10/2000
Rev. 2.1

Addendum to

MPC860 PowerQUICC™ User's Manual
MPC860P, MPC850SAR and MPC855T AAL2 Microcode Specification

This supplement to the *MPC860 PowerQUICC™ User's Manual* describes the features of a microcode package that implements the ATM adaptation layer type 2 (AAL2) on the MPC860P, the MPC850SAR, and the MPC855T. The *MPC860 PowerQUICC™ User's Manual*, the *MPC8xx ATM Supplement to the MPC860/MPC850 PowerQUICC ™ User's Manual*, and the *MPC860P Supplement to the MPC860 PowerQUICC™ User's Manual* should be consulted for any features not described in this document.

To locate any published errata or updates for this document, refer to the website at http://www.mot.com/netcomm.

1.1 AAL2 Features and Restrictions

Features of the AAL2 microcode are as follows:

- Implements the ITU I363.2 specification
- Runs on the communications processor (CP) of the MPC860P, the MPC850SAR, and the MPC855T
- Supports up to 224 AAL2 channels. Channel numbers 32–255 are available for AAL2. Channels 0-31 and channel numbers greater than 255 can be used for non-AAL2 channels.

freescale™ semiconductor

- The microcode is executed internally from dual-port RAM.
- Built-in mechanism for Timer CU support without host intervention
- After the host initializes the AAL0 BDs and buffers associated with the AAL2 channels, the AAL2 microcode handles the interface with AAL0 operation.
- Optional masking of the APC overrun (APCO) event
- Optional enabling of RISC timers when using the timer CU mechanism
- Reassembly:
  - Sequence number (SN) and parity check for each CPS-Packet
  - Length indication (LI) and CRC5 check for each cell
  - OSF of the STF check (valid value is less than 48)
- Segmentation:
  - Perform CPS-PDU padding as needed
  - Sequence number (SN) and parity generation for each cell
  - CRC5 generation for the CPS-PH

The following restrictions apply when the AAL2 microcode is used:

- Serial ATM is not available on SCC4. (UTOPIA can still be used.)
- AAL0 buffers should start on a 64-byte-aligned address.
- DSP functionality is not available.
- No other microcode RAM packages or patches can be used.

1.2 Overview of the AAL2 Microcode

The AAL2 microcode package implements the ITU I.363.2 specification. The AAL2 microcode builds upon AAL0 operation (AAL0 microcode in ROM) and existing ATM data structures, such as connection tables. For each AAL2 channel, the AAL2 microcode converts data from AAL2 packets into AAL0 cells and vice versa. After initialization, the AAL0 operation is transparent to the AAL2 host application.

Table 1-1 lists acronyms and abbreviations used in this document.

Table 1-1. Acronyms and Abbreviated Terms

Term	Meaning
8xxSAR	Represents the MPC860P, MPC850SAR or MPC855T
AAL2 channel	The logical connection through which ATM communicates. Each channel corresponds to a specific VPI/VCI combination. The channel can also be called a VCC (virtual channel connection).
AAL2_CT	AAL2 connection table. Each active AAL2 channel has a connection table consisting of an AAL2_RCT and AAL2_TCT.
AAL2_RCT	AAL2 receive connection table (one RCT for each AAL2 channel)
AAL2_Rx_Queue	Queue of RPDs (functionally equivalent to an RxBD table)
AAL2_TCT	AAL2 transmit connection table (one TCT for each AAL2 channel)
AAL2_Tx_Queue	Queue of TPDs (functionally equivalent to a TxBD table)
AAL2_TxWait_table	The wait table used to implement the Timer CU mechanism
AAL2_TxWait_PTR	The wait pointer marking expired active buffers for the Timer CU mechanism
Active BD	AAL0 BD that points to the active buffer

Table 1-1. Acronyms and Abbreviated Terms (Continued)

Term	Meaning
Active buffer	The next AAL0 buffer to be processed (or the buffer currently being processed) by the AAL2 microcode. The active buffer may be empty or partially filled.
APC	ATM pace control for scheduling transmission of cells
CPS-Packet	Common parts sublayer packet format for AAL2 data as defined in the ITU I.363.2 specification. A packet consists of a packet header (PH) and a packet payload (PP).
DPR	Dual-port internal RAM of the 8xxSAR. The AAL2 microcode is located here.
Host	The user's AAL2 application running on the PowerPC core of 8xxSAR
µs	Microseconds
ms	Milliseconds
OSF	Offset field
Partially filled cell	An ATM cell (AAL0 buffer) which is not completely filled with CPS-Packets
RPD	Receive packet descriptor. Describes the CPS-Packet for receiving (similar to an RxBD)
Rx	Receive (Data direction is toward the 8xxSAR.)
STF	Start field
Timer CU	The timer (combined use) mechanism to limit the time allowed for filling partially filled cells
TPD	Transmit packet descriptor. Describes the CPS-Packet for transmitting (similar to a TxBD)
Tx	Transmit (Data direction is away from the 8xxSAR.)

1.2.1 AAL2 Transmit Overview

The host generates CPS-Packets with each packet consisting of a CPS-PH (CPS packet header) and CPS-PP (CPS packet payload). The HEC field in the CPS-PH is generated by the AAL2 microcode.

For each CPS-Packet to be sent, the host prepares a TPD and adds it to the AAL2_Tx_Queue of the AAL2 channel. Similar to a TxBD, a TPD contains control information and the pointer to the CPS-Packet.

For each AAL2 channel, the AAL2 microcode prepares AAL0 buffers for transmission. Each AAL0 buffer consists of a 4-byte header and 48-byte payload. Ready AAL0 buffers packed with AAL2 channels are transmitted (along with other ATM channels) by the 8xxSAR according to the APC scheduling. Note that the ATM host command APC BYPASS can also be used for direct scheduling of AAL2 channels.

For each AAL2 channel, the host should set the I bit in the associated AAL0 TxBDs so that after each cell transmission is completed, the 8xxSAR attempts to generate an interrupt to the host. However, the interrupt does not reach the host because trap hardware intercepts the interrupt at the beginning of the service routine. This is the activation mechanism for the AAL2 microcode to attempt to pack ready CPS-Packets from the channel's AAL2_Tx_Queue into an available AAL0 buffer. Note that the first time the AAL2 microcode is activated (and whenever the host or APC unit attempts to transmit a cell when the buffer is not ready), an idle cell is sent.

For each active AAL0 buffer, the AAL2 microcode transmitter does the following:

- Copies the 4-byte cell header from the channel's AAL2_TCT
- Generates the STF field of the CPS-PDU
- Fills the payload (CPS-PDU) with CPS-Packets

- If the AAL2 microcode packs the whole CPS-Packet into the AAL0 buffer, it marks the TPD as available (R=0). If TPD[INT] is set and the interrupt is not masked by AAL2_TCT[TPI], the AAL2 microcode adds an entry to the exception queue (with the AAL2 bit and TXB bit set) to signal the host that this CPS-Packet has been processed. The global interrupt count (INT_CNT) is also decremented. If INT_CNT reaches zero, the global interrupt (GINT) bit in the event register is set and an interrupt is generated to the host. (The INT_CNT is then restored to the INT_ICNT value.) If the exception queue is already full, the microcode sets the IQOV bit in the event register—entries in the exception queue are not overwritten. The host may then service the exception queue and process the appropriate AAL2_Tx_Queues by filling available CPS-Packets and marking their TPDs as ready (R=1).
- If the AAL2 microcode cannot pack the whole CPS-Packet into the active buffer (because the AAL0 buffer became full), the current CPS-Packet parameters are stored in the AAL2_TCT. The next time the AAL2 microcode is activated to service this channel (to fill a new active AAL0 buffer), it resumes operation from where it stopped.
- Generates the HEC field for each CPS-PH
- If the active buffer is filled with CPS-Packets, its TxBD[R] is set so that the AAL0 microcode will process it for actual transmission.

Note that the host application does not interface directly with the AAL0 buffers. The host interfaces with the AAL2 microcode, which in turn interfaces with the AAL0 transfer mechanism of the 8xxSAR.

AAL2 microcode provides a built-in Timer CU mechanism which makes sure that CPS-Packets in partially filled AAL0 buffers are not delayed by the AAL2 microcode more than a fixed time. The host can program the maximum delay for each channel in the AAL2_TCT.

If there are not enough ready CPS-Packets in the AAL2_Tx_Queue of this channel to fill the (partially filled) active buffer, the AAL2 microcode temporarily stores the buffer's parameters in the AAL2_TCT. If the Timer CU mechanism is enabled for this channel, the Timer CU count down for this AAL2 channel is also started. If the Timer CU counter expires before the active buffer becomes full, the AAL2 microcode activates in order to pad the buffer with zeros and mark its BD as ready. This buffer (along with other ready AAL0 buffers of the channel) is then transmitted in a separate process by the 8xxSAR according to the APC scheduling or host scheduling (using APC BYPASS).

If the AAL2 microcode is activated to process a channel's partially filled active buffer before its Timer CU counter expired, the active buffer is filled with CPS-Packets, the active BD is closed (R bit is set) and the channel's Timer CU counter is deactivated. However, if the active buffer is still not full, the Timer CU counter continues its count down.

1.2.2 AAL2 Receive Overview

The 8xxSAR receives AAL0 cells and places them in the external memory. Because all of the RxBDs associated with AAL2 channels have the I bit set (like the TxBDs), the 8xxSAR attempts to generate an interrupt to the host. However, trap hardware intercepts the interrupt at the beginning of the service routine and activates the AAL2 microcode.

For each activation, the AAL2 microcode receiver processes one active AAL0 buffer. The AAL2 microcode does the following:

- Checks the STF field of the CPS-PDU
- Breaks the CPS-PDU into CPS-Packets
- Verifies the HEC for each CPS-PH
- Copies each CPS-Packet to external memory according to the pointer in the current RPD of the channel's AAL2_Rx_Queue

- Checks the CPS-PDU for errors. If an error is detected, an indication is written to the associated RPD and the CPS-Packet is discarded. See Section 1.5.1, “Receive Packet Descriptor (RPD),” for a description of the possible errors.
- If a complete CPS-Packet is retrieved from the active buffer, the AAL2 microcode marks the RPD as available to the host (E = 0). If RPD[INT] is set and the interrupt is not masked by AAL2_RCT[RPI], the AAL2 microcode adds an entry to the exception queue (with the AAL2 bit and RXB bit set) to signal the host that a CPS-Packet has been received. The global interrupt count (INT_CNT) is also decremented. If INT_CNT reaches zero, the global interrupt (GINT) bit in the event register is set and an interrupt is generated to the host. (The INT_CNT is then restored to the INT_ICNT value.) If the exception queue is already full, the microcode sets the IQOV bit in the event register—entries in the exception queue are not overwritten. The host may then service the exception queue and process the appropriate AAL2_Rx_Queues and mark their RPDs as empty (E=1).
- If only a partial CPS-Packet is retrieved from the AAL0 buffer, the AAL2 microcode stores the CPS-Packet parameters in the channel’s AAL2_RCT. When the AAL2 microcode is next activated to serve this channel (after a new cell has been placed in an AAL0 buffer), it resumes operation from where it stopped.
- If the complete AAL0 buffer is processed, its BD is marked empty. If, however, the AAL2 microcode could not finish processing the AAL0 buffer (because there are not enough available RPDs in the AAL2_Rx_Queue), the AAL2 microcode discards the CPS-PDU and adds an entry to the exception queue (with the AAL2 bit and OVF bit set) to indicate an AAL2_Rx_Queue overflow.

If the 8xxSAR attempts to receive a cell, but there is no available AAL0 buffer (the next RxBD of the channel is not empty), the 8xxSAR generates an exception entry with a BSY indication. Because this indication is directly related to AAL0 operation, the AAL2 bit in the exception entry is not set. This type of exception indicates that the AAL2 microcode is too slow to process the AAL0 buffers that were received previously by the 8xxSAR. The system load could be so great that the AAL2 microcode cannot maintain the data rate. Also, the channel’s RxBD table could be too small; that is, the latency of the AAL2 microcode in processing the active buffers is greater than the delay tolerance provided by the RxBD table. The host should reconfigure the system to prevent such exceptions.

1.3 AAL2 Microcode Interface with the Host

The host application interacts with the AAL2 microcode using packet descriptors (PDs) placed in queues. Inside each queue, the PDs are processed first in first out (FIFO). Each AAL2 channel has one transmit queue (AAL2_Tx_Queue) and one receive queue (AAL2_Rx_Queue).

The transfer of control between the host and the AAL2 microcode is managed through the ready bit (R) in the TPD and the empty bit (E) in the RPD. This is the same mechanism used for SCC TxBDs and RxBDs except that the ATM exception queue is used instead of an event register to signal when the microcode is ready to transfer control back to the host.

A TPD with R = 0 (not ready for transmission) is available to the host. The AAL2 microcode processes a TPD and its packet only after the host has set TPD[R]. The microcode then clears TPD[R] after processing the packet and can signal the host by adding an entry in the exception queue.

An RPD with E = 0 (not empty) is available to the host. The AAL2 microcode uses a RPD to retrieve a packet only after the host has set RPD[E]. The microcode then clears RPD[E] after retrieving the packet and can signal the host by adding an entry in the exception queue.

1.4 AAL2 Microcode Data Structures for Transmit

This section describes the AAL2 microcode’s data structures used for transmitting AAL2 packets.

1.4.1 Transmit Packet Descriptor (TPD)

Similar to buffer descriptors, each transmit packet descriptor (TPD), shown in Figure 1-1, contains control bits and the pointer to the CPS_Packet. However, the TPD does not contain status bits. The status feedback from the CP is given through the exception queue; see Section 1.9, “AAL2 Microcode Exceptions.” The host should program all TPD fields during initialization.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Offset + 0	**R**	—	**W**	**INT**	—	—	—	**CM**	—	—	—	—	—	—	—	—
Offset + 2	—															
Offset + 4	**TP_PTR**															
Offset + 6																

Figure 1-1. Transmit Packet Descriptor (TPD)

Table 1-2 describes the TPD fields.

Table 1-2. TPD Field Descriptions

Offset	Bits	Name	Description
0x00	0	**R**	Ready. Determines whether the CPS-Packet is ready for transmission. 0 - The CPS-Packet associated with this TPD is not ready for transmission. Host is free to manipulate the TPD or its associated CPS-Packet. 1 - The CPS-Packet associated with this TPD has not been packed or currently is packed into the active AAL0 buffer. Host may not write into this TPD or its associated CPS-Packet once this bit is set. Host sets this bit after the CPS-Packet associated with this TPD is ready for transmission. AAL2 microcode clears this bit after the CPS-Packet has been packed into the active AAL0 buffer. During initialization, the host should clear the ready bit in all TPDs.
0x00	1	—	Reserved, should be cleared.
0x00	2	**W**	Wrap. Determines whether this is the last TPD in the AAL2_Tx_Queue. 0 - This is not the last TPD in the AAL2_Tx_Queue. 1 - This is the last TPD in the AAL2_Tx_Queue. After the CPS-Packet associated with this TPD is packed into the active AAL0 buffer, the AAL2 microcode returns to the beginning of the queue (using the TPD pointed to by AAL2_TQ_BASE). The number of TPDs in the queue is programmable and is determined only by the W bit and the 256 KBytes of memory allocated for the transmit queues. Host writes this field during initialization. AAL2 microcode does not modify this field.
0x00	3	**INT**	Determines whether interrupt is generated after this CPS-Packet is packed into the AAL0 buffer. 0 - No interrupt is generated after this CPS-Packet has been packed 1 - After the CPS-Packet is packed (and if AAL2_TCT[TPI] is set), the AAL2 microcode adds an entry to the exception queue with the AAL2 bit and TXB bit set. The global interrupt count (INT_CNT) is decremented. If the counter reaches zero, SCCE[GINT] or IDSR1[GINT] is set, and an interrupt is generated to the host. Host writes this field during initialization. AAL2 microcode does not modify this field.
0x00	4-6	—	Reserved, should be cleared.

Table 1-2. TPD Field Descriptions (Continued)

Offset	Bits	Name	Description
0x00	7	**CM**	Continuous mode. Determines whether the CPS-Packet associated with this TPD is marked as available to the host (R = 0) after processing by the CP. 0 - Normal operation. 1 - The R bit is not cleared after the TPD is closed, allowing the associated CPS-Packet to be automatically retransmitted the next time the CP accesses this TPD. Host writes this field during initialization. AAL2 microcode does not modify this field.
0x00	4-15	—	Reserved, should be cleared.
0x02	0–15	—	Reserved, should be cleared.
0x04	0–31	**TP_PTR**	Transmit CPS-Packet pointer. TP_PTR points to the beginning of the CPS-Packet which reside in external memory. TPD pointer must be burst aligned (TP_PTR[28–31] = 0). Host writes this field during initialization. AAL2 microcode does not modify this field.

1.4.2 AAL2_Tx_Queue

Similar to a TxBD table, a channel's AAL2_Tx_Queue consists of a circular table of TPDs. AAL2_Tx_Queue parameters are defined in each channel's AAL2_TCT. The queue pointer (AAL2_TQ_PTR) is advanced by the AAL2 microcode.

Figure 1-2 shows an example AAL2_Tx_Queue with six TPDs. The third and fourth TPDs (shaded) are ready for transmission (R=1). Because the INT bits are set in the third and sixth TPDs, the AAL2 microcode will generate an interrupt to the host each time a CPS-Packet associated with these TPDs has been packed.

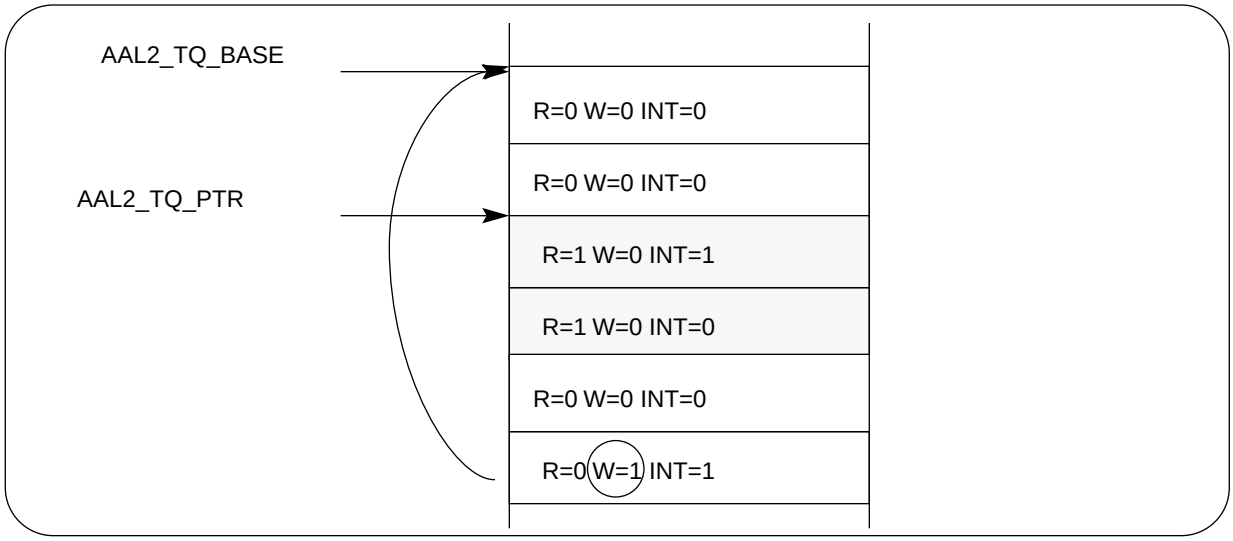


Figure 1-2. AAL2_Tx_Queue Example

1.4.3 AAL2 Transmit Connection Table (AAL2_TCT)

Each channel's AAL2_TCT, shown in Figure 1-3, contains all the parameters used by the AAL2 microcode to pack the channel's CPS-Packets. The host should configure all AAL2_TCT fields during initialization.

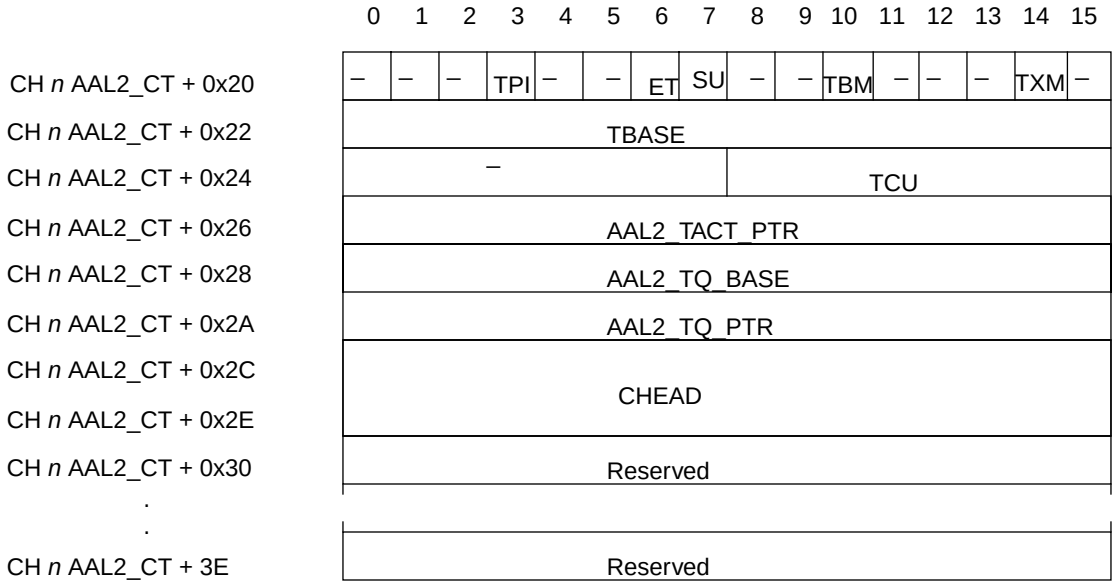


Figure 1-3. AAL2 Transmit Connection Table

Table 1-3 describes the AAL2 TCT fields.

Table 1-3. AAL2 Transmit Connection Table Field Descriptions

Offset	Bits	Name	Description
0x20	0-2	—	Reserved, should be cleared.
	3	TPI	Transmit packet interrupt. This bit controls whether the AAL2 microcode generates an interrupt to the host after processing a TPD in which the INT bit set. 0 - Interrupt generation based on INT bit is disabled. 1 - Interrupt generation based on INT bit is enabled. Host writes this field. AAL2 microcode does not modify this field.
	4-5	—	Reserved, should be cleared.
	6	ET	Enable Timer CU mechanism for this channel. 0 - Timer CU mechanism for this channel is disabled. 1 - Timer CU mechanism for this channel is enabled. Host writes this field. AAL2 microcode does not modify this field.
	7	SU	Start-up bit. Indicates to the AAL2 microcode that this is the first time this TCT has been accessed. (The AAL2 microcode clears this bit.) The host must set this field during initialization.
	8-9	—	Reserved, should be cleared.
	10	TBM	AAL2 TBSY interrupt mask. This bit allows the user to mask the AAL2 TBSY interrupt (AAL2 = 1 and TBSY = 1 in the exception queue entry). 0 - AAL2 TBSY interrupt is disabled. Exception entry with AAL2 and TBSY indication will not be generated. 1 - AAL2 TBSY interrupt is enabled. Host writes this field. AAL2 microcode does not modify this field.
	11-13	—	Reserved, should be cleared.
	14	TXM	AAL2 TXB interrupt mask. This bit allows the user to mask the AAL2 TXB interrupt (AAL2 = 1 and TXB = 1 in the exception queue entry). 0 - AAL2 TXB interrupt is disabled. Exception entry with AAL2 and TXB indication will not be generated. 1 - AAL2 TXB interrupt is enabled. Host writes this field. AAL2 microcode does not modify this field.
	15	—	Reserved, should be cleared.
0x22	-	TBASE	Transmit BD base. Holds the pointer to the first BD in the AAL0 BD table of this channel. TBASE represents bits [14–29] of an offset from TBDBASE. Bits [30–31] are always 00. This field should be the same value programmed in the TBASE as defined in the AAL0 TCT of this channel. TBDBASE is defined in the AAL0 parameter RAM of the SCC that runs this AAL2 channel. Host writes this field during initialization. AAL2 microcode does not modify this field.

Table 1-3. AAL2 Transmit Connection Table Field Descriptions (Continued)

Offset	Bits	Name	Description
0x24	0-7	—	Reserved, should be cleared.
	8-15	TCU	Determines the maximum wait time of CPS-Packets for this channel. The wait time is defined in units of RISC timer ticks (as defined in RCCR[TIMEP]). If the user uses the Timer CU mechanism (ET=1), the TCU field should be in the range of 1-255 Host writes this field during initialization. AAL2 microcode does not modify this field.
0x26	-	AAL2_TACT_PTR	Active transmit BD pointer. Points to the AAL0 BD currently being processed by AAL2 microcode. The address of the current BD in the table is (AAL2_TACT_PTR * 4) + TBDBASE. AAL2_TACT_PTR field provides bits [14-29] of the offset; bits [30-31] are always 00. Host should write TBASE to this field during initialization. This pointer is advanced by the AAL2 microcode.
0x28	-	AAL2_TQ_BASE	Pointer to the first TPD in the AAL2_Tx_Queue. The actual address of the first TPD is [(AAL2_TQ_BASE * 4) + AAL2_TPD_BASE]. AAL2_TPD_BASE is the base pointer to the TPD space and is defined in the AAL2 parameter area common to all AAL2 channels; see Section 1.7.1, “AAL2 Parameter RAM.” The AAL2_TQ_BASE field provides bits [14-29] of the word-aligned offset from AAL2_TPD_BASE; bits [30-31] are always 00. Host writes this field during initialization. AAL2 microcode does not modify this field.
0x2A	-	AAL2_TQ_PTR	Pointer to the current TPD in the AAL2_Tx_Queue. This is the TPD that the AAL2 microcode is currently processing (or will process next) in AAL2_Tx_Queue. The address of the current TPD in the AAL2_Tx_Queue is (AAL2_TQ_PTR * 4) + AAL2_TPD_BASE. AAL2_TQ_PTR field provides bits [14-29] of the offset; bits [30-31] are always 00. Host should write AAL2_TQ_BASE to this field during initialization. AAL2 microcode advances this pointer during normal operation.
0x2C	-	CHEAD	Channel header. This field contains the full (4 bytes) AAL0 buffer header of this channel. AAL2 microcode appends the CHEAD field to the CPS-PDU to create a complete AAL0 buffer. The byte ordering of this field is big endian. Host writes this field during initialization. AAL2 microcode does not modify this field.
0x30 - 0x3E	-	—	Reserved, should be cleared.

1.4.4 Example of AAL2 Transmit

Figure 1-4 shows how CPS-Packets are packed into AAL0 buffers.

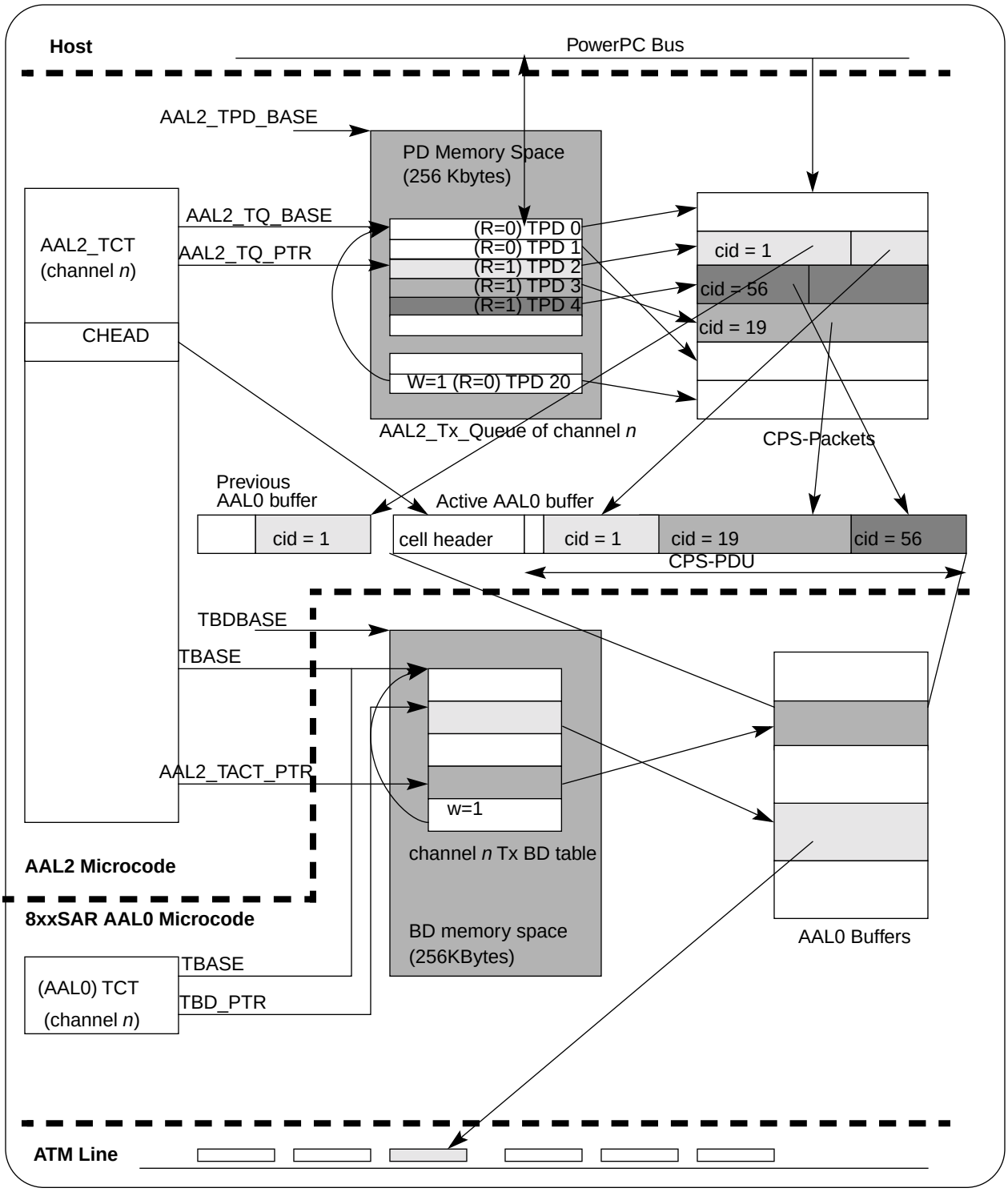


Figure 1-4. Example of the Transmit Data Flow of an AAL2 Channel

A description of the data flow shown in Figure 1-4 is as follows:

- The first part of the CPS-Packet (CID=1) has already been packed into the previous AAL0 buffer.
- The tail of CPS-Packet (CID=1) is packed into the active AAL0 buffer of this channel. TPD 2 is marked available (R=0).
- CPS-Packet (CID=19) is copied to the AAL0 buffer pointed to by TPD 3. TPD 3 is marked available.
- Only the first part of CPS-Packet (CID=56) has room in the active AAL0 buffer. The rest of the CPS-Packet remains in external memory to be placed in the next AAL0 buffer when the AAL2 microcode is activated again for this channel.
- The TxBD (pointed to by AAL2_TACT_PTR) that points to the active buffer is closed (R bit is set). (The 8xxSAR transmits this buffer later according to the APC or host scheduling.)
- Note that while the AAL2 microcode fills the active AAL0 buffer, the 8xxSAR (AAL0 microcode) uses TBD_PTR to actually transmit the AAL0 buffers prepared earlier.

1.4.5 Built-in Timer CU Support

The Timer CU mechanism is designed to guarantee a minimum packet throughput for an AAL2 channel. It ensures that transmit CPS-Packets are not delayed by the AAL2 microcode beyond the quality of service requirements for a channel.

In AAL2 operation, if a channel does not have enough CPS-Packets to fill a partially filled active buffer, the AAL2 microcode waits to send the buffer. If the active buffer is not filled with new CPS-Packets the next time this channel is activated, the microcode still delays sending the buffer in hopes of achieving a better utilization of the transmission line. The Timer CU mechanism limits this delay.

The AAL2 microcode implements the Timer CU mechanism using a wait table (AAL2_TxWait_table allocated by the host) and the CP's internal RISC timer software table. The wait table holds the remaining wait time for all partially filled active buffers. Each "tick" of the RISC timer activates the AAL2 microcode to pad (fill with zeros) and send all partially filled buffers which have expired. The duration between ticks is defined in the TIMEP field of the RISC controller configuration register (RCCR). The RISC timers are enabled in RCCR[TIME]. See the discussion of the RISC timer table and the RCCR in the communications processor chapter of the *MPC860 User's Manual*.

For each AAL2 channel using the Timer CU mechanism, the host should set AAL2_TCT[ET] and initialize AAL2_TCT[TCU]. Once initialized, the Timer CU mechanism does not require host intervention.

1.4.5.1 Algorithm Description

This section describes the algorithm used to implement the Timer CU mechanism for timer-enabled channels.

1.4.5.1.1 Starting the Timer for a Partially Filled Active Buffer

When the AAL2 microcode is unable to fill a channel's active buffer, it does the following:

- Starts the count down for the channel's partially filled active buffer by setting the AAL2_TxWait_table[X,Y] bit, where:

  X = (TCU + AAL2_TxWait_PTR) modulo (the number of rows in the wait table)

  Y = channel number.
- Stores X in the channel's AAL2_TCT (so that it can later clear the bit once the active buffer is sent, either full or partially filled).

The TCU field is defined in AAL2_TCT of each channel. The wait pointer (AAL2_TxWait_PTR) is defined in the AAL2 parameter RAM area; see Section 1.7.1, "AAL2 Parameter RAM."

1.4.5.1.2 Count Down Mechanism and Detection of Expired Buffers

Each advance of the wait pointer through the rows of the wait table corresponds to one RISC timer tick. The wait pointer marks the current row in the wait table. All bits that are set in the current row represent expired channels. Each time the RISC timer expires, it asserts a request to the CPM. This request in turn activates the AAL2 microcode to do the following:

- For each bit set in the current row, the AAL2 microcode:
  - Pads the partially filled active buffer with zeros.
  - Closes its BD (R=1). (The cell is sent the next time the channel is scheduled by the APC or the host issues an APC BYPASS command for the channel.)
  - Clears the channel's bit in the wait table.
- The wait pointer advances to the next row of the wait table, thereby decreasing the wait time of all remaining partially filled active buffers by one RISC timer tick.

When the wait pointer reaches the end of the table, it advances back to the beginning. The size of the wait table is limited to 256 rows, which means that the maximum delay of an active buffer is (255 * RISC timer period).

Note that if all the bits in the current row are 0 (which may happen in most cases), the AAL2 microcode simply advances the wait pointer. All channels in the wait table then become one row closer to the wait pointer, which means that one RISC timer period is subtracted from the wait time of all the partially filled active buffers.

The host controls the count down mechanism in each channel's AAL2_TCT:

- AAL2_TCT[ET] defines whether the Timer CU mechanism for this channel is enabled.
- AAL2_TCT[TCU] determines the maximum delay for the channel's partially filled active buffer (TCU * RISC timer period).

For example, assuming the following:

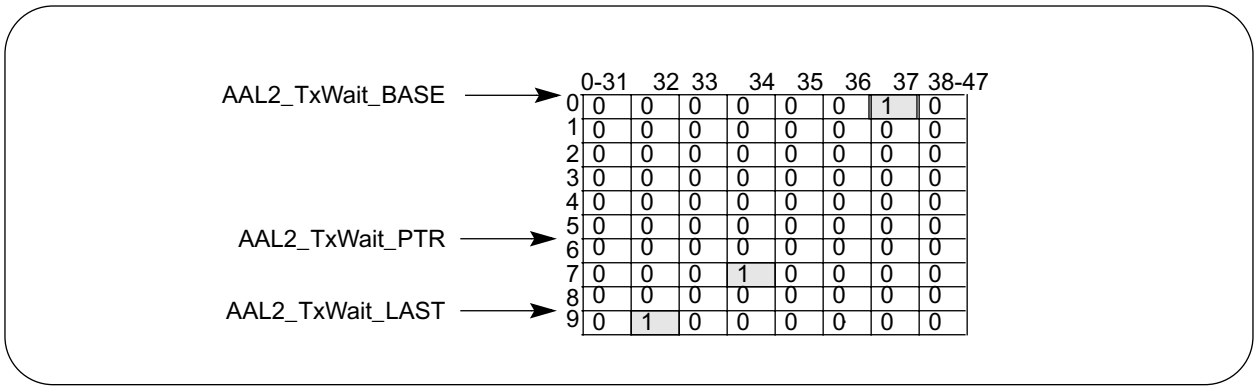
- RISC timer expires every 0.5 ms,
- TCU = 6 (for channel *n*), and
- AAL2_TxWait_PTR points to row 3,

then, if the active buffer for channel *n* is only partially filled, the AAL2 microcode sets the *n*th bit in the 9th row of the wait table (6 rows from the position of the wait pointer).

1.4.5.2 Example of Timer CU Implementation

In Figure 1-5, the wait table is implemented with 10 rows. There are six AAL2 channels (channels 32-37). The width of the wait table is three half-word (48 bits). Columns 0-31, 38-47 of the wait table are unused (must be cleared by the host during initialization). The table width is programmable in 16-bit increments.

Figure 1-5. Example of the Wait Table



The wait pointer points to row 6. Channels 32, 34 and 37 have partially filled active buffers:

- Channel 34 expires in one RISC timer period
- Channel 32 expires in three RISC timer periods
- Channel 37 expires in four RISC timer periods

Note that if the partially filled active buffer is filled before its time expires, the AAL2 microcode clears the channel's bit in the wait table.

1.4.6 Optional APC Overrun (APCO) Event Mask

When the user oversubscribes a channel, APC overrun events are generated by the CP. The AAL2 microcode provides the option to mask these interrupts by programming trap register four (RCTR4) as shown in Table 1-4.

Table 1-4. RCTR4 Initialization for APCO Masking

Trap Register	850SAR Rev. A and B	860P and 855T
RCTR4	0x933D	0x933F

The APC overrun mask bit (APCOM) is located in two previously reserved entries of the APC parameter table, as shown in Table 1-5.

Table 1-5. APC Parameter Table Additions for APCO Masking

Offset	Name	Width	Description	User Writes
APC Par + 0x18	APCLST1	HalfWord	APC level status for the first priority service	User defined
APC Par + 0x38	APCLST2	HalfWord	APC level status for the second priority service	User defined

Table 1-6 describes the APCLST field.

Table 1-6. APCLST Description

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>APCLSTn</td><td>APC status (first and second level). Contains the APC overrun mask.
<table>
<tr><td>Bit</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr>
<tr><td>Field</td><td>APCOM</td><td colspan="15">—</td></tr>
</table>
The APCOM bit is described below.
<table>
<tr><td>Bits</td><td>Name</td><td>Description</td></tr>
<tr><td>0</td><td>APCOM</td><td>APC scheduling table overrun event mask for this APC level.
0 APCO interrupts are enabled.
1 APCO interrupts are disabled.</td></tr>
<tr><td>1–15</td><td>—</td><td>Reserved</td></tr>
</table>
</td></tr>
</table>

1.5 AAL2 Microcode Data Structures for Receive

This section describes the AAL2 microcode's data structures used for receiving AAL2 packets.

1.5.1 Receive Packet Descriptor (RPD)

Similar to buffer descriptors, each receive packet descriptor (RPD), shown in Figure 1-6, contains control bits and the pointer to the CPS_Packet. However, the RPD does not contain status bits. The status feedback from the CP is given through the exception queue; see Section 1.9, "AAL2 Microcode Exceptions." The host should program all RPD fields during initialization.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Offset + 0	**E**	—	**W**	**INT**	—	—	—	**CM**	—	—	—	—	—	—	—	—
Offset + 2	—	—	—	—	—	—	E0	E1	E2	E3	E4	—	E6	E7	—	—
Offset + 4	**RP_PTR**															
Offset + 6																

Figure 1-6. Receive Packet Descriptor (RPD)

Table 1-7 describes the RPD fields.

Table 1-7. RPD Field Descriptions

Offset	Bits	Name	Description
0x00	0	**E**	Empty. Determines whether the CPS-Packet is accessible to the host or the AAL2 microcode. 0 - The CPS-Packet associated with this RPD has been filled with the received packet, or receiving has been aborted due to an error. Host is free to examine or to write to the fields of this RPD. The AAL2 microcode will not access this RPD while the E bit is 0. Host should set this bit after processing the CPS-Packet and RPD. 1 - The CPS-Packet associated with this RPD is empty, or is currently being filled by the AAL2 microcode. This RPD and its associated CPS-Packet can be used by the AAL2 microcode. Once the E bit is set, the host should not modify any fields of this RPD. AAL2 microcode clears this bit after it retrieves the CPS-Packet and updates the RPD. During host initialization, E bit should be set in all RPDs.
0x00	1	—	Reserved, should be cleared.
0x00	2	**W**	Wrap. Determines whether this is the last RPD in the AAL2_Rx_Queue. 0 - This is not the last RPD in the AAL2_Rx_Queue. 1 - This is the last RPD in the AAL2_Rx_Queue. After the CPS-Packet associated with this RPD is retrieved from the active AAL0 buffer, the AAL2 microcode returns to the beginning of the queue (using the RPD pointed to by AAL2_RQ_BASE). The number of RPDs in the queue is programmable and is determined only by the W bit and the 256 KBytes of memory allocated for the transmit queues. This bit is configured by the host during initialization and is not modified by the AAL2 microcode.
0x00	3	**INT**	Determines whether AAL2 interrupt is generated after this CPS-Packet is retrieved by the AAL2 microcode. 0 - No interrupt is generated after this CPS-Packet is filled. 1 - After the CPS-Packet is retrieved (and if AAL2_RCT[RPI] is set), the AAL2 microcode adds an entry to the exception queue with the AAL2 bit and RXB bit set. The global interrupt count (INT_CNT) is decremented. If the counter reaches zero, SCCE[GINT] or IDSR1[GINT] is set, and an interrupt is generated to the host. This bit is configured by the host and is not modified by the AAL2 microcode.
0x00	4-6	—	Reserved, should be cleared.
0x00	7	**CM**	Continuous mode. Determines whether the CPS-Packet associated with this RPD is marked as available to the host (E=0) after processing by the CP. 0 - Normal operation. 1 - The E bit is not cleared after the RPD is closed, allowing the associated CPS-Packet to be automatically rewritten the next time the CP accesses this RPD. Host writes this field during initialization. AAL2 microcode does not modify this field.
0x00	4-15	—	Reserved, should be cleared.
0x02	0-5	—	Reserved, should be cleared.

Table 1-7. RPD Field Descriptions (Continued)

Offset	Bits	Name	Description
0x02	6	E0	Indicates parity error on STF field. 0 - No parity error on STF field. 1 - Parity error was detected on STF field. The complete CPS-PDU is discarded. When this error is indicated, the CPS-Packet associated with this RPD is not valid. Host should clear this bit after processing the RPD. Host should clear this bit during initialization.
0x02	7	E1	Indicates whether sequence number of the STF is valid. 0 - Sequence number of the STF is valid. 1 - Sequence number of the STF is not valid; if the OSF is less than 47, the AAL2 microcode begins processing at the octet pointed by the OSF (uses the next RPD in the queue), otherwise the complete CPS-PDU is discarded. When this error is indicated, the CPS-Packet associated with this RPD is not valid. Host should clear this bit after processing the RPD. Host should clear this bit during initialization.
0x02	8	E2	Indicates whether the expected number of octets for a CPS-Packet overlapping into this CPS-PDU matches the STF. 0 - The expected number of overlapping octets matches the STF. 1 - The expected number of overlapping octets does not match the STF; if the OSF is less than 47, processing starts at the octet pointed to by the OSF. When this error is indicated, the CPS-Packet associated with this RPD is not valid. The host should clear this bit after processing the RPD. Host should clear this bit during initialization.
0x02	9	E3	Indicates whether the OSF of the STF contains a value of 48 or greater; 0 - OSF of the STF has value smaller than 48. 1 - OSF of the STF contains a value of 48 or greater; the complete CPS-PDU is discarded. When this error is indicated, the CPS-Packet associated with this RPD is not valid. Host should clear this bit after processing the RPD. Host should clear this bit during initialization.
0x02	10	E4	Indicates whether a HEC error was detected in the CPS-Packet header. 0 - No HEC error was detected. 1 - HEC error was detected; the CPS-PDU is discarded. When this error is indicated, the CPS-Packet associated with this RPD is not valid. Host should clear this bit after processing the RPD. Host should clear this bit during initialization.
0x02	11	—	Reserved, should be cleared.

Table 1-7. RPD Field Descriptions (Continued)

Offset	Bits	Name	Description
0x02	12	E6	Indicates whether the CPS-Packet was discarded due to errors detected before the reassembly of the CPS-Packet was completed. 0 - No errors were detected before the reassembly of the CPS-Packet was completed. (The CPS-Packet was not discarded.) 1 - The CPS-Packet was discarded due to errors detected before the reassembly of the CPS-Packet was completed. When this error is indicated, the CPS-Packet associated with this RPD is not valid. Host should clear this bit after processing the RPD. Host should clear this bit during initialization.
0x02	13	E7	Indicates whether a HEC error occurred in the CPS-Packet header of a packet overlapping a CPS-PDU boundary. 0 - No HEC error occurred for the overlapping packet. 1 - A HEC error occurred for the overlapping packet; if the value of the OSF is less than 47, processing starts at the octet pointed to by the OSF. When this error is indicated, the CPS-Packet associated with this RPD is not valid. Host should clear this bit after processing the RPD. Host should clear this bit during initialization.
0x02	14-15	—	Reserved, should be cleared.
0x04	-	**RP_PTR**	Pointer to the receive CPS-Packet. RP_PTR points to the beginning of the CPS-Packet which reside in external memory. This pointer is specified by the host, and must be burst aligned (RP_PTR[28–31]=0). AAL2 microcode does not modify this field

1.5.2 AAL2_Rx_Queue

Similar to an RxBD table, a channel's AAL2_Rx_Queue consists of a circular table of RPDs. AAL2_Rx_Queue parameters are defined in each channel's AAL2_RCT. The queue pointer (AAL2_RQ_PTR) is advanced by the AAL2 microcode.

Figure 1-7 shows an example AAL2_Rx_Queue with six RPDs. The second and third RPDs (shaded) have been filled (E=0). Because the INT bits are set in the third and sixth RPDs, the AAL2 microcode will generate interrupts to the host after the CPS-Packets associated with these RPDs have been retrieved.

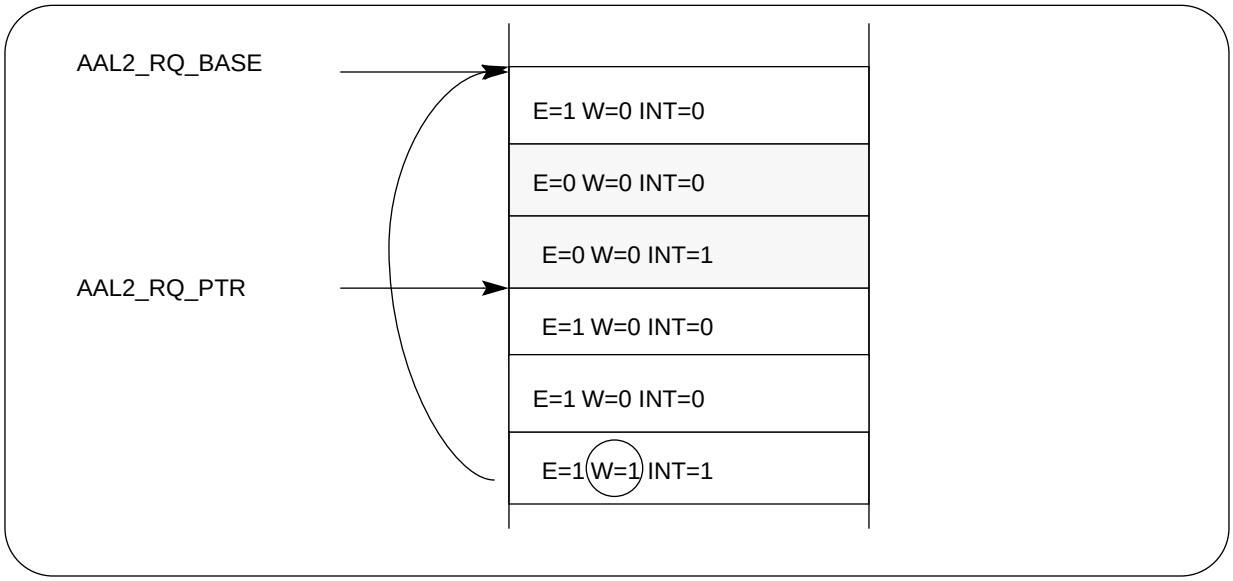


Figure 1-7. AAL2_Rx_Queue Example

1.5.3 AAL2 Receive Connection Table (AAL2_RCT)

Each channel's AAL2_RCT, shown in Figure 1-8, contains all the parameters used by the AAL2 microcode to retrieve the channel's CPS-Packets. The host should configure all AAL2_RCT fields during initialization.

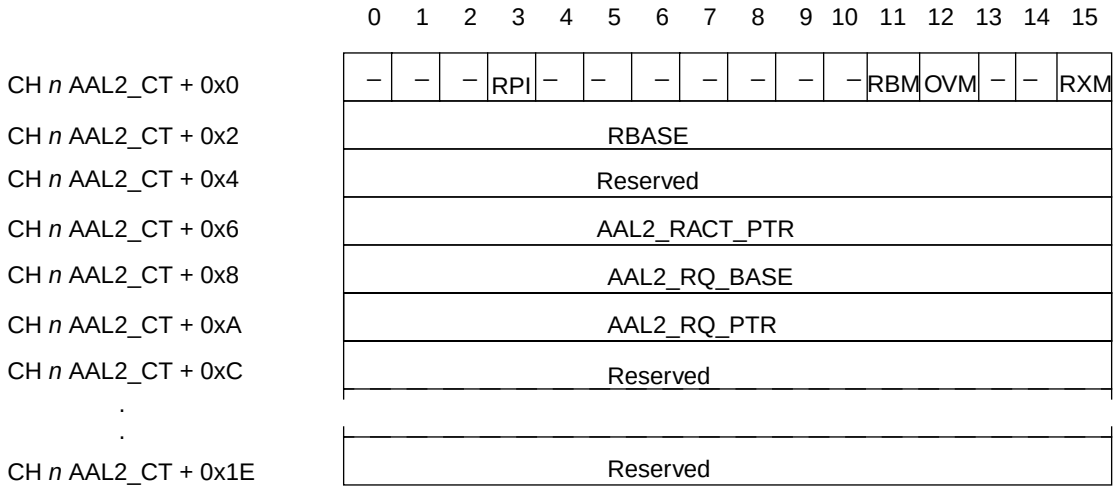


Figure 1-8. AAL2 Receive Connection Table

Table 1-8 describes the AAL2 RCT fields.

Table 1-8. AAL2 Receive Connection Table Field Descriptions

Offset	Bits	Name	Description
0x00	0-2	—	Reserved, should be cleared.
	3	RPI	Receive packet interrupt. This bit controls whether the AAL2 microcode generates an interrupt to the host after processing a RPD in which the INT bit is set. 0 - Interrupt generation based on INT bit is disabled. 1 - Interrupt generation based on INT bit is enabled. This field is configured by host and is not modified by the AAL2 microcode.
	4-10	—	Reserved, should be cleared.
	11	RBM	AAL2 RBSY interrupt mask. This bit allows the user to mask the AAL2 RBSY interrupt (AAL2 = 1 and RBSY = 1 in the exception queue entry. 0 - AAL2 RBSY interrupt is disabled. Exception entry with AAL2 and RBSY indication will not be generated. 1 - AAL2 RBSY interrupt is enabled. Host writes this field. AAL2 microcode does not modify this field.
	12	OVM	AAL2 OVF interrupt mask. This bit allows the user to mask the AAL2+OVF interrupt (AAL2 = 1 and OVF = 1 in the exception queue entry. 0 - AAL2 OVF interrupt is disabled. Exception entry with AAL2 and OVF indication will not be generated. 1 - AAL2 OVF interrupt is enabled. Host writes this field. AAL2 microcode does not modify this field.
	13-14	—	Reserved, should be cleared.
	15	RXM	AAL2 RXB interrupt mask. This bit allows the user to mask the AAL2 RXB interrupt (AAL2 = 1 and RXB = 1 in the exception queue entry. 0 - AAL2 RXB interrupt is disabled. Exception entry with AAL2 and RXB indication will not be generated. 1 - AAL2 RXB interrupt is enabled. Host writes this field. AAL2 microcode does not modify this field.
0x02	-	RBASE	Receive BD base. Holds the pointer to the first BD in the AAL0 BD table of this channel. RBASE represents bits [14–29] of an offset from RBDBASE. Bits [30–31] are always 00. This field should be the same value programmed in the RBASE as defined in the AAL0 RCT of this channel. RBDBASE is defined in the AAL0 parameter RAM of the SCC that runs this AAL2 channel. This field is configured by host and is not modified by the AAL2 microcode
0x04	-	—	Reserved, should be cleared.

Table 1-8. AAL2 Receive Connection Table Field Descriptions (Continued)

Offset	Bits	Name	Description
0x06	-	AAL2_RACT_PTR	Active receive BD pointer. Points to the AAL0 BD currently being processed by AAL2 microcode. The address of the current BD in the table is (AAL2_RACT_PTR * 4) + RBDBASE. AAL2_RACT_PTR field provides bits [14-29] of the offset; bits [30-31] are always 00. Host should write RBASE to this field during initialization. This pointer is advanced by the AAL2 microcode.
0x08	-	AAL2_RQ_BASE	Pointer to the first RPD in AAL2_Rx_Queue. The actual address of the first TPD is [(AAL2_RQ_BASE * 4) + AAL2_RPD_BASE]. AAL2_RPD_BASE is the base pointer to the RPD space and is defined in the AAL2 parameter area common to all AAL2 channels; see Section 1.7.1, “AAL2 Parameter RAM.” The AAL2_RQ_BASE field provides bits [14-29] of the word-aligned offset from AAL2_RPD_BASE; bits [30-31] are always 00. This field is configured by host and is not modified by the AAL2 microcode.
0x0A	-	AAL2_RQ_PTR	Pointer to the current RPD in AAL2_Rx_Queue. This is the RPD that the AAL2 microcode is currently processing (or will process next) in AAL2_Rx_Queue. The address of the current RPD in the AAL2_Rx_Queue is [(AAL2_RQ_PTR * 4) + AAL2_RPD_BASE]. The AAL2_RQ_PTR field provides bits [14-29] of the offset; bits [30-31] are always 00. Host should write AAL2_RQ_BASE to this field during initialization. AAL2 microcode advances this pointer during normal operation.
0x10-0x1E	-	—	Reserved, should be cleared.

1.5.4 Example of AAL2 Receive

Figure 1-9 shows how CPS-Packets are retrieved from AAL0 buffers.

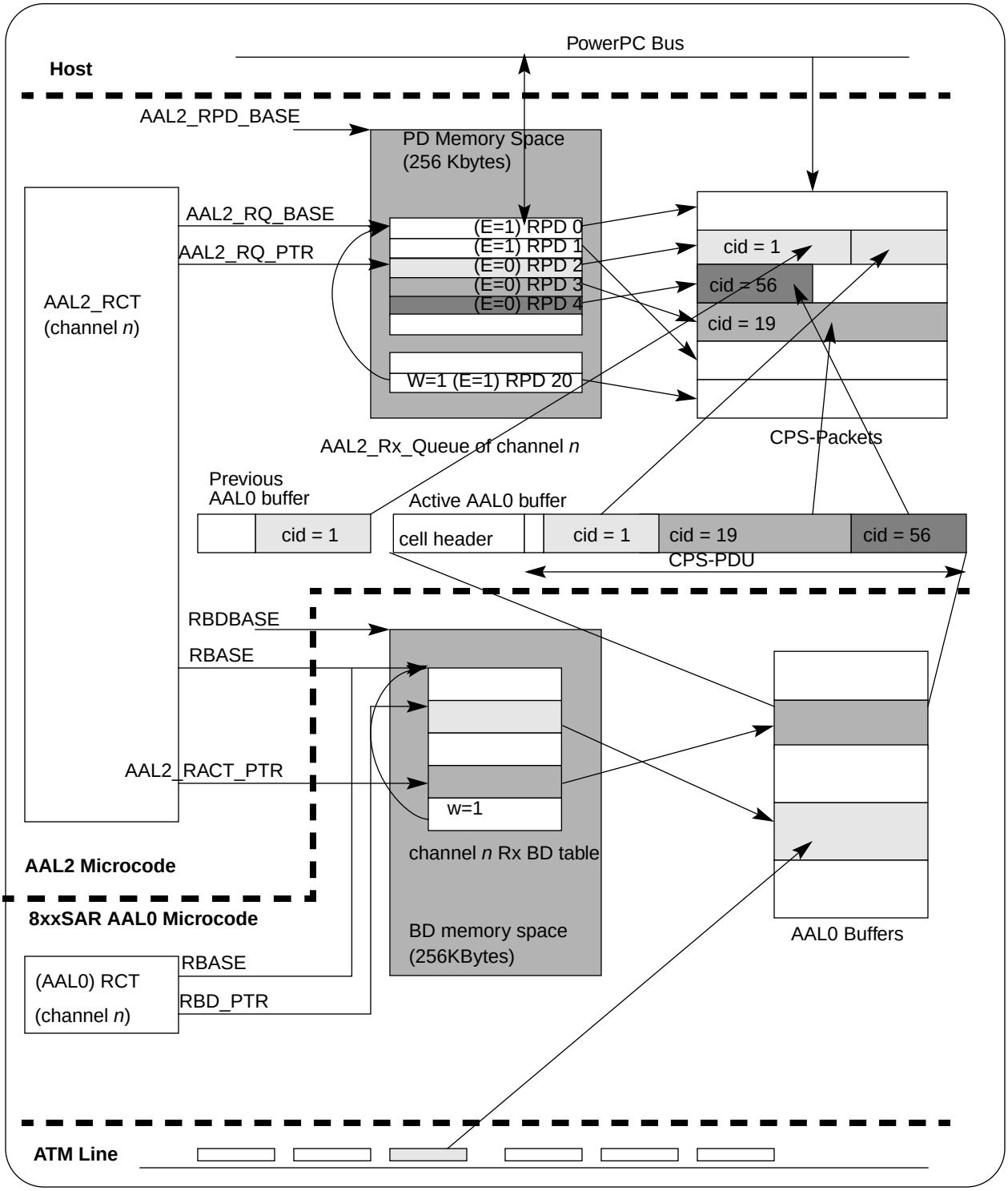


Figure 1-9. Example of the Receive Data Flow of an AAL2 Channel

A description of the data flow shown in Figure 1-9 is as follows:

- The head of CPS-Packet (CID=1) has already been received (using the previous AAL0 buffer) and has been stored in memory using the packet pointer in RPD 2. The tail of this packet (CID=1), now in the first part of the active AAL0 buffer, is appended to the head, and RPD 2 is marked full (E=0).
- Then, CPS-Packet (CID=19) is copied to memory using RPD 3, and RPD 3 is marked full (E=0).
- Only the first part of CPS-Packet (CID=56) is in the active buffer, and it is stored in memory using RPD 4.
- Note that while the AAL2 microcode processes the active AAL0 buffer (pointed to by AAL2_RACT_PTR), the 8xxSAR AAL0 microcode may be filling another AAL0 buffer (pointed to by RBD_PTR) to be processed later.

1.6 Systems Restrictions When Using AAL2 Microcode

Any combination of SCCs can run AAL2 channels; that is, all SCCs can activate the AAL2 microcode. However, the following restrictions apply:

- Because the AAL2 microcode uses only one set of global AAL2 structures, each SCC should handle a unique set of AAL2 channel numbers. No two SCCs should control the same AAL2 channel *n*.
- AAL2 channel numbers can only be in the range 32–255.
- If the AAL2 microcode is enabled, channels 32-255 are dedicated to AAL2. Channels 0-31 and channel numbers greater than 255 can be used for non-AAL2 channels.

NOTE:

In serial mode or single-PHY mode, configure RCT0 as the raw cell queue. In multi-PHY mode, configure RCT0, RCT4, RCT8, and RCT12 as the raw cell queues for PHY 0–3, respectively.

1.7 Global AAL2 Data Structures

The global AAL2 structures consist of the AAL2 parameter RAM, two programmable scratch areas for internal use (AAL2_SCRATCH1 and AAL2_SCRATCH2), the CRC tables. the external CT pointer table (AAL2_ECT_PTR_table), and the optional wait table (AAL2_TxWait_table). All the SCCs running AAL2 microcode share this one set of global AAL2 data structures. The following subsections discuss the AAL2 parameter RAM and the external CT pointer table.

1.7.1 AAL2 Parameter RAM

All global AAL2 parameters and the pointers to global data structures are defined in the 64-byte AAL2 parameter RAM area, starting at location (IMMR+0x3FC0—which is offset 0x1FC0 from the beginning of dual-port RAM) in the parameter RAM. See Figure 1-10.

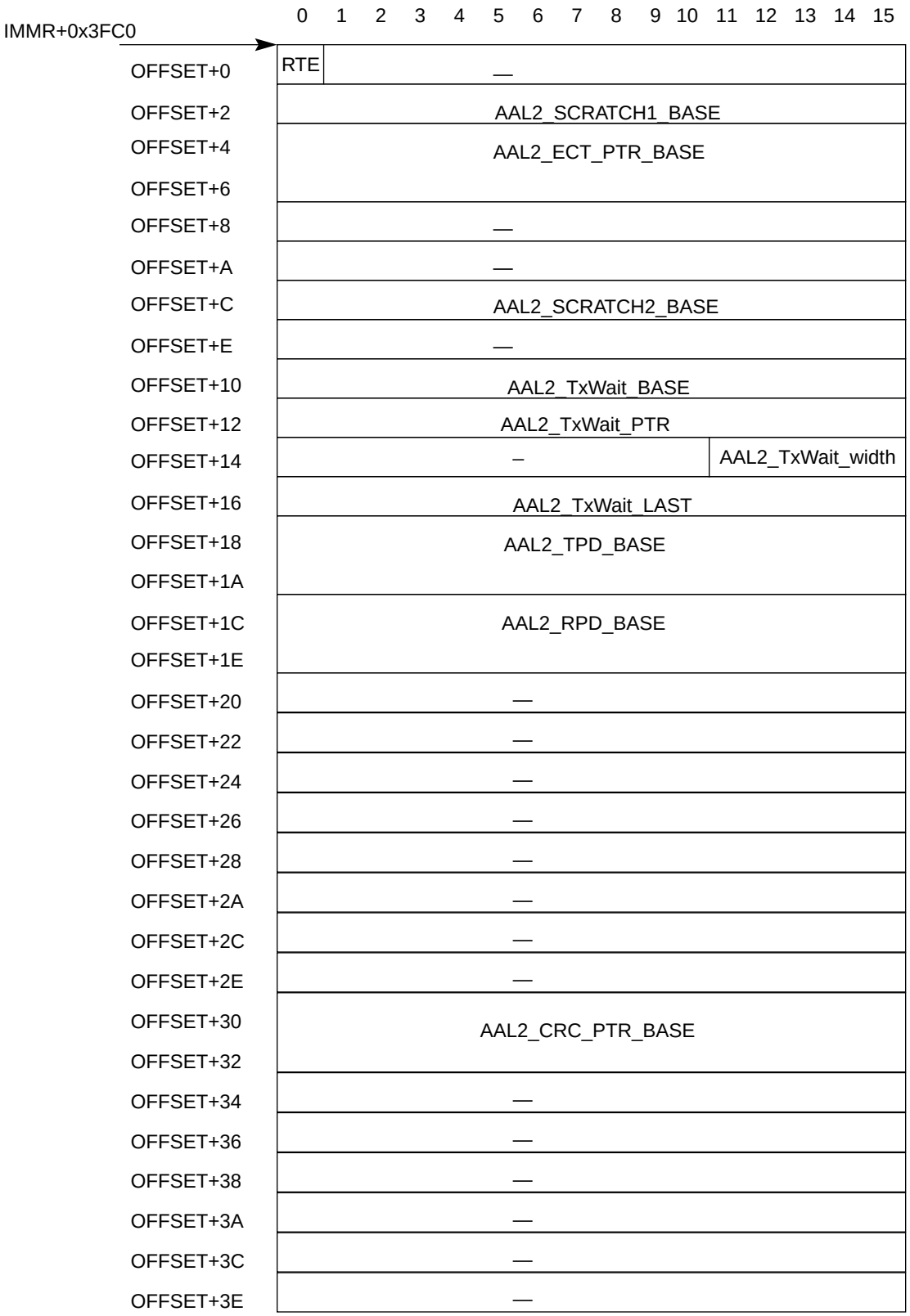


Figure 1-10. AAL2 Parameter RAM

Table 1-9 describes the global AAL2 parameters.

Table 1-9. AAL2 Parameter RAM Memory Map

Offset	Bits	Name	Description
0x00	0	RTE	RISC timers enable. To enable the RISC timers when using the timer CU mechanism set RTE. 0 - RISC timers are disabled. 1 - RISC timers are enabled.
	1-15	—	Reserved, should be cleared.
0x02	-	AAL2_SCRATCH1_BASE	16-bit pointer to a 512-byte scratch area in the DPR used by the AAL2 microcode during AAL2 operation. This pointer should be 64-byte aligned (AAL2_SCRATCH1_BASE[26–31]=00_0000). Note: This pointer is offset from IMMR (and not from IMMR+0x2000 which is the beginning of the DPR). For example, if AAL2_SCRATCH1 starts at IMMR+0x2C00, the user should program 0x2C00 in AAL2_SCRATCH1_BASE (not 0x0C00). Host writes this field and clears the 512 bytes during initialization. AAL2 microcode does not modify this field.
0x04	-	AAL2_ECT_PTR_BASE	AAL2_ECT_PTR_BASE points to the first entry of AAL2_ECT_PTR_table. AAL2_ECT_PTR_table should start on a word boundary (AAL2_ECT_PTR_BASE[30-31]=00). Host writes this field during initialization; see Section 1.7.2, “Mapping the AAL2 Connection Tables in External Memory.” AAL2 microcode does not modify this field
0x08 - 0x0A	-	—	Reserved, should be cleared.
0x0C		AAL2_SCRATCH2_BASE	16-bit pointer to a 256-byte scratch area in the DPR used by the AAL2 microcode during AAL2 operation. This pointer should start on a half-word boundary (AAL2_SCRATCH2_BASE[31]=0). Note: This pointer is offset from IMMR (and not from IMMR+0x2000 which is the beginning of the DPR). For example, if AAL2_SCRATCH2 starts at IMMR+0x3800, the user should program 0x3800 in AAL2_SCRATCH2_BASE (not 0x1800). Host writes this field and clears the 256 bytes during initialization. AAL2 microcode does not modify this field.
0xE	0-15	—	Reserved, should be cleared.
0x10	-	AAL2_TxWait_BASE	Points to the first row of the AAL2_TxWait_table in the DPR. This pointer should be on a word boundary (AAL2_TxWait_BASE[30-31]=00). Note: This pointer is offset from IMMR (and not from IMMR+0x2000 which is the beginning of the DPR). For example, if the wait table starts at IMMR+0x2930, the user should program 0x2930 in AAL2_TxWait_BASE (not 0x0930). Host writes this field during initialization. AAL2 microcode does not modify this field
0x12	-	AAL2_TxWait_PTR	Points to the current row in the AAL2_TxWait_table. This parameter is used by the AAL2 microcode to count down the time for partially-filled active AAL0 buffers. Host should initialize AAL2_TxWait_PTR to AAL2_TxWait_BASE. AAL2 microcode advances this pointer during normal operation

Table 1-9. AAL2 Parameter RAM Memory Map (Continued)

Offset	Bits	Name	Description
0x14	0-10	—	Reserved, should be cleared.
	11-15	AAL2_TxWait_width	Determines the row width of the AAL2_TxWait_table in half-words (A half-word is 16 bits). AAL2_TxWait_width should be in the range [1–16] half-words. Each half-word can support 16 channels (1 bit per channel). Host writes this field during initialization. AAL2 microcode does not modify this field
0x016	-	AAL2_TxWait_LAST	Points to the last row of the AAL2_TxWait_table. This pointer should be on a half-word boundary (AAL2_TxWait_LAST[31]=0). Note that AAL2_TxWait_LAST must be within 256 rows of AAL2_TxWait_BASE. The size of each row is AAL2_TxWait_width half-words. Host writes this field during initialization. AAL2 microcode does not modify this field
0x18	-	AAL2_TPD_BASE	32-bit base pointer to the global TPD area. Defines the starting location in external memory for a space of up to 256 KBytes where the TPDs of all AAL2_TxQueues reside. The base pointer to a specific channel's AAL2_TxQueue is located in the channel's AAL2_TCT. Host writes this field during initialization. AAL2 microcode does not modify this field
0x1C	-	AAL2_RPD_BASE	32-bit base pointer to the global RPD area. Defines the starting location in external memory for a space of up to 256 KBytes where the RPDs of all AAL2_RxQueues reside. The base pointer to a specific channel's AAL2_RxQueue is located in the channel's AAL2_RCT. Host writes this field during initialization. AAL2 microcode does not modify this field
0x20 -0x2E	-	—	Reserved, should be cleared.
0x30	-	AAL2_CRC_PTR_BASE	32-bit base pointer to the CRC tables. The AAL2 microcode uses the CRC tables to calculate the HEC field of the CPS-Packet header. The user has to allocate 8 Kbytes in external memory (8-Kbyte aligned) (AAL2_CRC_PTR_BASE[19-31]=0). These tables are generated by a C program called 'fill_crc19' which is provided in the example code. During initialization, the host should call this program with the base address: fill_crc19(AAL2_CRC_PTR_BASE) Host writes this field during initialization. AAL2 microcode does not modify this field
0x34 -0x3E	-	—	Reserved, should be cleared.

1.7.2 Mapping the AAL2 Connection Tables in External Memory

Figure 1-11 shows the structure of the external CT pointer table (AAL2_ECT_PTR_table) used to map the AAL2 connection tables in external memory. Note that the external CT pointer table entries should be 64-byte aligned.

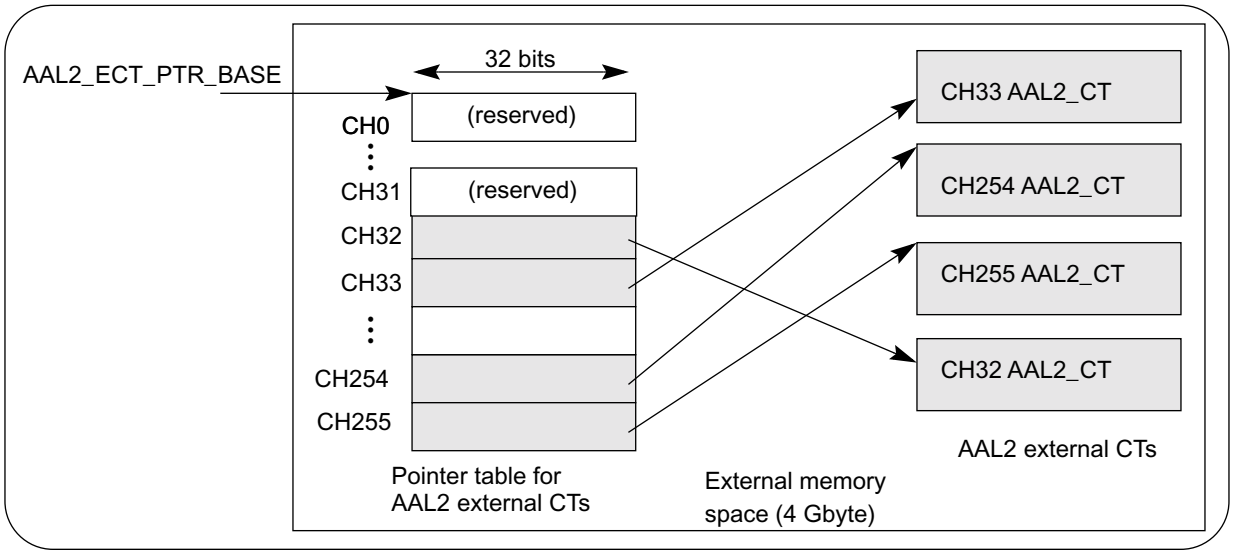


Figure 1-11. Example Using the External CT Pointer Table to Allocate AAL2 CTs

Once defined by the user, the pointer table is used by the AAL2 microcode to locate CTs. The CT pointer of channel *n* is located *n* words from AAL2_ECT_PTR_BASE. Note that the first 32 words of the pointer table are reserved.

1.8 DPR Memory Map with AAL2 Microcode

Figure 1-12 shows where the AAL2 microcode resides in the dual-port RAM.

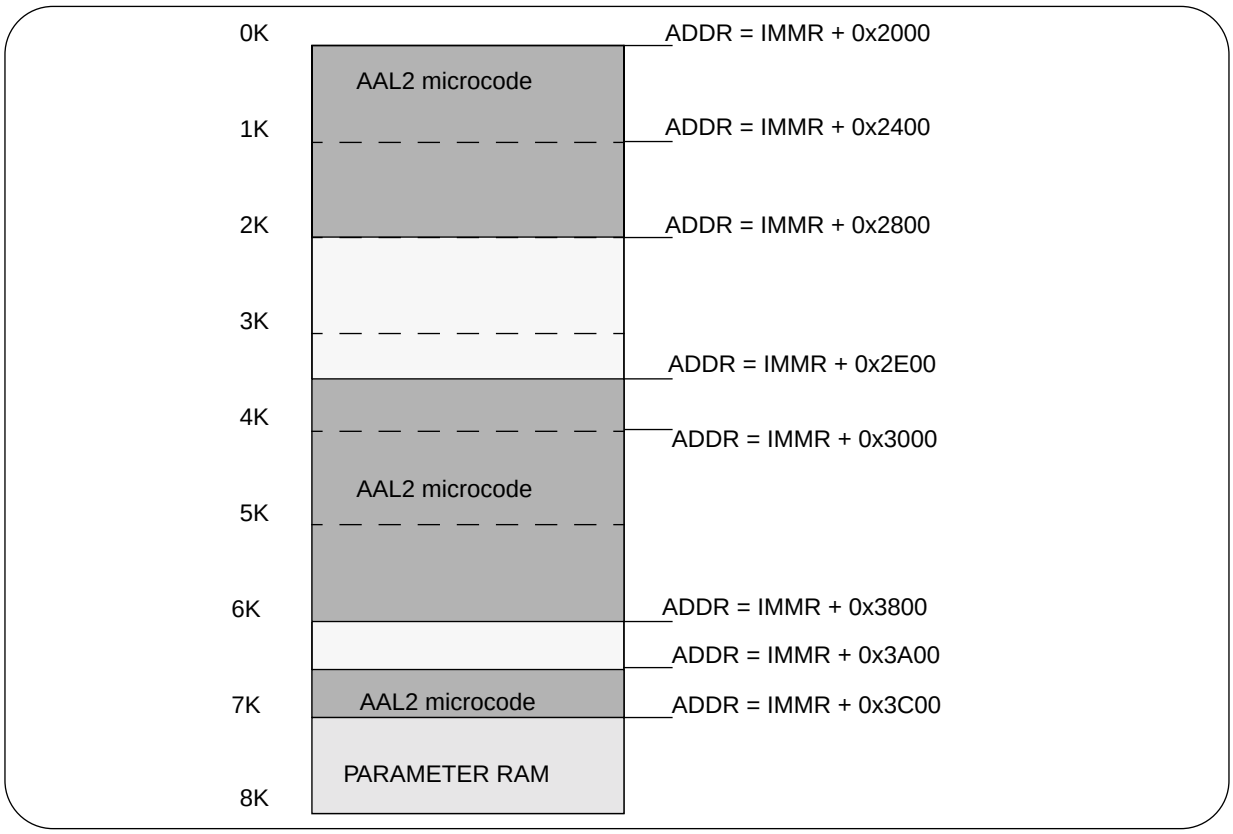


Figure 1-12. DPR Mapping for an AAL2 Application

AAL2 microcode occupies a total of 5 Kbytes of the 8-Kbyte DPR space: 0x2000 - 0x27FF, 0x2E00 - 0x37FF and 0x3A00 - 0x3BFF.

Other data structures that must be in the DPR are as follows:

- AAL2_SCRATCH1 area (512 bytes—cleared during initialization)
- AAL2_SCRATCH2 area (256 bytes—cleared during initialization)
- AAL2_TxWait_table (if Timer CU mechanism is used)
- APC scheduling tables and parameters, the transmit queue, and the raw cell queue (AAL0 channel 0 CT)

Any remaining empty space in the DPR can be used for other structures.

1.9 AAL2 Microcode Exceptions

During AAL2 microcode execution, errors and indications are sent to the host. As for all ATM adaptation layers in the 8xxSAR, these indications are reported in the 8xxSAR exception queue.

All exceptions generated by the AAL2 microcode set bit 4 (AAL2 microcode indication). An AAL2-specific exception entry is defined as in Figure 1-13.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x00	**V**	**W**	—		AAL2	—					TBSY	RBSY	OVF	—	TXB	RXB
0x02	CHNUM															

Figure 1-13. AAL2-Specific Exception Entry Format

Table 1-10 describes the AAL2-specific exceptions.

Table 1-10. AAL2-Specific Exceptions

Offset	Bits	Name	Description
0x00	0	**V**	Valid bit. Indicates that this entry contains a valid interrupt. AAL2 microcode sets the V bit upon generating the exception. Host clears the V bit immediately after it reads the interrupt flags of the entry. The V bits in all entries must be cleared by the host during initialization
	1	**W**	Wrap bit. Indicates the last entry in the circular exception queue. The next event's entry in the queue is written to the address contained in INTBASE. During initialization, the host should set the W bit in the queue's last entry; all others should be cleared. AAL2 microcode does not modify this field.
	2-3	—	Reserved, should be cleared.
	4	AAL2	AAL2 microcode interrupt. This interrupt was generated by the AAL2 microcode and indicates an exception event related to AAL2 microcode execution. Host should examine bits 10,11,12,14,15 to determine the type of the interrupt.
	5-9	—	Reserved, should be cleared.
	10	TBSY	(AAL2=1 and TBSY=1) indicates that AAL2 microcode cannot maintain the current high rate of AAL2 transmit data throughput. The internal structures of the AAL2 microcode are being overwritten. The host should reconfigure the system.
	11	RBSY	(AAL2=1 and RBSY=1) indicates that AAL2 microcode cannot keep up with the AAL2 receive data throughput. The CPS-PDU of the channel described in the CHNUM field may be lost. The AAL2 data rate at which the application tries to receive is too high. The host should reconfigure the system.
	12	OVF	(AAL2=1 and OVF=1) indicates that the channel's AAL2_Rx_Queue has overflowed. The CPS-PDU of the channel indicated in the CHNUM field was discarded. Either the host did not empty the AAL2_Rx_Queue quickly enough, or there are not enough RPDs in the AAL2_Rx_Queue. The host should reconfigure the system.
	13	—	Reserved, should be cleared.
	14	TXB	(AAL2=1 and TXB=1) indicates that a CPS-Packet (whose TPD[INT] = 1) has been packed into the active AAL0 buffer. See Section 1.9.1, "AAL2 Transmit Exceptions." The host may examine the AAL2_Tx_Queue and fill all available TPDs (R=0) and their associated CPS-Packets.
	15	RXB	(AAL2=1 and RXB=1) indicates that a CPS-Packet (whose RPD[INT] = 1) has been retrieved from the active AAL0 buffer. See Section 1.9.2, "AAL2 Receive Exceptions." Host may examine the AAL2_Rx_Queue and process all full RPDs (E=0)and their associated CPS-Packets
0x02	0-15	CHNUM	Channel number. Identifies the channel number of the interrupt source.

Note that exceptions not generated by the AAL2 microcode may also occur on channels that run AAL2. These exceptions will not set the entry's bit 4 (the AAL2 bit). A non-AAL2 entry will also have different field names and definitions. The host should process the exception entry as defined in the 8xxSAR specification.

1.9.1 AAL2 Transmit Exceptions

After the AAL2 microcode packs a transmit CPS-Packet, it generates an interrupt to the host and adds an entry to the exception queue if TPD[INT] and the TPI bit in the channel's AAL2_TCT are both set. The exception entry will have the AAL2 bit and TXB bit set. The AAL2 microcode also clears TPD[R] (not ready to transmit) to make the CPS-Packet available to the host.

1.9.2 AAL2 Receive Exceptions

After the AAL2 microcode retrieves a receive CPS-Packet, it generates an interrupt to the host and adds an entry to the exception queue if RPD[INT] and the RPI bit in the channel's AAL2_TCT are both set. The exception entry will have the AAL2 bit and RXB bit set. The AAL2 microcode also clears RPD[E] (not empty) to make the CPS-Packet available to the host.

1.10 Initialization of 8xxSAR for AAL2 Operation

This section describes the initialization process for ATM channels running AAL2 on the 8xxSAR. Because the AAL2 microcode is built upon AAL0 operation, the AAL0 data structures have to be in place to support AAL2 channels. The user also must configure the trap registers and the data structures specific to the AAL2 microcode itself. The Timer CU structure should also be initialized if any transmit channel uses the timer.

Example code is provided with the microcode package.

1.10.1 AAL2 RAM Microcode Memory Allocation

Because the AAL2 microcode occupies 5 KBytes of the DPR, the user should allocate 5 KBytes for program space by configuring RCCR and RMDS as follows:

- Program RCCR[ERAM] (bits 14:15) to 0b11.
- Program RMDS[ERAM4K] (bit 0) to 0b1.

1.10.2 Initialization of AAL0 Structures for AAL2 Channels

During AAL2 operation, the host does not deal directly with the AAL0 BDs and buffers associated with the AAL2 channels. However, the host does need to initialize these AAL0 structures for the AAL2 microcode to use. Therefore, for each AAL2 channel, the host should do the following:

- Allocate 64-byte buffers. The first 52 bytes of each buffer are used to hold the cell header and payload. Each buffer should start on a 64-byte-aligned address.
- Define TxBD and RxBD tables.
  - — Point each BD to a 64-byte buffer
  - — Set the interrupt bit (I=1) in each BD to trigger the AAL2 microcode

  For better performance, it is recommended to allocate 4—10 BDs for the RxBD table and for the TxBD table.
- All TCT and RCT parameters should be programmed for AAL0 operation

1.10.3 Initialization of Address Translation Mechanism

The AAL2 microcode requires the host application to use either address compression or an external CAM as the address translation mechanism. The unextended channel mode is not supported.

1.10.4 Initialization of Trap Hardware

The trap register programming depends on the specific device being used. The host should initialize the trap registers as shown in Table 1-11.

Table 1-11. Trap Register Initialization

Trap Register	850SAR Rev. A	860P, 855T, and 850 Rev B
RCTR1	0x8034	0x8034
RCTR2	0x9F09	0x9F0C
RCTR3	0x803C	0x803C

To optionally disable APC overrun events, the user should also initialize RCTR4. See Section 1.4.6, “Optional APC Overrun (APCO) Event Mask.”

1.10.5 Initialization of AAL2 Microcode

During AAL2 initialization, the host needs to do the following:

- Initialize the AAL2 parameter RAM area (64 bytes).
- Clear the AAL2_SCRATCH1 area (512 bytes).
- Clear the AAL2_SCRATCH2 area (256 bytes).
- ¥ Allocate buffers for transmit and receive CPS-Packets

  Each buffer allocated for the CPS-Packet should start on burst-aligned address (bits [28—31] = 0000). The size of each buffer should be a multiple of 16 bytes. For example, if the maximum possible packet size for a certain channel is 22 bytes, the host should allocate 32 bytes for each CPS-Packet in external memory.
- Define AAL2_Tx_Queue and AAL2_Rx_Queue for each AAL2 channel.

  Each PD in the queue should point to an empty burst-aligned buffer in external memory. The AAL2 microcode uses this pointer to access the CPS-Packet.

  — Wrap bit should be set in the last PD (W=1) in the receive and transmit PD queues.

  — The PD interrupt bit can be set (INT=1) in certain PDs. When INT=1 bit, the PD will trigger the host to write/read data to/from the AAL2_Tx_Queue/AAL2_Rx_Queue.

  For better performance, it is recommended to allocate at least 10 PDs for each queue.
- Initialize AAL2_TCT and AAL2_RCT for each AAL2 channel.
- Build the AAL2_ECT_PTR_table at the address programmed in AAL2_ECT_PTR_BASE in the AAL2 parameter RAM area.
- Initialize each channel’s ALL2_CT.
- Allocate 8 KBytes of external memory for the CRC tables. The tables must be on an 8-KByte boundary.
- In the initialization program, the user should call the C program fill_crc19(AAL2_CRC_PTR_BASE). The fill_crc19 program calculates the CRC tables and writes them to the 8-KByte block pointed to by AAL2_CRC_PTR_BASE. This program is provided in the example code.

1.10.6 Reconfiguring AAL2 Channels

When reconfiguring an AAL2 channel, the user should also clear specific parts of the AAL2 scratch area as follows:

- For transmit channels, clear the halfword (16 bits) at the offset AAL2_SCRATCH1_BASE + 0x1C.
- For receive channels, clear the halfword (16 bits) at the offset AAL2_SCRATCH1_BASE + 0x1E.

1.10.7 Initialization of Timer CU Mechanism (Optional)

If the built-in Timer CU mechanism is used, the host should do the following:

- Initialize the TIMEP field in the RCCR
- Set the TIME bit in the RCCR
- ¥ Define the wait table parameters in the AAL2 parameter RAM and clear the AAL2_TxWait_table
- ¥ Set AAL2_TCT[ET] for the channels using the Timer CU mechanism

1.11 Performance Estimation

Table 1-12 represents the estimated performance of the AAL2 microcode using 100% of the CPM bandwidth. The CPM utilization scales linearly with the data throughput. The performance numbers represent the aggregate data rate throughput of the AAL2 microcode for both transmit and receive. Note that the performance analysis was done with SDRAM; usage of EDO will result in a 5–10% performance reduction.

Table 1-12. Estimated AAL2 Performance

ATM Interface	System Freq [MHz]	Average Packet Size of 10 Bytes	Average Packet Size of 20 Bytes	Average Packet Size of 30 Bytes	Average Packet Size of 40 Bytes
Serial	50	4.5 Mbps	6.5 Mbps	7 Mbps	8 Mbps
Serial	80[1]	4 Mbps	7 Mbps	9 Mbps	10 Mbps
UTOPIA	50	5 Mbps	8 Mbps	9 Mbps	10 Mbps
UTOPIA	80[1]	4.5 Mbps	9 Mbps	12 Mbps	13 Mbps

[1] The PowerPC bus frequency is 40 MHz (half the speed of the system frequency).

In order to evaluate the total CPM utilization of a system, the user should add the expected CPM utilization needed for the AAL2 microcode to the calculated CPM utilization using the CPM utilization tool. To locate the utilization tool see the website at http://www.mot.com/netcomm/tools.

For example, if a 50-MHz system is running AAL2 at 4 Mbps (aggregate) using a UTOPIA interface with an average packet size of 20 bytes, the AAL2 channels utilize 50% of the CPM (4 Mbps out of the possible 8 Mbps). In this case, the user has 50% of the CPM bandwidth available for other purposes.

1.12 PHY Interface

The AAL2 microcode can support only T1/E1 rates per PHY line. Faster input bursts of data, such as from a 25-Mbps line, will cause the receive buffers to overflow. The AAL2 microcode can support multiple T1/E1 connections depending on the system's frequency, CPM load and ATM interface; see Section 1.11, "Performance Estimation."

1.13 Additional Recommendations

The host application should process the AAL2_Tx_queues and AAL2_Rx_queues on an interrupt-driven basis. Accessing the AAL2_Tx_queues and AAL2_Rx_queues using polling may result in lower performance of the AAL2 microcode.

Upon start-up of the AAL2 microcode or when dynamically adding new AAL2 channels, the AAL2 microcode may detect errors before synchronizing with the incoming data. For example, if the sequence number of the first CPS-PDU is set, the AAL2 microcode detects a STF error. When handling the first exception event for a new channel in the interrupt service routine, the host application should ignore these errors. All errors detected thereafter are valid.

Home Page:
www.freescale.com
email:
support@freescale.com
USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescale.com
Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com
Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 2666 8080
support.japan@freescale.com
Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com
For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com