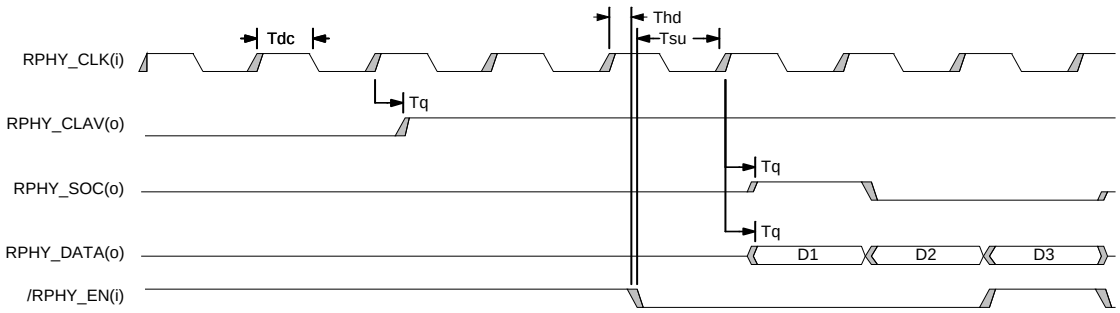


Figure 64. TUTOPIA SPHY Timing

Symbol	Parameter	Signal	Min	Max	Unit
	RPHY_CLK frequency			33	MHz
Tdc	RPHY_CLK duty cycle		45	55	%
Thd	RPHY_CLK hold time	/RPHY_EN	1		ns
Tq	RPHY_CLK-to-output delay	RPHY_CLAV, RPHY_SOC	2	12	ns
Tsu	RPHY_CLK setup time	/RPHY_EN	5		ns
Tq	RPHY_CLK-to-output delay	RPHY_DATA	2	13	ns

6.3.3 TUTOPIA as the PHY Layer Device in Multi-PHY (MPHY) Mode

The TUTOPIA block functions as a receive UTOPIA block when in MPHY mode. The receive MPHY UTOPIA timing signals are compatible with the ATM Forum UTOPIA Level 2 MPHY specification. Table 18 on page 98 indicates the AAL1gator II receive MPHY UTOPIA signal names and their corresponding UTOPIA designations. The AAL1gator II will not drive RPHY_CLAV unless /RPHY_ADDR was low the previous cycle. The value driven is dependent on whether or not the AAL1gator II has a complete cell to send. RPHY_DATA and RPHY_SOC are tristate, except for the cycles following ones in which /RPHY_EN is active and the AAL1gator II was selected. The AAL1gator II is selected only if on the falling edge of /RPHY_EN, /RPHY_ADDR is low. RPHY_SOC indicates the start of a new cell. RPHY_CLAV goes inactive one cycle after the last data byte has been output if /RPHY_ADDR is low at the time. Figure 65 shows the data output timing for the TUTOPIA block in MPHY mode. The timing parameters used in Figure 65 are defined in the table following the figure. All output timing delays assume a capacitive loading of 50 pF. Refer to section 3.4 “Transmit UTOPIA Interface Block (TUTOPIA)” on page 41 for additional information.

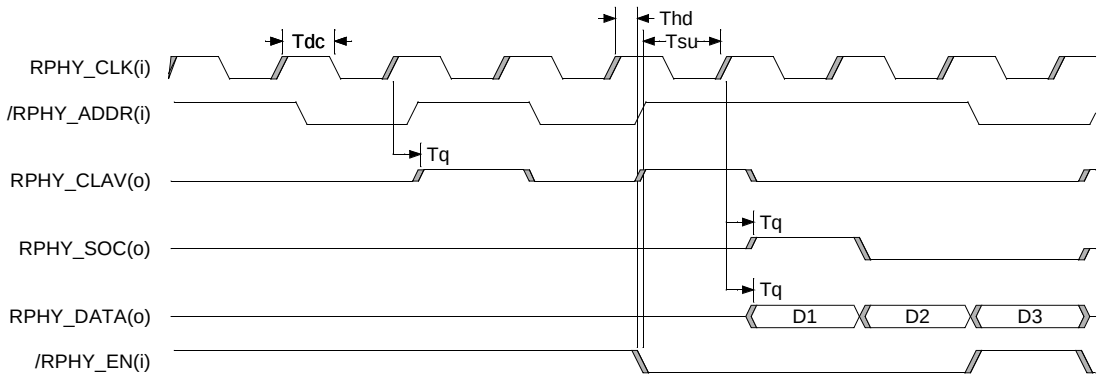


Figure 65. TUTOPIA MPHY Timing

.

Symbol	Parameter	Signal	Min	Max	Unit
	RPHY_CLK frequency			33	MHz
Tdc	RPHY_CLK duty cycle		45	55	%
Thd	RPHY_CLK hold time	/RPHY_ADDR, /RPHY_EN	1		ns
Tq	RPHY_CLK-to-output delay	RPHY_CLAV, RPHY_SOC	2	12	ns
Tsu	RPHY_CLK setup time	/RPHY_ADDR, /RPHY_EN	5		ns
Tq	RPHY_CLK-to-output delay	RPHY_DATA	2	13	ns

6.4 Receive UTOPIA Timing

6.4.1 RUTOPIA as the ATM Layer Device

The receive ATM UTOPIA timing signals are compatible with the UTOPIA Level 1 byte-by-byte specification (refer to Appendix B, “References”, on page 203). Table 19 indicates the receive UTOPIA signal names and their corresponding UTOPIA designations. The device will not assert /RATM_EN unless it can receive an entire cell without needing to deassert the /RATM_EN signal. Refer to section 3.5 “Receive UTOPIA Interface Block (RUTOPIA)” on page 44 for additional information.

If the receive FIFO will become full after the AAL1gator II reads the current cell, and /RATM_EMPTY is asserted during byte 51 through byte 53 of that cell, the AAL1gator II will toggle its /RATM_EN until those bytes become available from the external device, or the other cell in the external receive FIFO is read. Since the read data is pipelined, this action ensures the internal receive FIFO will not overflow.

Table 19. Receive Signal Names and Corresponding UTOPIA Designations

Signal Name	UTOPIA Name
RATM_DATA	RxData
RATM_SOC	RxSOC
/RATM_EN	RxEnb*
/RATM_EMPTY	RxEmpty*
RATM_CLK	RxClk

Figure 66 illustrates the receive ATM UTOPIA timing.

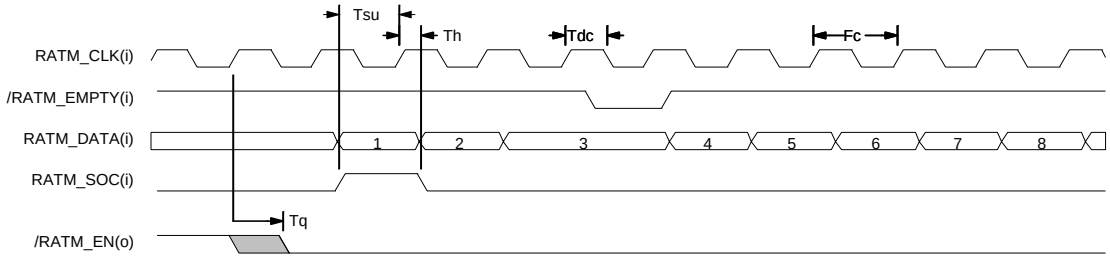


Figure 66. Receive UTOPIA ATM Timing

Symbol	Parameter	Signals	Min	Max	Unit
Fc	Clock frequency	RATM_CLK		33	MHz
Tdc	Clock duty cycle	RATM_CLK	45	55	%
Tsu	Clock setup	/RATM_EMPTY, RATM_DATA, RATM_SOC	5		ns
Th	Clock hold	/RATM_EMPTY, RATM_DATA, RATM_SOC	1		ns
Tq	Clock-to-output delay	/RATM_EN	2	12	ns

6.4.2 RUTOPIA as the PHY Layer Device in Single-PHY (SPHY) Mode

The RUTOPIA block functions as a transmit UTOPIA block when in SPHY mode. The transmit SPHY UTOPIA timing signals are compatible with the UTOPIA Level 1 cell-level hand-shaking specification (refer to Appendix B, “References”, on page 203). Table 20 indicates the transmit UTOPIA signal names and their corresponding UTOPIA designations. The device will not assert TPHY_CLAV unless it can receive an entire cell. When ready, the device will accept data for any cycle that /TPHY_EN is active. If the current cell will fill the internal FIFO, then TPHY_CLAV will be deactivated the same cycle it is receiving the 49th byte of data. Figure 67 shows the data input timing for the RUTOPIA block in SPHY mode. The timing parameters used in Figure 67 are defined in the table following the figure. Refer section 3.5 “Receive UTOPIA Interface Block (RUTOPIA)” starting on page 44 for additional information.

Table 20. Transmit Signal Names and Corresponding UTOPIA Designations

Signal Name	UTOPIA Name
/TPHY_ADDR	TxAddr
TPHY_DATA	TxData
TPHY_SOC	TxSOC
TPHY_CLAV	TxClav
/TPHY_EN	TxEnb*
TPHY_CLK	TxClk

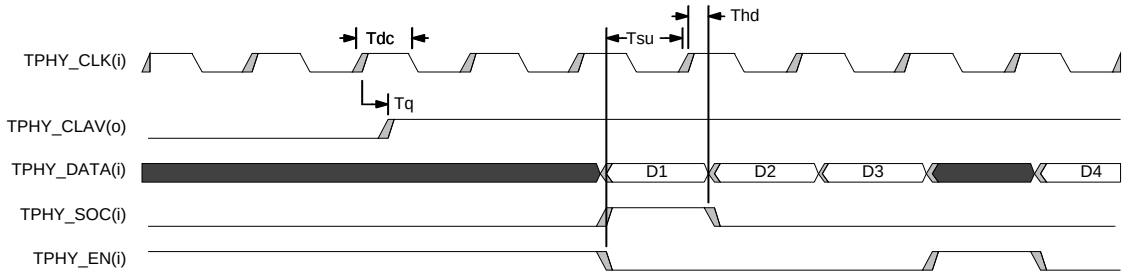


Figure 67. RUTOPIA SPHY Timing

Symbol	Parameter	Signal	Min	Max	Unit
	TPHY_CLK frequency			33	MHz
Tdc	TPHY_CLK duty cycle		45	55	%
Thd	TPHY_CLK hold time	TPHY_DATA, TPHY_SOC, /TPHY_EN	1		ns
Tq	TPHY_CLK-to-output delay	TPHY_CLAV		12	ns
Tsu	TPHY_CLK setup time	TPHY_DATA, TPHY_SOC, /TPHY_EN	5		ns

6.4.3 RUTOPIA as the PHY Layer Device in Multi-PHY (MPHY) Mode

The RUTOPIA block functions as a transmit UTOPIA block when in MPHY mode. The transmit MPHY UTOPIA timing signals are compatible with the UTOPIA Level 2 MPHY specification (refer to Appendix B, “References”, on page 203). Table 20 on page 101 indicates the transmit UTOPIA signal names and their corresponding UTOPIA designations. The AAL1gator II will not drive TPHY_CLAV unless /TPHY_ADDR was low the previous cycle. The value driven will be dependent on whether or not it has room for a complete cell. TPHY_DATA and TPHY_SOC are captured when /TPHY_EN is active and the AAL1gator II was selected. The AAL1gator II is selected only if /TPHY_ADDR is low and /TPHY_EN is high during one clock cyle, and /TPHY_EN is asserted low in the following clock cycle. TPHY_SOC indicates the start of a new cell. If the current cell will fill the internal FIFO, then TPHY_CLAV will be deactivated the same cycle it is receiving the 49th byte of data if /TPHY_ADDR is low on the previous cycle. Figure 68

shows the data input timing for the RUTOPIA block in MPHY mode. The timing parameters used in Figure 68 are defined in the table following the figure. Refer to section 3.5 “Receive UTOPIA Interface Block (RUTOPIA)” starting on page 44 for additional information.

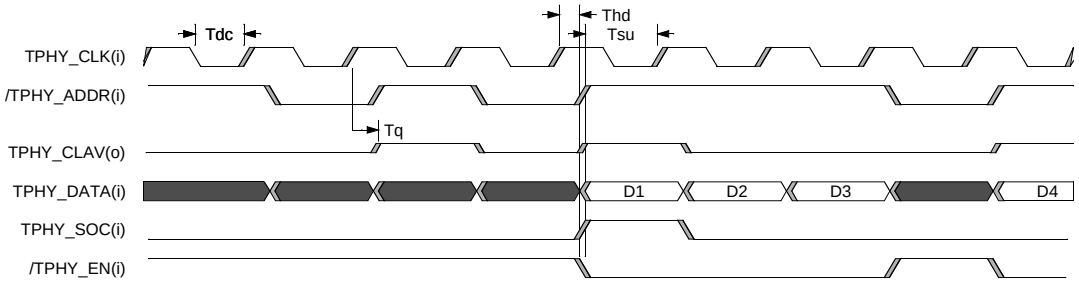


Figure 68. RUTOPIA MPHY Timing

Symbol	Parameter	Signal	Min	Max	Unit
	TPHY_CLK frequency			33	MHz
Tdc	TPHY_CLK duty cycle		45	55	%
Thd	TPHY_CLK hold time	/TPHY_ADDR, TPHY_DATA, TPHY_SOC, /TPHY_EN	1		ns
Tq	TPHY_CLK-to-output delay	TPHY_CLAV		12	ns
Tsu	TPHY_CLK setup time	/TPHY_ADDR, TPHY_DATA, TPHY_ SOC, /TPHY_EN	5		ns

6.5 RAM and Microprocessor Timing

6.5.1 RAM Timing

The RAM interface is designed to work with 12 ns SRAMs, which have a write data setup time of 7 ns or less, when SYS_CLK is 40 MHz (the maximum frequency). This interface is asynchronous and the timing parameters are given in this section. If the interface is used at a lower frequency, equations have been provided to calculate the RAM interface timing parameters. The timing is very dependent on the pulse width of SYS_CLK. The /MEM_WE signals are derived from the high pulse width of the SYS_CLK input. The high pulse width affects the pulse width of /MEM_WE and the setup and hold time of MEM_DATA to the rising edge of /MEM_WE.

Refer to section 8.6 “Board Requirements for the SRAM Interface” on page 174 for important information about interfacing to different speed SRAMS.

Figure 69 shows the timing information for all AAL1gator II-initiated RAM write cycles.

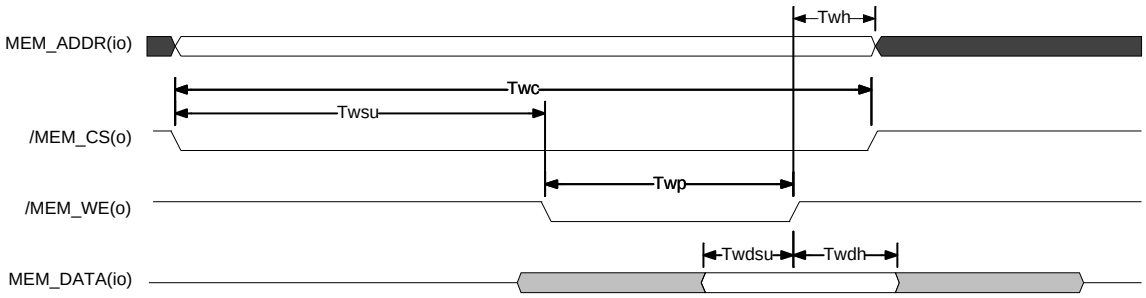


Figure 69. RAM Write Cycle Timing

Symbol	Parameter	Signals	Min	Max	Unit
Twc	Write cycle	/MEM_CS	Tp-2		ns
Twdh	Write data hold	MEM_DATA	Tp-Tch-10		ns
Twdsu	Write data setup	MEM_DATA	Tch - 4.3		ns
Twh	Write hold	/MEM_CS, MEM_ADDR, /MEM_WE	Tp -Tch -10		ns
Twsu	Write setup	MEM_ADDR, /MEM_CS	1		ns
Twp	Write pulse width	/MEM_WE	Tch-1.3	Tch+0.3	ns

NOTES:
- Tch and Tp are the clock high time and clock period as measured at 1.5 V. See Figure 80 on page 119 and refer to section 8.6 "Board Requirements for the SRAM Interface" on page 174.
- Test conditions are: /MEM_WE(0) and /MEM_WE(1) at 15 pF; and MEM_DATA, SP_DATA_EN, SP_DATA_CLK, SP_DATA_DIR, and /MEM_CS at 30 pF; MEM_ADDR and SP_ADDR_EN at 40 pF.
- All outputs are measured at 1.5 V, -40 to 85°C, 4.75 - 5.25 V.

Figure 70 shows the timing information for all RAM read cycles.

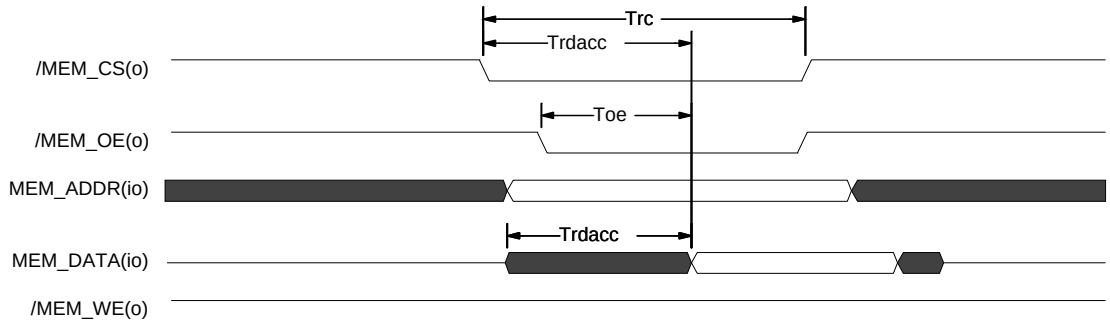


Figure 70. RAM Read Cycle Timing

Symbol	Parameter	Signals	Min	Max	Unit
Toe	RAM output enable delay	MEM_DATA	0	7	ns
Trc	Read cycle	/MEM_CS	Tp-2		ns
Trdacc	Read access time	MEM_DATA, /MEM_CS, /MEM_ ADDR		16 + Tsc	ns

NOTES:
- Tsc = (1 ÷ Fc) - (1 ÷ 38.87 MHz). For Fc and Tp values, see Figure 80 on page 119.
- Test conditions are: /MEM_WE(0) and /MEM_WE(1) at 15 pF; and MEM_DATA, SP_DATA_EN, SP_ DATA_CLK, SP_DATA_DIR, and /MEM_CS at 30 pF; MEM_ADDR and SP_ADDR_EN at 40 pF.
- All outputs are measured at 1.5 V, -40 to 85°C, 4.75 - 5.25 V.

6.5.2 Microprocessor Timing

Microprocessor accesses are controlled by the AAL1gator II. To properly access the AAL1gator II and the external memory, the AAL1gator II must be supported with a tristatable address buffer and a bidirectional tristatable data latch.

6.5.2.1 Microprocessor RAM Write Cycle Timing

Figure 71 on page 107 shows the timing for a microprocessor-initiated write cycle. It shows the 2-cycle write operation occurring immediately after a non-processor read cycle. RAM writing is not honored if higher priority internal functions request the memory, or the holdoff from a previous microprocessor transfer has not expired. /PROC_CS and /PROC_WR are double sampled (1 and 2) with the rising edge of SYS_CLK, and at (3) ADDR17 is sampled to distinguish between a command register write and a RAM write.

As long as HOLDOFF is not high, /SP_ADD_EN, /SP_DATA_EN, and SP_DATA_DIR are activated at the next clock cycle (3) allowing the microprocessor address and data to pass through the address and data buffer to the RAM. To minimize bus conflicts when the microprocessor access follows an AAL1gator II-initiated access, /SP_ADD_EN and /SP_DATA_EN signals are delayed.

At the next rising edge of SYS_CLK (4), /MEM_CS and /MEM_WE are activated. The /MEM_WE is delayed to provide sufficient setup time for /MEM_CS and MEM_DATA.

At the following clock cycle (5), /PROC_ACK is activated and /SP_DATA_EN, /SP_ADD_EN, and /MEM_CS are deactivated. The relative skew of these signals guarantees sufficient hold time. To eliminate bus contention, a recovery cycle is inserted between the microprocessor access and any subsequent access. /PROC_ACK is held active until /PROC_CS is deactivated.

Cycles (1) and (2) are grouped together in Figure 71 for the sake of convenience. These are normally two separate clock cycles.

NOTE: The timing characteristics (indicated by asterisks in the table following Figure 71) are based on external component requirements.

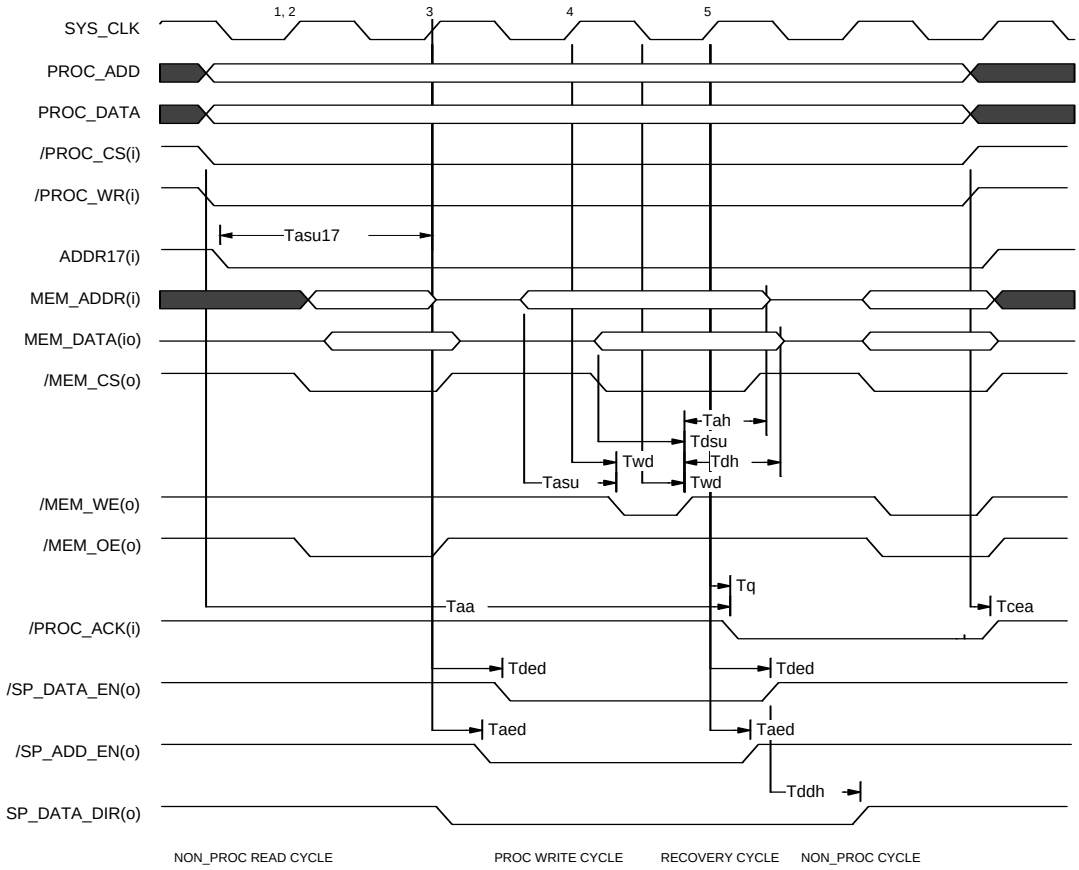


Figure 71. Microprocessor RAM Write Cycle Timing

Symbol	Parameter	Signals	Min	Max	Unit
Taa (See note below)	Acknowledge assertion after /PROC_ CS or /PROC_WR; whichever comes last	/PROC_ACK	5	29	SYS_CLK periods
Tasu	Address setup to write pulse*	MEM_ADDR, /MEM_WE	10		ns
Tcea	/PROC_CS deassertion to /PROC_ACK deassertion	/PROC_ACK	2	15	ns
Twd	Write pulse delay	/MEM_WE	8	25	ns
Tdsu	Data setup to write pulse off*	MEM_DATA	15		ns
Tded	Data enable delay	/SP_DATA_EN	7	25	ns
Taed	Address enable delay	/SP_ADD_EN	5	20	ns
Tq	Clock-to-output delay	SP_DATA_DIR	2	15	ns

Symbol	Parameter	Signals	Min	Max	Unit
Tq	Clock-to-output delay	/MEM_CS	2	18	ns
Tq	Clock-to-output delay	/PROC_ACK	2	18	ns
Tah	Address hold time*	MEM_ADDR, /SP_ADD_EN	5		ns
Tdh	Data hold time*	MEM_DATA, /SP_DATA_EN	5		ns
Tddh	Data direction delay hold time	/SP_DATA_EN, SP_DATA_DIR	10		ns
Tasu17	Address setup to SYS_CLK	ADDR17	2		ns

* These parameters are dependent on external components and assume that the requirements from Table 26 on page 195 are met.

NOTE: Taa is dependent on the HOLDOFF signal. If HOLDOFF is not asserted when the access begins, Taa will be a maximum of five SYS_CLK periods. If the access occurs immediately after another access, then Taa will be 23 to 29 SYS_CLK periods. Refer to section 6.5.3 “Microprocessor Holdoff Timing” on page 116 for a description of the HOLDOFF activity.

6.5.2.2 Microprocessor RAM Read Cycle Timing

Figure 72 on page 109 shows the timing for a microprocessor-initiated read cycle. It shows the two-cycle read operation occurring immediately after a non-microprocessor write cycle. RAM reading is not honored if higher priority internal functions request the memory, or the holdoff from a previous microprocessor transfer has not expired. For processor read operations, /SP_DATA_EN goes active when both /PROC_CS and /PROC_RD are active. /PROC_CS and /PROC_RD are double sampled (1 and 2) at the rising edge of SYS_CLK; and at (3) ADDR17 is sampled to distinguish between a command register read and a RAM read.

The /SP_ADD_EN is activated at the next clock cycle (3), as long as HOLDOFF is not high, to allow the microprocessor address to pass through to the RAM. Also /MEM_CS and /MEM_OE are activated to enable the RAM. In this case, /MEM_CS was already active due to the access from the previous cycle. SP_DATA_DIR remains high, allowing RAM data to pass through to the processor. At this time, SP_DATA_CLK also goes low in preparation of going high in a few cycles to latch the RAM data into the data latch. The /SP_ADD_EN is delayed to minimize bus conflicts when the microprocessor access follows a non-microprocessor access. The control signals remain constant through the next clock cycle (4) allowing the read data to stabilize.

In the following clock cycle (5), /SP_ADD_EN is deactivated and SP_DATA_CLK goes high, which clocks the RAM data into the data latch. The skew of /SP_ADD_EN guarantees sufficient hold time for the data to be latched. In the following clock cycle (6), /MEM_CS and /MEM_OE are deactivated and /PROC_ACK is activated.

To eliminate conflicts for this access sequence, a recovery cycle is inserted between the microprocessor access and any subsequent access. /PROC_ACK is driven until /PROC_CS is deactivated. /SP_DATA_EN is driven until either /PROC_CS or /PROC_OE are deactivated.

Cycles (1) and (2) are grouped together in Figure 72 for the sake of convenience. These are normally two separate clock cycles.

NOTE: The timing characteristics (indicated by asterisks in the table following Figure 72) are based on external component requirements.

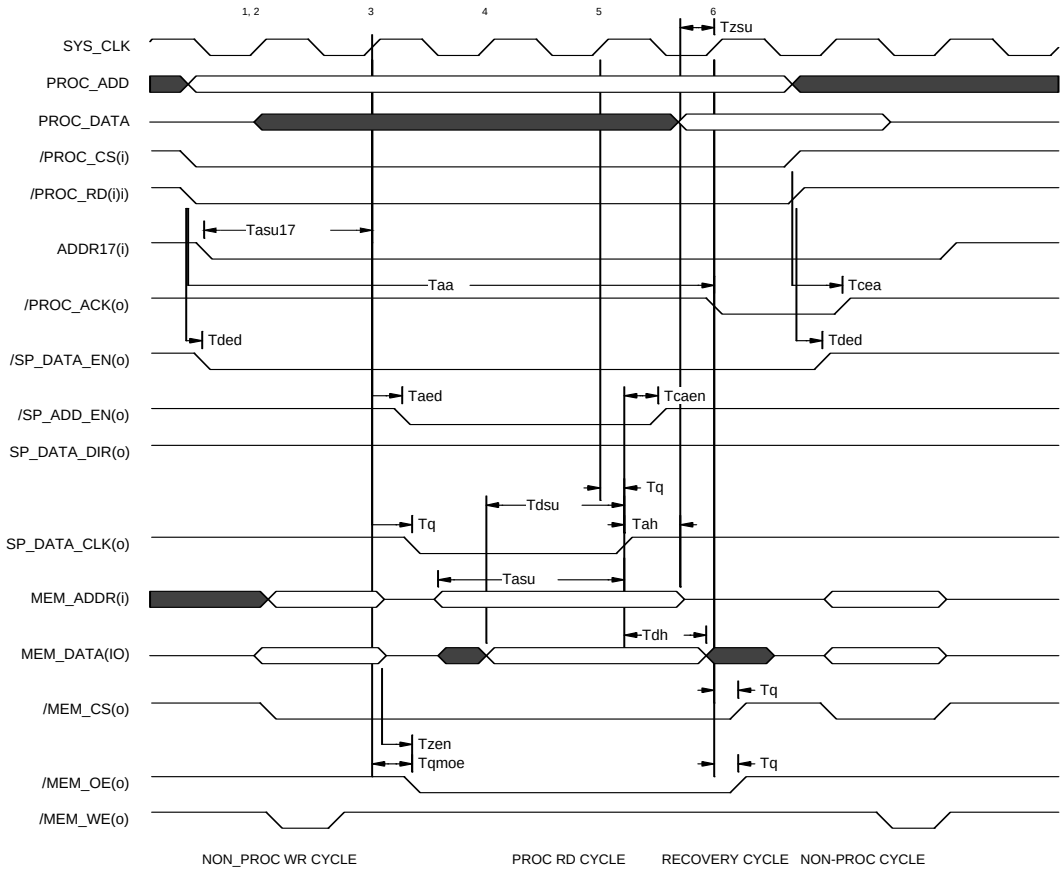


Figure 72. Microprocessor RAM Read Cycle Timing

Symbol	Parameter	Signals	Min	Max	Unit
Taa (Refer to NOTE below)	Acknowledge assertion after /PROC_CS or /PROC_RD; whichever comes last	/PROC_ACK	5	29	SYS_CLK periods
Tzen	Time from z state to enable	MEM_DATA, /MEM_OE	1**		ns

Symbol	Parameter	Signals	Min	Max	Unit
Tq	Clock-to-output delay	SP_DATA_CLK,/MEM_OE (deactivate)	2	15	ns
Tq	Clock-to-output delay	/MEM_CS	2	18	ns
Tqmoe	Clock-to-output delay for activation of /MEM_OE	MEM_OE (activate)	2	25	ns
Tasu17	Address setup to SYS_CLK	ADDR17	2		ns
Taed	Address enable delay	/SP_ADD_EN	5	20	ns
Tasu	Address setup to SP_DATA_CLK*	MEM_ADDR, SP_DATA_CLK	26		ns
Tded	Data enable delay from /PROC_CS and /PROC_RD	/SP_DATA_EN, /PROC_CS, /PROC_RD	2**	15**	ns
Tcea	/PROC_CS deassertion to /PROC_ ACK deassertion	/PROC_ACK	2	15	ns
Tdsu	Data setup to SP_DATA_CLK*	MEM_DATA	11		ns
Tdh	Data hold from SP_DATA_CLK*	MEM_DATA, SP_DATA_CLK	3		ns
Tcaen	SP_DATA_CLK high to /SP_ADD_EN high	SP_DATA_CLK, /SP_ADD_EN	0		ns
Tah	MEM_ADDR hold time from SP_ DATA_CLK*	MEM_ADD, SP_DATA_CLK	1		ns
Tzsu	Z state setup-to-clock	MEM_ADDR	3**		ns

* These parameters are dependent on external components and assume that the requirements from Table 26 on page 195 are met.
** These parameters are typical only.

NOTE: Taa is dependent on the HOLDOFF signal. If HOLDOFF is not asserted when the access begins, Taa will be a maximum of six SYS_CLK periods. If the access occurs immediately after another access, then Taa will be 24 to 30 SYS_CLK periods. Refer to section 6.5.3 “Microprocessor Holdoff Timing” on page 116 for a description of the HOLDOFF activity.

6.5.2.3 Microprocessor Write Command Register Timing

Figure 73 on page 112 shows the write command register timing. Writing to the internal command register is not honored if higher priority internal functions request the memory, or if the holdoff from a previous microprocessor transfer has not expired.

/PROC_CS and /PROC_WR are double sampled (1 and 2) at the rising edge of SYS_CLK, and at (3) ADDR17 is sampled to distinguish between a command register write and a RAM write.

As long as HOLDOFF is not high, /SP_ADD_EN, /SP_DATA_EN, and SP_DATA_DIR are activated at the next clock cycle (3), allowing the microprocessor address and data to pass through the address and data buffer to the AAL1gator II. The /SP_ADD_EN and the /SP_DATA_EN signals are delayed to minimize bus conflicts when the microprocessor access follows an

AAL1gator II-initiated access. Since all address bits, except ADDR17, are ignored for command register write operations (unless PROC_TEST_ACCESS is set (refer to "(Reserved)" on page 165)), the timing of the lower 16 address bits is not critical for this operation and is not shown.

Internally, a delayed version of SYS_CLK is used to latch the data. Since this internal clock is not visible on the outside, no setup or hold times are given. As long as the output enable and output disable delay of the external data buffer meet the parameters in the following table, the setup and hold time of the internal clock will be met.

At the following clock cycle (4), /PROC_ACK is activated and /SP_DATA_EN and /SP_ADD_EN are deactivated. The relative skew of these signals guarantees sufficient hold time. To eliminate bus contention, a recovery cycle is inserted between the microprocessor access and any subsequent access. /PROC_ACK is held active until /PROC_CS is deactivated.

Cycles (1) and (2) are grouped together in Figure 73 for the sake of convenience. These are normally two separate clock cycles.

NOTE: The timing characteristics (indicated by asterisks in the table following Figure 73) are based on external component requirements.

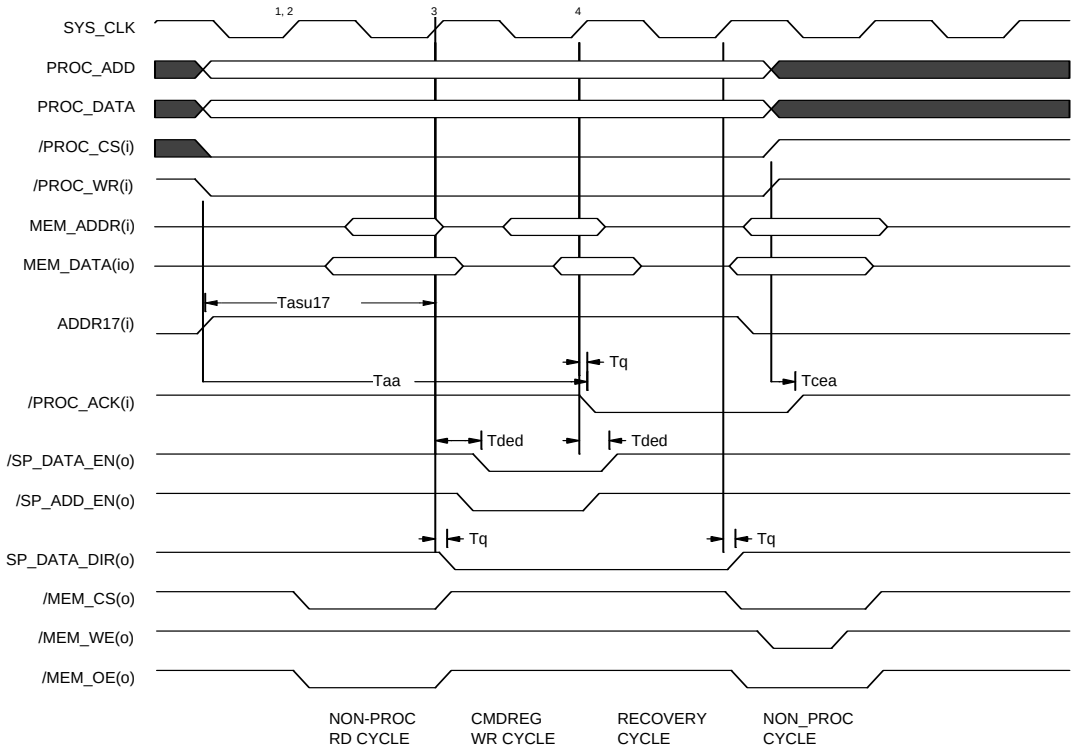


Figure 73. Microprocessor Write Command Register Timing

Symbol	Parameter	Signals	Min	Max	Unit
Taa (Refer to NOTE below)	Acknowledge assertion after /PROC_CS or /PROC_WR; whichever occurs last	/PROC_ACK	5	29	SYS_CLK periods
Tasu17	Address setup to SYS_CLK	ADDR17	2		ns
Tcea	/PROC_CS deassertion to /PROC_ACK deassertion	/PROC_ACK, /PROC_CS	2	15	ns
Tded	Data enable delay from SYS_CLK	/SP_DATA_EN, SYS_CLK	7	25	ns
Tq	Clock-to-output delay	SP_DATA_DIR	2	15	ns
Tq	Clock-to-output delay	/PROC_ACK	2	18	ns

NOTE: Taa is dependent on the HOLDOFF signal. If HOLDOFF is not asserted when the access begins, Taa will be a maximum of four SYS_CLK periods. If the access occurs immediately after another access, then Taa will be 23 to 29 SYS_CLK periods. Refer to section 6.5.3 “Microprocessor Holdoff Timing” on page 116 for a description of the HOLDOFF activity.

6.5.2.4 Microprocessor Read Command Register Timing

Figure 74 on page 115 shows the read command register timing. Four SYS_CLK cycles are required to read the internal command register. However, the read operation is not honored if higher priority internal functions request the memory, or the holdoff from a previous microprocessor transfer has not expired.

The /PROC_CS and /PROC_RD signals are double sampled (1 and 2) at the rising edge of SYS_CLK, and at (3) ADDR17 is sampled to distinguish between a command register read and a RAM read. For microprocessor read operations, /SP_DATA_EN goes active when both /PROC_CS and /PROC_RD are active, allowing data to pass through the data buffer to the microprocessor.

As long as HOLDOFF is not high, /SP_ADD_EN is activated at the next clock cycle (3) allowing the microprocessor address to pass through the address buffer to the AAL1gator II and SP_DATA_CLK is also driven low. /SP_ADD_EN is delayed to minimize bus conflicts when the microprocessor access follows an AAL1gator II-initiated access. Since all address bits, except for ADDR17 are ignored for command register read operations (unless PROC_TEST_ACCESS is set), the timing of the lower 16 address bits is not critical for this operation and is not shown.

At the following clock cycle (4), /SP_ADD_EN is deactivated but /SP_DATA_EN remains active. At clock cycle (5) the AAL1gator II begins driving valid data that is latched into the data latch in cycle (6) when SP_DATA_CLK is driven high.

At the following clock cycle (7), /PROC_ACK is activated and MEM_DATA is deactivated. A recovery cycle is inserted between the microprocessor access and any subsequent access to eliminate bus contention. /PROC_ACK is held active until /PROC_CS is deactivated. /SP_DATA_EN is held active until /PROC_CS or /PROC_RD is deactivated to ensure the microprocessor reads the data.

Cycles (1) and (2) are grouped together in Figure 74 for the sake of convenience. These are normally two separate clock cycles.

NOTE: The timing characteristics (indicated by asterisks in the table following Figure 74) are based on external component requirements.

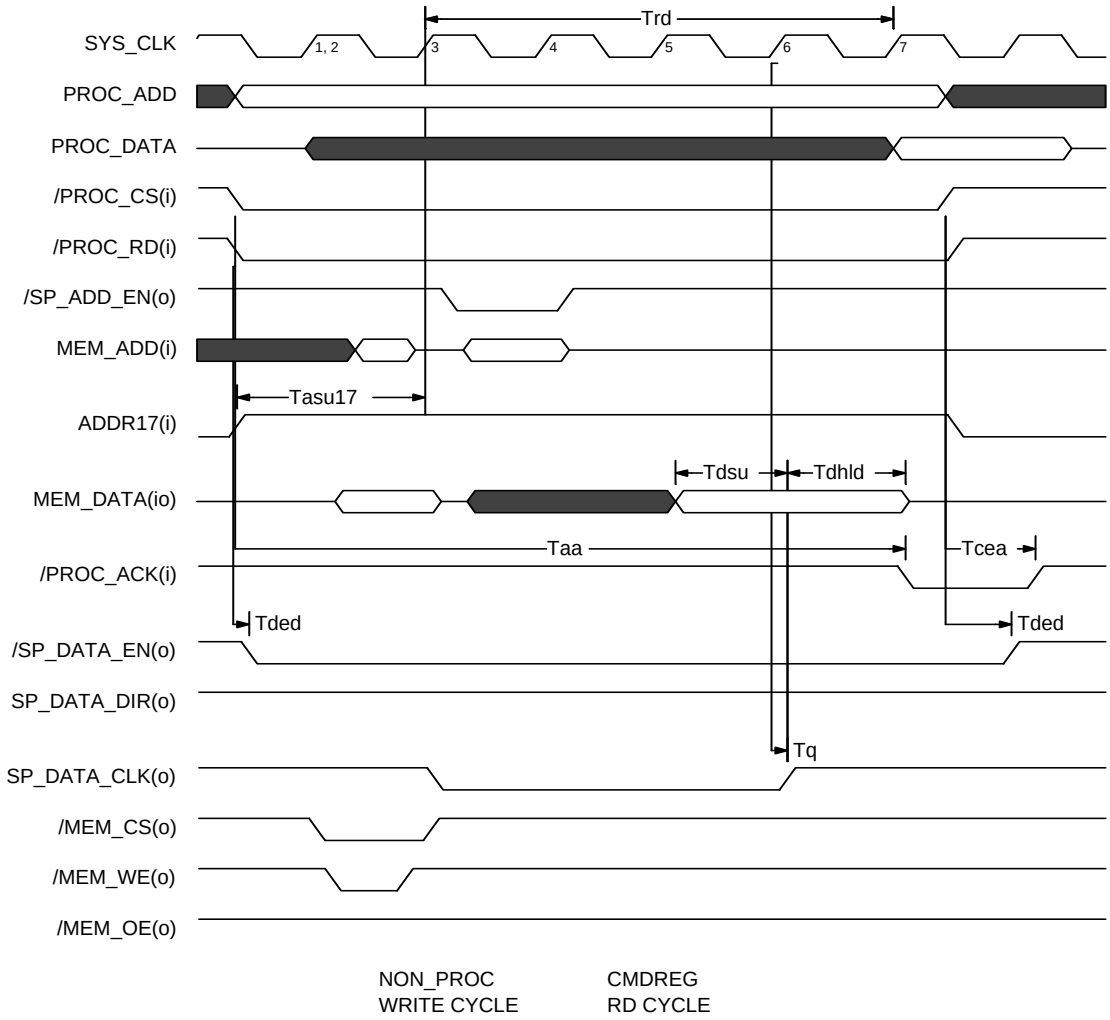


Figure 74. Microprocessor Read Command Register Timing

Symbol	Parameter	Signals	Min	Max	Unit
Taa (Refer to NOTE below)	Acknowledge assertion after /CS or /RD, whichever comes last	/PROC_ACK	5	29	SYS_CLK periods
Tasu17	Address setup time	MEM_ADDR	2		ns
Trd	Read operation time		4	4	SYS_CLK periods
Tq	Clock-to-output delay	SP_DATA_CLK	2	15	ns
Tded	Data enable delay from /MEM_ CS and /MEM_RD	/SP_DATA_EN, /MEM_CS, /MEM_RD	2	15	ns
Tdsu	Data setup to SP_DATA_CLK	MEM_DATA	15		ns
Tdhld	Data hold from SP_DATA_CLK	MEM_DATA	15		ns
Tcea	/PROC_CS deassertion to /PROC_ACK deassertion	/PROC_ACK	2	15	ns

NOTE: Taa is dependent on the HOLDOFF signal. If HOLDOFF is not asserted when the access begins, Taa will be a maximum of seven SYS_CLK periods. If the access occurs immediately after another access, then Taa will be 23 to 29 SYS_CLK periods. Refer to section 6.5.3 “Microprocessor Holdoff Timing” for a description of HOLDOFF activity.

6.5.3 Microprocessor Holdoff Timing

Figure 75 on page 116 shows the microprocessor holdoff timing. After the microprocessor accesses the AAL1gator II, the holdoff circuit prevents the microprocessor from obtaining another back-to-back access for 20 SYS_CLK cycles. However, holdoff does not apply while SW_RESET, in the command register is set. The AAL1gator II holdoff counter is activated after every processor access that occurs while the device is not in the RESET state. The holdoff counter counts up to 20 and then freezes at that count. Microprocessor accesses are not honored unless this count has completed.

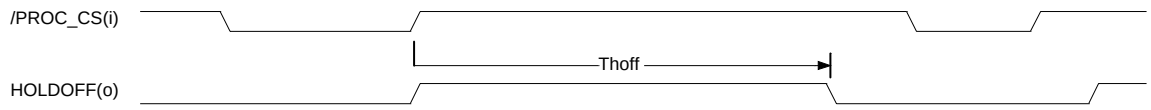


Figure 75. Microprocessor Holdoff Timing

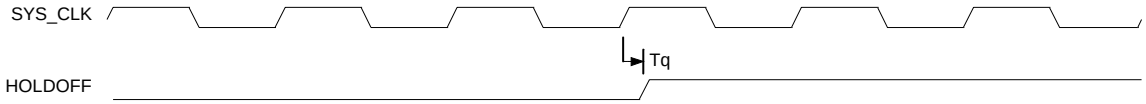


Figure 76. Microprocessor Output Delay Timing

Symbol	Parameter	Signals	Min	Max	Unit
Thoff	Holdoff time	HOLDOFF	20	20	SYS_CLK periods
Tq	Clock-to-output delay	HOLDOFF	2	16	ns

6.6 Interrupt Timing

The PROC_INT goes high when an OAM cell has been received and the OAM_INT_MASK bit in CMDREG is cleared. PROC_INT goes inactive when the CLR_RX_OAM_LATCH bit is set in CMDREG or the mask bit is set. See Figure 77 for interrupt timing.

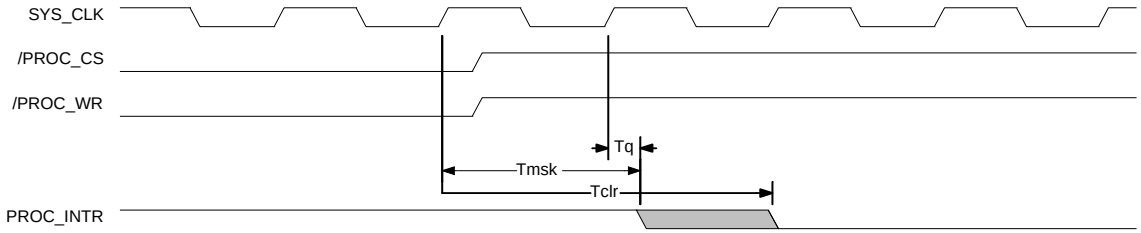


Figure 77. Interrupt Timing

Symbol	Parameter	Signals	Min	Max	Unit
Tq	Clock-to-output delay	SYS_CLK, PROC_INTR	2	17	ns
Tmsk	Clear intr due to mask	PROC_CS, PROC_INTR	1	1	SYS_CLK cycles
Tclr	Clear intr due to clear latch bit	PROC_CS, PROC_INTR	2	2	SYS_CLK cycles

6.7 SRTS Timing

The SRTS interface timing requirements for the low-speed and high-speed interfaces are shown respectively in Figure 78 and Figure 79. Low-speed SRTS timing is for UDF-ML, SDF-FR, and SDF-MF modes; and high-speed SRTS timing is used for UDF-HS mode.

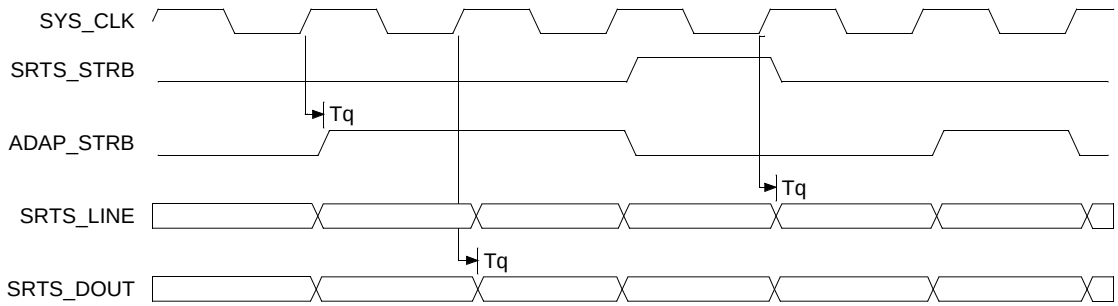


Figure 78. Low-Speed SRTS Timing

Symbol	Parameter	Signals	Min	Max	Unit
Tq	Clock-to-output delay	SRTS_LINE, SRTS_DOUT, SRTS_STRB, ADAP_SRTB	2	19	ns

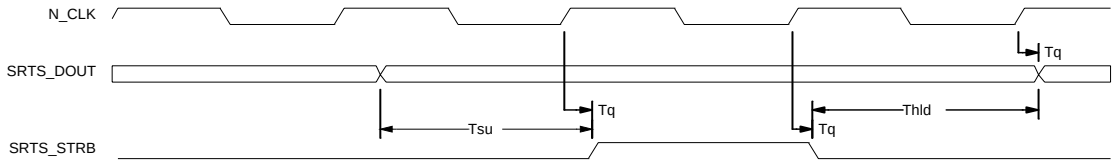


Figure 79. High-Speed SRTS Timing

Symbol	Parameter	Signals	Min	Max	Unit
Tq	Clock-to-output delay	SRTS_DOUT, SRTS_STRB		18	ns
Tsu	Data setup	SRTS_DOUT	1	1	N_CLK cycle
Thld	Data hold	SRTS_DOUT	1	1	N_CLK cycle

6.8 Miscellaneous Timing

6.8.1 SYS_CLK Timing

Figure 80 shows the timing for the SYS_CLK signal.

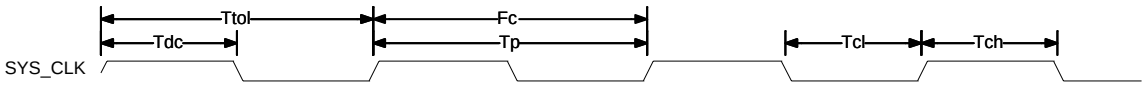


Figure 80. SYS_CLK Timing

Symbol	Parameter	Signals	Min	Max	Unit
Fc	SYS_CLK frequency	SYS_CLK		40.00	MHz
Tch	SYS_CLK high time	SYS_CLK	**		ns
Tcl	SYS_CLK low time	SYS_CLK	**		ns
Tp	SYS_CLK period	SYS_CLK	25.7		ns
Tdc	SYS_CLK duty cycle	SYS_CLK	**	**	%
Ttol	SYS_CLK tolerance	SYS_CLK*		50	ppm

NOTES:
- The 8-line throughput is guaranteed only at Fc minimum of 38.87 MHz. For each line that is not used, the minimum frequency can be decreased by 4.5 MHz, if SRTS is not used.
- *If TL_CLK synthesis is not used, this value can be relaxed to 200 ppm.
- ** The SYS_CLK pulse width and duty cycle is dependent on frequency, and external component selection. Refer to section 8.6 “Board Requirements for the SRAM Interface” on page 174 for more detailed requirements.

6.8.2 RESET Timing

Figure 81 shows the timing for the RESET signal.

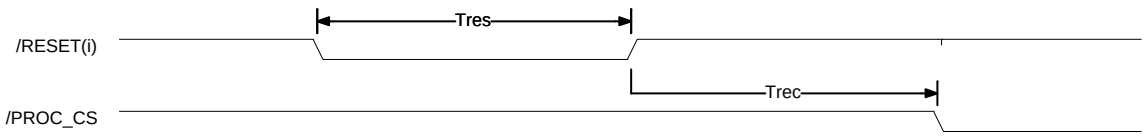


Figure 81. Reset Timing

Symbol	Parameter	Signals	Min	Max	Unit
Tres	Reset assertion time	/RESET	Minimum /RESET pulse width must be more than three clock times of the slowest active RL_CLK, active TL_CLK, or SYS_CLK.		
Trec	Reset recovery time	/RESET	Minimum /RESET recovery time must be more than three clock times of the slowest active RL_CLK, active TL_CLK, or SYS_CLK.		

6.8.3 JTAG Timing

Figure 82 provides timing information regarding the JTAG port.

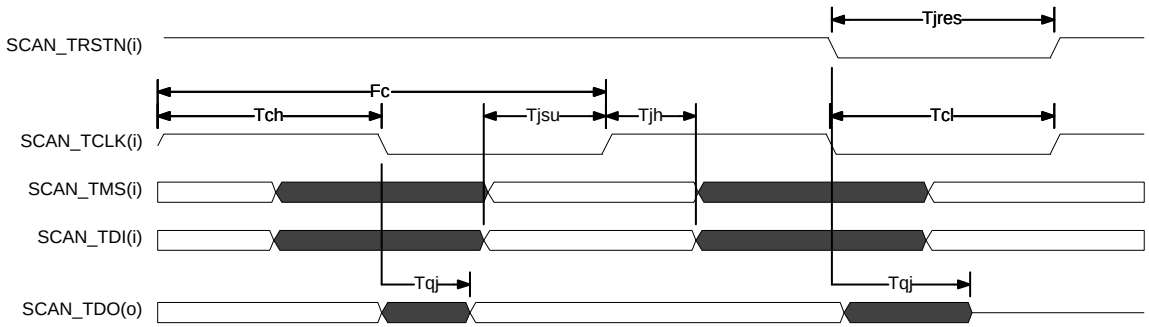


Figure 82. JTAG Timing

Symbol	Parameter	Signals	Min	Max	Unit
Fc	SCAN_TCLK frequency	SCAN_TCLK		5	MHz
Tch	SCAN_TCLK high period	SCAN_TCLK	80		ns
Tcl	SCAN_TCLK low period	SCAN_TCLK	80		ns
Tjh	SCAN_TCLK hold time	SCAN_TMS, SCAN_TDI	40		ns
Tjres	/SCAN_TRST low	/SCAN_TRST	80		ns
Tjsu	SCAN_TCLK setup time	SCAN_TMS, SCAN_TDI	40		ns
Tqj	SCAN_TCLK-to-output delay /SCAN_TRST-to-output delay	SCAN_TDO	2	40	ns

7 CONTROL REGISTERS AND DATA STRUCTURES

7.1 General

The initial programming for the AAL1gator II is performed by loading the external memory with specified information while a SW_RESET (refer to "SW_RESET" on page 165) is applied. The SW_RESET state is entered after a hardware reset is removed, or it can be asserted by writing the command register. After the memory is initialized, the CMD_REG_ATTN bit (refer to "CMD_REG_ATTN" on page 166) should be set so the configuration data can be read. Then 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). Since there are 128K words of memory and the memory itself is only 16 bits, complete byte pointers cannot be stored in 1-word memory locations. To achieve 1-word pointers, all structures except the receive multiframe buffers are stored in the first half of the memory. The receive buffers, which occupy 64K words, are the only structures located in the second half of the memory.

Figure 83 on page 122 shows the distribution of the data structures within the AAL1gator II. All registers except CMDREG are stored in the SRAM and all are readable.

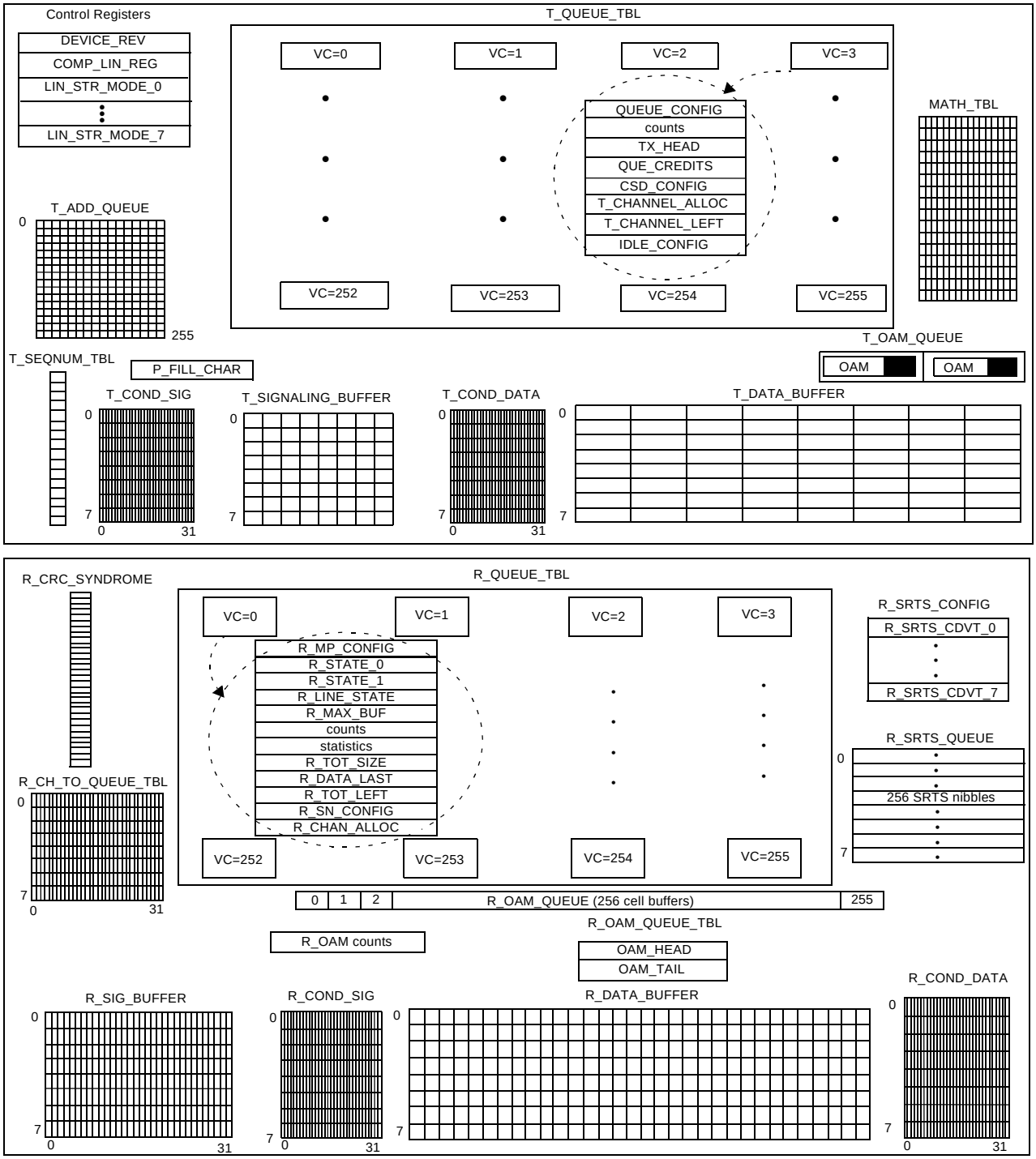


Figure 83. Transmit and Receive Data Structures

7.2 Initialization

The memory must be initialized to 0 (unless otherwise indicated) before the software reset is released. The number of data structures used by the device in reserved areas depends on this initialization.

7.3 Control Registers Summary

Table 21. Control Register Summary

Address	Name	R/W	Organization	Amount of Memory	Description
$0\ 0000_h$	DEVICE_REV	R/W	1 word	2 bytes	The Device Revision register indicates the revision of the device.
$0\ 0001_h$	COMP_LIN_REG	R/W	1 word	2 bytes	The Composite Line Register provides overall mode information.
$0\ 0010_h$	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.
$0\ 0011_h$	LIN_STR_MODE_1	R/W	1 word	2 bytes	
$0\ 0012_h$	LIN_STR_MODE_2	R/W	1 word	2 bytes	
$0\ 0013_h$	LIN_STR_MODE_3	R/W	1 word	2 bytes	
$0\ 0014_h$	LIN_STR_MODE_4	R/W	1 word	2 bytes	
$0\ 0015_h$	LIN_STR_MODE_5	R/W	1 word	2 bytes	
$0\ 0016_h$	LIN_STR_MODE_6	R/W	1 word	2 bytes	
$0\ 0017_h$	LIN_STR_MODE_7	R/W	1 word	2 bytes	

7.4 Control Register Descriptions

NOTES:

- All ports marked as “Reserved” must be initialized to 0 at initial setup. Software modifications to these locations after setup will cause incorrect operation.
- All read/write port bits marked “Not used” must be written with the value 0 to maintain software compatibility with future versions.

- 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.

7.4.1 DEVICE_REV

Organization: 1 word
Base address: 0_h
Type: Read/Write
Hardware Reset Value: Undefined.
Function: Stores the device revision.
Format: Refer to the following table.

Field (Bits)	Description
DEVICE_REV (15:0)	This register is undefined until the microprocessor asserts CMD_REG_ATTN, at which time the AAL1gator II writes it with $121A_h$. The revision of the device. This field should be initialized to 0.

7.4.2 COMP_LIN_REG

Organization: 1 word
Base address: 1_h
Type: Read/Write
Function: Stores the global configuration.
Format: Refer to the following table.

Field (Bits)	Description
Reserved (15:14)	Initialize to 0.
SHIFT_VCI (13)	Selects the VCI address range. 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.
Reserved(CT) (12)	Write with a 0 to maintain future software compatibility.
Not used (11:7)	Write with a 0 to maintain future software compatibility.
MIXED_MODE_EN (6)	Enables each line to operate in a different mode. 0 All lines use the mode defined by T1_MODE. 1 Device is in mixed mode. LINE_MODE bits are enabled in LIN_STR_MODE register. Initialize to the proper value.
Not used (5)	Write with a 0 to maintain future software compatibility.
SPHY_EN (4)	An active high signal that enables single PHY UTOPIA mode. This signal is valid only when PHY_ENABLE input is tied high.
Not used (3)	Write with a 0 to maintain future software compatibility.
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.
Reserved (1)	Initialize to 0.
T1_MODE (0)	The device is in the T1 mode. 0 Device is in E1 mode. 1 Device is in T1 mode. Initialize to the proper value.

NOTE: To operate lines in the device in UDF-ML mode, set the T1_MODE bit of this register to 0 or to 1 as appropriate. This will allow other lines to operate in the desired T1 or E1 mode. Then, in the LIN_STR_MODE register (refer to section 7.4.3 “LIN_STR_MODE” on page 126), set the FR_STRUCT field of each line to 10.

7.4.3 LIN_STR_MODE

Organization: Eight words.
Base address: 10_h
Index: 1_h
Type: Read/Write
Function: Stores the per-line configuration.
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.
5_h	LIN_STR_MODE_5	Line structure mode for line 5.
6_h	LIN_STR_MODE_6	Line structure mode for line 6.
7_h	LIN_STR_MODE_7	Line structure mode for line 7.

Field (Bits)	Description
HS_TX_COND (15)	Send cells with all 1s when in high-speed mode (line 0 only).
HS_RX_COND (14)	Fetch receive conditioning data from the R_COND_DATA buffer for line 0, channels 0 and 1. High-speed mode (line 0 only).
T1_MODE (13)	Determines mode of the line. This bit is valid only if the MIXED_MODE_EN bit in the COMP_LIN_REG is set. 1 Line is in T1 mode. 0 Line is in E1 mode.
E1_WITH_T1_SIG (12)	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. 0 Use E1 signaling.
Not used (11:6)	Write with a 0 to maintain future software compatibility.
CLK_SOURCE (5:4)	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. 00 Use external clock. (TL_CLK is an input). 01 LOOPED - Use RL_CLK as the clock source. 10 NOMINAL - Generate a clock of the nominal (T1 or E1) frequency from SYS_CLK. 11 SRTS - Generate a clock frequency based on the received SRTS values.
Not used (3)	Write with a 0 to maintain future software compatibility.

Field (Bits)	Description
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. 0 The CSI bits of the odd transmit AAL1 cells are set to 0.
FR_STRUCT (1:0)	Frame structure mode. 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 × line. For example, line 3 uses T_QUEUE_ TBL and R_QUEUE_TBL entry 96. 00 Not used. Initialize to the proper value.

7.5 Transmit Data Structures Summary

Table 22. Transmit Data Structures Summary

Name	R/W	Organization	Amount of Memory	Address	Description
P_FILL_CHAR	R/W	1 word	2 bytes	$0\ 0004_h$	The empty bytes in a partially filled cell are filled with P_FILL_CHAR.
T_ADD_QUEUE	R/W	16 words	32 bytes	$0\ 0030_h$ - $0\ 003F_h$	Bit table of queues to be added to the calendar queue after a CSD_ATTN assertion.
T_SEQNUM_TBL	R/W	16 words	32 bytes	$0\ 0020_h$ - $0\ 002F_h$	The Transmit Sequence Number Table is initialized according to a table.
T_COND_SIG	R/W	32 bytes × 8 lines	256 bytes	$0\ 0400_h$ - $0\ 047F_h$	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 × 8 lines	256 bytes	$0\ 0480_h$ - $0\ 04FF_h$	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	$0\ 0700_h$ - $0\ 07FF_h$	Reserved (Frame Advance FIFO).
Reserved	R/W	8 × 128 × 2 words	4 kBytes	$0\ 0800_h$ - $0\ 0FFF_h$	Reserved (Transmit Calendar).
Reserved	R/W	8 × 256 bytes	2 kBytes	$0\ 1000_h$ - $0\ 13FF_h$	Reserved (Transmit Signaling Buffer).
T_OAM_QUEUE	R/W	2 × 32 words	128 bytes	$0\ 1400_h$ - $0\ 143F_h$	The Transmit OAM Queue contains the OAM cells to be transmitted.
T_QUEUE_TBL	R/W	256 × 32 words	16 kBytes	$0\ 2000_h$ - $0\ 3FFF_h$	The Transmit Queue Table contains all pointers and variables that are queue-dependent.
Reserved	R/W	8 × 2 K words	32 kBytes	$0\ 4000_h$ - $0\ 7FFF_h$	Reserved (Transmit Data Buffer).
MATH_TBL	R/W	8 K words	16 kBytes	$0\ C000_h$ - $0\ DFFF_h$	The math table is initialized from formulas.

7.6 Transmit Data Structures Descriptions

NOTE:

- All ports marked as “Reserved” must be initialized to 0 at initial setup. Software modifications to these locations after setup will cause incorrect operation.
- All read/write port bits marked “Not used” must be written with the value 0 to maintain software compatibility with future versions.
- 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.

7.6.1 P_FILL_CHAR

Organization: One word
Base address: 4_h
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.

7.6.2 T_ADD_QUEUE

Organization: 16 words
Base address: 30_h
Index: 2_h
Type: Read/Write
Function: Bit table of queues to be added to the calendar queue. To add queues, set the proper bits in this table to 1. Ensure all other bits in this table are set to 0, then assert CSD_ATTN in the CMDREG. This table will be scanned in its entirety after the CSD_ATTN is detected. Bits left asserted from earlier add activities will be added back in with undesirable results.
Initialization: Initialize to 0.

Offset	Description
0_h	Queues 15:0 (line 0)
1_h	Queues 31:16 (line 0)
2_h	Queues 47:32 (line 1)
3_h	Queues 63:48 (line 1)
4_h	Queues 79:64 (line 2)

Offset	Description
5_h	Queues 95:80 (line 2)
6_h	Queues 111:96 (line 3)
7_h	Queues 127:112 (line 3)
8_h	Queues 143:128 (line 4)
9_h	Queues 159:144 (line 4)
A_h	Queues 175:160 (line 5)
B_h	Queues 191:176 (line 5)
C_h	Queues 207:192 (line 6)
D_h	Queues 223:208 (line 6)
E_h	Queues 239:224 (line 7)
F_h	Queues 255:240 (line 7)

7.6.3 T_SEQNUM_TBL

Organization: 16 words
Base address: 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 SRAM 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	$00FF_h$

7.6.4 T_COND_SIG

Organization: 32 bytes × 8 lines
Base address: 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 = ((channel -1) ÷ 2) + line × 16.
T_COND_SIG_B_H (10)	Transmit conditioned B bit for: Offset = ((channel -1) ÷ 2) + line × 16.
T_COND_SIG_C_H (9)	Transmit conditioned C bit or A bit if T1 SF for: Offset = ((channel -1) ÷ 2) + line × 16.
T_COND_SIG_D_H (8)	Transmit conditioned D bit or B bit if T1 SF for: Offset = ((channel -1) ÷ 2) + line × 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 × 16.
T_COND_SIG_B_L (2)	Transmit conditioned B bit for: Offset = (channel ÷ 2) + line × 16.

Field (Bits)	Description
T_COND_SIG_C_L (1)	Transmit conditioned C bit or A bit if T1 SF for: Offset = (channel ÷ 2) + line × 16.
T_COND_SIG_D_L (0)	Transmit conditioned D bit or B bit if T1 SF for: Offset = (channel ÷ 2) + line × 16.

7.6.5 T_COND_DATA

Organization: 32 bytes × 8 lines
Base address: 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 × 16.
T_COND_DATA_L (7:0)	Transmit conditioned data offset = (channel ÷ 2) + line × 16.

7.6.6 RESERVED (Transmit Signaling Buffer)

NOTE: 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 × 32 DS0s × 8 lines. 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 84 displays the contents of the transmit signaling buffer.

T_SIG_BUF_n Word Format

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

NOTE: The upper nibble of each byte is 0.
T1 format uses word addresses 0 to 11.

Figure 84. SDF-MF Format of T_SIGNALING_BUFFER

7.6.7 T_OAM_QUEUE

Organization: 2 cells × 32 words
Base address: 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.	

7.6.8 T_QUEUE_TBL

Organization: 256 × 32 words
Base address: 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_SUPPRESSED_CELL_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.
8_h	TX_HEAD(2:1)	Header byte 1 in bits 15:8, header byte 2 in bits 7:0.
9_h	TX_HEAD(4:3)	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	IDLE_CONFIG	Controls transmission of data.
13_h-$1F_h$	Not used	Initialize to 0 each time this queue is initialized.

NOTE: All registers are right justified, with any unused bits set to 0.

T_COND_CELL_CNT Word Format

Field (Bits)	Description
T_COND_CELL_CNT (15:0)	A 16-bit rollover count of conditioned cells transmitted.

T_SUPPRESSED_CELL_CNT Word Format

Field (Bits)	Description
T_SUPPRESSED_CELL_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 (refer to "SUPPRESS_TRANSMISSION" on page 139) is set.

QUEUE_CONFIG Word Format

Field (Bits)	Description
TX_COND (15)	Sends data and signaling from the transmit conditioned data area. Initialize to the proper value.
TX_ACTIVE (14)	Enables this queue. To enable sending cells on a UDF-HS connection, assert this bit. To enable non-UDF-HS connections: 1. Assert this bit. 2. Set the bit corresponding to this queue in the T_ADD_QUEUE table (refer to section 7.6.2 "T_ADD_QUEUE" on page 128). 3. Assert the CSD_ATTN bit in the CMD_REG (refer to "CMDREG Word Format" on page 165). To disable non-UDF_HS connections, clear this bit. This queue is then removed from the calendar queue the next time a cell would have been sent. This queue must not be returned to the calendar queue until enough time has elapsed for this to have occurred. NOTE: When reactivating a previously active queue, be sure to reinitialize all the registers in the queue table for that queue.

Field (Bits)	Description
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_WITH_T1_SIG modes, the 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 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. NOTES: • For channels with a single DS0, encode the value 48 as one multiframe and 24 frames. • 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.
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 (refer to Appendix B, "References", on page 203) when using a single DS0 in SDF-FR mode, no pointer should be used.
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 value must be 47. 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. NOTE: 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.

T_CELL_CNT Word Format

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

TX_HEAD(1:2) Word Format

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

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

Field (Bits)	Description
TX_HEAD(5) (15:8)	Fifth header byte that contains the precalculated HEC word. Initialize to the proper value.
Not used (7:0)	Write with a 0 to maintain compatibility with future software versions.

QUE_CREDITS Word Format

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 00_b.
Not used (13:10)	Write with a 0 to maintain compatibility with future software versions.
QUEUE_CREDITS (9:0)	A 10-bit quantity representing the number of credits(bytes) accumulated for the queue. It is measured in eighths (three LSBs are fractional bits). Initialize to 47×8 (178_h) for UDF modes full cells and Single-DS0-no-pointer mode full cells, 46.875×8 (177_h) for SDF modes full cells, or to the partially filled cell length $\times 8$.

CSD_CONFIG Word Format

Field (Bits)	Description
NUM_CHAN (15:10)	A 6-bit integer specifying the number of DS0s carried by this queue. If a queue serves seven DS0s, initialize this field to 7. It is not used in UDF-ML and UDF-HS modes.
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 or Single-DS0-no-pointer mode. For partial cells, this value is the same as the partially filled value × 8. This field is not used in UDF-HS mode.

T_CHANNEL_ALLOC(15:0) Word Format

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 and UDF-HS modes.

T_CHANNEL_ALLOC(31:16) Word Format

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. Initialize to the proper value for SDF-MF and SDF-FR modes and to $FFFF_h$ for UDF-ML and UDF-HS modes.

T_CHANNEL_LEFT(15:0) Word Format

Field (Bits)	Description
T_CHANNEL_LEFT (15:0)	Initialize to the same value as T_CHAN_ALLOC(15:0).

T_CHANNEL_LEFT(31:16) Word Format

Field (Bits)	Description
T_CHANNEL_LEFT (31:16)	Initialize to the same value as T_CHAN_ALLOC(31:16).

IDLE_CONFIG Word Format

Field (Bits)	Description
SUPPRESS_TRANSMISSION (15)	Set to 1 to suppress the generation of cells for this queue. Cells continue to be scheduled, but are not built. This bit can be used to temporarily stop the generation of cells while a queue is reconfigured, if the line configuration and number of timeslots assigned to the queue do not change.
Not used (14:0)	Write with a 0 to maintain future software compatibility.

7.6.9 RESERVED (Transmit Data Buffer)

NOTE: 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 × 8 lines - Each line is allocated a separate 128 frame buffer memory. For E1 applications, this is large enough to store eight multiframes (32 DS0s × 16 frames × 8 multiframes = 4096 bytes). In T1 mode, 96 frames or four multiframes are stored (24 × 24 × 4 = 2880 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.

NOTE: If E1_with_T1_SIG is set (refer to bit 12 in section 7.4.3 “LIN_STR_MODE” starting on page 126), data is arranged as if in T1 mode.

Base address: 04000_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_n Word Format

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

7.6.10 MATH_TBL

Organization: 8 K words
Base address: $0C000_h$
Index: 1_h
Type: Read/Write
Function: The math table must be loaded into SRAM on every power cycling, except in UDF-HS mode, in which case the math table is not used. The math table stores the results of five different mathematical operations.

- DIV_OP performs the integer divide and roundup-to-next integer operation. The number of integer credits (six bits) is divided by the number of channels (five bits). These two fields are concatenated to form an 11-bit address into the table. The result is an 8-bit integer.
- MOD_ADD_OP adds the results of the frame location to the frame offset. In the case of T1 structured data, the mod is 120. All others use a module 128. The result is a 7-bit integer. A 7-bit frame value is concatenated with a 6-bit frame increment value to form a 13-bit address.
- MULT_OP determines the number of earned integer credits for next service time. It is obtained by multiplying the number of channels by the frame increment value. A 6-bit frame value is concatenated with a 5-bit channel count to create an 11-bit address.
- T1_MF_DIV_OP determines the number of 1/8 multiframes and the remainder of the division given a frame count for the T1 mode of operation.
- E1_MF_DIV_OP determines the number of 1/8 multiframes and the remainder of the division given a frame count for the E1 mode of operation.

Format: The four values resulting from the math operations are stored in the same table, as shown in Figure 85.

Field (Bits)	Description
This table is available in the Software Driver available at http://www.pmc-sierra.com.	

NOTE: The AAL1gator II uses a different math table than previous versions of the chip. The

same table is used for both E1 and T1 modes.

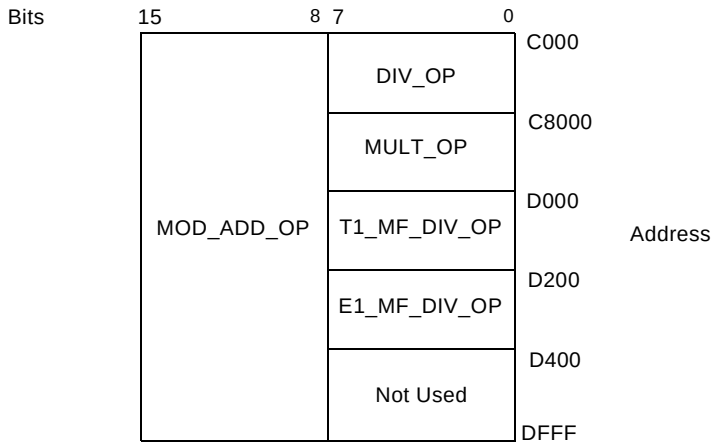


Figure 85. Math Operation Results

7.7 Receive Data Structures Summary

Table 23 lists the data structures unique to the receive side of the AAL1gator II.

Table 23. Receive Data Structures Summary

Data Structure Description	Organization	Amount of Memory	Address	Description
R_OAM_QUEUE_TBL	2 words	4 bytes	$0\ 8000_h$ - $0\ 8001_h$	Receive OAM head and tail pointers.
R_OAM_CELL_CNT	1 word	2 bytes	$0\ 8002_h$	Count of received OAM cells.
R_DROPPED_OAM_CELL_CNT	1 word	2 bytes	$0\ 8003_h$	Count of dropped OAM cells.
Reserved	16 words	32 bytes	$0\ 8020_h$ - $0\ 802F_h$	Reserved (SRTS Queue Pointers).
R_SRTS_CONFIG	2 bytes × 8 lines	16 bytes	$0\ 8038_h$ - $0\ 803F_h$	Receive SRTS configuration.
R_CRC_SYNDROME	128 words	256 bytes	$0\ 8080_h$ - $0\ 80FF_h$	Mask of bits. Initialized from a table.
R_CH_TO_QUEUE_TBL	128 words	256 bytes	$0\ 8200_h$ - $0\ 827F_h$	Receive channel to queue table.
R_COND_SIG	16 × 8 bytes	256 bytes	$0\ 8400_h$ - $0\ 847F_h$	Receive signaling conditioning values.
R_COND_DATA	32 × 8 bytes	256 bytes	$0\ 8480_h$ - $0\ 84FF_h$	Receive data conditioning values.
Reserved	8 × 256 words	4 kBytes	$0\ 8800_h$ - $0\ 8FFF_h$	Reserved (Receive SRTS Queue).
Reserved	8 x 32 × 16 words	8 kBytes	$0\ 9000_h$ - $0\ 9FFF_h$	Reserved (Receive Signaling Buffer).
R_QUEUE_TBL	256 × 32 words	16 kBytes	$0\ A000_h$ - $0\ BFFF_h$	Receive queue table.
R_OAM_QUEUE	256 × 64 bytes	16 kBytes	$0\ E000_h$ - $0\ FFFF_h$	Receive OAM queue.
Reserved	8 × 512 × 32 bytes	128 kBytes	$1\ 0000_h$ - $1\ FFFF_h$	Reserved (Receive Data Buffer).

7.8 Receive Data Structures Descriptions

This section describes the structures used by the receive side of the AAL1gator II. For structures that are common to both receive and transmit, refer to section 7.5 “Transmit Data Structures Summary” on page 127.

NOTES:

- All ports marked as “Reserved” must be initialized to 0 at initial setup. Software modifications to these locations after setup will cause incorrect operation.
- All read/write port bits marked “Not used” must be written with the value 0 to maintain software compatibility with future versions.
- 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.

7.8.1 R_OAM_QUEUE_TBL

Organization: 2 words
Base address: 0 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 (refer to "CMDREG Word Format" on page 165).

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.

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.

7.8.2 R_OAM_CELL_CNT

Organization: 1 word
Base address: 0 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_CELLCNT Word Format

Field (Bits)	Description
R_OAM_CELLCNT (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.

7.8.3 R_DROPPED_OAM_CELL_CNT

Organization: 1 word
Base address: 0 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_DROPPED_OAM_CELLCNT Word Format

Field (Bits)	Description
R_DROPPED_OAM_CELLCNT (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.

7.8.4 R_SRTS_CONFIG

Organization: 2 bytes × 8 lines
Base address: 0 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.

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 (4:0)	Receive SRTS CDVT.

7.8.5 R_CRC_SYNDROME

Organization: 128 words
Base address: 0 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
RX_CRC_SYNDROME (4:0)	Mask of bits to change.

Mask Bit Table

SAR-PDU Header	Offset	Data (Hex)
00	00	00
	01	10
	02	10
	03	01
	04	10
	05	08
	06	02
	07	04
	08	01
	09	10
	0A	10
17	0B	00
	0C	04
	0D	02
	0E	08
	0F	10
	10	02
	11	04
	12	10
	13	08
	14	10
	15	01
2D	16	00
	17	10
	18	08
	19	10
	1A	04
	1B	02
	1C	10

SAR-PDU Header	Offset	Data (Hex)
	40	08
	41	10
	42	04
	43	02
	44	10
8B	45	00
	46	01
	47	10
	48	02
	49	04
	4A	10
	4B	08
	4C	10
	4D	01
9C	4E	00
	4F	10
	50	01
	51	10
	52	10
A6	53	00
	54	04
	55	02
	56	08
	57	10
B1	58	00
	59	10
	5A	10
	5B	01
	5C	10

SAR-PDU Header	Offset	Data (Hex)
3A	1D	00
	1E	01
	1F	10
	20	04
	21	02
	22	08
	23	10
	24	01
	25	10
	26	10
4E	27	00
	28	10
	29	08
	2A	02
	2B	04
59	2C	00
	2D	10
	2E	10
	2F	01
	30	10
63	31	00
	32	01
	33	10
	34	08
	35	10
	36	04
	37	02
	38	10
	39	01
74	3A	00
	3B	10
	3C	02
	3D	04
	3E	10
	3F	08

SAR-PDU Header	Offset	Data (Hex)
	5D	08
	5E	02
	5F	04
	60	10
	61	01
C5	62	00
	63	10
	64	02
	65	04
	66	10
	67	08
	68	10
D2	69	00
	6A	01
	6B	10
	6C	08
	6D	10
	6E	04
	6F	02
	70	10
	71	08
	72	02
	73	04
E8	74	00
	75	10
	76	10
	77	01
	78	04
	79	02
	7A	08
	7B	10
	7C	01
	7D	10
	7E	10
FF	7F	00

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)

7.8.6 R_CH_TO_QUEUE_TBL

Organization: 128 words (8 lines × 32 DS0s)
Base address: 08200_h
Index: 1_h
Type: Read/Write
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. Initialize to the proper values. The AAL1gator II processes two bytes at a time, so the values in the following table are in pairs. For unstructured 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 II reads data for the line from the first word of the R_COND_DATA_0 table (refer to section 7.8.8 "R_COND_DATA" on page 151).
Format: Refer to the following table.

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

CH_TO_QUEUE Word Format

Field (Bits)	Description
RX_COND_H (15:14)	Determines the type of data to be played out: NOTE: Options "00", "01", and "11" are executed only when the queue is in an underrun or resume state. 00 When the queue is in underrun, read signaling for this channel from multiframe 0 and the data for this channel from the R_COND_DATA table (refer to section 7.8.8 "R_COND_DATA" on page 151). 01 When the queue is in underrun, read signaling for this channel from multiframe 0 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 Read signaling for this channel from the R_COND_SIG table (refer to section 7.8.7 "R_COND_SIG" on page 150) and the data for this channel from the R_COND_DATA table. 11 When the queue is in underrun, read the signaling from multiframe 0 and play out the contents of the buffer.
Not used (13)	Write with a 0 to maintain future software compatibility.

Field (Bits)	Description
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 = (channel - 1) ÷ 2 + line × 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: NOTE: 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, read signaling for this channel from multiframe 0 and the data for this channel from the R_COND_DATA table (refer to section 7.8.8 “R_COND_DATA” on page 151). 01_b When the queue is in underrun, read signaling for this channel from multiframe 0 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 (refer to section 7.8.7 “R_COND_SIG” on page 150) and the data for this channel from the R_COND_DATA table. 11_b When the queue is in underrun, read the signaling from multiframe 0 and play out the contents of the buffer.
Not used (5)	Write with a 0 to maintain future software compatibility.
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 × 16.

7.8.7 R_COND_SIG

Organization: 16 words × 8

Base address: 08400_h

Index: 10_h

Type: Read/Write

Function: This table stores the signaling to be used when RX_COND_H or RX_COND_L equals "10" 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.

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 = (channel - 1) ÷ 2 + line × 16.
R_COND_B_H (10)	Receive conditioned B signaling bit for: Offset = (channel - 1) ÷ 2 + line × 16.
R_COND_C_H (9)	Receive conditioned C signaling bit or A bit if T1 SF for: Offset = (channel - 1) ÷ 2 + line × 16.
R_COND_D_H (8)	Receive conditioned D signaling bit or B bit if T1 SF for: Offset = (channel - 1) ÷ 2 + line × 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 × 16.
R_COND_B_L (2)	Receive conditioned B signaling bit for: Offset = channel ÷ 2 + line × 16.

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

7.8.8 R_COND_DATA

Organization: 16 words × 8

Base address: 08480_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_h", "01_h", or "10_h".

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.

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 = (channel - 1) ÷ 2 + line × 16.
R_COND_DATA_L (7:0)	Receive conditioned data for: Offset = channel ÷ 2 + line × 16.

7.8.9 RESERVED (Receive SRTS Queue)

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

Organization: 64 words × 8 lines. Each line is allocated a separate 64-entry queue to store the SRTS receive nibbles.
Base address: 8800_h
Index: 100_h
Type: Read/Write
Function: The receive SRTS 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 (15:4)	Write with a 0 to maintain future software compatibility.
R_SRTS (3:0)	Receive SRTS data for line = offset ÷ 64.

7.8.10 RESERVED (Receive Signaling Buffer)

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

Organization: 32 × 32 DS0s × 8 lines. Each line is allocated a separate 32 × 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 transmit 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: 09000_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 (15:12)	Write with a 0 to maintain compatibility with future software versions.
R_SIG_H (11:8)	Receive signaling data for: Channel = (offset mod 16) × 2 + 1. Multiframe = (offset mod 512) ÷ 16. Line = offset ÷ 512. Offset = line × 512 + multiframe × 16 + (channel − 1) ÷ 2.
Not used (7:4)	Write with a 0 to maintain compatibility with future software versions.
R_SIG_L (3:0)	Receive signaling data for: Channel = (offset mod 16) × 2. Multiframe = (offset mod 512) ÷ 16. Line = offset ÷ 512. Offset = line × 512 + multiframe × 16 + channel ÷ 2.

7.8.11 R_QUEUE_TBL

Organization: 256×32 words
Base address: $0A000_h$
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.

Offset	Name	Description
0_h	R_STATE_0	Cell receiver state 0.
1_h	R_MP_CONFIG	Bytes per cell and CDVT constant.
2_h	R_STATE_1	Cell receiver state 1.
3_h	R_LINE_STATE	Line state.
4_h	R_MAX_BUF	Receive maximum buffer size.
5_h	R_SEQUENCE_ERR	16-bit rollover count of SN errors.
6_h	R_INCORRECT_SNP	16-bit rollover count of cells with incorrect SNP.
7_h	R_CELL_CNT	16-bit rollover count of played out cells.
8_h	R_ERROR_STKY	Receive sticky bits.
9_h	R_TOT_SIZE	Total bytes in structure.
A_h	R_DATA_LAST	Number of signaling bytes in structure.
B_h	R_TOT_LEFT	Number of bytes remaining in the structure. Initialize to 0 each time this queue is initialized.
C_h	Not used	Initialize to 0 each time this queue is initialized.
D_h	R_SN_CONFIG	Configures sequence number processing algorithm.
E_h	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.
F_h	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.
10_h	Reserved (CHNLEFTL)	Initialize to 0 each time this queue is initialized.
11_h	Reserved (CHNLEFTH)	Initialize to 0 each time this queue is initialized.
12_h	R_DROPPED_CELLS	16-bit rollover count of cells that were received but dropped. Initialize to 0.
13_h	R_UNDERRUNS	16-bit rollover count of the occurrences of underrun on this queue. Initialize to 0.
14_h	R_LOST_CELLS	16-bit rollover count of the number of lost cells for this queue. Initialize to 0.
15_h	R_OVERRUNS	16-bit rollover count of the occurrences of overrun on this queue. Initialize to 0.
16_h	R_PTR_REFRAMES	16-bit rollover count of the occurrences of pointer reframes. Initialize to 0.
17_h	R_PTR_PAR_ERR	16-bit rollover count of the occurrences of pointer parity errors. Initialize to 0.
18_h	R_MISINSERTED	16-bit rollover count of the occurrences of misinserted cells. Initialize to 0.

Offset	Name	Description
19_h-$1F_h$	Not used	Initialize to 0 each time this queue is initialized.
NOTE: All of these locations must be initialized whenever the queue is initialized.		

R_STATE_0 Word Format

NOTE: This word is read-only and is maintained by the RALP.

Field (Bits)	Description
Not used (15)	Initialize to 0 to maintain future software compatibility.
R_STRUCT_FOUND (14)	Indicates that the receiver structure was found. Initialize to 0.
Reserved(OLDUNDRN_N) (13)	Initialize to 0 to maintain future software compatibility.
Reserved(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

NOTE: 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.
Not used (8)	Write with a 0 to maintain future software compatibility.

Field (Bits)	Description
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-MF, E1_WITH_T1_SIG, or SDF-FR mode, the 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 µs periods.

R_STATE_1 Word Format

NOTE: This word is read-only and is maintained by the RALP.

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.
Not used (10:9)	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.

R_LINE_STATE Word Format

NOTE: This word is read-only after initialization and is maintained by the RALP and RFTC.

Field (Bits)	Description
R_UNDERRUN (15)	Indicates that this queue is currently in underrun. Initialize to 1. Not used in UDF-HS mode.
R_RESUME (14)	Indicates that this queue is currently in resume state. Initialize to 0. Not used in UDF-HS mode.
R_SIG_RESUME (13)	Indicates that this queue is currently in signal resume state. Initialize to 0. Not used in UDF-HS mode.

Field (Bits)	Description
Reserved (12:9)	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_MAX_BUF Word Format

NOTE: 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 (refer to Appendix B, "References", on page 203) when using a single DS0 in SDF-FR mode, no pointer should be used.
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.
Not used (12:9)	Write with 0 to maintain future software compatibility.
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 $1FE_h$. R_MAX_BUF should be set to at least 2 * R_CDVT, or R_CDVT + 2 * (number of frames per cell), whichever is greater.

R_SEQUENCE_ERR Word Format

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

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

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

NOTE: Receive sticky bits should be used for statistics gathering purposes only as there is no means of clearing them without the possibility of missing an occurrence. Initialize to 0.

Field (Bits)	Description
TRANSFER (15)	This bit is read then written with the same value each time the AAL1gator II receives a cell. This feature allows the processor to determine if the AAL1gator II 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 II has had its sticky word update interrupted.
CELL_RECEIVED (14)	A cell was received.
Not used (13:12)	Write with a 0 to maintain future software compatibility.
ALLOC_TBL_BLANK (11)	A cell was dropped because of a blank allocation table or because R_CHAN_DISABLE (refer to "R_MAX_BUF Word Format" on page 157) was asserted.
POINTER_SEARCH (10)	A cell was dropped because a valid 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 overruns and pointer mismatches.
SN_CELL_DROP (8)	A cell was dropped in accordance with the "Fast SN Algorithm" (as specified in ITU-T Recommendation I.363.1).
POINTER_RECEIVED (7)	A valid pointer was received.
PTR_PARITY_ERR (6)	A cell was received with a pointer parity error.
SRTS_RESUME (5)	An SRTS resume has occurred. A valid SRTS value was received and stored in the SRTS FIFO.

Field (Bits)	Description
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. This cell will be played out after 1 CDVT.
PTR_MISMATCH (2)	A cell was dropped because of a pointer mismatch. This event causes a forced underrun condition.
OVERRUN (1)	A cell was dropped due to overrun. The receive buffer exceeded the maximum allowed depth. This event causes a forced underrun condition.
UNDERRUN (0)	A cell was received while the queue was in underrun.

R_TOT_SIZE Word Format

NOTE: This word is maintained by the microprocessor.

Field (Bits)	Description
Not used (15:10)	Write with a 0 to maintain future software compatibility.
R_TOT_SIZE (9:0)	Total number of bytes in the structure, minus one (for example, for an E1 SDF-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 DS0s − 1 For T1 SDF-MF and E1_W_T1_SIG: $R\_TOT\_SIZE = (24 \times \text{no. of DS0s}) + \frac{(\text{no. of DS0s} + 1)}{2} - 1$ 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

NOTE: 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 bit table (refer to "R_CHAN_ALLOC(15:0) Word Format" and "R_CHAN_ALLOC(31:16) Word Format" on page 161). Not used in UDF-ML or UDF-HS modes.
Not used (7:6)	Write with a 0 to maintain future software compatibility.

Field (Bits)	Description
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 or UDF-HS mode. $R\_DATA\_LAST = \frac{(no.\ of\ DS0s + 1)}{2} - 1$

R_TOT_LEFT Word Format

NOTE: This word is read-only and is maintained by RALP.

Field (Bits)	Description
Not used (15:10)	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.

R_SN_CONFIG Word Format

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.
Not used (7)	Write with a 0 to maintain future software compatibility.
INSERT_DATA (6:5)	Controls the format of the data inserted for lost cells: 00_b Insert FF_h. 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, sequence number processing is disabled. Statistics will still be kept but no cells will be dropped due to SN errors.

Field (Bits)	Description
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. Refer to section 3.6 "Receive Adaptation Layer Processor (RALP)" starting on page 46 for more information. 0 When SN_STATE (refer to "SN_STATE" on page 155) equals "000_b", any received cell will be dropped. 1 When SN_STATE (refer to "SN_STATE" on page 155) equals "000_b", any received cell with valid SNP will be accepted.
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

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 and UDF-HS modes, initialize to $FFFF_h$. (DS0 15 is in bit 15).

R_CHAN_ALLOC(31:16) Word Format

Field (Bits)	Description
R_CHAN_ALLOC (15:0)	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 and UDF-HS modes, initialize to $FFFF_h$. (DS0 31 is in bit 15).

R_DROPPED_CELLS Word Format

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 (refer to "R_CHAN_ALLOC(15:0) Word Format" on page 161). • SN processing. • Structured cell received while in underrun but structure start has not been found yet.

R_UNDERRUNS Word Format

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 (refer to "R_LINE_STATE Word Format" on page 156). If this bit is set, then increment the underrun count by one to get the current count.

R_LOST_CELLS Word Format

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

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

Field (Bits)	Description
R_POINTER_REFRAMES (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

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

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.

7.8.12 R_OAM_QUEUE

Organization: 256 cells × 64 bytes
Base address: $0E000_h$
Index: 20_h
Type: Read/Write
Function: The receive OAM queue stores the OAM cells and non-data cells received from the UTOPIA interface. Refer to section 3.6, "Receive Adaptation Layer Processor (RALP)", on page 46 for the definitions of OAM and non-data cells.
Initialization: It is not necessary to initialize this structure.
Format: Two data bytes per word.

Offset	Name	Description
00000_h	R_OAM_CELL_0	Receive OAM cell 0
00020_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 5 (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)	

7.8.13 RESERVED (Receive Data Buffer)

NOTE: 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: 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: It is not necessary to initialize this structure.

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) × 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 × 8192 + multiframe(E1) × 256 + frame(E1) × 16 + (chan-1) ÷ 2. T1 offset = line × 8192 + multiframe(T1) × 512 + frame(T1) × 16 + (chan-1) ÷ 2.
R_DATA_L (7:0)	Receive data for: Channel = (offset mod 16) × 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 × 8192 + multiframe(E1) × 256 + frame(E1) × 16 + channel ÷ 2. T1 offset = line × 8192 + multiframe(T1) × 512 + frame(T1) × 16 + channel ÷ 2.

7.9 CMDREG (Command Register)

Organization: 1 register
Base address: 20000_h
Index: 1_h
Type: Read/Write
Hardware Reset Value: $xx21_h$
Function: Allows the microprocessor to signal events to the AAL1gator II.
Format: Refer to the following table.

CMDREG Word Format

Field (Bits)	Description
Not used (15:8)	Write with a 0 to maintain compatibility with future software versions.
(Reserved) PROC_TEST_ACCESS (7)	Initialize to 0. When set, destructively allows access to the internal FIFOs for testing purposes. The device should be reset with either the /RESET pin or the SW_RESET register after this operation is performed. For production test use only.
CLR_RX_OAM_LATCH (6)	When set, causes the receive OAM interrupt latch to be cleared. On read: 1 means an OAM interrupt is present. 0 means an OAM interrupt is not present.
SW_RESET (5)	When set, causes all of the device except the microprocessor interface to be held in reset. While set, the external SRAM may be accessed. When switching to or from UDF-HS mode, this bit must be asserted. Resets to 1.

Field (Bits)	Description
CSD_ATTN (4)	When written with a 1, causes the device to "OR" the T_ADD_QUEUE table with the first entry in the calendar queue, thereby starting up VCs. Resets to 0. Reads as 0 when the operation is complete. This operation normally takes up to 3 ms, but is blocked if the TALP is not allowed to build new cells. In the presence of severe backpressure at the transmit UTOPIA port, the CSD_ATTN bit may not clear.
CMD_REG_ATTN (3)	When written with a 1, causes the device to write the device revision code into the DEVICE_REV location, read the COMP_LIN_REG location, and finally read the eight LIN_STR_MODE locations. Reads as a 0 when the operation is complete. Resets to 0.
SEND_OAM_1 (2)	A write of 1 causes the cell in the TX OAM buffer 1 to be sent. Reads as a 0 when the cell has been sent. Resets to 0.
SEND_OAM_0 (1)	A write of 1 causes the cell in the TX OAM buffer 0 to be sent. Reads as a 0 when the cell has been sent. Resets to 0.
OAM_INT_MASK (0)	When set, disables the OAM receive interrupt. Resets to 1.

NOTES:

- SEND_OAM _0 has a higher priority than SEND_OAM_1. Therefore, if the correct order of OAM cell transmission is important, both bits should not be set at the same time. After SEND_OAM_0 is requested to be sent, the software can build the second cell in memory, but should not set the SEND_OAM_1 bit until it has detected that the SEND_OAM_0 bit has been cleared.
- Both attention bits (CSD_ATTN and CMD_REG_ATTN) cannot be set simultaneously.

Dual port RAM testing is enabled when the PROC_TEST_ACCESS bit is asserted. All processor accesses will then be directed to the internal RAMs. Address bits 6:0 are used to provide the address for the write port and bits 13:7 are used to provide the address for the read port. Bit 15 and 14 select the respective RAM based upon the following code.

Bits (15:14)	Description
00	Not used.
01	Signal data FIFO.
10	Receive FIFO.
11	Transmit FIFO.

7.10 Activating a New Line After Reset

The microprocessor can activate or deactivate a new line at any time without affecting lines already in service. The microprocessor should change the line and then set the line CMD_REG_ ATTN bit in the microprocessor attention register, informing the AAL1gator II that it must interrogate the memory for line status changes. After the requested change has been performed, the AAL1gator II clears the attention bit. The microprocessor must not request another change

until it recognizes the attention bit is no longer set. Failure to observe this requirement can result in loss of attention requests. All parameters must be set up before CMD_REG_ATTN is asserted.

7.11 Activating a New Queue on an Active Line

Before a new queue on an active line can be activated, all transmit queue table structures must be properly set up. Refer to section 7.6.8 “T_QUEUE_TBL” on page 134. The proper bit in the T_ADD_QUEUE table is then set, followed by an assertion of the CSD_ATTN bit.

NOTES:

- **Only** bits for new queues in the T_ADD_QUEUE table should be set when the CSD_ATTN bit is set. If bits are left set from previous add queue operations, the cell generation process will be corrupted.
- Before reactivating a previously active queue, be sure to reinitialize all the registers in the queue table for that queue.
- If backpressure on the transmit UTOPIA port is severe enough to prevent the TALP from building cells when they are scheduled, the CSD_ATTN bit will not clear until the TALP has a chance to build those cells. Under this condition, the CSD_ATTN bit may not clear until the backpressure stops. Note that a VC is not considered to be started until the CSD clears the CSD_ATTN bit.

In UDF-HS mode, the CSD_ATTN bit is not used. Simply being in UDF-HS mode and not in software reset and having the TX_ACTIVE bit set for queue 0 will activate the queue.

7.12 Making Changes to an Active Queue

Before reconfiguration, a queue should be deactivated by clearing the ACTIVE bit in the transmit queue table (refer to section 7.6.8 “T_QUEUE_TBL” on page 134). The queue will not be deactivated until the next entry in the calendar is encountered. The time for the next calendar entry to be encountered is:

$$(\text{FRAMES\_PER\_CELL} + 1) \times \text{the frame rate}$$

NOTES:

- The frame rate is typically 125 µs. Refer to “FRAMES_PER_CELL” on page 136.
- Before reactivating a previously active queue, be sure to reinitialize all the registers in the queue table for that queue.

8 APPLICATION NOTES

8.1 DS1 Application

Figure 86 shows a typical application of the AAL1gator II in a network. Data is received from the DS3 network and microprocessor and split into 28 DS1 lines. The AAL1gator II segments this data into ATM cells and transmits it to the UTOPIA. The ATM Routing Table (WAC-187-X) appends a 6-byte routing tag and transmits the data to an ATM switch fabric as 118 nibbles.

In the transmit to DS3 direction, the ATM Routing Table (WAC-187-X) strips the 6-byte routing tag and presents the data to the AAL1gator II in a 53-byte cell format. The AAL1gator II then translates the VPI/VCI information into the logical channel numbers for the T1 line and plays the data out at the appropriate time.

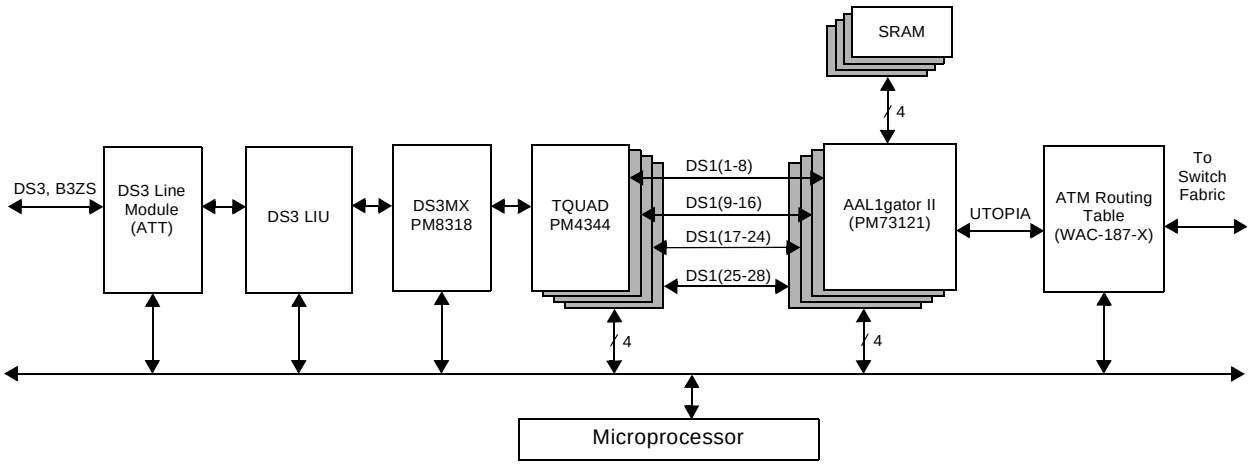


Figure 86. Typical DS1 Application

8.2 Interface Circuit with Typical Framer

The AAL1gator II can interface with a variety of T1 and E1 framers. A typical T1 Quad PMC interface is shown in Figure 87

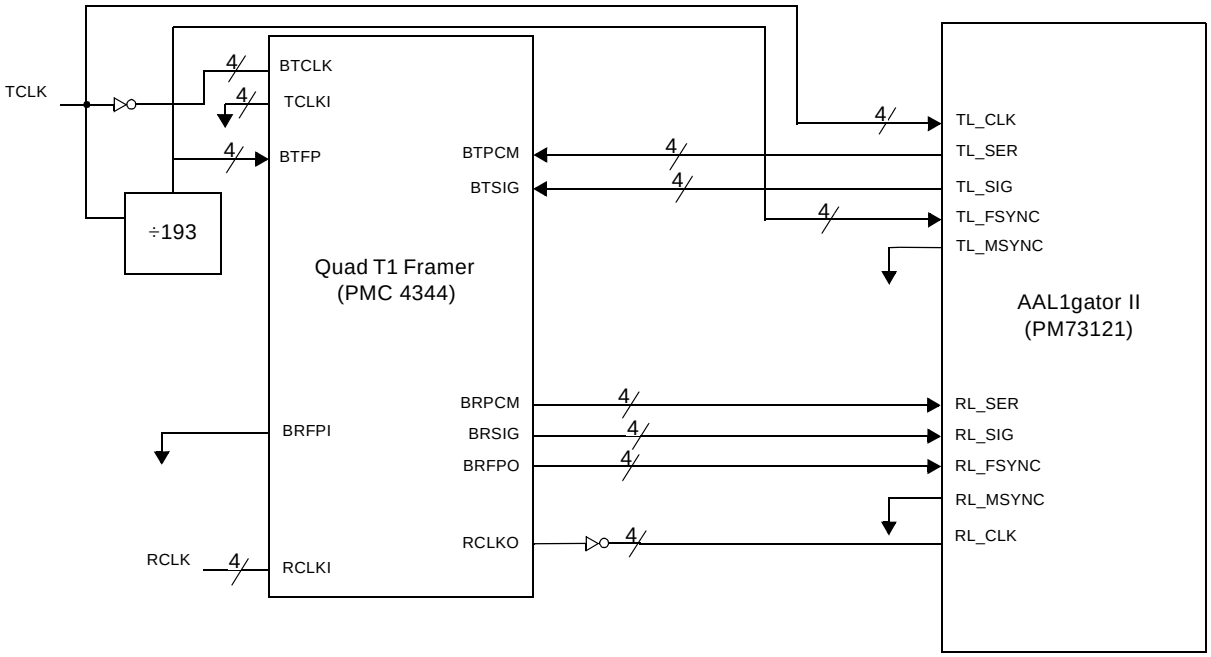


Figure 87. Typical PMC Quad Framer Interface

8.3 DS3 Application

For DS3 applications, the AAL1gator II can handle one DS3 or E3 line with one VCI in an unstructured data format. Figure 88 shows the AAL1gator II in a DS3 application.

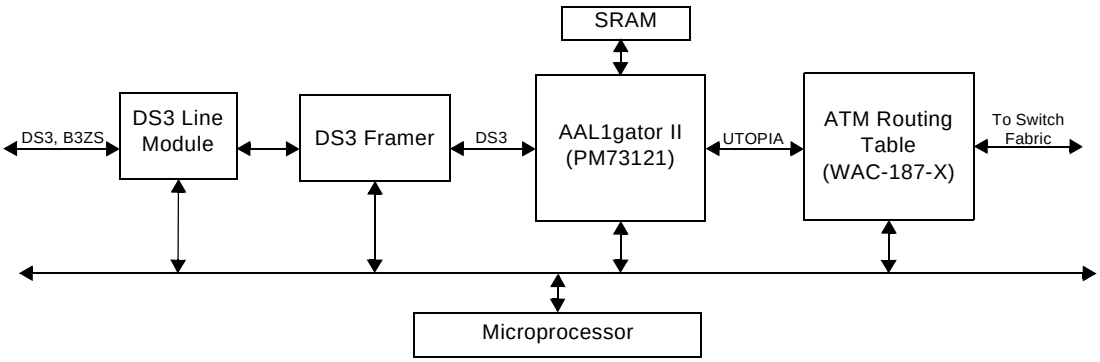
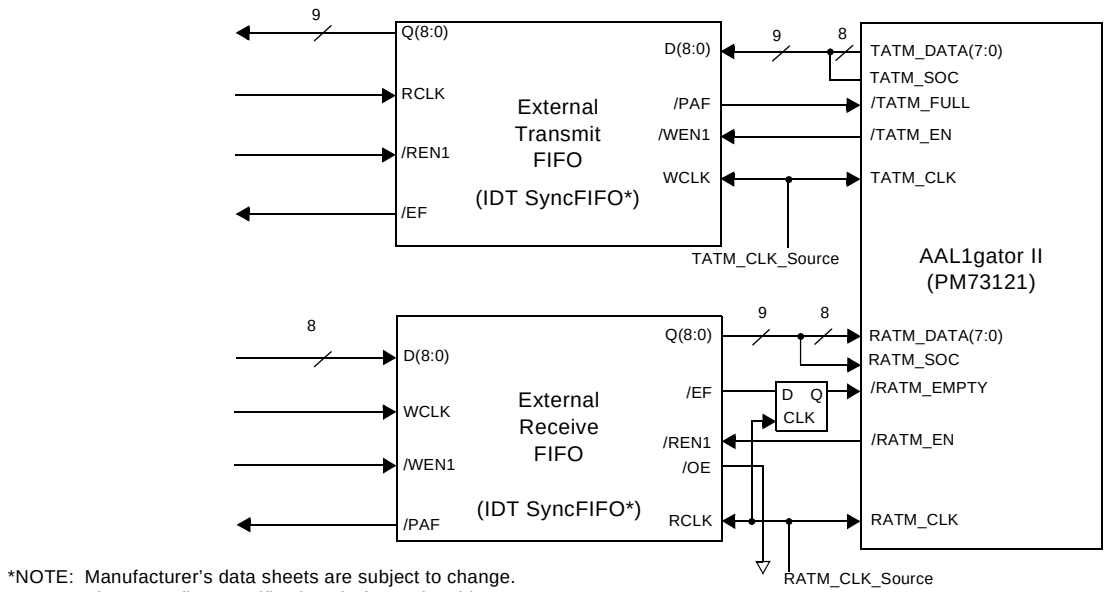


Figure 88. Typical DS3 Application

8.4 External FIFO Application

Figure 89 shows how to connect the AAL1gator II to two 9-bit Integrated Device Technology SyncFIFOs™. On the RATM interface, a D-type flip-flop is required on the /RATM_EMPTY signal since the SyncFIFO /EF signal does not conform to UTOPIA timing. On the TATM interface, the SyncFIFO must be able to accept four more data bytes after it asserts the /TATM_FULL signal. This is accomplished by using the SyncFIFO /PAF flag and its full-7 bytes default offset.



*NOTE: Manufacturer's data sheets are subject to change.
Please confirm specifications before using this part.

Figure 89. Typical External FIFO Application

8.5 External SRTS-Based Clock Recovery Application

While the AAL1gator II has built-in SRTS, to remain backward compatible with previous versions of the device, external based SRTS is still supported. External SRTS-based clock recovery can be accomplished with the circuitry shown in Figure 90 on page 173. The circuit produces a clock around the nominal T1 or E1 frequency derived from the network clock. This clock frequency is then increased or decreased based on the SRTS difference produced by the AAL1gator II.

The following requirements apply to the 38.88 MHz clock:

- If SRTS is being used, the clock generation logic and AAL1gator II must use the same 38.88 MHz clock. The SRTS information passes from the AAL1gator II to the clock generator logic across a synchronous boundary.
- If a nominal frequency clock is generated and the generated clock needs to be locked to a network clock, the clock generation logic must use a network-derived 38.88 MHz clock only.
- If a locally generated 38.88 MHz clock is used, the clock must be within the tolerance of the network clock requirements. For E1, this is 50 ppm (G.703) and for T1, it is 130 ppm (T1.403). Other specifications exist for different modes of operation. Refer to Appendix B, “References”, on page 203 for a list of references.

PMC-Sierra, Inc. has secured an agreement with Bellcore which allows PMC-Sierra to freely market and sell components that contain the SRTS function, such as the AAL1gator II product. However, the Bellcore-PMC-Sierra agreement does not provide any patent infringement protection to any users (that is, equipment manufacturers) of AAL1gator II products. Any equipment manufacturer which makes use of the SRTS function (by using an AAL1gator II product or via some other implementation) needs to determine if it may be necessary to establish a separate licensing agreement with Bellcore.

NOTE: As PMC-Sierra understands it, Bellcore will not license the SRTS patent to silicon manufacturers. Instead, it is Bellcore’s desire to license the SRTS patent under a royalty arrangement only to equipment manufacturers. The ATM Forum states that Bellcore must make this patent available under fair and equitable conditions. Bellcore believes they are satisfying this requirement by offering the license to the equipment manufacturers rather than to the silicon manufacturers.

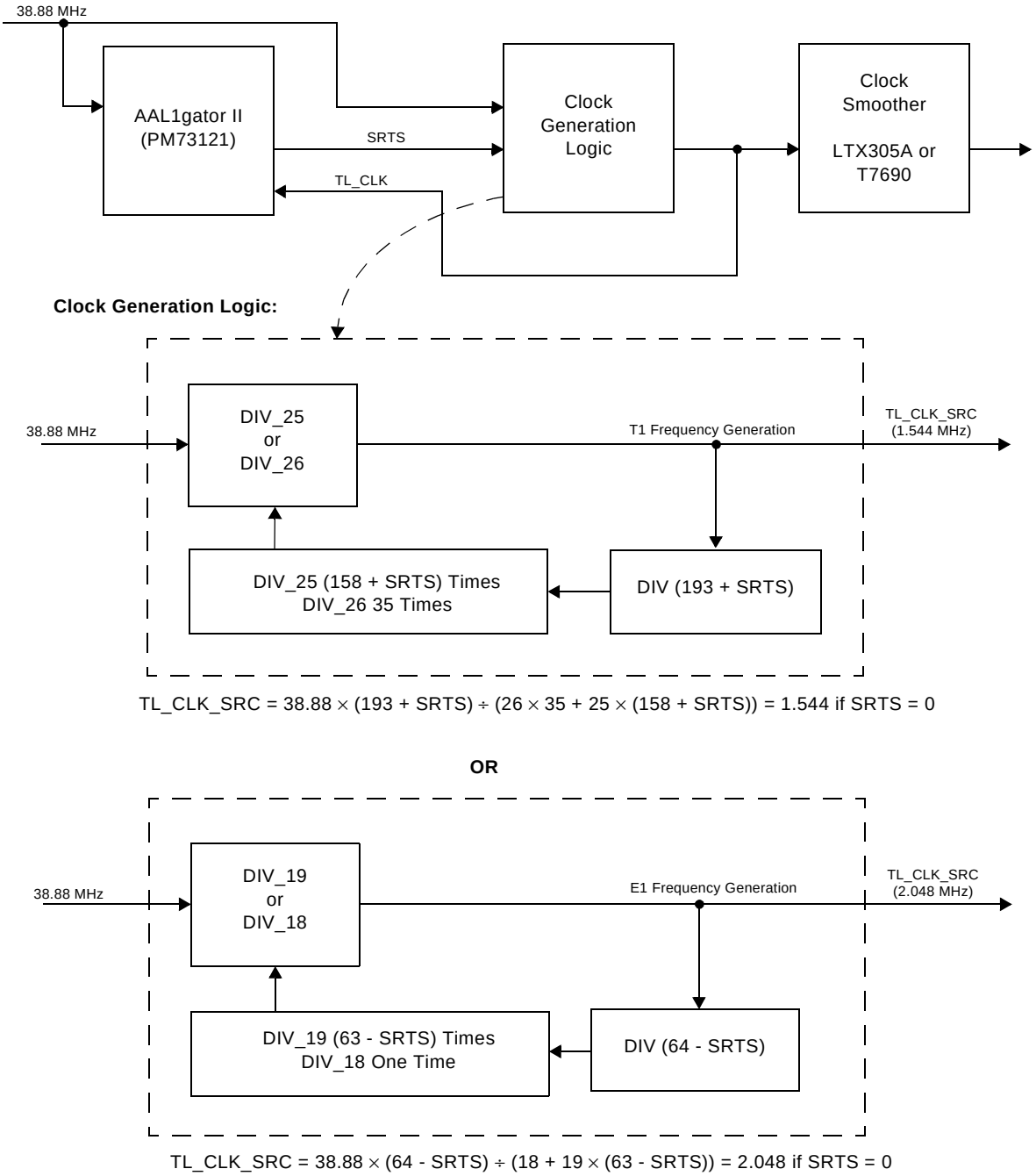
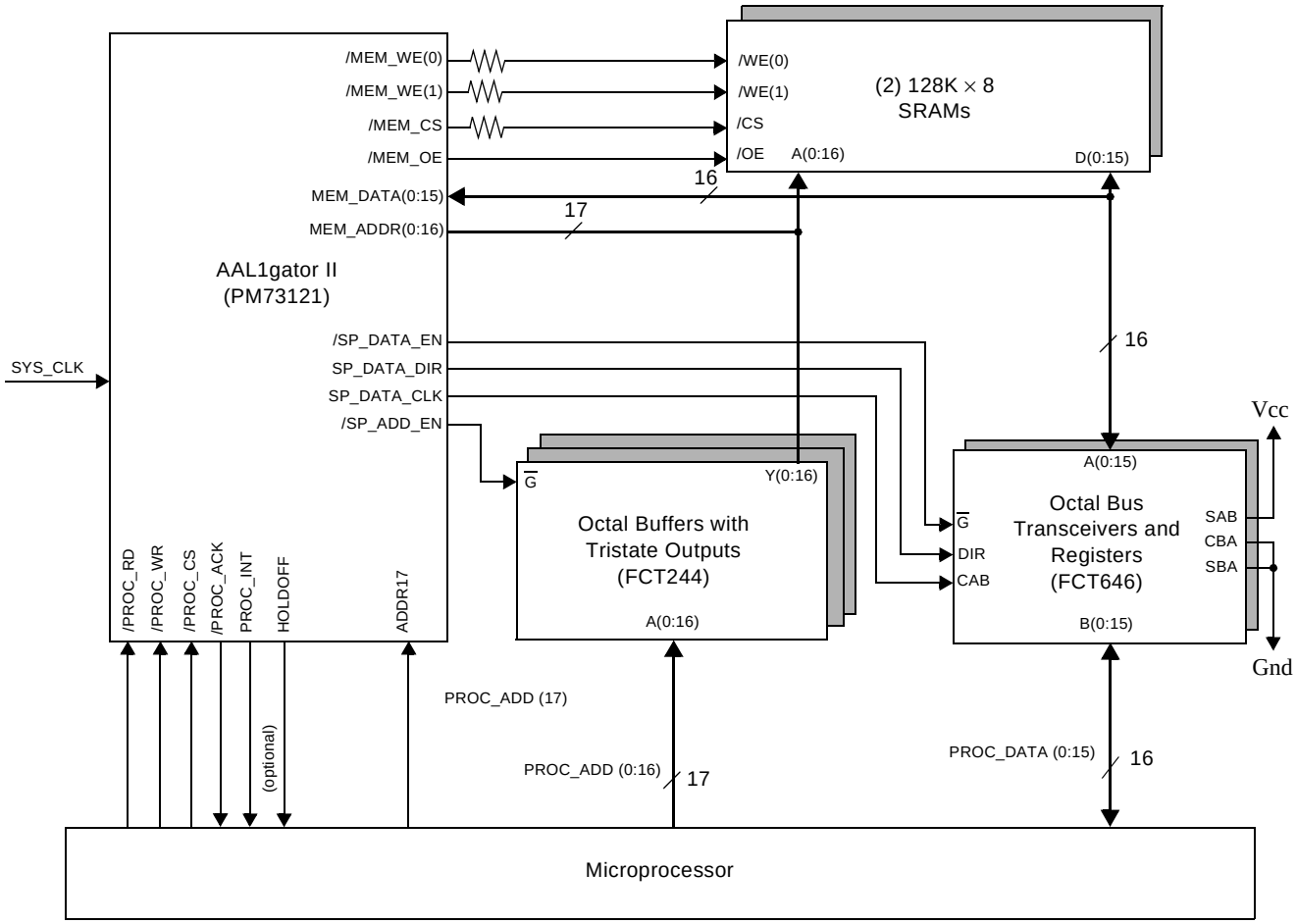


Figure 90. SRTS-Based Clock Recovery Circuitry

8.6 Board Requirements for the SRAM Interface

The AAL1gator II can use industry-standard asynchronous SRAMs for its external memory. The AAL1gator II provides control signals that control all data accesses to the SRAM. Figure 91 is an example of how the SRAM should be incorporated into the AAL1gator II system design.

Table 26 on page 195 lists the recommended worst-case parameters for each component. Parts must meet these requirements to run this interface at full speed.



NOTES: Series terminating resistors are 33 Ω to 100 Ω. See this application note for requirements for /MEM_WE resistors.
*Manufacturer's data sheets are subject to change. Please confirm specifications before using this part.

Figure 91. Suggested AAL1gator II Memory Interface

SRAMs must be 12 ns or faster and must have a write data setup of 7 ns or faster to operate at the maximum system clock rate.

The following section details how to ensure the SRAM interface will work over all commercial environmental conditions. Examples are also provided for a few different scenarios. The generic equations are provided so other scenarios can be evaluated.

Due to its tight requirements, the write data setup and write data hold are the most critical parameters for SRAM selection. Both of these parameters are derived from the high pulse width of the SYS_CLK input.

The setup and hold times of the SRAM are based on:

- The high time of SYS_CLK
- The series termination resistor value for /MEM_WE

Adding a series resistor will increase the rise/fall time due to an increase in the RC constant. Based on the slew rate output of the driver, the value of the resistor, and a 15 pF load, the delay at 1.5 V will be increased by the amount shown in Table 24.

Table 24. Delay Values for Different Resistors

Resistor (Ω)	Delay (ns)
33	0.3
50	0.4
75	0.6
100	0.7

Using a resistor value greater than 100 Ω is not recommended, since this will cause rise/fall times that are too slow and will not match the board impedance close enough, which can cause reflections.

The high pulse width of SYS_CLK, plus the delay through the resistor must ensure the rising edge of /MEM_WE is slow enough to provide sufficient setup time, but fast enough to provide enough hold time.

The generic equations for determining the SRAM setup and hold requirements are:

$$\text{SRAM write setup} = Tch_{min} - 4.3 + Rs$$
$$\text{SRAM write data hold} = Tp - Tch_{max} - Rs - 10$$

where: Tch is the high pulse width of SYS_CLK at 1.5 V,
Tp is the clock period, and
Rs is the delay through the resistor.

To determine Tch_{min} and Tch_{max}, the minimum and maximum duty cycle must be defined. Also if a CMOS driver is being used, the minimum and maximum rise and fall times of the driver must be defined.

Since high-speed SRAMS use TTL input thresholds, all output timing parameters are measured at 1.5 V. The SYS_CLK input, like all the AAL1gator II inputs, uses TTL input levels. Due to tight timing requirements on the minimum pulse width required for SYS_CLK, a CMOS driver can be used to generate SYS_CLK to improve the pulse width. The rise/fall time of the signal will cause the high pulse width measured at 1.5 V to be larger then the pulse width measured at 2.5 V.

The following sections provide four case examples:

- Using an SRAM with 7 ns write data setup and a TTL clock.
- Using an SRAM with 7 ns write data setup and a CMOS clock.
- Using an SRAM with 8 ns write data setup and a TTL clock.
- Using an SRAM with 8 ns write data setup and a CMOS clock.

8.6.1 SRAM with 7 ns Write Data Setup and a TTL Clock

Referring back to the setup and hold equations:

SRAM write setup = Tch_{min} - 4.3 + Rs
SRAM write data hold = Tp - Tch_{max} - Rs -10

Assuming we want to have a 0.5 ns margin on both setup and hold time, the requirements to meet are:
Tch_{min} - 4.3 + Rs ≥ 7.5 ns
Tp - Tch_{max} - Rs -10 ≥ 0.5 ns

A 5% duty cycle TTL clock source at 38.88 MHz has a Tch_{min} of 11.6 ns and a Tch_{max} of 14.1 ns at 1.5 V.

Replacing Tch_{min} and Tch_{max} in the equations above gives:
Rs ≥ 7.5 + 4.3 -11.6 ≥ 0.2 ns
Rs ≤ 25.7 - 14.1 - 10 - 0.5 ≤ 1.1 ns

Referring to Table 24 on page 175, selecting a 50 Ω resistor would meet both of these requirements.

In summary, one possible solution when using an SRAM with 7 ns setup is to use a 5% duty cycle TTL clock source and a 50 Ω series resistor. To increase the margin, the tolerance on the clock

could be tightened, or a CMOS clock driver could be used. The rise/fall time of the clock should be less than 1.5 ns.

8.6.2 SRAM with 7 ns Write Data Setup and a CMOS Clock

When using a CMOS clock, the rise and fall time of the clock needs to be taken into account to determine what the pulse width will be at a TTL level. The benefit of using a CMOS clock is that the pulse width will be wider at the TTL level then it will be at the CMOS level. Rise/fall times for CMOS outputs are usually given from the 20% to the 80% level or across a 3 V range. Dividing this number by 3 gives the approximate delay between the 1.5 V level and the 2.5 V level. Since the delay occurs at both rising and falling edges, this value is doubled to give the increase in setup time. The hold time is usually the critical factor with a CMOS clock driver.

For instance, if the rise/fall time is 3 ns, there will be a 1 ns delay from a transition at 1.5 V to the transition at 2.5 V and, a 2 ns increase in setup time.

This time we will have 1 ns margin on setup and 0.5 ns margin on hold. The requirements to meet are:

Tch_{min} - 4.3 + Rs ≥ 8.0 ns

Tp - Tch_{max} - Rs -10 ≥ 0.5 ns

A 2.5% duty cycle CMOS clock source at 38.88 MHz has a Tch_{min} of 12.2 ns and a Tch_{max} of 13.5 ns at 2.5 V.

A rise/fall time of 0.5 ns (from 20-80%) results in a gain of 0.33 ns at 1.5 V and would change Tch_{min} to 12.53 ns.

A rise/fall time of 2 ns (from 20-80%) results in a gain of 1.33 ns at 1.5 V and would change Tch_{max} to 14.83 ns.

Replacing Tch_{min} and Tch_{max} in the equations above gives:

Rs ≥ 8.0 + 4.3 -12.53 ≥ - 0.23 ns

Rs ≤ 25.7 - 14.83 - 10 - 0.5 ≤ 0.37 ns

Referring to Table 24 on page 175 shows that selecting a 33 Ω resistor would meet both requirements. Note that the negative resistance value in the first equation indicates additional margin. In this situation, the hold time is more critical and is dependent on the clock duty cycle. Selecting a 5% clock would create a hold time problem unless either the required margin was reduced or the maximum rise time was reduced.

In summary, one possible solution when using an SRAM with 7 ns setup is to use a 2.5% duty cycle CMOS clock (with a minimum rise/fall time of 0.5 ns and a maximum rise/fall time of 2 ns) and a 33 Ω series resistor.

8.6.3 SRAM with 8 ns Write Data Setup and a TTL Clock

When using a TTL clock with an SRAM with a 8 ns setup requirement, the clock duty cycle requirement is very tight. One way to tighten up the duty cycle of a clock is to derive the clock from a higher frequency. Since each edge will be evenly spaced, the only way to affect symmetry is to adjust the difference in the rise and fall time at 1.5 V.

Assuming we want to have a 0.5 ns margin on both setup and hold time, the requirements to meet are:
Tch_{min} - 4.3 + Rs ≥ 8.5 ns
Tp - Tch_{max} - Rs -10 ≥ 0.5 ns

A 1% duty cycle TTL clock source at 38.88 MHz has a Tch_{min} of 12.6 ns and a Tch_{max} of 13.1 ns.

Replacing Tch_{min} and Tch_{max} in the equations above gives:
Rs ≥ 8.5 + 4.3 -12.6 ≥ 0.2 ns
Rs ≤ 25.7 - 13.1 - 10 - 0.5 ≤ 2.1 ns

Referring to Table 24 on page 175 shows that selecting a 50 Ω resistor would meet both requirements.

In summary, one possible solution when using an SRAM with 8 ns setup is to use a 1% duty cycle TTL clock source and a 50 Ω series resistor. To increase the margin, a higher value resistor or a CMOS clock driver could be used. A resistor value greater than 100 Ω should not be used since this will cause rise/fall times that are too slow and will not match the board impedance close enough. The rise/fall time of the clock should be less than 1.5 ns.

8.6.4 SRAM with 8 ns Write Data Setup and a CMOS Clock

When using a CMOS clock, the rise and fall time of the clock needs to be considered to determine the pulse width at a TTL level. The benefit of using a CMOS clock is that the pulse width will be wider at the TTL level than at the CMOS level. Rise/fall times for CMOS outputs are usually given from the 20% to the 80% level, or across a 3 V range. Dividing this number by 3 gives the approximate delay between the 1.5 V level and the 2.5 V level. Since the delay occurs at both rising and falling edges, this value is doubled to give the increase in setup time. Usually the critical factor with a CMOS clock driver is the hold time.

For example, if the rise/fall time is 3 ns, there will be a 1 ns delay from a transition at 1.5 V to the transition at 2.5 V, and a 2 ns increase in setup time.

Assuming we want a 0.5 ns margin on both setup and hold time, the requirements to meet are:
Tch_{min} - 4.3 + Rs ≥ 8.5 ns

Tp - Tch_{max} - Rs -10 ≥ 0.5 ns

A 2.5% duty cycle CMOS clock source at 38.88 MHz has a Tch_{min} of 12.2 ns and a Tch_{max} of 13.5 ns at 2.5 V.
A rise/fall time of 0.5 ns (from 20-80%) results in a gain of 0.33 ns at 1.5 V and would change Tch_{min} to 12.53 ns.
A rise/fall time of 2 ns (from 20-80%) results in a gain of 1.33 ns at 1.5 V and would change Tch_{max} to 14.83 ns.

Replacing Tch_{min} and Tch_{max} in the equations above gives:
Rs ≥ 8.5 + 4.3 -12.53 ≥ 0.27 ns
Rs ≤ 25.7 - 14.83 - 10 - 0.5 ≤ 0.37 ns

Referring to Table 24 on page 175, selecting a 33 Ω resistor would meet both of these requirements.

In summary, a possible solution when using an SRAM with 8 ns setup is to use a 2.5% duty cycle CMOS clock (with a minimum rise/fall time of 0.5 ns and a maximum rise/fall time of 2 ns) and a 33 Ω series resistor. To improve margin, either the duty cycle needs to be tightened or the maximum rise time needs to be reduced, which would allow a larger resistor.

8.6.5 Layout

The delay information provided assumes a 15 pF load on the /MEM_WE signals that have one load and a 30 pf load on the MEM_DATA outputs that have two loads. The trace capacitance is approximately 2 pF per inch and the input pin capacitance is about 7 pF. Therefore, the SRAM trace lengths should be approximately four inches long.

Also, the series resistor should be placed as close as possible to the source pin.

8.7 UDF-HS Mode SRTS-Based Clock Recovery Application for DS3

While the AAL1gator II has built-in SRTS clock recovery for T1/E1 rates, external circuitry is required for DS3/E3 rates in UDF-HS mode. SRTS-based clock recovery can be accomplished with the circuitry shown in Figure 92. The circuit generates a clock at the 44.736 MHz DS3 frequency.

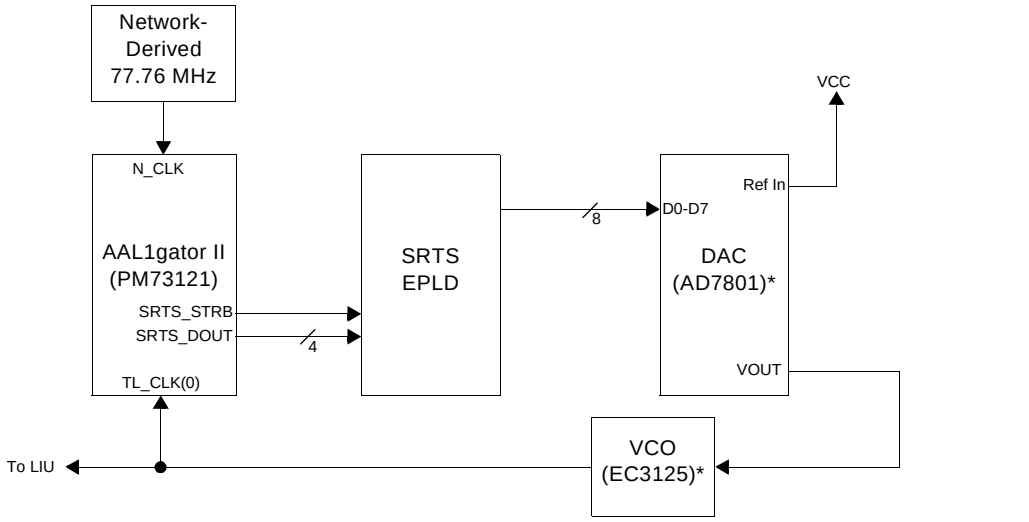


Figure 92. SRTS-Based Clock Recovery Circuit

This circuit functions as follows. The AAL1gator II asserts SRTS_STRB indicating a new SRTS nibble is available on SRTS_DOUT. In UDF-HS mode, the SRTS_STRB and SRTS_DOUT signals are driven from the rising edge of N_CLK, thus SRTS_STRB should be used to clock the SRTS_DOUT nibble in D-type flip flops. Once latched, a lookup table (refer to Table 25) is used to convert the SRTS_DOUT nibble into an 8-bit code to drive an Analog Devices' AD7801 8-bit Digital-to-Analog Converter (DAC). The DAC output voltage then controls the ECLIPTEK® EC3125 Voltage Controlled Oscillator (VCO), which has a 44.736 MHz center frequency. The resulting DS3 clock rate is then fed to the AAL1gator II TL_CLK(0) input and the LIU. The N_CLK must be network derived, but the SYS_CLK does not need to be network derived.

Table 25. Memory Interface System Clock Operating Conditions

SRTS_DOUT (Binary)	8-Bit DAC Code (Hex)
0111	E7
0110	CD
0101	C0

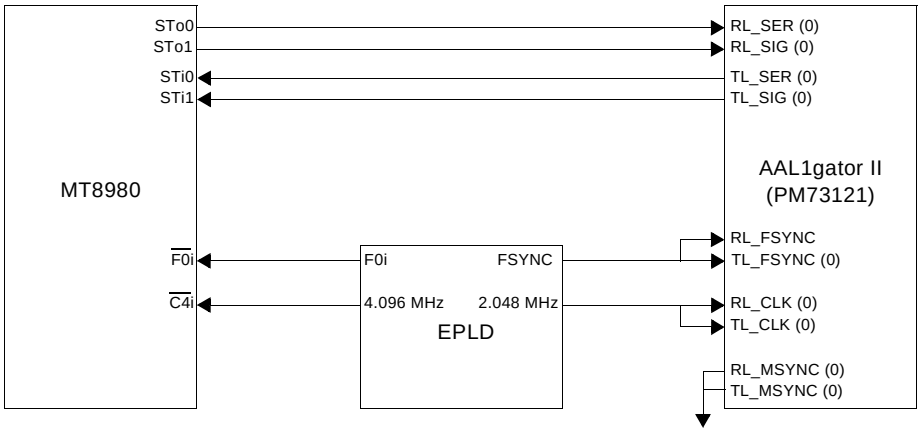
Table 25. Memory Interface System Clock Operating Conditions (Continued)

SRTS_DOUT (Binary)	8-Bit DAC Code (Hex)
0100	B3
0011	A6
0010	9A
0001	8D
0000	80
1111	73
1110	66
1101	5A
1100	4D
1011	40
1010	33
1001	26
1000	19
XXXX	80

Typically, SRTS nibbles delivered by the AAL1gator II oscillate between two values, such as 1100_b and 1011_b. To smooth out the resulting clock jitter, a more sophisticated mapping from SRTS nibbles to 8-bit DAC codes can be used. Some examples of such mappings are averages and running averages that access a lookup table with finer resolution than that provided by the SRTS nibbles.

8.8 Interfacing with the Mitel MT8980 Digital Switch

Figure 93 shows the AAL1gator II interface to the MITEL® MT8980 Digital Switch and Figure 94 shows the required relationship among the timing signals that should be generated in the EPLD.



*NOTE: Manufacturer's data sheets are subject to change.
Please confirm specifications before using this part.

Figure 93. Interfacing with the Mitel MT8980

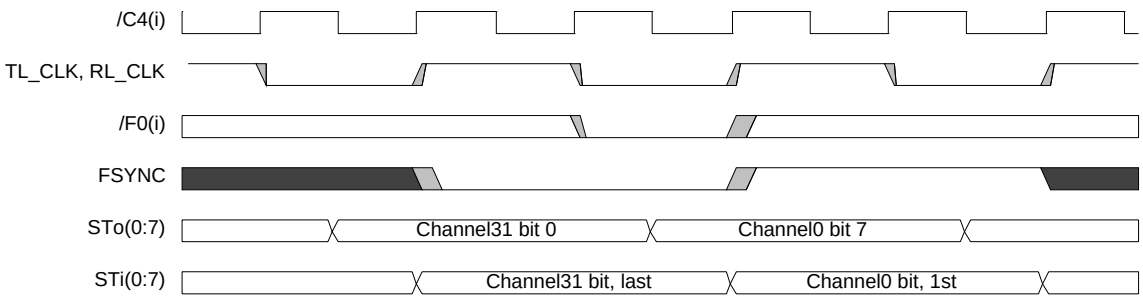


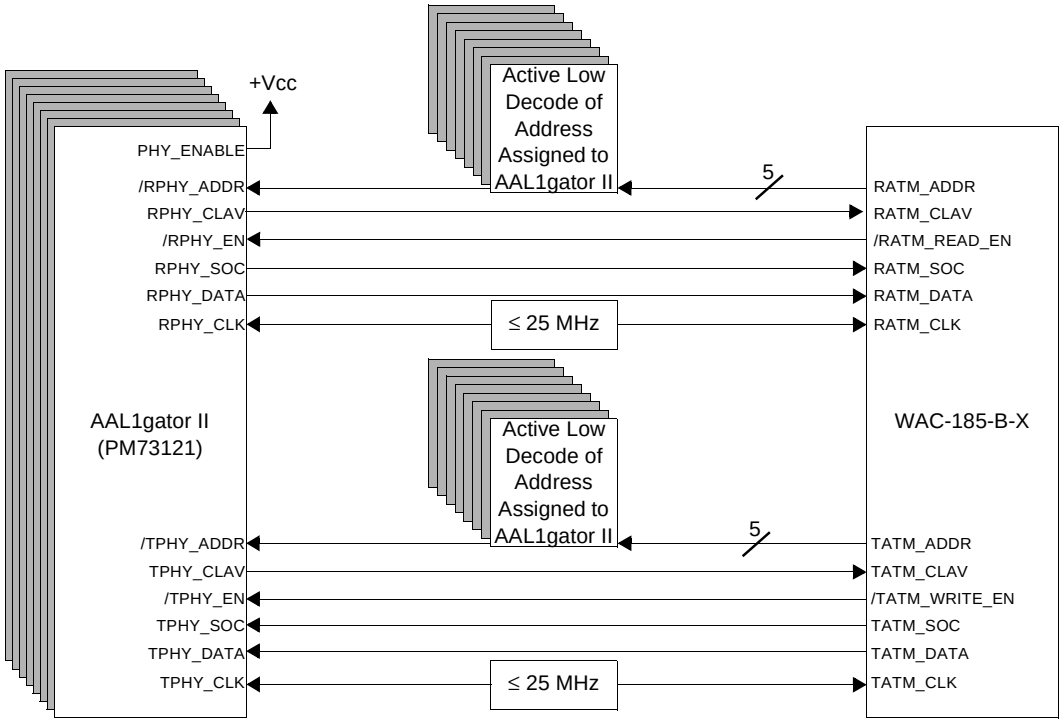
Figure 94. Interfacing Timing with the Mitel MT8980

8.9 Interfacing with the ATM Cell Multiplexer (WAC-185-B-X)

Figure 95 shows the interface for connecting eight AAL1gator IIs (as PHY layer devices) to the ATM Cell Multiplexer (WAC-185-B-X), an ATM layer device. Configure the AAL1gator IIs in Multi-PHY UTOPIA mode by connecting the PHY_ENABLE input pin to V_{CC}, and setting the SPHY_EN bit to "0" in the COMP_LIN_REG memory word (refer to section 7.4.2 "COMP_LIN_REG" on page 125). Configure the WAC-185-B-X in Multi-PHY UTOPIA mode by setting the ATM_MPHY_EN bit to "1" in the GEN_5 microprocessor port. In this mode, the WAC-185-B-X will drive the eight addresses x00, x01, x02, x03, x04, x05, x06, and x07, and the null address x1F onto the RATM_ADDR/TATM_ADDR busses. The decoding logic shown in Figure 95 should decode each of these addresses to an active low enable signal for each of the AAL1gator IIs.

If only four, or fewer, AAL1gator IIs are to be connected to the WAC-185-B-X, the external decode logic can be eliminated by implementing a single-PHY UTOPIA bus instead of a Multi-PHY UTOPIA bus. In this case, configure the AAL1gator IIs in single-PHY UTOPIA mode by connecting the PHY_ENABLE input pin to V_{CC} and setting the SPHY_EN bit to "1" in the COMP_LIN_REG memory word. Configure the WAC-185-B-X in single-PHY UTOPIA mode by setting the ATM_MPHY_EN bit to "0" and the ATM_CELL_EN bit to "1" in the GEN_5 microprocessor port. (The ATM_CELL_EN is set because the AAL1gator II implements cell-level handshaking due to its cell-based internal FIFOs.) In this mode the WAC-185-B-X does

not drive addresses, so the RATM_ADDR/TATM_ADDR should be left unconnected, and each AAL1gator II is essentially connected to its own single-PHY UTOPIA bus.



*NOTE: Manufacturer's data sheets are subject to change.
Please confirm specifications before using this part.

Figure 95. Connecting Eight AAL1gator IIs to a WAC-185-B-X

8.10 Jitter Characteristics of Clock Synthesis Logic

This section shows the results of jitter analysis of the clock synthesis circuitry built into the AAL1gator II. Jitter was measured using a FIREBERD 6000A with the E1 or T1 jitter analyzer option and the jitter spectrum analysis option added. The clock synthesis logic is intended to be used with an external jitter attenuator. Results are shown with and without the jitter attenuator in a Level One LXT305A LIU enabled.

8.10.1 Nominal T1 Clock

The nominal T1 clock jitter was measured using a 2^{20}-1 Pseudorandom Bit Sequence (PRBS) pattern in unstructured mode. The total maximum jitter without the jitter attenuator was 0.40 Unit Interval (UI). With a jitter attenuator the maximum jitter was 0.08 UI. See Figure 96 and Figure 97 for the jitter spectrum versus the G.824 mask.

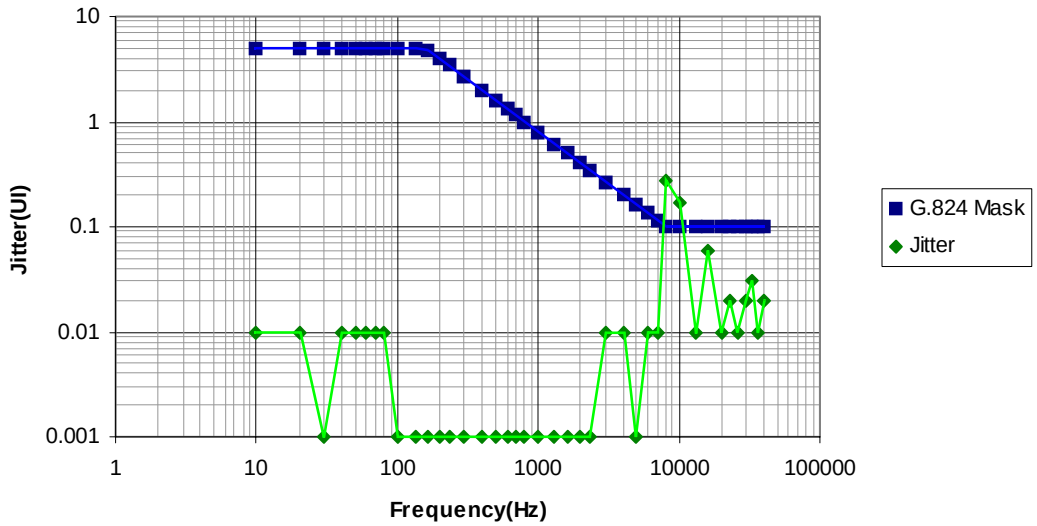


Figure 96. Nominal T1 Clock with no Jitter Attenuator

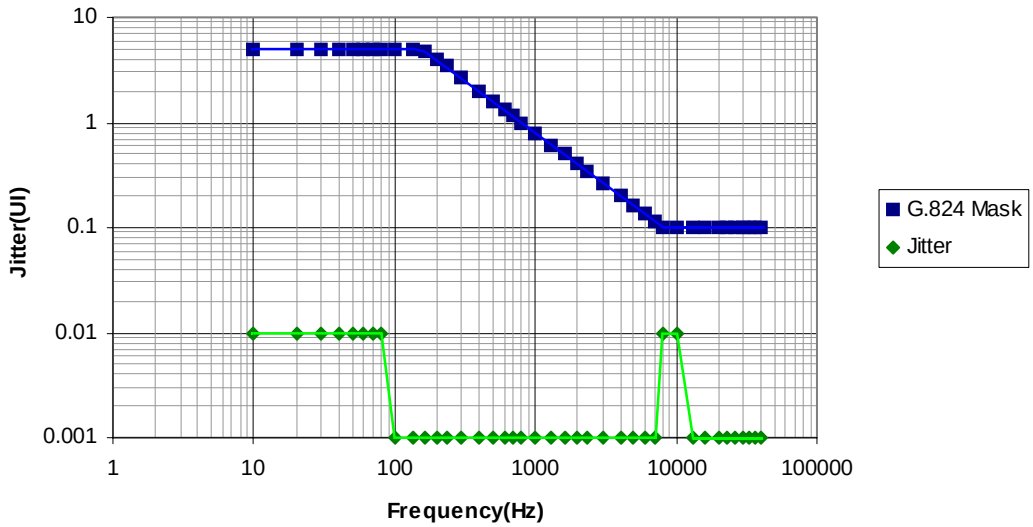


Figure 97. Nominal T1 Clock with Jitter Attenuator

8.10.2 Nominal E1 Clock

The nominal E1 clock jitter was measured using a 2^{15}-1 PRBS pattern in unstructured mode. The total maximum jitter without the jitter attenuator was 0.11 UI. With a jitter attenuator the maximum jitter was 0.065 UI. See Figure 98 and Figure 99 for the jitter spectrum versus the G.823 mask.

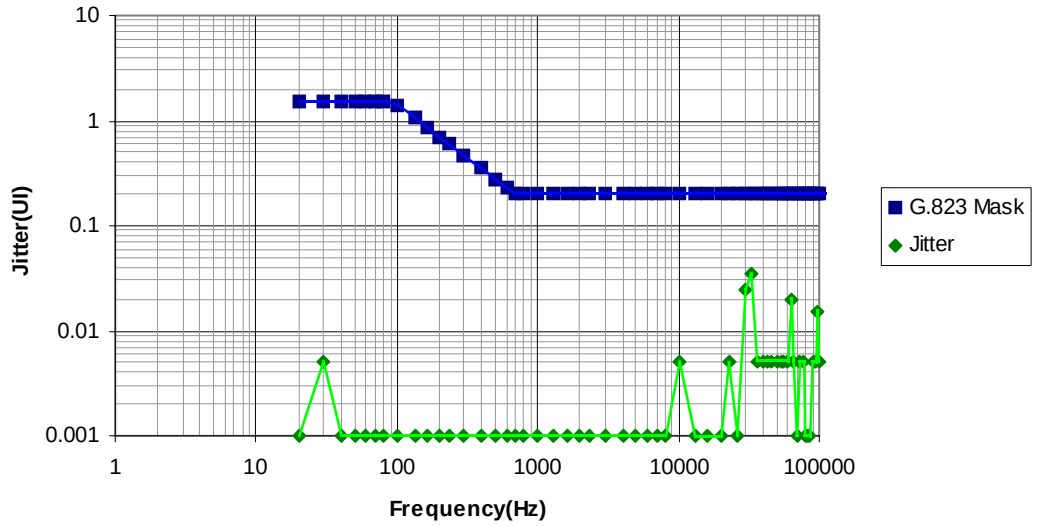


Figure 98. Nominal E1 Clock with no Jitter Attenuator

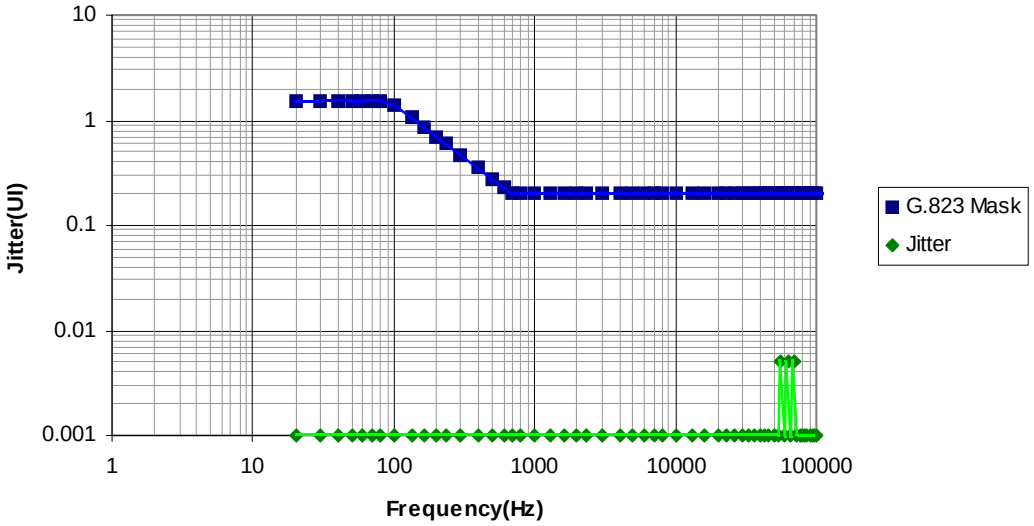


Figure 99. Nominal E1 Clock with Jitter Attenuator

8.10.3 SRTS T1 Clock

The T1 SRTS clock jitter was measured using a 2^{20}-1 PRBS pattern in unstructured mode.

Two AAL1gator IIs were connected to each other across a point to point ATM connection. One AAL1gator II was used to generate SRTS values, and the other AAL1gator II was used to receive the values. The source AAL1gator II was connected to a FIREBERD, which used an external signal generator as its clock source. The clock source was stepped across the entire ± 200 ppm range and the jitter was measured at the FIREBERD connected to the AAL1gator II at the receiving end. A jitter spectrum was then taken at the frequency that exhibited the worst jitter.

The total maximum jitter without the jitter attenuator was 1.91 UI. With a jitter attenuator the maximum jitter was 0.26 UI, other than a spike at - 200 ppm. See Figure 100 through Figure 103 on page 190 for jitter versus frequency and the jitter spectrum versus the G.824 mask.

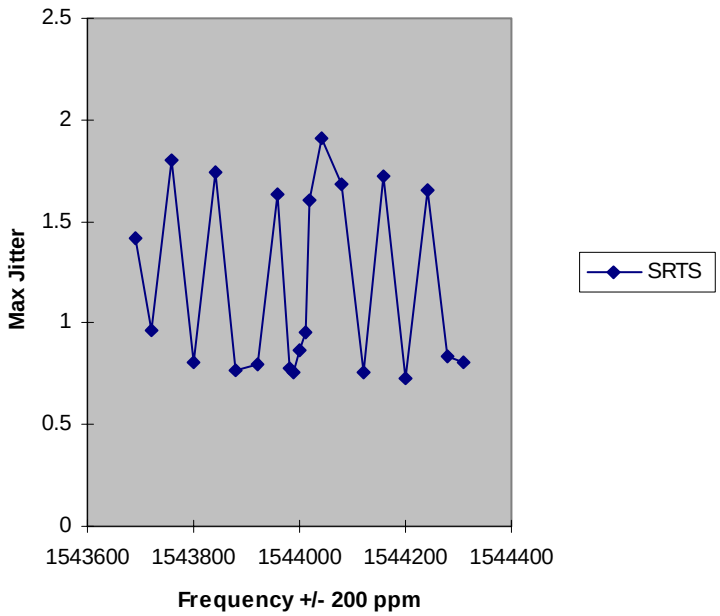


Figure 100. Maximum T1 SRTS Jitter with no Jitter Attenuator

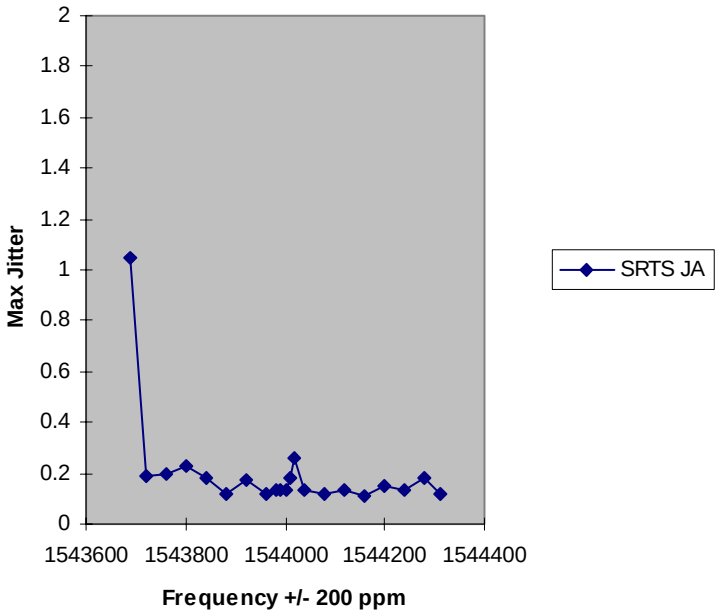


Figure 101. Maximum T1 SRTS Jitter with Jitter Attenuator

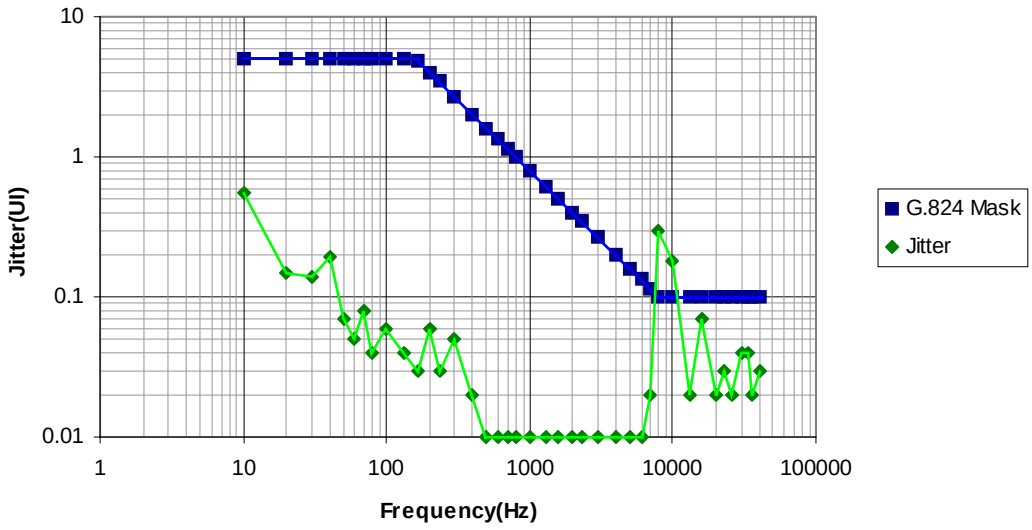


Figure 102. T1 SRTS Clock with no Jitter Attenuator

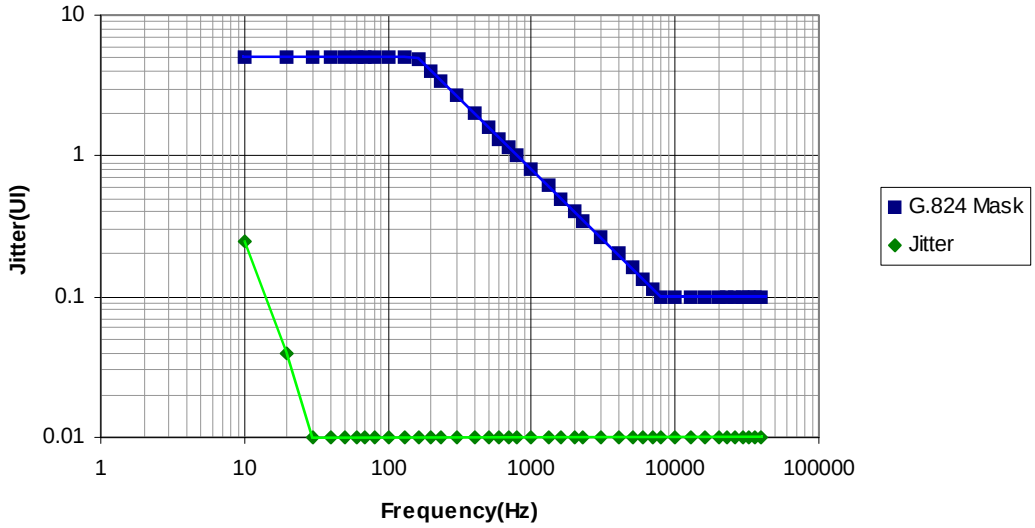


Figure 103. T1 SRTS Clock with Jitter Attenuator

8.10.4 SRTS E1 Clock

The E1 SRTS clock jitter was measured using a 2^{15}-1 PRBS pattern in unstructured mode.

Two AAL1gator IIs were connected to each other across a point-to-point ATM connection. One AAL1gator II was used to generate SRTS values, and the other AAL1gator II was used to receive the values. The source AAL1gator II was connected to a FIREBERD, which used an external signal generator as its clock source. The clock source was stepped across the entire ± 90 ppm range and the jitter was measured at the FIREBERD connected to the AAL1gator II at the receiving end. A jitter spectrum was then taken at the frequency that exhibited the worst jitter.

The total maximum jitter without the jitter attenuator was 0.845 UI. With a jitter attenuator the maximum jitter was 0.42 UI. See Figure 104 through Figure 107 on page 193 for jitter versus frequency and the jitter spectrum versus the G.824 mask.

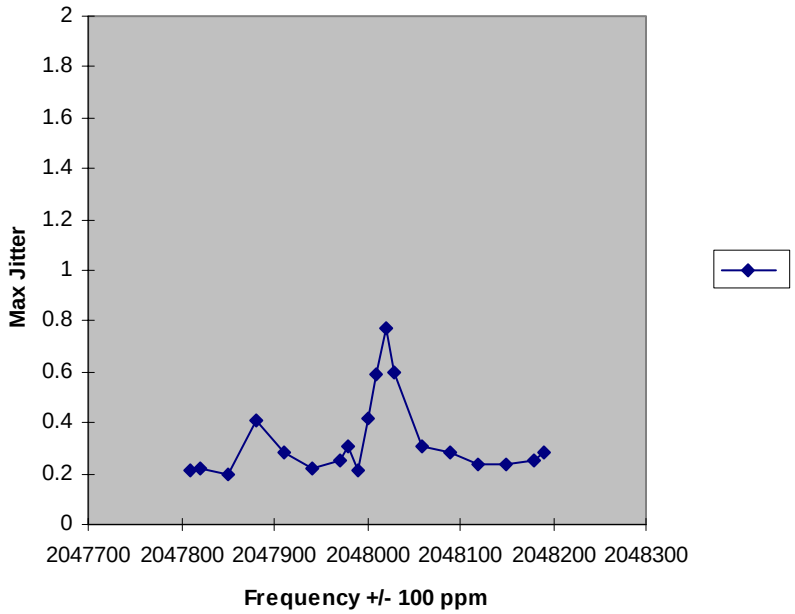


Figure 104. Maximum E1 SRTS Jitter with no Jitter Attenuator

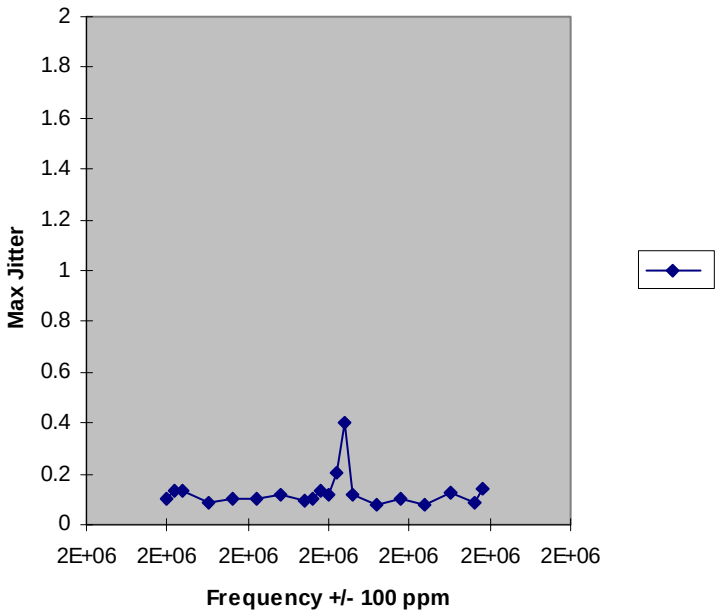


Figure 105. Maximum E1 SRTS Jitter with Jitter Attenuator

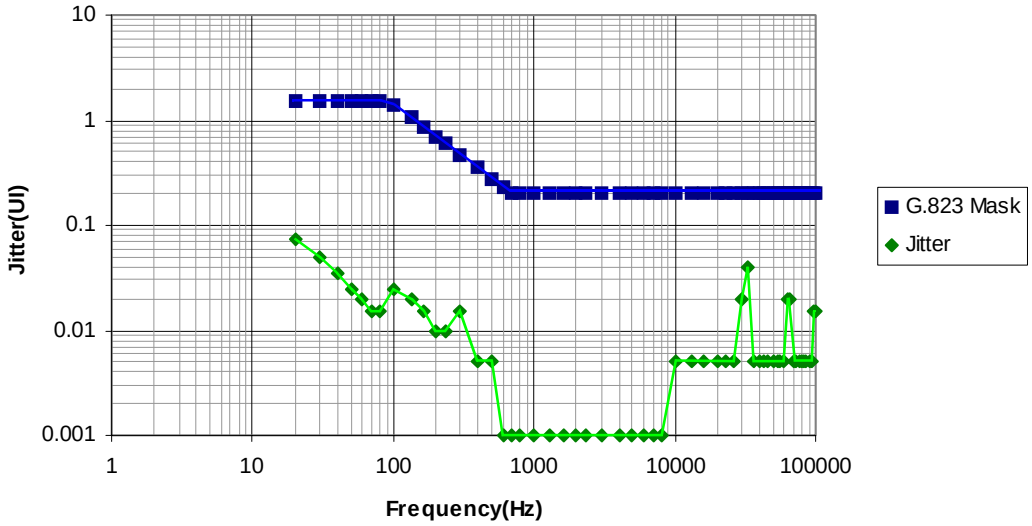


Figure 106. E1 SRTS Clock with no Jitter Attenuator

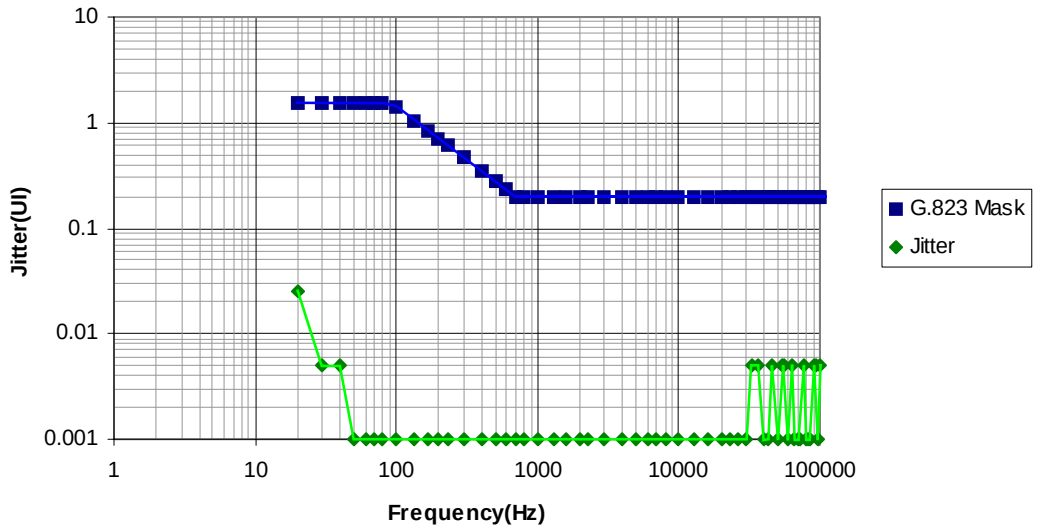


Figure 107. E1 SRTS Clock with Jitter Attenuator

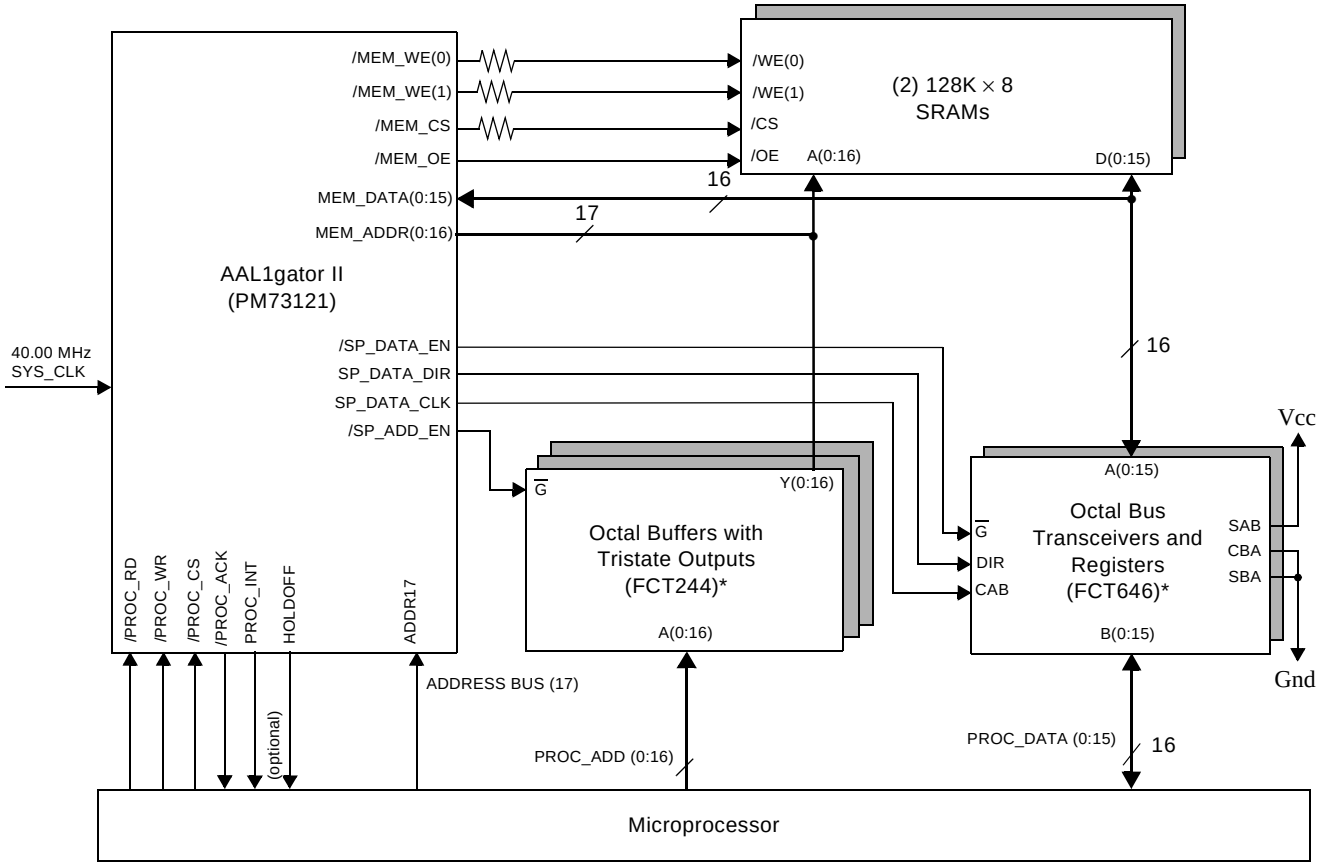
8.11 Timing Requirements On External Logic for RAM and Microprocessor Interface

Microprocessor accesses are controlled by the AAL1gator II. To properly access the AAL1gator II and the external memory, the AAL1gator II must be supported with a tristatable address buffer and a bidirectional, tristatable data latch.

The RAM and microprocessor interface has been optimized to work with a pair of 128K x 16 SRAMs, a pair of FCT646s, and a few FCT244s, as shown in Figure 108 on page 195. Due to the asynchronous timing of these parts, the speed of the interface, and the interleaving of different types of read and write operations, the address, data, and control signals are skewed with respect to each other to provide the delicate balance that is required.

It is strongly recommended that the architecture shown in Figure 108 on page 195 is followed, especially if this interface is running near maximum speed. The components should also be placed near each other. If slower speeds or faster parts are used, alternative architectures can be used, as long as all timing parameters are met. Note that the capacitance for the timings given assumes the architecture shown. Other configurations may have different loadings. **Refer to section 8.6 "Board Requirements for the SRAM Interface" on page 174 for important timing information regarding this interface.**

If using the architecture shown in Figure 108 at maximum frequency (40.00 MHz), the recommended worst-case parameters for each part are listed in Table 26 on page 195. At lower speeds these numbers can be adjusted.



NOTES: Series terminating resistors are 33Ω to 100Ω. See application notes for guidelines regarding /MEM_WE resistor.
* Manufacturer's data sheets are subject to change. Please confirm specifications before using this part.

Figure 108. Suggested AAL1gator II Memory Interface

Table 26. Recommended Worst-Case Parameters for Suggested Memory Interface

Number	Parameter	Min	Max	Unit
1	maximum tristatable address buffer output enable time (FCT244)	1	8	ns
2	maximum tristatable address buffer output disable time (FCT244)	1	8	ns
3	maximum tristatable address buffer propagation delay (FCT244)	1	8	ns
4	maximum bidirectional tristatable data latch output enable time (FCT646)		10	ns
5	maximum bidirectional tristatable data latch output disable time (FCT646)		10	ns

Table 26. Recommended Worst-Case Parameters for Suggested Memory Interface (Continued)

Number	Parameter	Min	Max	Unit
6	minimum bidirectional tristatable data latch data setup to clock (FCT646)		5	ns
7	minimum bidirectional tristatable data latch data hold to clock (FCT646)		3	ns
8	maximum bidirectional tristatable data latch propagation delay (FCT646)		10	ns
9	maximum RAM access delay from address stable and chip select active		15	ns
10	minimum RAM data hold from address change	2		ns
10	minimum RAM data hold from /OE or /CS high	0		ns
11	maximum RAM output enable delay		7	ns
12	minimum RAM data setup to write clock		7	ns
13	minimum RAM data hold to write clock		0	ns
14	minimum RAM address hold to write clock		0	ns
15	minimum RAM write enable minimum pulse width		10	ns
16	minimum RAM address setup to write start		1	ns
17	minimum RAM address setup to write end		12	ns
18	maximum RAM output disable delay		6	ns

The numbers in Table 26 are used in Figure 109 to Figure 112 on page 197 to show the suggested memory interface timing relationships.

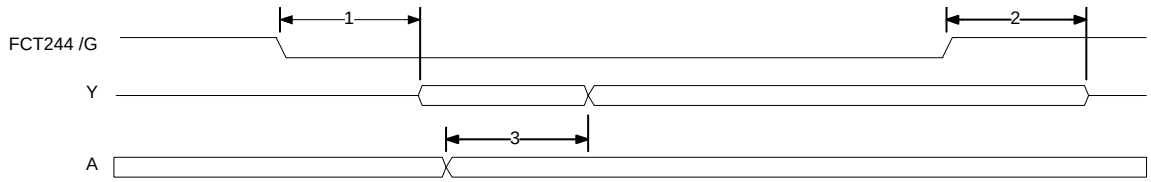


Figure 109. Address Buffer (FCT244) Timing

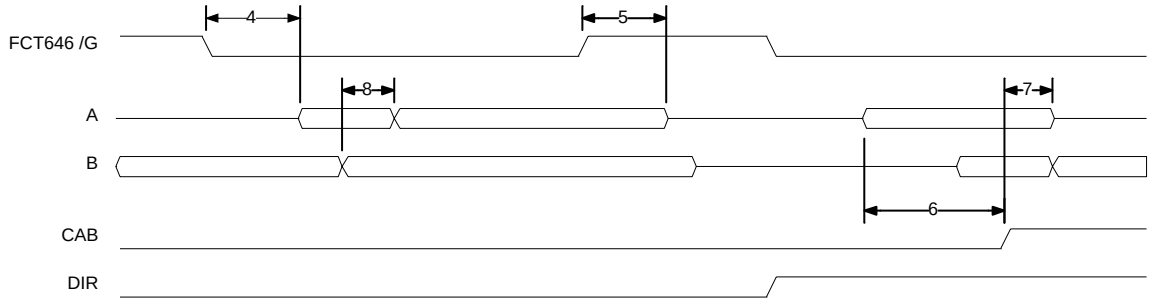


Figure 110. Bidirectional Data Latch (FCT646) Timing

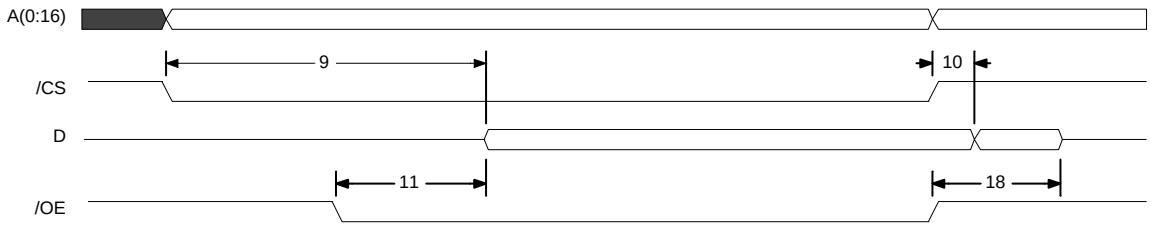


Figure 111. RAM Read Timing

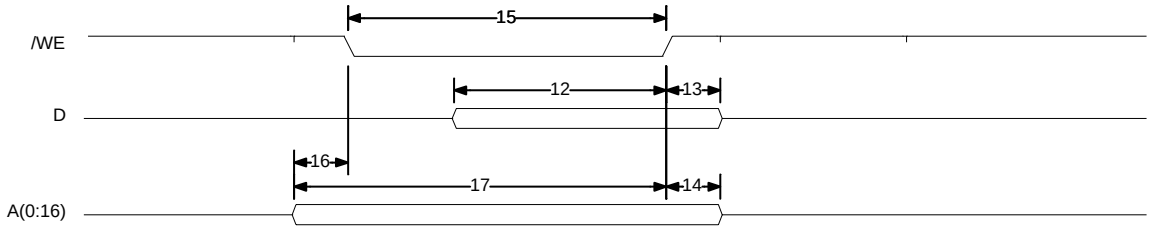


Figure 112. RAM Write Timing

APPENDIX A NOMENCLATURE

A.1 Definitions

Transmit Signals: all signals related to processing the data heading towards the optical/electrical layer.
Receive Signals: all signals related to processing the data heading towards the ATM layer.

A.2 Signal Name Prefixes

All pins and alarm names in the AAL1gator II have prefixes indicating the functional layer they implement (refer to Table 27).

Table 27. Prefixes and Associated Functional Layers

Pin Name Prefix	Associated Functional Layer
TATM	Transmit UTOPIA ATM Layer. These signals should be connected to the PHY layer.
RATM	Receive UTOPIA ATM Layer. These signals should be connected to the PHY layer.
TPHY	Transmit UTOPIA ATM Layer. These signals should be connected to the ATM layer.
RPHY	Receive UTOPIA ATM Layer. These signals should be connected to the ATM layer.

A.3 Numbers

Hexadecimal numbers are followed by the suffix "$_h$" and binary numbers are followed by the suffix "$_b$". For example: 101_b or $2C_h$. Decimal numbers appear without a suffix.

A.4 Glossary of Abbreviation

Table 28. Standard Abbreviations

Abbreviation	Description
AAL1	ATM Adaptation Layer 1
ACS	ATM Circuit Steering
AMON	ATM Monitoring
AMS	Audio/Visual Multimedia Service
ANSI	American National Standards Institute
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 Service
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
E4	European Digital Signal Level 4
ESF	Extended Super Frame

Table 28. Standard Abbreviations (Continued)

Abbreviation	Description
FCT	Fairchild TTL Compatible
FIFO	First-In, First-Out
FPGA	Field Programmable Gate Array
FXO	Foreign Exchange Office
FXS	Foreign Exchange Subscriber
GFC	Generic Flow Control
HEC	Header Error Check
LI	Line Interface
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
MPEG2	Motion Picture Experts Group 2
MPHY	Multi-PHY
MSB	Most Significant Bit
MULDEM	Multiplexer/Demultiplexer
OAM	Operations, Administration, and Maintenance
OC-1	Optical Carrier Level 1
OC-3	Optical Carrier Level 3
PBX	Private Branch Exchange
PCR	Peak Cell Rate
PDU	Protocol Data Unit
PDH	Plesiochronous Digital Hierarchy
PHY	Physical
PLL	Phase-Locked Loop
PCM	Pulse Coded Modulation

Table 28. Standard Abbreviations (Continued)

Abbreviation	Description
PRBS	Pseudorandom Bit Sequence
PTI	Payload Type Indicator
RALP	Receive Adaptation Layer Processor
RATM	Receive UTOPIA ATM Layer
RFTC	Receive Frame Transfer Controller
RL	Receive Line
RMON	Remote Monitoring
RUTOPIA	Receive UTOPIA
RX	Receive
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
SONET	Synchronous Optical Network
SP	Supervisory Processor
SPHY	Single PHY
SRAM	Static Random Access Memory
SRTS	Synchronous Residual Time Stamp
TALP	Transmit Adaptation Layer Processor
TATM	Transmit UTOPIA ATM Layer
TDM	Time Division Multiplexing
TFTC	Transmit Frame Transfer Controller
TL	Transmit Line
TLIP	Transmit Line Interface Processor

Table 28. Standard Abbreviations (Continued)

Abbreviation	Description
TTL	Transistor-to-Transistor Logic
TUTOPIA	Transmit UTOPIA
TX	Transmit
UDF	Unstructured Data Format
UDF-HS	Unstructured Data Format, High Speed
UDF-ML	Unstructured Data Format, Multiple Line
UI	Unit Intervals
UNI	User Network Interface
UPC	Usage Parameter Control
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

APPENDIX B REFERENCES

- ANSI T1 Recommendation T1.403, *Network-to-Customer Installation - DS1 Metallic Interface*, NY, NY, 1995.
- ANSI T1 Recommendation T1.630, *Broadband ISDN-ATM Adaption Layer for Constant Bit Rate Services, Functionality and Specification*, NY, NY, 1993.
- ATM Forum, *ATM User Network Interface (UNI) Specification*, V 3.1, Foster City, CA USA, September 1994.
- ATM Forum, *Circuit Emulation Service - Interoperability Specification (CES-IS)*, V. 2.0, Foster City, CA USA, August 1996.
- ATM Forum, *UTOPIA, an ATM-PHY Layer Specification*, Level 1, V. 2.01, Foster City, CA USA, March 1994.
- ATM Forum, *UTOPIA, an ATM-PHY Layer Specification*, Level 2, V. 1.0, Foster City, CA USA, June 1995.
- *E3 Framer (EAC-030) User's Manual*.
- ITU-T Recommendation G.703, *Physical/Electrical Characteristics of Hierarchical Digital Interfaces*, April 1991.
- ITU-T Recommendation I.363.1, *B-ISDN ATM Adaptation Layer (AAL) Specification*, July 1995.

NOTES

ORDERING INFORMATION

Ordering Information

Table 29 lists the ordering information.

Table 29. Ordering Information

Part Number	Description
PM73121-RI	240-pin Quad Flat Pack

CONTACT INFORMATION

Contacting PMC-Sierra, Inc.

PMC-Sierra, Inc.
105-8555 Baxter Place
Burnaby B.C. Canada V5A 4V7

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

Document Information:	document@pmc-sierra.com
Corporate Information:	info@pmc-sierra.com
Application Information:	apps@pmc-sierra.com (604) 415-4533
Web Site:	http://www.pmc-sierra.com

PMC-980620 (R3) Issue date: January 1999