IBM®

The information contained in this document is subject to change without notice. The products described in this document are NOT intended for use in implantation or other life support applications where malfunction may result in injury or death to persons. The information contained is this document does not affect or change IBM's product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of IBM or third parties. All the information contained in this document was obtained in the specific environments, and is presented as an illustration. The results obtained in other operating environments may vary.

INFORMATION IN THIS DOCUMENT IS PROVIDED ON AN "AS IS" BASIS WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILIY, FITNESS FOR A PARTICULIAR PURPOSE, OR NON-INFRINGEMENT. In no event will IBM be liable for any damage arising directly or indirectly from any use of the information contained in this document.

PCI Interface Support

The following list contains the PCI features and functions supported by the ATM 25Mbps PCI Controller interface:

- The ATM 25Mbps PCI Controller is a 32-bit Bus Master, which means REQ64, ACK64, and PAR64 are not implemented, nor are the optional D[63..32] and BE#[7:4] connections.
- The ATM 25Mbps PCI Controller does not implement cache support, which means SBO and SBDONE are not implemented.
- INTA is the only interrupt implemented by the ATM 25Mbps PCI Controller, which means INTB, INTC, and INTD are not implemented.
- JTAG is not implemented by the ATM 25Mbps PCI Controller, which means there is no connection to TCK, TRST#, and TMS; however, TDI is connected to TDO to maintain the shift chain in the PCI machine.
- Only a +5-V PCI interface will be implemented, rather than the +3.3-V or the PCI/Universal. The on-card regulator will power the ATM 25Mbps PCI Controller at 3.3V.
- LOCK is not supported by the ATM 25Mbps PCI Controller; hence, LOCK# is not implemented on the PCI interface.
- The power consumption on the PCI adapter is assumed to be less than 7.5 watts (5.25V @ 1.42 amp). This means that PRESENT_1# and PRESENT_2# will be grounded (logic 0). These signals originate on the adapter card and do not come from the ATM 25Mbps PCI Controller.
- The ATM 25Mbps PCI Controller is compliant with the PCI Version 2.1 specification.
- The maximum data rate of the ATM 25Mbps PCI Controller is 4 bytes per 30 ns.
- The ATM 25Mbps PCI Controller supports bus slave functions; however, this controller supports only single data transfers. The ATM 25Mbps PCI Controller activates STOP# after the first transfer.

The ATM 25Mbps PCI Controller does not support 64-bit addressing.

Introduction

General Description

The ATM 25Mbps PCI Controller provides the same functions as the ATM 25Mbps ISA Controller design but attaches to the Peripheral Component Interconnect (PCI) instead of the ISA bus. In order to maintain software compatibility, an attempt has been made to keep the same register addresses, memory structure formats, queue formats and operations as were used in the ATM 25Mbps ISA Controller design.

The ATM 25Mbps PCI Controller is a highly integrated, low cost interface controller for connection of Peripheral Component Interconnect (PCI) machines to Asynchronous Transfer Mode (ATM) networks. This CMOS VLSI device incorporates functions of the ATM Physical Layer, the ATM Layer, and ATM Adaptation Layer (AAL), and an attachment to the PCI system bus. When combined with the ATM 25Mbps PMD analog transceiver and Control/Data Memory SRAM, the ATM 25Mbps PCI Controller forms a complete solution for attachment of PCI machines to an ATM network. A serial EEPROM is required for product data and can contain user defined configuration data. The system block diagram is shown in Figure 1 on page 4.

The ATM 25Mbps PMD transceiver provides the interface between an Unshielded Twisted Pair Category 3 (UTP-3), Category 4 (UTP-4), Category 5 (UTP-5),and Shielded Twisted Pair (STP) network media and the ATM 25Mbps PCI Controller device. The ATM 25Mbps PMD provides the appropriate line signaling format for transmission and performs clock and data recovery for reception. This transceiver operates at a signaling rate of 32 Megabaud. The line coding uses a 4 of 5 code that results in a data rate of 25.6 Mbps.

The ATM 25Mbps PCI Controller can be decomposed into four major functional blocks:

- ENDEC (TC Sublayer Support)
- ATM Layer and AALs (1 and 5)
- External Memory Interface
- PCI System Interface.

The ENDEC performs the functions of the Transmission Convergence (TC) Sublayer. For transmission, it inserts the appropriate control symbol sequences, performs scrambling of the data to reduce radiation, encodes each 4 bits of data into a 5-bit symbol, serializes the symbol stream, and converts the Non-Return to Zero (NRZ) data to a Non-Return to Zero Invert (NRZI) format for transmission over the media.

For reception, the ENDEC performs the reverse operations. The NRZI symbol stream is converted to NRZ data and de-serialized, the control sequences are interpreted and removed, then the 5 bit symbol stream is converted into 4 bit data and de-scrambled. The data is then passed to the ATM Layer function.

The ATM Layer function provides for the multiplexing and de-multiplexing of user cells from various sources. The controller can support up to 4096 cell streams simultaneously. The exact number of cell streams supported is a function of the memory space available for the associated logical channel tables. An example of the memory allocation is given later in his document.

The multiplexing of cells on the transmit side is controlled by the traffic shaping function. The traffic shaping function uses a leaky bucket algorithm and places limits on the peak cell rate, the average cell rate, and the maximum allowable credit.

The controller supports eight different peak cell rates which can be divided between two service priorities. At intervals associated with each peak rate, the corresponding cell streams are analyzed to determine if there is credit to send a cell. Credits are accumulated at the average cell rate until the maximum allowable credit is achieved. One credit is consumed each time a cell is transmitted.

The de-multiplexing function is provided by separating the received cell stream into logical channels. This is accomplished by analyzing the VPI and VCI values of the received cells and directing cells with like values to a common data structure.

The AAL performs segmentation and reassembly of Common Part Convergence Sublayer Protocol Data Units (CPCS-PDUs).

The controller supports both AAL Type 5 (AAL5) PDUs and AAL Type 1 (AAL1) cells. AAL5 provides connection oriented, variable bit rate service for data traffic not requiring timing information from the data source.

The controller will append the appropriate CPCS header and trailer prior to segmentation of transmitted cells. The Cell Header is then appended and the cell is passed to the ENDEC for transmission. Received cells will have the Cell Header, and CPCS header and trailer removed before moving the cell data to system memory where reassembly occurs.

AAL1 provides connection oriented constant bit rate service for data traffic requiring timing information from the data source. The controller will append the required header information at the transmitter and remove the ATM Cell header and perform AAL-1 SAR header checking at the receiver.

The External Memory Interface provides a path for movement of user data to and from the ATM network, storage of control information associated with each ATM connection, and a communications path for controlling and synchronizing operations between the controller and the PCI system processor.

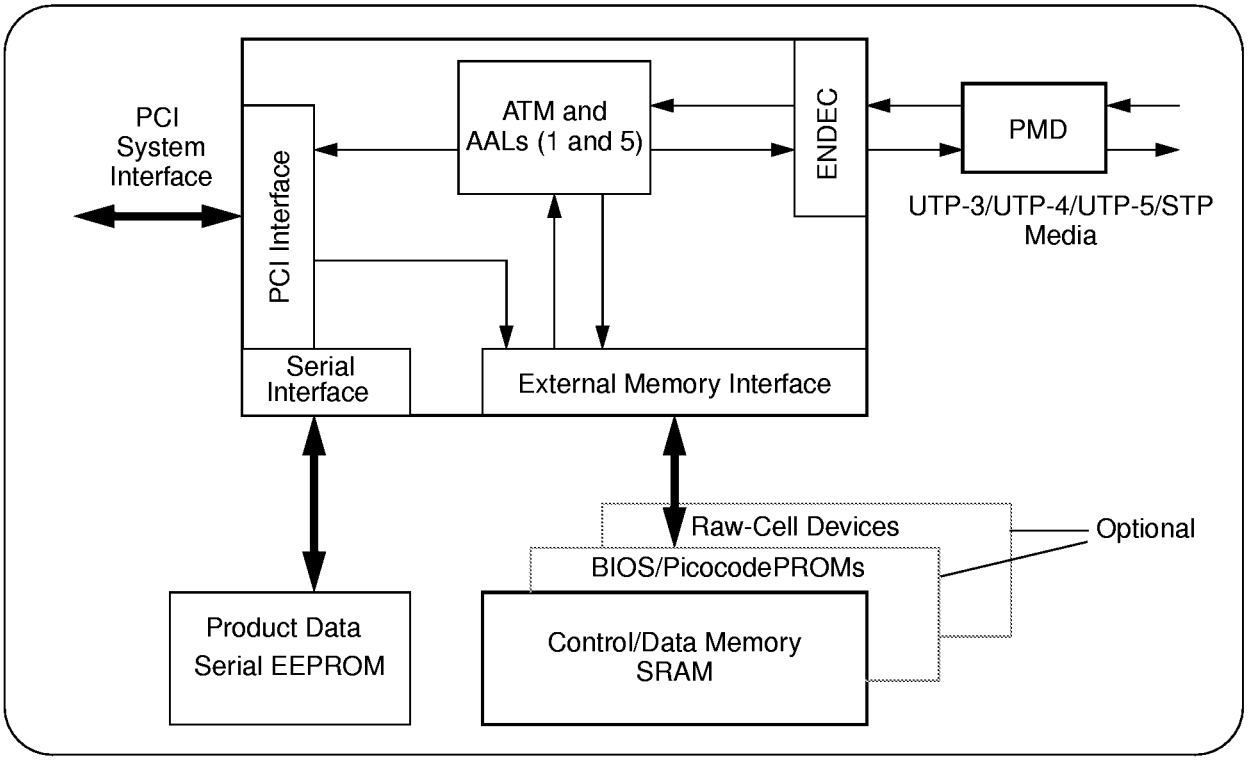
Under control of the traffic shaping function, the controller will acquire cell payload information from buffers in system memory when constructing AAL-5 cells for transmission.

The controller supports a linked list of buffers for each logical channel. In this way, the system can construct a data structure consisting of one or more USER-PDUs in a series of non-contiguous data buffers. This buffer structure is passed to the controller which removes the USER-PDU from the buffers and builds the CPCS-PDUs for transmission. For received cells, the controller will acquire buffers from a free pool maintained by the system and build a linked list of buffers to contain USER-PDUs.

The PCI system interface is designed for attachment to the PCI System Bus. A serial interface to an external EEPROM (required) that contains product and configuration data, as well as default PCI configuration parameters is provided.

The controller acts as a PCI Bus Master for transferring data and control information between the system memory and the internal controller storage elements, and as a Bus Slave for receiving system commands and providing status to the system software.

Figure 1. System Diagram



Features

- Programmable Picoprocessors control most functions
  - Full AAL-5 support
    - Header insertion and removal
    - Pad insertion and removal
    - Protocol checking
  - AAL-1 support
  - Linked list of system buffers per logical channel
    - Gather read and scatter write support
  - Extensive performance statistics and monitoring
  - Supports Backward Explicit Congestion Notification (BECN)
  - 4096 concurrent segmentation and reassembly processes
  - GFC decoding
  - Operations and Maintenance (OAM) cell processing
- Hardware controls other functions:
  - 25.6-Mbps Transmission Convergence (TC) Sublayer
  - High and low priority service for transmitted cells
  - Source traffic shaping (peak and average rate control)
  - Attachment to PCI bus, complete with on-chip drivers
    - Complete Bus Master capability

Chip Functions

Figure 2 shows the different functions performed by the controller chip. A detailed explanation of these functions is found in “Functional Description” on page 8 for details.

Receive Functions

ATM Logical Channel Resolution

A logical channel is an abstraction that is used to identify a stream of information as an entity. Segments of an information stream have a common logical channel identifier. In this way, distinct information streams may be segmented, multiplexed together for transmission, and later separated to reconstruct these individual information streams. The scope of this logical channel is from one end-user, through various intermediate nodes, to another end-user. See Figure 7. on page 14 for details.

ATM Logical Channel Validation

Logical Channel (LC) Validation is the process of validating a channel. A channel is initialized during set-up and is checked during LC Resolution to assure the channel is valid. Invalid channels are programmed into the VP and Logical Channel Identifier (LCI) Tables upon initialization. See "ATM LC Validation" on page 13 for details.

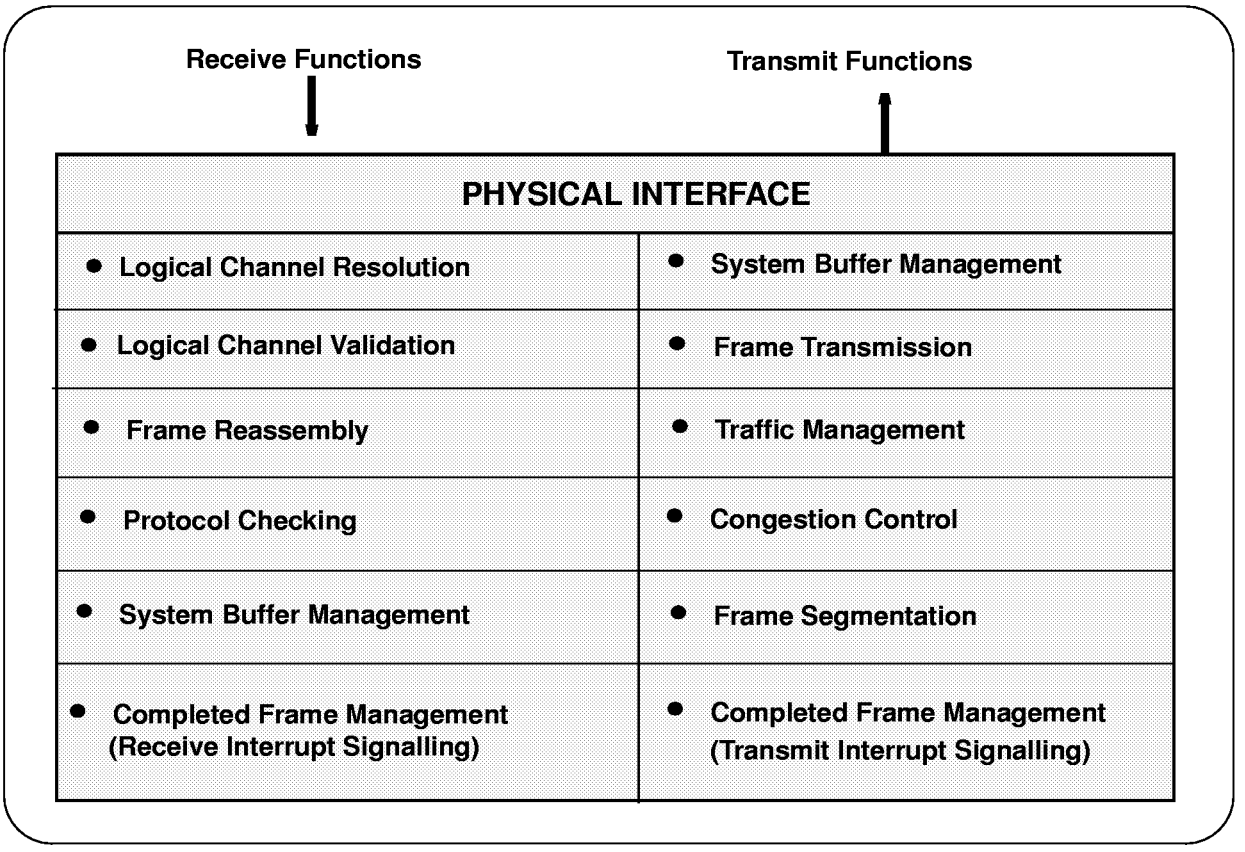
Temporary Cell Storage

After LC Resolution and LC Validation, valid cells are stored in either Receive Cell Buffers or AAL-1External Raw-Cell Device depending on the AAL type. The LC number is stored with the cell payload to identify the cell when it is removed for AAL Protocol Processing. See "Cell Storage" on page 13.

AAL Protocol Processing

AAL Protocol Processing refers to the process of *stripping* the trailers and pad units from the raw cells as they are retrieved from Receive Cell Buffers. See "Protocol Processing/Checking" on page 13 for details.

Figure 2. Chip Functions



AAL Protocol Checking

Protocol Checking refers to a series of checks performed on the cell data to ensure proper transmission to the system buffers. These checks verify the integrity of the control, the length, and the cyclic redundancy check (CRC) data in the cell. See "Protocol Processing/Checking" on page 13 for details.

Frame Reassembly and System Buffer Management

The protocol processed and checked cells are reassembled into system buffers. These system buffers may be link listed to form a frame. See "Frame Reassembly and System Buffer Management" on page 5 for details.

Completed Frame Management

Completed Frame Management is the process by which the controller chip interrupts the system to indicate that a frame has been reassembled. See "Completed Frame Management" on page 6 for details.

Transmit Functions

System Buffer Management

The System Buffer Management function assumes that the system has built a linked list of frame descriptors in system memory. Once this structure is built, the controller chip uses the Transmit Frame Descriptor to locate each frame of data to be segmented and transmitted by the controller. Each transmitting LC has its own linked list of descriptors for data frames waiting to be segmented into ATM cells and transmitted by the controller chip. The system is responsible for linking frame descriptors together in the same order that their associated frames are to be transmitted. See "AAL5 System Buffer Management" on page 22 for details.

Transmitting a Frame

A frame can be transmitted once an LC is established. The system utilizes a Transmit Ready Queue (TRQ) which is used to pass frame descriptors to the controller. The TRQ queue contains both a 13 bit LC pointer and a 32-bit Frame Descriptor Address (FDA). Once the system writes the TRQ data, the controller detects that a frame is ready to be transmitted. See "Transmitting an AAL5 Frame" on page 26 for details.

Traffic Management

Traffic management involves peak and average metering of LCs to control the rate at which they are able to transmit ATM cells. See "AAL5 Traffic Control" on page 22 for details.

Congestion Control

Some networks provide a backward explicit congestion notification (BECN) scheme for congestion control. Networks that support BECN communicate congestion through OAM F5 end-to-end cells. When a congestion notification OAM cell is received, the peak cell transmission rate of the specified LC is throttled. See "AAL5 Congestion Control" on page 25 for details.

Frame Segmentation

Frame Segmentation refers to the appending of CPCS Pad and Trailer onto the USER-PDU as well as separating the resulting CPCS-PDU into individual 48-byte payloads and adding a 5-byte header to the front to generate the standard 53-byte cell. See "AAL5 Frame Segmentation" on page 25 for details.

Completed Frame Management

Completed Frame Management is the process by which the controller chip interrupts the system to indicate that a frame has been transmitted and that the system buffers are empty. See "AAL5 Management of Completed Frames" on page 28 for details.

Chip Compatibility and Compliance

At the network side, the controller complies with the ATM standards and recommendations defined by ITU-TS (formerly CCITT), ANSI, and the ATM Forum. The following is a list of standard documents on which the controller design point is based:

- ITU Rec. I-361–B-ISDN ATM Layer Specification
- ITU *Rec. I.362–B-ISDN ATM Adaptation Layer (AAL) Functional Description*
- ITU Rec. I.363–B-ISDN ATM Adaptation (AAL) Specification
- ITU Rec. I.371–B-ISDN Traffic Congestion Control
- ITU Rec. I.413–B-ISDN User-Network Interface
- ITU Rec. I-432–B-ISDN User-Network Interface— Physical Layer Specification
- ITU Rec. I-610–OAM Principles of B-ISDN Access
- ANSI T1.ATM–199x Draft, Broadband ISDN— ATM Layer Functionality and Specification
- ANSI T1.CBR-199x Draft, Broadband ISDN— ATM Adaptation Layer for Constant Bit Rate Service Functionality and Specification
- *ATM Forum 93-620R2–ATM User-Network Interface Specification* Version 3.1 (April,1994)
- ATM Forum 94-10008R5—Physical Interface Specification for 25.6 Mbps over Twisted Pair Cable

At the system interface side, the controller complies with the following PCI -bus architecture:

- *PCI Local Bus Specification* Revision 2.0. Dated April 30, 1993.

Functional Description

The controller provides Transmission Convergence (TC) Sublayer, ATM Layer, and AAL5 and AAL1 facilities for the transmission and reception of ATM cells over a 25.6-Mbps UNI. Cell streams from as many as 8191 simultaneous sources are multiplexed for transmission and de-multiplexed for reception by the controller's ATM Layer function of the controller.

A more detailed description of the Transmit and Receive processes are contained in the remaining sections of this chapter. Both AAL1 and AAL5 are covered in this chapter.

Receive Data Flow

"Cell Arrival and Processing" explains the overall flow of Receive Data Traffic through the controller. Subsequent subsections add details to this flow.

To receive a frame, the data structures in Table 1 are required. All of these structures are built by the system software—except the Receive Ready List (RRL), which is built by the controller. The functions performed by each of these structures are described in detail in the following subparagraphs.

Table 1: Receive Data Structures

Structure Name	Location
Buffer Descriptors	Located in the system memory
VP Table	Located in the controller Control/Data memory
LCI Table	Located in the controller Control/Data memory
LC Table	Located in the controller Control/Data memory
Receive Free List	Located in the system memory
Receive Ready List	Located in the system memory

Cell Arrival and Processing

As cells arrive from the network, the ATM header is checked for errors. Cells with ATM header errors are counted and discarded. If the header is correct, then the Virtual Path Identifier (VPI) and Virtual Channel Identifier (VCI) values are examined to determine if the channel is valid. Cells with an invalid VPI or VCI are counted and discarded. The ATM header is removed from cells with valid channel information. The AAL type is determined, and further processing is modified based on this AAL type.

AAL5 Processing

For AAL5, the raw cell (with ATM HEC removed and the Logical Channel [LC] number attached) is stored in AAL5 Receive Cell Buffers in Control/Data memory. The Receive Picoprocessor moves this raw cell into system memory buffers to reassemble the frame, and performs further AAL5 processing while the reassembly is taking place. This AAL5 processing includes the removal of the CPCS Trailer and Pad information as required. During the reassembly process, each cell is checked as required for protocol violations. If an AAL5 protocol violation occurs, the reassembly process is terminated and the status information associated with reassembly indicates the form of the error. The controller obtains system data buffers as required from a free pool of buffers maintained by the system processor, and links these buffers together as required by the reassembly process. When the reassembly process is complete, status is written into the Receive Buffer Descriptor, and the system processor is signaled through the use of a bit in the System Interrupt Status Register (SISR).

AAL1 Processing—System Memory

For AAL1 to system memory traffic, the raw cell (with HEC removed) is stored in AAL1 Receive Cell Buffers in Control/Data memory. The decision to store the received cell in the system memory is based on the contents of the LC Identifier (LCI) Table. The Receive Picoprocessor moves this raw cell into system memory, giving it priority over AAL5 cells. The received AAL1 cell can also be directed to

a storage device other than the system memory, such as an External Raw-Cell device, which is memory-mapped into the System Memory Address Space. The controller performs sequence checking and synchronous time stamp recovery for received cells. Any additional processing of errors or time stamps must be provided by the User.

AAL1 Processing—Break Out Interface

The raw cell, with HEC removed, is transferred from the ATM network to the Raw-Cell interface. This interface uses the data lines of the External Memory interface and special control signals to transfer received AAL1 cells to an external device. The decision to store the received cell in an external Raw-Cell device is based on the contents of the LCI table. The user must process the received AAL1 cell and perform any required checking. The Raw-Cell interface provides a bypass of the controller AAL processing and a fast path around the system memory. No interrupt or status information is provided to the system processor concerning these cells.

OAM F4 Receive Processing

To receive an OAM F4 cell, the system software must establish an LC entry in the LC Table for the desired VP and VC3 used to carry the OAM F4 cells. The format for the LC table entry is the same as for AAL5 cells. When an OAM F4 cell is received from the network and processed by the controller, the cell is moved to a buffer located in the Host memory. The associated buffer descriptor is then linked onto the RRL4 and a host interrupt is generated. All OAM F4 received cells are linked to RRL4 regardless of the setting in the LC Table. All cells received on this VP/VC are assumed to be OAM F4 cells.

AAL5 Receive Processing

The following subsections pertain to AAL5 receive processing. Some subsections also pertain to AAL1 as noted.

ATM Logical Channel Resolution

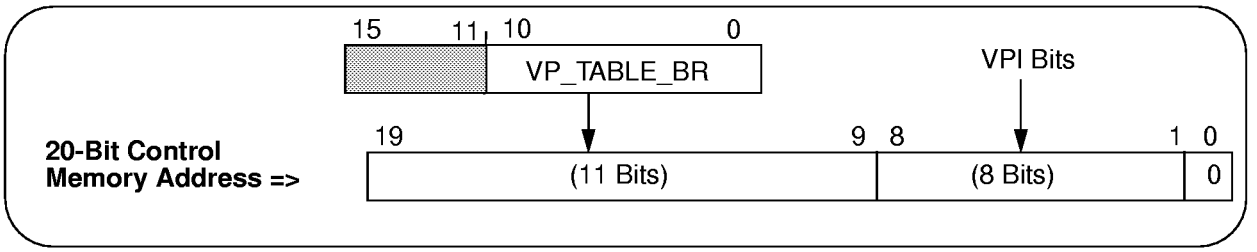
This subchapter applies to AAL5, AAL1 to system memory, and AAL1 to Break Out interface traffic.

An LC is an abstraction that identifies a stream of information as an entity. Segments of an information stream have a common LCI. In this way, distinct information streams may be segmented, multiplexed together for transmission, and later separated to reconstruct the individual information streams. The scope of this LC is from one end user, through various intermediate nodes, to another end user. In an ATM network, there is a relationship between these LCs and the VPI and VCI values used on each intermediate link taken along the path between end nodes. The addressing capability achieved through the combined use of VPI and VCI values generally exceeds the end node's LC capacity. Therefore, the end node must convert the VPI and VCI values to a corresponding LCI. This process is referred to as Logical Channel Resolution.

The resolution of the LCI from the ATM cell header is accomplished by using two Table Look-up operations. The first Table Look-up operation uses the Virtual Path (VP) Table. The second Table Look-up operation uses the LCI Table. The LC resolution produces an LC number that locates the LC entry in the LC Table. See Figure 6 on page 12 for a summary of the LC resolution process.

The VP Table contains one entry for each of the 256 possible VPI values defined at the User Network Interface (UNI). The address for accessing the VP Table is generated by concatenating the VP Table Base Address with the VPI bits from the cell header (see Figure 3). Each entry in the VP Table is 2 bytes in length and is located on an Even boundary. As shown in Figure 3, Bit 0 of the generated address is set to b'0' and Bits 15:11 of the VP Table Base register are not used.

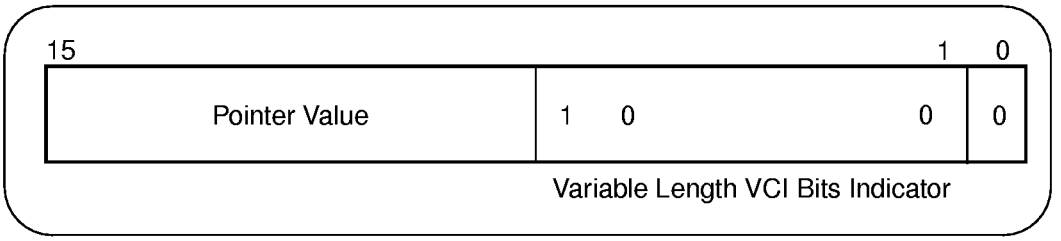
Figure 3. VP Table Address Generation



Each entry in the VP Table contains a variable length pointer value and a variable length VCI bit selection field, as shown in Figure 2. The LCI Table Index value is generated by concatenating the pointer value with the VCI from the ATM cell header. The number of VCI bits used is indicated by the VCI bits selection field; the total length of the index is 15 bits. This index is used to locate the LC numbers in the LCI Table that are associated with the received VPI/LCI.

The right-most bit of the pointer value is determined by scanning the entry bits from right to left, starting at bit position 1 until the first b'1' bit is found. The pointer value is located to the left of this first 1 bit. The number of bits in the VP Table entry to the right of the pointer value (excluding bit position 0, which is the VCI selection field) determine the number of VCI bits that are needed to generate the LCI index. A typical VP Table entry is shown in Figure 2. Bit 0 is always equal to b'0' for the controller.

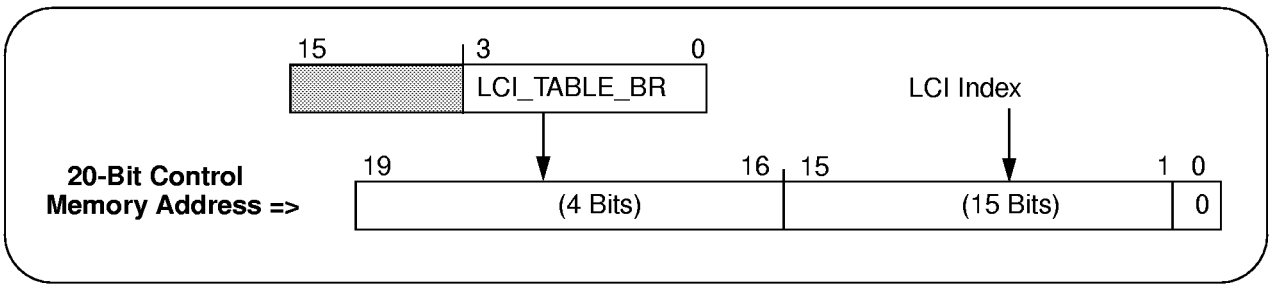
Table 2: VP Table Entry



The second Table Look-up operation, using the LCI Table, uses the LCI index value generated in the previous process to build the LCI Table Address.

The LCI Table Address is generated by concatenating the LCI Base register and the LCI index.

Figure 4. LCI Table Address Generation

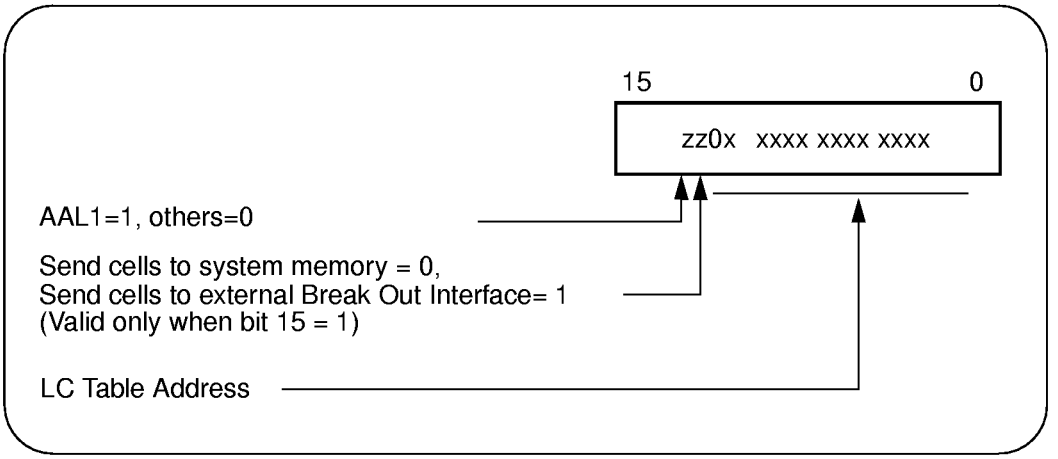


The LCI Table contains up to 32,768 entries, each having 16 bits. This table is located in the controller's Control/Data memory, and is initialized by the system software when the controller is initialized.

Each entry contains information pertaining to an individual LC. (See Figure 5 for bit definitions of the LCI Table entries.) Bits 12:0 contain the Logical Channel Table Address. Bit 13 is unused and must

be set to b'0'. Bit 14 indicates where to route the cell. Bit 15 determines if the VP/VCI received is for an AAL1 cell. For invalid LCs, a value of X'0000' is programmed into the table location.

Figure 5. LCI Table Entry



A summary of the LCI Table Address generation using the VP Table entry and the VCI bits received in the cell header is given in Figure 6 on page 12. This method of generating the LCI Table Address allows a variable number of Virtual Channels (VCs) to be associated with each VP. Figure 6 on page 12 shows the VP Table, the LCI Table, and the LC number generation process. After the LC number has been obtained from the LCI Table, it is used to access the LC Table each time information is needed to reassemble or segment cells for a given VP/VC.

Figure 6. VP/VC Resolution Summary

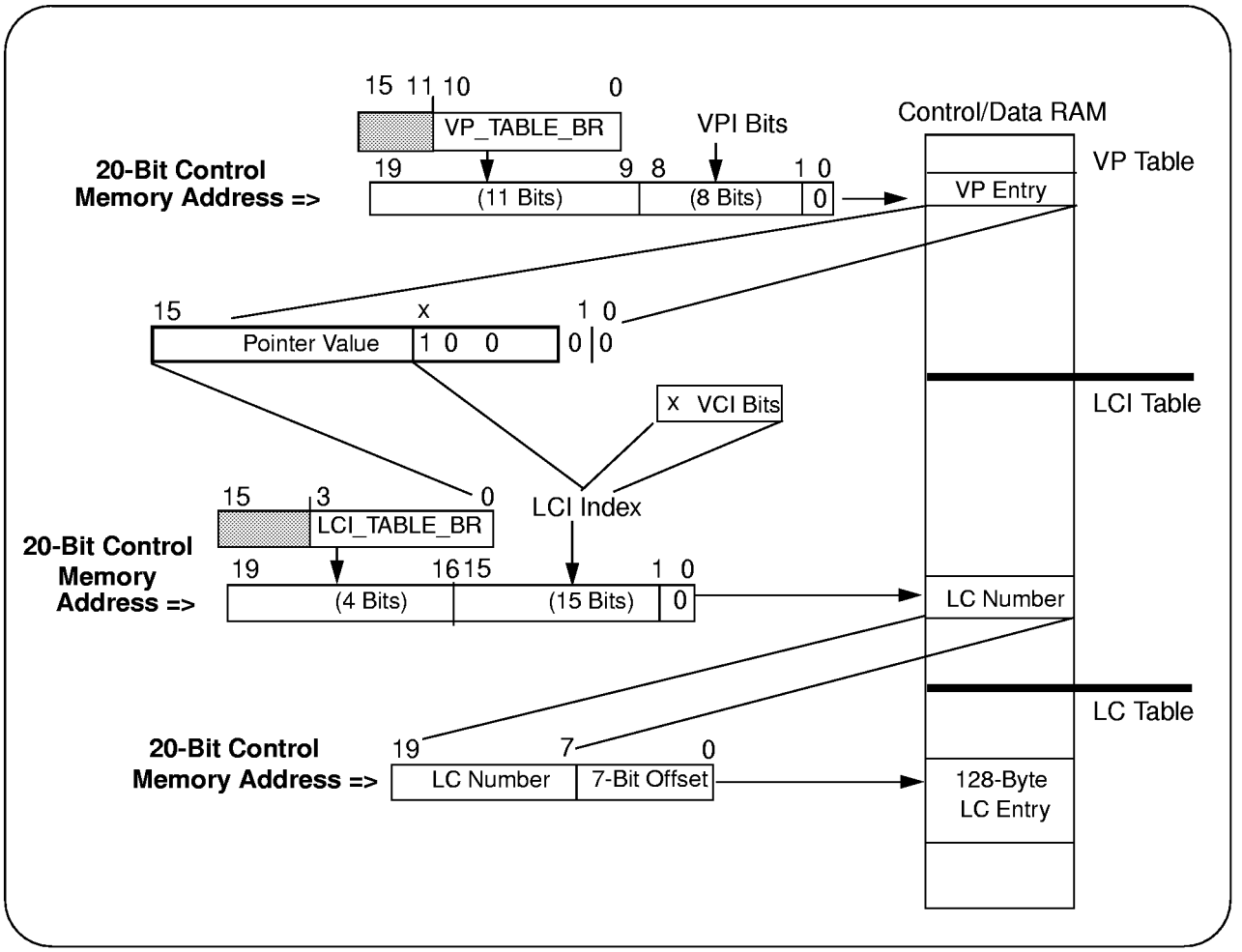


Table 3: Number of VCI Bits Used in Address Generation

VP Table Entry	Number of VCI Bits in Address Pointer	VP Table Entry	Number of VCI Bits in Address Pointer
xxxx xxxx xxxx xxx1	Reserved	xxxx xxx1 0000 0000	8
xxxx xxxx xxxx xx10	1	xxxx xx10 0000 0000	9
xxxx xxxx xxxx x100	2	xxxx x100 0000 0000	10
xxxx xxxx xxxx 1000	3	xxxx 1000 0000 0000	11
xxxx xxxx xxx1 0000	4	xxx10000 0000 0000	12
xxxx xxxx xx10 0000	5	xx10 0000 0000 0000	13
xxxx xxxx x100 0000	6	x100 0000 0000 0000	14
xxxx xxxx 1000 0000	7	1000 0000 0000 0000	15

Table 4: Receive Control/Data Memory Data Structure Sizes

Data Structure	Entry Size	Number of Entries (Min.)	Number of Entries (Max.)
VP Table	16 Bits	256	256
LCI Table	15 Bits	2	32,768
LC Table	128 Bytes	0	< 8,192[a]

a. Actual number based on User-configured memory.

ATM LC Validation

This subchapter applies to AAL5 traffic only.

LCs are setup by software at initialization by programming the VP and LCI Tables. A channel is invalidated by programming the LCI Table with zero for a particular VP/VC combination.

Any cell received by the controller with an invalid VP/VC combination is discarded and is counted in a mis-routed cells counter. This counter is used in performance monitoring statistics for the network.

Cell Storage

All AAL5 cells are temporarily stored in receive cell buffers located in the Control/Data RAM. These cells are stored in the order they are received without regard to their LC number. The LC number, the Bit Interleave Parity-16, the ATM cell header (excluding the HEC byte), and the 48-byte cell payload are stored in the Receive Cell Buffer.

The number of AAL5 receive buffers and their location in the Control/Data RAM is controlled by the system software.

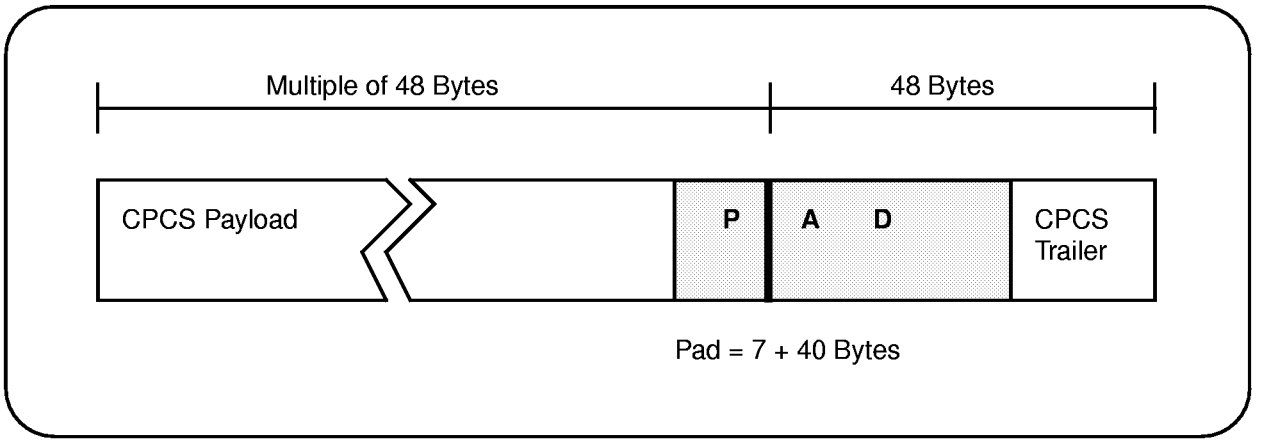
Protocol Processing/Checking

The controller's Receive Picoprocessor is interrupted when an AAL5 cell is placed in an AAL5 Receive Cell Buffer. The picoprocessor reads the cell header from the cell buffer and begins protocol processing.

The ATM cell header contains a 3-bit payload type of which one of the bits is the ATM user-to-user Identifier. When the ATM user-to-user Identifier is a b'1', it is the last ATM cell in the CPCS-PDU, and contains the CPCS Trailer. The last 8 bytes of this cell make up the CPCS Trailer. The Trailer is removed from the CPCS-PDU, and is not passed along to the system data buffers. The other 2 bits of the Payload Type Identifier (PTI) field indicate congestion and Operations and Maintenance (OAM) cells. For a complete description of the PTI field, refer to *ITU Rec. I-361–B-ISDN ATM Layer Specification*

As each cell is processed for a particular CPCS-PDU, an accumulated length value is increased by the length of a cell payload. When the last cell is received, the length in the CPCS trailer is subtracted from the accumulated length to determine the Pad length. The maximum length of a CPCS-Service Data Unit (CPCS-SDU) is 65,535 bytes. When transmitting, the CPCS Trailer must reside in the last 8 bytes of the last ATM cell. If the entire 8-byte trailer does not fit into this cell, Pad must be added to complete the 48-byte cell payload. The CPCS trailer is then placed in the last 8 bytes of the next ATM cell preceded by 40 bytes of Pad. In Figure 7, the Pad field is contained in both the last and next to last cell in the frame. The Pad contained in the last received cell is not passed along to the System Receive Data Buffers.

Figure 7. CPCS-PDU With Pad



There are two checks performed on the AAL5 trailer: the Length Field Check and the Cyclic Redundancy Check (CRC). Presently, the Control field is reserved, and the only valid encoding is all 0s; however, this field is not checked by the controller. The protocol check compares the Length field and the accumulated length for the particular CPCS-PDU. If the values of the lengths differ, a length violation has occurred, and the CPCS-PDU is aborted. A length violation is also detected when the accumulated length of a CPCS-SDU is greater than the maximum length set for that VP/VC. This value is set in the LC Table. When a length violation occurs, the CPCS-PDU is aborted, and anymore cells received for the frame are discarded. The second check compares the received CPCS-PDU's accumulated CRC with the received CRC located in the CPCS-PDU's trailer. If the CRCs differ, a CRC error has occurred, and the CPCS-PDU is aborted.

When a CPCS PDU is aborted, the Status field in the System Receive Buffer Descriptor, located in system memory, is marked as aborted, and a reason for the abort is included in the abort status.

The Reassembly Timer function is also supported by the controller. The Reassembly Timer function limits the amount of time a particular LC is receiving ATM cells for a CPCS-PDU. If the CPCS-PDU is not received within the specified amount of time, the CPCS-PDU is aborted and the status is posted in the System Receive Buffer Descriptor.

Frame Reassembly and System Buffer Management

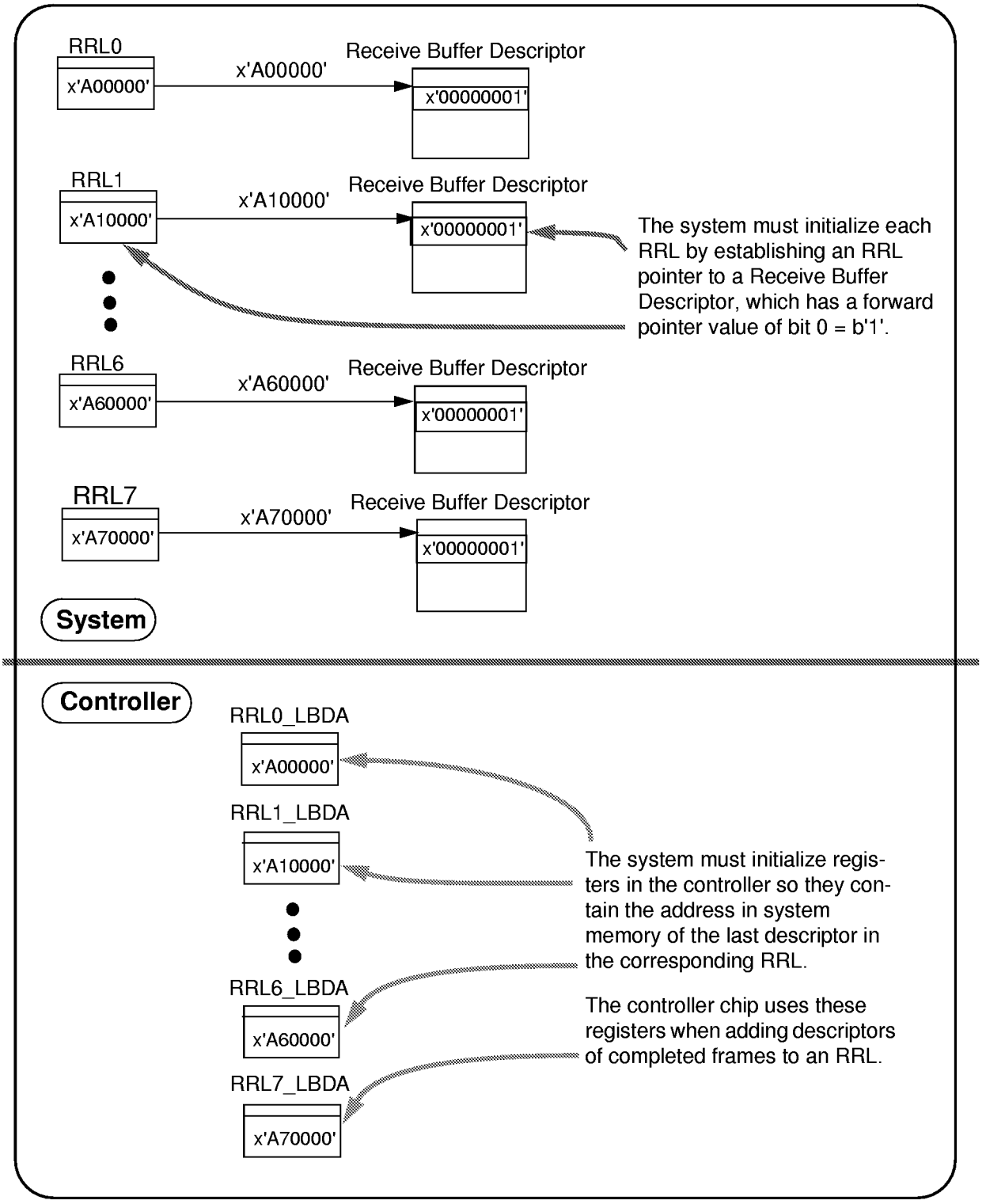
There are two system software structures required in the Receive process: the data buffers and the eight RRLs. Both of these structures are built by the system software in system memory upon initialization. The available Receive System Buffers and their descriptors are linked together via their forward pointers to form a Receive Free List (RFL), such as the free buffer-pool. The last buffer descriptor must have a forward pointer with bit 0 set to b'1' to indicate that it is the last descriptor in the free list. The eight RRLs are used by the controller to pass information to the system software concerning frames which have been received and their status. Each RRL is a linked-list of buffer descriptors with the capability of linking multiple frames.

The Free Buffer Pool is generated by the system software to be used by the controller for storing received network data. As data is received, the controller obtains free buffers from this pool and links them together, via their descriptors, to form a linked-list of buffers containing the received frame. After the complete frame has been received, the controller chains the list of buffers containing the frame to the appropriate RRL and interrupts the system software to present status for the received frame. After the system software has processed the received frame, the Receive Buffers are returned to the Free Buffer Pool.

There are eight RRLs located in the system memory to allow completion notification to pass from the controller to the system software. Pointers to these lists

are maintained both in the controller and the system memory. The controller appends buffer descriptors to one of these lists when a complete frame has been assembled in the system buffers. The system software removes the descriptors from the list after they are processed and returns the buffers to the Free Buffer Pool. All received OAM cells are appended to RRL4, which is reserved for OAM traffic. Non-OAM cells received and placed in either the AAL5 or AAL1 received cell buffers are appended to the RRL specified in the LC Table.

Figure 8. The System's Role in the Initialization of the RRL Pointers



Completed Frame Management

The responsibility for processing the received frames is divided between the controller and the system software. The following subparagraphs describe the procedures by which both the controller and the system software inter-operate to manage the flow of cells from the network to the system buffers.

The controller chip receives the ATM cells from the network and reassembles them into frames, using the system buffers taken from the Free Buffer List. The fact that the frame has been reassembled in system memory is conveyed to the system software using the SISR. The SISR contains a status bit for each of the eight RRLs maintained by the system software. By having multiple RRLs, the system software can assign priorities to the received frames. Upon reassembly completion, the controller attaches the chain of system buffers containing the reassembled frame to one of the eight lists as defined by the associated entry in the LC Table. Then, the controller sets the appropriate RRLx bit in the SISR, interrupting the system processor. Cells placed in the OAM Receive Cell Buffers are attached to RRL4.

Once the system software reads the SISR in the controller and discovers that frames have been received from the network, it is the software's responsibility to propagate through the RRL and process all frames until it reaches *the end of the RRL*. The system reaches the end of the RRL when the Forward Pointer Address field, Bit 0, in the buffer descriptor, is equal to b'1'. (Valid addresses are located on a 4-byte boundary.) "Processing a Frame," on page 19 for details of the roles of system software and the controller in frame processing, and the remaining steps required in the frame processing sequence.

When the System software has reached the 'end of RRL,' it will retain the address of the last Receive Buffer Descriptor processed for the RRL in system memory.

The system software must update its copy of the RRL pointer. This pointer indicates the last buffer descriptor processed as a result of the controller interrupt. By comparing the system software's copy of the pointer with the controller's associated RRL register, the system software can determine if additional received frames are available for processing.

It is the system software's responsibility to update the RFL as buffers in system memory are released. The system software must update the forward pointer of the descriptor which is currently at the end of the RFL to point to the first of a list of descriptors being added (descriptors may be added to the RFL one at a time or several at a time). The last descriptor in this list must have a forward pointer indicating "End of List" (a forward pointer value of bit 0 = b'1').

An example of the software's initialization process is shown in Figure 8 on page 16. Each of the eight pointers to the Receive Buffer List is initialized with the address of a buffer descriptor, and the forward pointers located in the buffer descriptors are set to b'1'. The "Controller" section in the lower half of the figure shows the controller RRL registers, which are also initialized with these same address values by the system software.

The controller maintains a view of the system's Receive Free List (RFL) via the Start of Receive Free List (SRFL) register. This 32-bit register, located in the controller, must be initialized by the system software to tell the controller where the Free Buffer List begins in system memory. When removing a buffer from the Free Buffer List, the controller updates the SRFL with the address of the next descriptor in the Free Buffer List. This address is located in the forward pointer field of the descriptor being removed. If the forward pointer bit 0 is b'1', the controller interrupts the system and indicates that the end of the Free Buffer List has been reached. The system must read the SRFL register to determine where the controller thinks the Free Buffer List ends, add buffers to the list, and re-initialize the SRFL before the controller can continue processing cells. Valid buffer descriptors must be located on 4-byte boundaries.

The controller uses one of the eight RRLs, located in the system memory, to notify the system software that a complete frame has been reassembled. These lists consist of a linked-list of Receive Buffer Descriptors whose associated buffers contain the received frame. The first descriptor of the chain associated with the received frame contains the status and length of the frame. Each descriptor in the

list points to the next descriptor until a pointer with Bit 0 set to a b'1' is reached, marking the end of the chain. Multiple received frames can be linked together while they are waiting to be processed by the system software. The eight RRLs form eight status queues, which contain the status of the received frames. The processing of these eight lists can be prioritized by the system software.

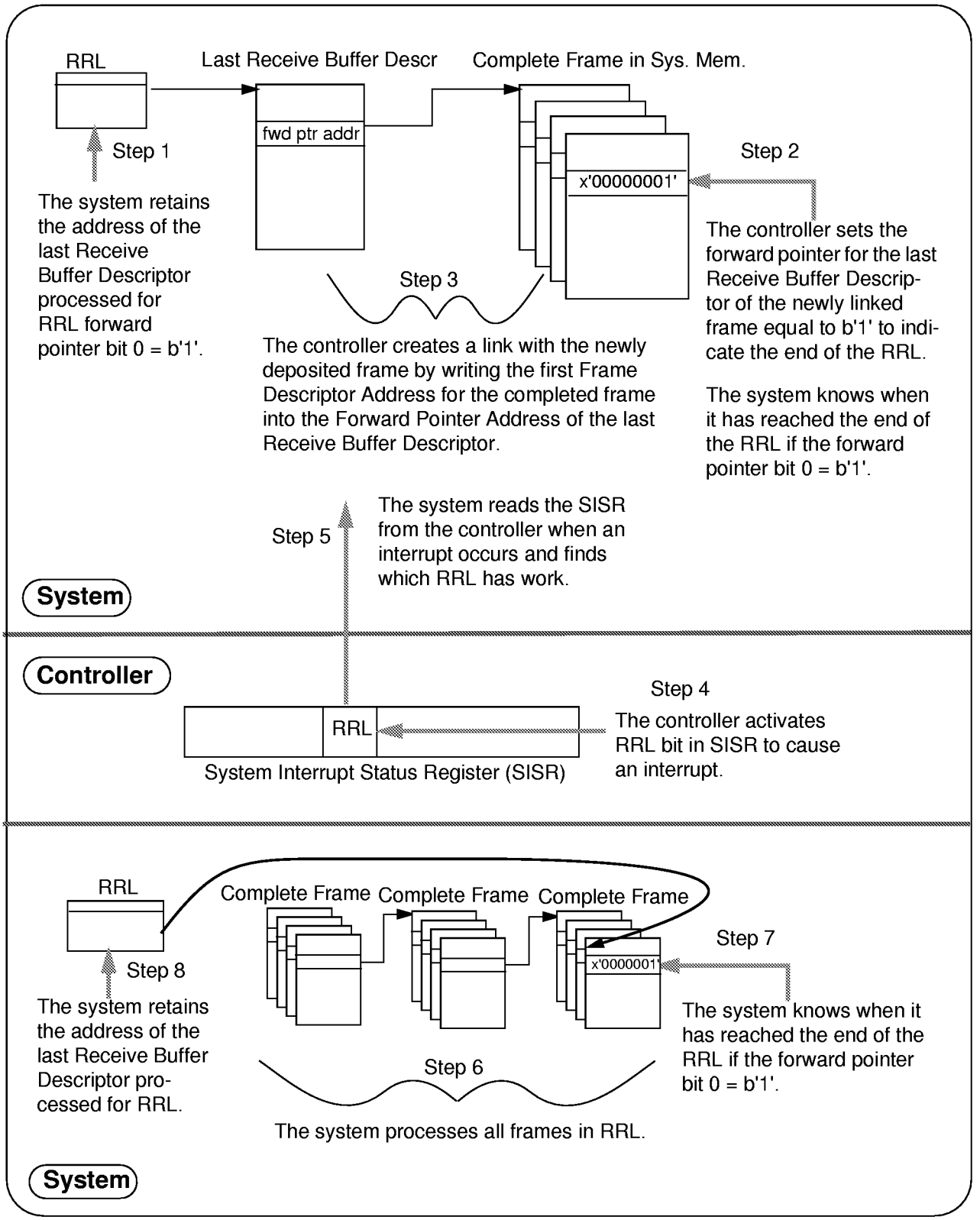
Pointers to the last processed entry of each of the eight RRLs are kept by the system software. The controller RRL registers contain the address of the last entry added to each list by the controller. The system software initializes the controller RRL registers and its own copy with the address of a buffer descriptor. At initialization, these descriptors (which may point to empty buffers), have their forward pointers bit 0 set to b'1'. These first eight buffers are not used by the controller to store data. Their descriptor addresses serve only as primers, and provide a starting descriptor address that the controller can use when linking the first received frame's descriptors.

To add an entry to one of the RRLs, the controller uses the associated RRL register to locate the last entry on the list, change the forward pointer of the entry to point to the first descriptor of the received frame, and set the forward pointer, bit 0, of the last descriptor of the chain to b'1'. The intermediate descriptors associated with this frame were linked, using their forward pointers, as the frame was being reassembled. After the controller links the received frame's descriptors to one of the eight RRLs, it sets a bit in the SISR to indicate the need to process received frames and then generates a system interrupt. RRL selection is based on a field contained in the LC Table.

The System is interrupted when a complete frame has been transferred from the network to system memory. A complete frame consists of one or more Receive Buffers. Since multiple Receive Buffers may exist for a complete frame, the controller chip is responsible for linking the Receive Buffers for a complete frame prior to the system's involvement.

For each buffer that the controller uses to reassemble a frame, it writes a status word into the associated buffer descriptor. The descriptor associated with the first buffer of the frame has the Start of Frame (SOF) bit set to b'1'. The descriptor associated with the last buffer of the frame has the End of Frame (EOF) bit set to b'1'. All intermediate buffers have both bits set to b '0'. If the received frame is fit into a single system buffer, that buffer's descriptor has both the SOF and EOF bits set to b'1'. The system software can use these status bits to locate the SOFs and EOFs when multiple frames are chained together within a single RRL.

Figure 9. Processing a Frame



AAL1—System Memory

The following subsections pertain to AAL1 to system memory traffic.

ATM Logical Channel Resolution

See "ATM Logical Channel Resolution" on page 9 for details.

ATM Logical Channel Validation

See "ATM LC Validation" on page 13 for details.

Cell Storage

Received AAL1 cells routed to the system memory are temporarily stored in Receive Cell Buffer Queues located in the Control/Data RAM. These cells are stored in the order they are received without regard to their LC number. The LC number, the Bit Interleave Parity-16, the ATM cell header (excluding the HEC byte), and the 48-byte cell payload are stored in the Receive Cell Buffer.

The number of AAL1 Receive Buffers, and their location in the Control/Data RAM, is controlled by the system software.

Protocol Processing/Checking

The controller Picoprocessor is interrupted when an AAL1 cell is placed in an AAL1 Receive Cell Buffer. The Picoprocessor reads the cell from the cell buffer and begins protocol processing.

When an AAL1 cell is received, it is checked for sequence and structure pointer errors. The controller also reassembles the Synchronous Remote Time Stamp (SRTS) data. As a result of this processing, bits are set in the AAL1 Receive System Data Structure.

The controller will process the AAL1 SAR header and the Structure Pointer, if one exists. The AAL1 SAR header processing includes Sequence Number Check, CRC 3 Check, and SAR Parity Check.

If the AAL1 Structure Pointer exists, the controller will perform a Parity Check on the Structure Pointer.

Cell Transfer to System Memory

For cells transferred to system, the controller will construct the AAL1 Receive Data Structure. This data structure is transferred to system memory using information contained in the Receive AAL1 portion of the LC Table.

All AAL1 traffic crossing the system bus is sent to a destination address as indicated by the receive portion of the LC Table.

AAL1 Processing—Break Out Interface

The following subsections pertain to the transfer of AAL1 traffic to the Break Out Interface.

ATM Logical Channel Resolution

See "ATM Logical Channel Resolution" on page 9 for details.

ATM Logical Channel Validation

See "ATM LC Validation" on page 13 for details.

Cell Transfer to AAL1 Break Out Interface.

For AAL1 cells transferred to the Break Out Interface, the controller strips the HEC from the received cell and transfers the remaining cell header bytes, along with the 48 byte payload, to the AAL1 Break Out Interface. No AAL1 protocol processing is performed on these cells and no interrupts to the system are generated. The transfer of cells to the Break Out Interface is controlled by a collection of control signals as described in "External Memory Interface," on page 37; the Receive operation is shown in Figure 23. on page 42.

Transmit Data Flow

To transmit a frame, the data structures in Table on page 20 are required. The Buffer Descriptor Structure and the Transmit Ready Queue are built by the system software and are used by the controller during the Transmit process. The Transmit Complete List Structure is built by the controller and is used by the system software. The functions performed by each of these structures is described in detail in the following subparagraphs.

Table 5: Transmit Data Structures

Structure Name	Location
Buffer Descriptors	Located in the system memory
Transmit Ready Queue for AAL1	Located in the controller Control/Data memory
Transmit Ready Queue for AAL5	Located in the controller Control/Data memory
Transmit Complete List	Located in the system memory

Introduction

Two different ATM formats are supported by the controller, AAL1 and AAL5. For AAL5 ATM Cells, payload data is transferred between the system memory and the ATM network. For AAL1 ATM Cells, payload data is transferred between the system memory and the ATM network, or RAW ATM Cells are transferred between the External Raw-Cell device and the ATM network.

AAL5 Processing

When there is an AAL5 frame to be transmitted to the ATM network, the system communications software places the frame in a series of data buffers pointed to by the Transmit Buffer Descriptor. (See Figure 10 on page 22.) The Transmit process is initiated by the system processor when it signals the controller by placing the address of the Transmit Frame Descriptor on the AAL5 Transmit Ready Queue (TRQ). If required, the controller links the associated channel into the proper Traffic Management Queue (TMQ).

Cell multiplexing is determined by the traffic shaping performed by the TMQs for AAL5 cells. Each of the TMQs is serviced at a rate which is established at initialization. The rate for each of the eight queues can be different. When a TMQ is serviced, each channel linked into the queue is examined to determine if it has credit to send a cell. Channels without credit are skipped. For channels that have credit, the controller moves one cell's worth of payload data from buffers in system memory into one of the Transmit Cell Buffers, and that channel's credit is reduced. As a cell is moved into the Transmit Cell Buffer, it is processed in accordance with AAL5 specifications. This processing includes the addition of the CPCS Trailer and Pad information. If, as a result of moving AAL5 cells, the EOF is encountered or if the frame is aborted, then the system is notified of the completion via an interrupt bit in the SISR and status in the Transmit Complete List.

Once the cell payload has been constructed, the ATM header is then added and the cell is forwarded to the TC Sublayer hardware. The controller produces a composite cell stream that conforms to the TC Sublayer specifications for the 25.6-Mbps UNI.

AAL1 Processing—System Memory

The system builds the AAL1 Transmit Data Structure in system memory and indicates a Transmit request by putting an entry in the AAL1 TRQ. This entry contains the address of the AAL1 Transmit Data Structure. This queue is serviced ahead of the AAL5 TRQ. The AAL1 cell is built by the controller in the Transmit Cell Buffer in accordance with AAL1 specifications. Once the cell has been transmitted, the controller signals completion by setting the OWN bit in the AAL1 Transmit Data Structure to b'1'.

AAL1 Processing—Break Out Interface

The system builds the AAL1 Cell, excluding the HEC, in the external Raw-Cell device. Next, the external Raw-Cell device must activate the TCBR_Avail signal on the Raw-Cell Interface to inform the controller to transmit the cell. Once this process is completed, the cell is transferred across the Break Out Interface and is transmitted onto the network. See Figure 23 on page 42 for details.

OAM F4 Processing

In the 25.6-Mbps environment, OAM F4 cells are used for Segment Loopback. To transmit a OAM F4 cell, the system software must establish an LC entry in the LC Table for the desired VP and VC3 used to carry the OAM F4 cells. The format for the LC table entry is the same as for AAL5 cells and the Transmit request is added to the AAL5 Transmit Ready Queue. To transmit an F4 OAM cell, the system

builds the OAM F4 cell in the system memory and indicates a Transmit request by putting an entry in the Transmit Ready Queue. The Transmit Buffer Descriptor is also built by the system software and must contain the correct LCI. The buffer count should be set to 1, and the buffer length to x'2F'. For OAM F4 Segment Loopback protocol, the PTI and CLP fields should be set to 0. The Control and SDU Length fields are ignored by the controller.

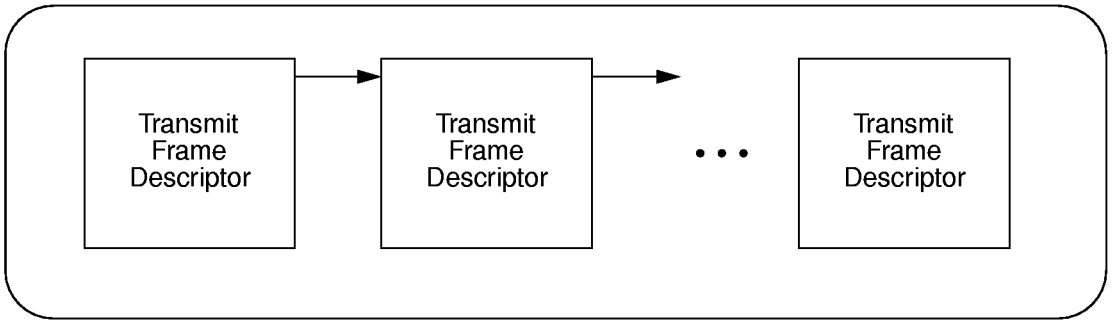
AAL5

The following sub-paragraphs describe the operation of the AAL5 transmit process.

AAL5 System Buffer Management

The buffer structure that the controller requires the system to build in system memory for each transmitting logical channel. AAL5 frames can consist of multiple system buffers linked together via the buffer descriptors, (AAL1 transmissions are performed on a cell basis; and chaining of system buffers is not allowed). For AAL5, a Transmit Frame Descriptor is associated with each frame of data. Each transmitting LC has its own linked list of Transmit Frame Descriptors for frames waiting to be segmented into ATM cells and transmitted by the controller chip, as depicted in Figure 10. The frame data can be stored in multiple system data buffers. The number of system data buffers is stored in the Transmit Frame Descriptor, along with the starting address and length of each of the buffers.

Figure 10. Logical Channel Linked Transmit Frame Descriptor List



The system is responsible for linking Transmit Frame Descriptors together for each LC. Transmit Frame Descriptors are linked in the order that the frames are to be transmitted for the associated LC. Transmit Frame Descriptors are passed to the controller chip through the TRQ. After a frame has been transmitted by the controller chip, the Transmit Frame Descriptor is passed back to the system via the Transmit Complete List.

AAL5 Traffic Control

Traffic control involves metering of LCs to control the rate at which they are able to transmit ATM cells. Each LC has a peak cell transmission rate and a sustainable cell transmission rate. The peak rate is the fastest rate at which an LC may transmit cells. Each LC is metered so that its average cell transmission rate does not exceed its specified sustainable cell transmission rate.

The implementation of the Average Metering function follows the *leaky bucket* algorithm. The leaky bucket algorithm introduces the idea of each LC having *cell credit*. As long as an LC has cell credit, it may transmit ATM cells at its peak cell transmission rate. One cell credit is taken away each time an LC transmits a cell. While the LC is idle, cell credits accumulate at a rate equal to the LC's sustainable cell rate, up to the LC's cell credit limit. An LC with no credit, that uses each credit it gains to transmit a cell, transmits cells at its sustainable cell rate.

The traffic parameters, as shown in Table 6, are used to define LC traffic characteristics:

Table 6: Traffic Parameters

Traffic Parameter	Definition
Peak Cell Rate	The reciprocal of the minimum allowable time interval spacing between cells transmitted on a particular LC.
Cell Delay Variation Tolerance	The upper bound on the cell clumping phenomenon due to the multiplexing of cells from multiple LCs.
Sustainable Cell Rate	The upper bound on the average cell transmission rate of an LC.
Burst Tolerance	The time it takes for an idle LC to accumulate a number of credits equal to its cell credit limit.

Refer to the ATM standards publications for a more detailed description of these four traffic parameters and how they are related according to the "Generic Cell Rate Algorithm".

The Cell Delay Variation Tolerance traffic parameter is fixed per LC by the controller. The other three traffic parameters are derived indirectly from the following three fields in a LC's Transmit LC Table:

Table 7: Logical Channel Table Traffic Parameters

LC Table Field	Number of Bits	Description
TMQ Number	3	This is the TMQ number of the LC's associated TMQ. The peak rate of an LC is the peak rate of its associated TMQ.
Peak-to-Sustainable Rate (PSR)	8	Specifies the ratio of the LC's peak to sustainable cell rates.
Burst Tolerance Specifier (BTS)	16	Specifies the burst tolerance of the LC.

The directions for setting these fields is given in the following subparagraphs.

TMQs

Each LC is associated with one of eight internal TMQs. An LC is only *linked* into its TMQ while it has frame data to send. Each TMQ has an associated peak rate. The peak cell transmission rate of an LC is the peak rate of the TMQ to which it is associated. In an LC Table, the TMQ number field indicates which of the eight TMQs the LC is associated with.

For each TMQ, there is a corresponding enable bit in the controller Configuration register (ACONFIG_REG). A TMQ, and all of the transmitting LCs associated with it, are shut down when the queue's enable bit is 0.

Priority and Oversubscription

For each TMQ, there is also a corresponding priority bit in the controller Configuration register. A priority bit setting of **1** means high priority, **0** means low priority. When all TMQs have the same priority, all high or all low, the LCs (in the enabled queues) with the faster peak rates are serviced first; for example, they transmit cells ahead of LCs in slower peak rate queues. This behavior may not be desirable under *oversubscribed* conditions.

Oversubscription occurs when the sum of the peak rates for all LCs over all enabled TMQs is greater than the maximum rate at which the controller may transmit cells on the media (25 Mbps). In the case of all queues having the same priority, this would mean that some of the LCs in the slowest peak rate queues may miss getting serviced until the LCs in the faster queues run out of credit or run out of data to transmit. If there is a slower peak rate queue whose LCs carry high priority traffic, and oversubscribed conditions are expected, the user may wish to give that queue high priority by setting its priority bit.

The priority bits can be changed at any time. High priority queues are serviced before low priority queues. Within each of the two priority groupings of queues (high and low), the fastest peak rate queues are serviced first. The controller can make better use of the media bandwidth if all queues have the same priority; so, if the user knows that oversub-

scription will not occur or that there are no higher-priority, slower peak rate LCs, then it is better to keep all queue priorities the same.

Under oversubscribed conditions, the extent to which LCs miss being serviced depends on:

- The amount by which the aggregate peak rate exceeds the controller bandwidth
- The magnitude of the peak-to-sustainable cell rate ratios
- Cell credit limits of the transmitting LCs currently linked into TMQs.

In the extreme best case, each of the currently transmitting LCs would have plenty of cell credit, or their sustainable cell rates would be equal to their peak rates. In the extreme worst case, enough bandwidth from the slowest, low priority LC is denied to account for the amount by which the aggregate peak rate exceeds the controller's maximum transmission rate. However, in the extreme best case, if all of the currently transmitting LCs have no cell credit or zero cell credit limits, it is possible that no LCs will be starved until the aggregate *sustainable* cell rate approaches the controller's maximum transmission rate.

Specifying the Peak Rate

The peak transmission rate of an LC is specified by its associated TMQ, and is determined by the following relationship:

$$\text{Peak Rate (Mbps)} = \left(424 \frac{\text{bits}}{\text{cell}}\right) * \left(\frac{\text{Network clock speed (MHz) / 2}}{\text{Clock Divider}}\right)$$

The value of the clock divider is set by the Configuration register of the LC's TMQ.

Example:

For a Network bit clock speed of 32 MHz to obtain a peak rate of 4 Mbps, the following equation calculates the clock divider value:

Clock Divider = 424 * (32 MHz / 2) / 4 Mbps = 1696

Cell Delay Variation Tolerance

The cell delay variation (CDV) tolerance is a characteristic of the hardware implementation used to multiplex ATM cells from different LCs. It takes into account the randomness introduced by multiplexing, which keeps an LC's cells from being transmitted at its exact peak rate time interval. For the controller chip, the cell delay variation tolerance of a particular LC is given as follows:

Cell delay variation tolerance =
(1 / peak cell rate) - (1 ATM cell time)

For a 25-Mbps ATM media speed, one ATM cell time is 16.96 µs (424 bits per cell / 25 Mbps).

Specifying the Sustainable Cell Rate

An LC's sustainable cell rate is equal to its peak cell rate divided by its peak-to-sustainable cell rate ratio. The peak cell rate of an LC comes from its associated TMQ. The peak-to-sustainable cell rate ratio of an LC is specified by the 8-bit PSR field in the LC Table. The peak-to-sustainable cell rate ratio is equal to the value of the PSR field. The peak-to-sustainable cell rate ratio can range from 1 (PSR = b'00000001') to 255 (PSR = b'11111111'). A PSR value of 0 is not valid.

Specifying the Burst Tolerance

The burst tolerance specifier is set to accommodate the desired cell credit limit that an LC is allowed to accumulate according to the leaky bucket algorithm. The cell credit limit is specified in multiples of 8 cells and can range from 0 to 2040 cells. By defining *CL/8* as the credit limit specified in multiples of 8 cells, the value of the 16-bit burst tolerance specifier (BTS) field in the LC Table is set by the following equation:

BTS = (peak cell rate / sustainable cell rate) * CL/8

The units of the BTS field are multiples of eight peak rate time intervals. A BTS field setting of 0 is not valid and should not be used.

The following equation relates the burst tolerance specifier to the burst tolerance traffic parameter:

Burst tolerance = (BTS * 8) / peak cell rate

Example:

An LC linked to a TMQ, with a peak cell rate of 5 Mbps, requires a sustainable cell rate equal to its peak cell rate divided by 18, with a credit limit of 1280 cells (CL/8 = 160).

PSR = (peak cell rate / sustainable cell rate) = 18

BTS = (peak cell rate / sustainable cell rate) * CL/8 = PSR * (1280/8)
= 18 * 160 = 2880

AAL5 Congestion Control

Some networks provide a Backward Explicit Congestion Notification (BECN) scheme for congestion control. Networks that support backward congestion notification communicate congestion through OAM F5 end-to-end cells. When a congestion notification OAM cell is received, the peak cell transmission rate of the specified LC is throttled.

Throttling and recovery of an LC's cell transmission rate can be controlled by either the controller chip or by the system, as specified by the Congestion Handling Mode (CHM) field in the Transmit portion of the LC Table. When the *SHC* or Software/Hardware Control bit of an LC's CHM field is set to 1, congestion control for that channel is controlled by the controller. When the SHC bit is set to 0, congestion control is performed by the system. The use of the Control Rate Divider (CRD) field depends on the setting of the CHM.

Controller Congestion Control

When the SHC bit of the CHM field is set to **1**, the controller handles congestion control. Each time a congestion notification cell is received, the controller reduces the specified LC's cell transmission rate by a factor of two. If the cell transmission rate has been throttled down by a factor of 16, the next congestion control notification will result in suspending transmission on that LC.

The controller recovery is performed by increasing the throttled LC's cell transmission rate by a factor of two, at a recovery rate specified by the *RATE_DIVIDER* portion of the CHM field. The RATE_DIVIDER indicates the number of consecutive cell transfers that must occur over the congested LC before its cell transmission rate can be increased by a factor of two. The number of consecutive cell transfers required to recover the cell rate is equal to the value of the RATE_DIVIDER field. These cell transfers must occur without any further congestion notifications being received. As long as no additional congestion notifications are received, an LC's cell transmission rate eventually recovers to its originally subscribed cell transmission rate.

Note: A RATE_DIVIDER value of 0, although meaningless, has the same effect as a RATE_DIVIDER value of **1**.

Recovery of a disabled LC, whose cell transmission rate has been throttled down to zero, proceeds as if the LC were transmitting cells at its original cell transmission rate decreased by a factor of 32—although no actual cells are transmitted. The first recovery stage of a disabled LC increases its cell transmission rate to its original cell transmission rate divided by 16.

System Congestion Control

When the SHC bit is set to **0**, congestion throttling and recovery is handled by the system software. When the system software detects that an LC is congested, it may throttle the LC's cell transmission rate by controlling the RATE_DIVIDER in the CHM field. In System Congestion Control mode, an LC's cell transmission rate is divided by the value of the RATE_DIVIDER field, except for the case when the RATE_DIVIDER equals b'11111'. When the RATE_DIVIDER equals b'11111', cell transmission is suspended. When the RATE_DIVIDER equals b'00001', the LC's cell transmission rate is unaffected (for example, the transmission rate is divided by 1). The rate at which the LC recovers is determined by how the system chooses to control the RATE_DIVIDER.

Note: A RATE_DIVIDER value of 0, although meaningless, has the same effect as a RATE_DIVIDER value of 1.

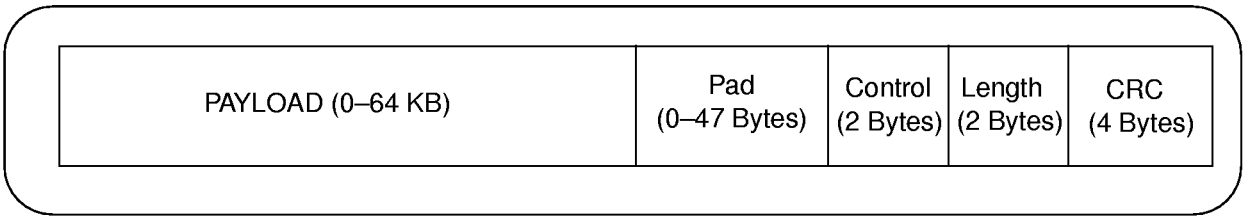
AAL5 Frame Segmentation

Appending of CPCS Header, Trailer, and Pad

For AAL5, a CPCS header does not exist. The CPCS-PDU is composed of a CPCS payload (up to 65,535 bytes), a Pad (0–47 bytes), and an 8-byte CPCS Trailer. The CPCS Trailer contains a 2-byte Control field, a 2-byte Length field, and a 4-byte CRC. The Control field is currently reserved, and it must contain all **0**s. The Length field is the CPCS-

SDU length in bytes, and can support a maximum length encoding of 65,535 bytes. The CPCS-SDU length resides in the Transmit Frame Descriptor, and is copied into the Transmit (TX) LC Table. The 32-bit CRC covers the CPCS payload, Pad, Control field, and Length field. The CRC is generated as each ATM cell in the CPCS-PDU is transmitted. Between the end of the CPCS payload and the CPCS trailer, there are 0–47 unused bytes that are called the Pad. The padding field complements the CPCS payload, Pad, and CPCS Trailer to an integral multiple of 48 bytes. The CPCS Trailer is always the last 8 bytes of the last cell in the frame. To ensure this, the next to last cell transmitted may contain up to 7 bytes of Pad. Each AAL5 ATM cell that is transmitted is composed of a 5-byte ATM header and 48 bytes of the CPCS-PDU. Figure 11 shows the CPCS PDU format.

Figure 11. CPCS-PDU Format



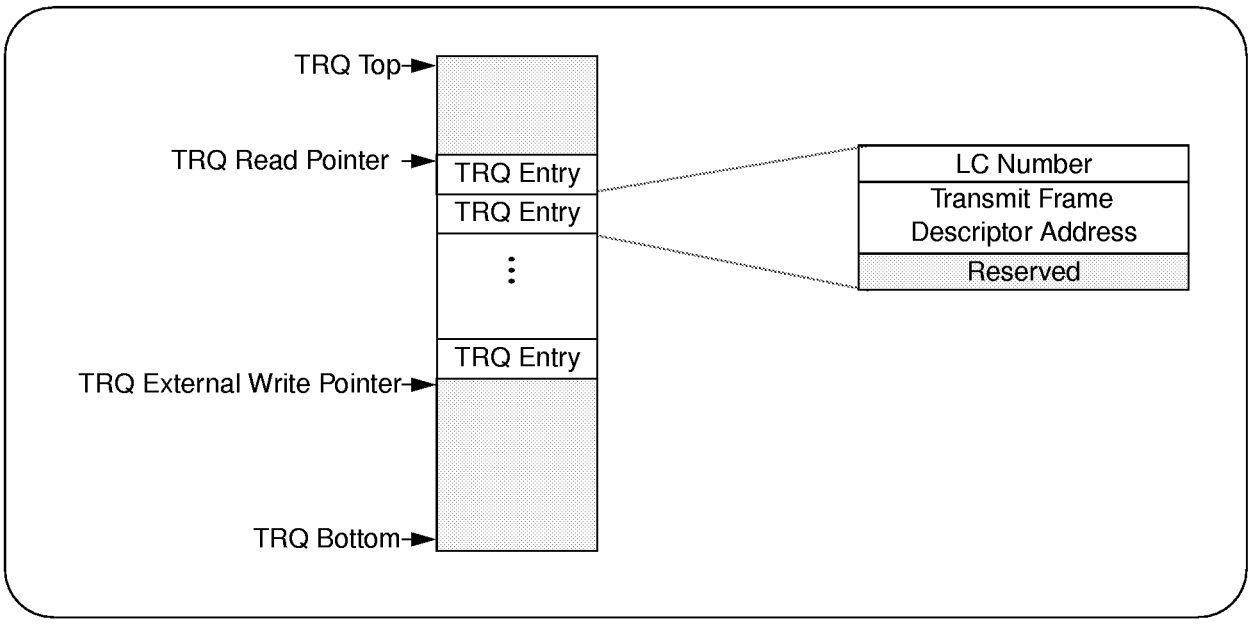
Transmitting an AAL5 Frame

After an LC has been set up, the system can begin building Transmit Frames and Transmit Frame Descriptors in the system memory. To start the controller frame transmission process, the system passes the LC number and the Transmit Frame Descriptor Address to the TRQ.

Note: The system can forward chain multiple frames for transmission and pass only the first frame in the chain to the TRQ.

The last frame descriptor in this chain has a forward pointer of bit 0 = b'1'. The TRQ is a data structure that exists in the controller's Control/Data memory. A functional diagram of the TRQ is given in Figure 12.

Figure 12. Transmit Ready Queue (TRQ)



The TRQ is a circular queue. The controller manages the queue for the system through the following five registers:

- TRQ Top Pointer
- TRQ Bottom Pointer
- TRQ Write Pointer
- TRQ Read Pointer
- TRQ Status Register

Each entry in the queue is 4 words (8 bytes). The system writes the LC number to the first word and the Transmit Frame Descriptor Address to the second and third words. The fourth word is unused. The total number of 4-word entries is determined by the TRQ Top and Bottom Pointers, and is set by the user.

At the controller initialization time, the system loads the TRQ Top Pointer (the TRQ Write Pointer and the TRQ Read Pointer automatically are loaded with the Top Pointer value), and all three pointers are initialized to the top of the queue. The system loads the TRQ Bottom Pointer with the bottom location of the queue. After this time, the controller completely manages the queue for the system.

When the system has an entry to add to the TRQ (one or more frames ready to be transmitted), it first reads the TRQ Status register in the controller to determine if the queue is full or not full. A single bit in the register indicates this status. If it is full, the system must poll the register until it becomes not full. When the queue is not full, the system writes the LC number and the Transmit Frame Descriptor Address to the address of the appropriate TRQ. The system always writes its entry to the same addresses because the controller internally manages the TRQ Write Pointer. Any time the controller detects an PCI Slave Write to the address of an entry in the TRQ, it internally replaces this address with the current TRQ Write Pointer Address. After each entry is written to the TRQ, the controller increments the TRQ Write Pointer or wraps it back to the top of the queue, whichever is necessary.

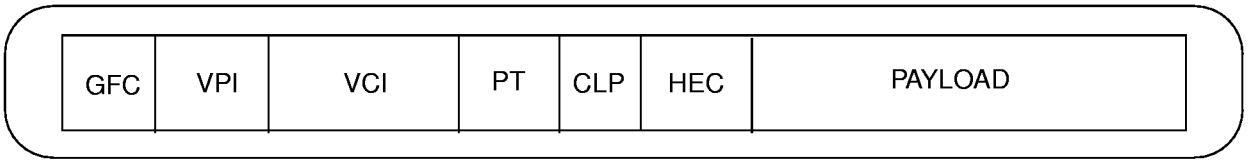
Appending ATM Cell Header

Segmentation is the process of creating one or more 48-byte ATM cell payloads from a CPCS-PDU, and attaching a 5-byte ATM cell header to each cell payload. The following fields reside in the ATM header:

- Generic Flow Control (GFC 4 bits)
- Virtual Path Identifier (VPI 8 bits)
- Virtual Channel Identifier (VCI 16 bits)
- Payload Type Identifier (PTI 3 bits)
- Cell Loss Priority (CLP 1 bit)
- Header Error Control field (HEC 8 bits).

The GFC, VPI, and VCI are written into the LC Table by the system processor across the System Interface when the LC connection is established. The PTI field indicates whether the cell contains user data or OAM information. For user data a bit in the PTI field represents the ATM user-to-user Identification. OAM cells may be inserted between user data cells on an LC basis. The PTI field is formed internal to the controller. The CLP identifies low- and high-priority cells; low-priority cells may be discarded when network congestion conditions exist. The Cell Loss Priority bit resides in the SDU Parameters field of the Transmit Frame Descriptor. The CLP bit is copied into the Transmit LC Table and is distinct for each CPCS-PDU. The HEC field is formed internal to the controller, and provides an 8-bit CRC on the ATM cell header. Figure 13 on page 27 shows the ATM cell format.

Figure 13. ATM Cell Format



AAL5 Management of Completed Frames

Completed Transmit Frame Processing

The controller chip reports status to the system on a frame-by-frame basis. The system is interrupted via the SISR each time a frame is transmitted. The controller chip links the completed Transmit Frame Descriptor to a Transmit Complete List (TCL) prior to the system's involvement.

When the controller has linked the completed Transmit Frame Descriptor into the TCL, the controller chip activates the SV_TCL bit in the SISR.

The system reads the SISR in the controller to make sure that at least one frame has been transmitted to the network. It is the system's responsibility to propagate through the TCL and process all frames until it reaches the end of the TCL. The system reaches the end of TCL when the TCL Forward Pointer Address field bit 0 is equal to b'1'.

Examples of linked Transmit Frame Descriptors prior to and after transmission are shown in Figure 14 and Figure 15 on page 29. In Figure 14, linked frame descriptor lists for three different LCs are shown. The last descriptor of each chain contains a forward pointer with bit 0 set to a b'1', marking the end of the chain. In Figure 15, the end of the TCL is indicated by a descriptor with a TCL forward pointer, with bit 0 set to a b'1'.

Figure 14. Linked Transmit Frame Descriptors in System Memory Prior to Any Transmissions by the Controller Chip

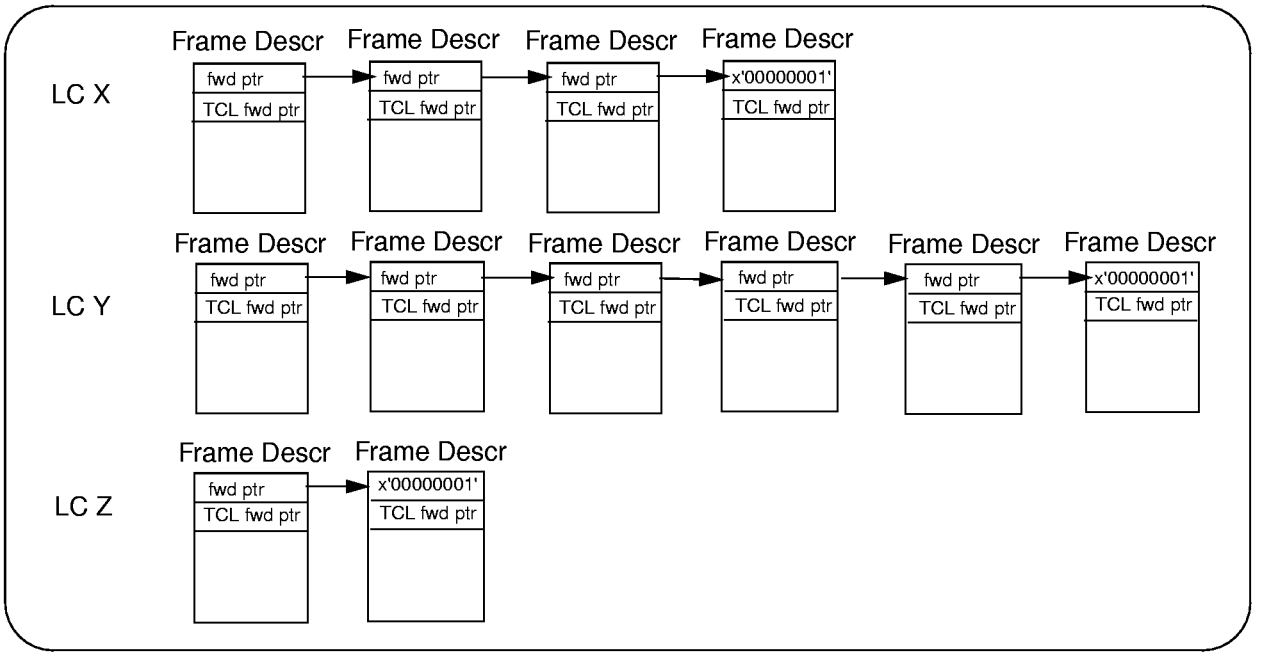
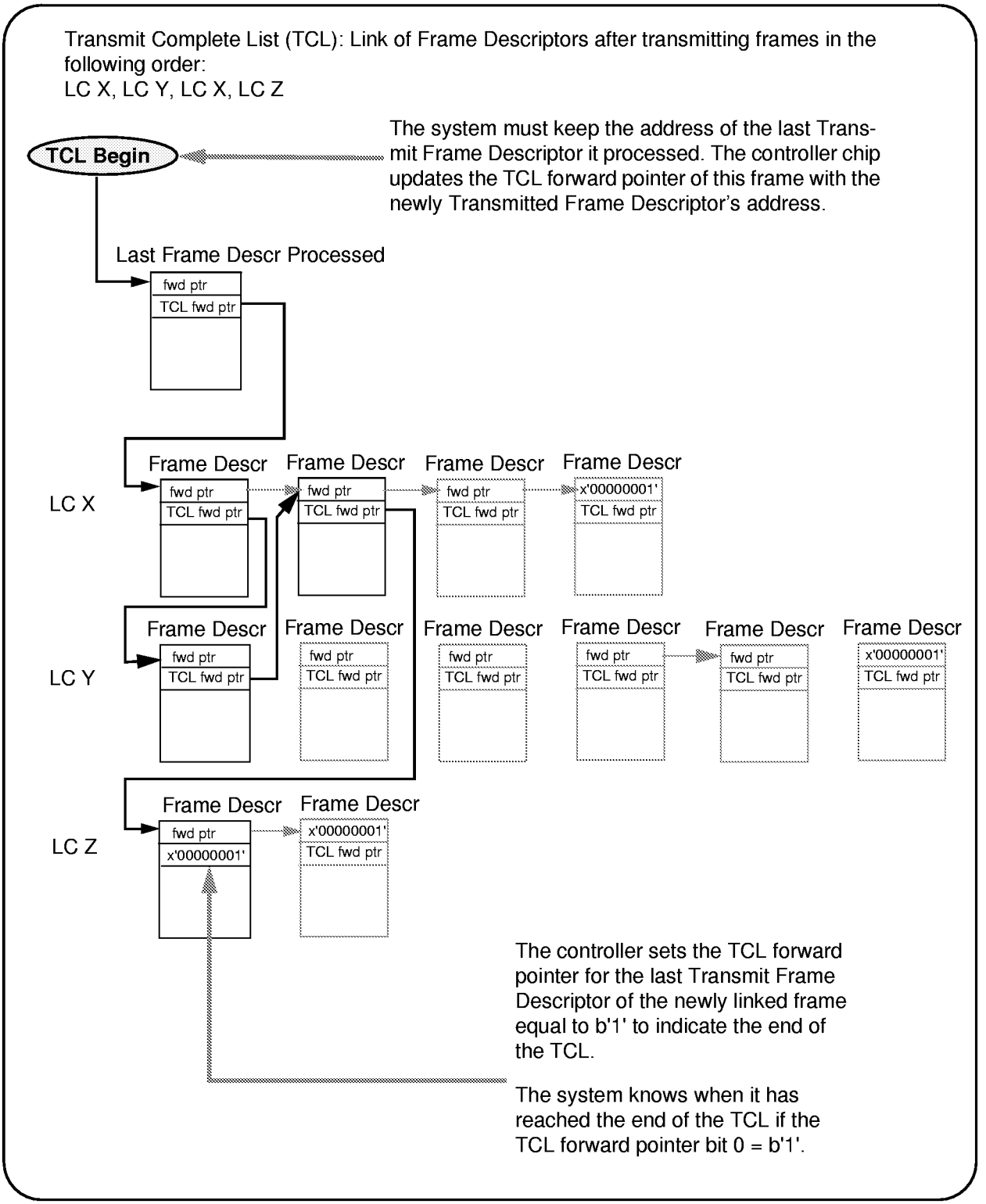


Figure 15. TCL: Linked Transmit Frame Descriptors in System Memory after Transmissions by the Controller Chip

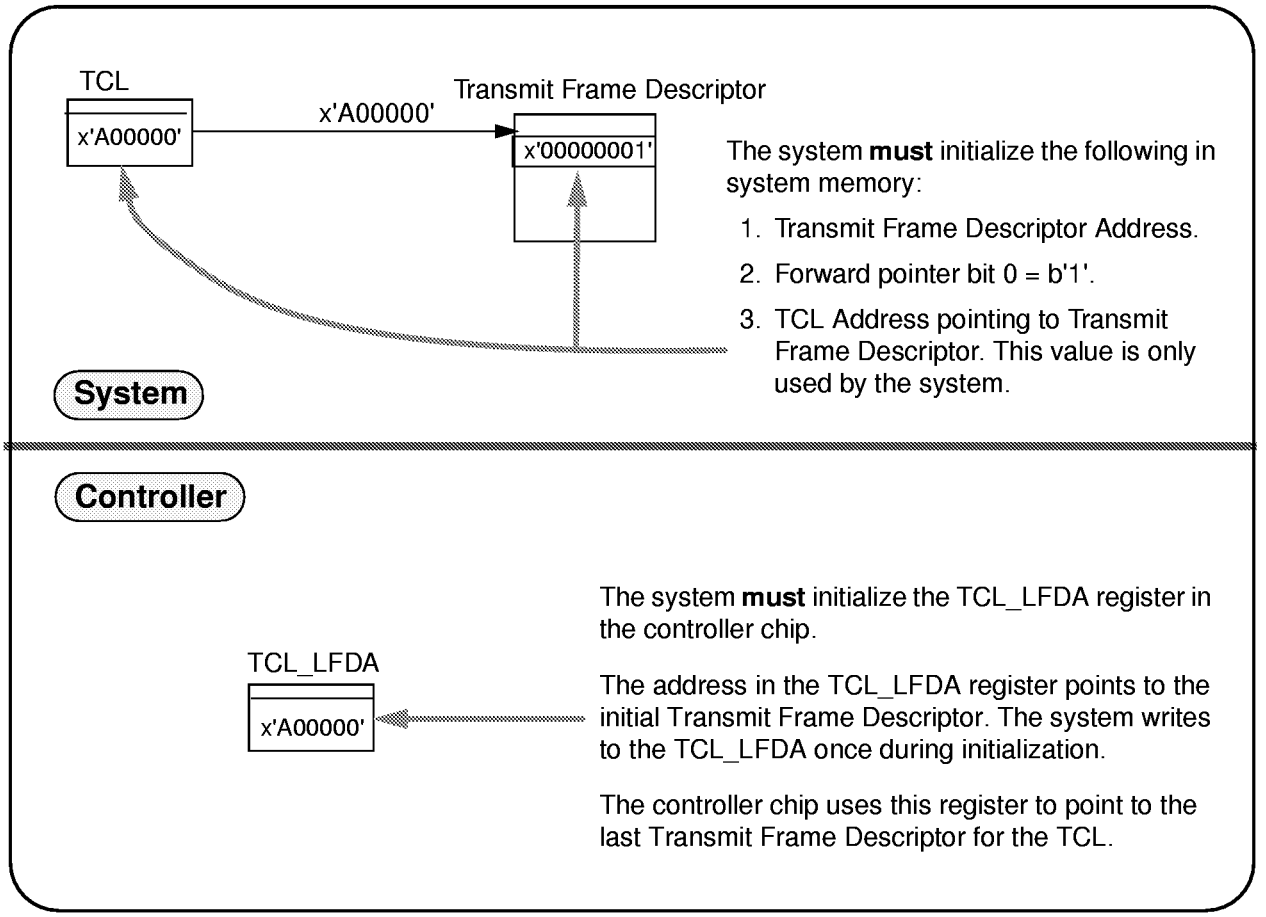


Transmit Complete List Initialization

At initialization, the system must retain an address that points to a Transmit Frame Descriptor for the TCL. The controller chip updates the TCL forward pointer of this Transmit Frame Descriptor with the newly transmitted Frame Descriptor's address.

To perform this function, the controller chip must have the Transmit Frame Descriptor Address for the TCL initialized by the system in register TCL_LFDA, as shown in Figure 16.

Figure 16. The System's Role in Initialization of System Space and Controller Space



Setting/Resetting TCL Bit in the SISR

The controller chip always sets the SV_TCL bit in the SISR after linking the Frame Descriptor to the TCL; and the controller chip resets the complete SISR (including the TCL bit in the SISR) when the system software reads the SISR.

AAL1 Traffic Control

The controller also provides support for AAL1 traffic. Because of the constant bit-rate nature of AAL1 traffic, cells associated with this AAL must be handled differently than those of AAL5. The cell transmission rate is determined by a time base associated with the data source. In general, this time base is different for each AAL1 traffic source. These traffic sources usually support an application with real-time constraints such as digitized audio. For this reason, cell transmission cannot be controlled by a TMQ, and must take precedence over AAL5 traffic. The

controller relies on the traffic source to provide cells at the correct rate. In other words, AAL1 cells must be transmitted on demand from the traffic source. AAL1 traffic is handled in a similar manner as AAL5 traffic except that cells are processed in accordance with AAL1 specifications and cell transmission is initiated by channel demand.

Three schemes are supported for AAL1 traffic. The source and destination of the traffic may be routed across either the system data bus into system memory or an external device memory mapped into the system space or the External Memory Interface's data bus. These three implementations allow freedom for system design.

When AAL1 traffic is ready in system memory, the system processor initiates transmission by placing the LC number and the system source address of the Transmit Data Structure on the AAL1 TRQ. The controller processes entries in this queue ahead of demand from the TMQs.

For each cell transmitted, the PTI and CLP provided in the Transmit Data Structure are used in the construction of the ATM cell header. These values are added to the end of the header information stored in the Transmit portion of the LC Table. In addition to the ATM cell header information, the Transmit portion of the LC Table contains information needed to provide OAM monitoring of the channel.

The controller provides cell sequence numbers and generates the 3-bit CRC and parity for transmitted cells. The controller completes the cell by adding the ATM cell header to the 48-byte payload supplied by the source. The ATM header, with the exception of the PTI and CLP, is stored in the Transmit portion of the LC Table entry that is associated with the LC identified by the LCI field of the Transmit Data Structure

AAL1 Processing—System Memory

The controller acts as a transmission scheduler and cell payload conveyor with limited AAL1 processing. This AAL1 processing is only performed for cells routed across the system data bus. In that case, a data structure that consists of a 4-byte header, followed by the 48-byte payload data for the cell, must be constructed. The HEC byte is generated by the controller.

The Transmit process is initiated by the system processor when it signals the controller by placing the address of the AAL1 Transmit Data Structure on the AAL1 TRQ.

Cell multiplexing of AAL1 traffic takes precedence over AAL5 TMQs. The controller moves one cell's worth of payload data from AAL1 Transmit Data Structures in system memory into one of the Transmit Cell Buffers. As a cell is moved into the Transmit Cell Buffer, it is processed in accordance with AAL1. This processing includes generation of the AAL1 SAR header and parity generation of the Structure Pointer byte, if applicable. After the AAL1 cell is transmitted, the controller sets the OWN bit of the AAL1 Transmit Data Structure to b'1', indicating that the cell was transmitted.

Once the cell payload has been constructed, the ATM header is added and the cell is forwarded to the TC Sub-layer hardware. The controller produces a composite cell stream that conforms to the TC Sublayer specifications for the 25.6-Mbps UNI.

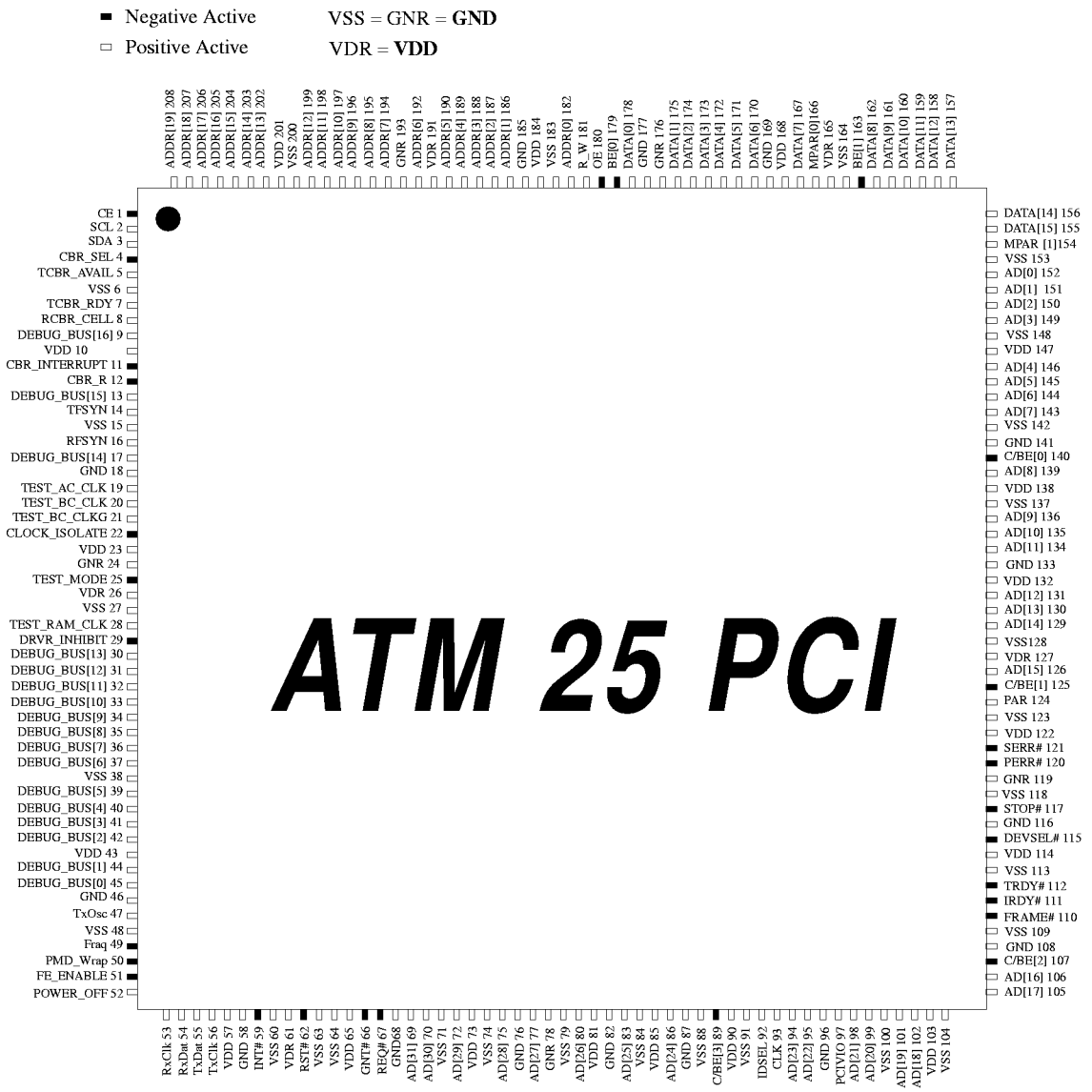
AAL1 Processing—Break Out Interface

The controller acts as a transmission scheduler and cell payload conveyor with no AAL1 processing. In this case, a data structure, which consists of a 4-byte ATM header (HEC not included), followed by the 48-byte payload for the cell, must be constructed by the AAL1 External Raw-Cell device.

The transfer of cells across the Break Out Interface is controlled by a collection of control signals described in "External Memory Interface," on page 37; the Transmit operation is shown in Figure 23 on page 42.

External Interfaces

Pin Diagram



Pin Descriptions

Table 8: Controller Pin Description

Pin #	Pin Name	Bits	I_{ol} (I_{oh}) ma	I/O	Description	PU/PD
PROM Interface						
1	-CE	1	4 (-)	O	Chip Enable line to Pico Code/BIOS PROM	None
Sub-Total			**1**			
Serial PROM Interface						
2	+SCL	1	4 (-)	O	Serial clock for serial EEPROM	None
3	+SDA	1	4 (-)	I/O	Serial data/address/control for serial EEPROM	None
Sub-Total			**2**			
Break Out Interface						
4	-CBR_SEL	1	4 (-)	O	The controllerdrives this signal active to select the External Raw-Cell Device for transmitting or receiving data on the External Memory Interface bus. This signal is active for each Data Transfer Bus cycle. The External Memory Interface's +R_W signal indicates whether the controller is reading or writing data.	None
5	+TCBR_AVAIL	1		I	This signal is an indication to the controller that the External Raw-Cell Device has a cell waiting to be transmitted. This signal is held active until after the controller has responded by raising the +TCBR_RDY signal.	PD
7	+TCBR_RDY	1	4 (-)	O	In response to the TCBR_AVAIL signal, the controller activates this signal to indicate that it is beginning to accept data under control of the -CBR_SEL and +R_W signals. This signal remains inactive (Low) for at least one cycle following the transfer of the last byte of the AAL1 Break Out Interface Data Structure.	None
8	+RCBR_CELL	1	4 (-)	O	The controller indicates to the External Raw-Cell Device that a cell is being transferred from the ATM Network to that device. This signal becomes active at least one bus clock cycle prior to the activation of the -CBR_SEL signal. This signal is deactivated for at least one bus cycle, beginning with the bus cycle following the last data transfer of the cell.	None
12	-CBR_R	1	4(-)	O	This signal is activated for one bus cycle if the controller is reading from the Raw-Cell Device.	None
	+R_W		4(-)	O	Read, Not Write. This is an External Memory Interface signal.	None
	+DATA		4 (-)	I/O	This bus is 16 bits + 2 parity bits. These are actually External Memory Interface data bus signals.	None
11	-CBR_Interrupt	1	4(-)	I	When pulled low by a CBR device, this signal produces a maskable system interrupt.	PU
Sub-Total			**6**			
Network Interface						
50	-PMD_Wrap	1	4 (-)	O	Diagnostic Control. Output to PMD. When low, PMD is placed in Wrap mode. See MODE_REG for details.	None

Table 8: Controller Pin Description

Pin #	Pin Name	Bits	I_{ol} (I_{oh}) ma	I/O	Description	PU/PD
54	+RxDat	1		I	Received Data. This is the retimed data derived from the network signal input. This signal has been sampled on the falling edge of the RxClk signal.	None
53	+RxClk	1		I	Received Clock. This is the clock derived from the network signal input.	None
47	TxOsc	1		I	Transmit Oscillator. This is a buffered version of the Crystal Oscillator signal.	None
55	+TxDat	1	4 (-)	O	Transmit Data. This is the NRZI-encoded data to be transmitted onto the network.	None
56	+TxClk	1	4 (-)	O	Transmit Clock. This is the Transmit Clock used by the IBM ATM 25Mbps PMD to sample TxDat.	None
49	-Fraq	1	4 (-)	O	Frequency Acquisition. This signal puts the IBM ATM 25Mbps PMD PLL into Frequency Acquisition mode. This mode sets the center frequency of the IBM ATM 25Mbps PMD VCO by locking the VCO to the local crystal oscillator. Low=Frequency Acquisition; High=Normal operation.	None
14	+TFSyn	1		I	Transmit Frame Sync. Input signal that causes Start of Frame. This comes from the 125 µs source when 125 µs frame synchronization is required. TFSyn is required to be synchronized to TBClk's active (High) level and to be on for the entire level, but not longer then TBClk's period. This line is internally pulled down to Vss through a 100-Kohm resister.	PD
16	+RFSyn	1	4 (-)	O	Receive Frame Sync. Frame sync control sequence has been received. (ESC symbol followed by a data symbol identifying the Frame Sync.) This output can be used for 125 µs clock/strobe recovery. Pulse is nominally active for 8 RxClk cycles.	None
51	-FE_Enable	1	4 (-)	O	This signal reflects the state of the Mode register Bit 4, and is used to enable the IBM ATM 25Mbps PMD chip. If set to a 0, PMD is enabled; when set to 1, PMD is in Standby mode.	None
Sub-Total			**10**			
External Memory Interface						
	+ADDR	20	4 (-)	O	Address Bus	None
182, 186–190,192, 194		Bits 0–7				
195–199, 202–204		Bits 8–15				
205–208		Bits 16–19				
	+DATA	16	4 (-)	I/O	Data Bus	None
178, 175–170, 167		Bits 0–7				
162–155		Bits 8–15				
	+M PAR	2	4 (-)	I/O	Odd Parity Pins	None
166, 154		Parity on Data Bits 0–7; Parity on Data Bits 8–15.				

Table 8: Controller Pin Description

Pin #	Pin Name	Bits	I_{ol} (I_{oh}) ma	I/O	Description	PU/PD
	-BE	2	4 (-)	O	Byte Enable	None
179, 163		Byte Enable 0; Byte Enable 1.				
181	+R_W	1	4 (-)	O	Read, Not Write	None
180	-OE	1	4 (-)	O	Output Enable	None
Sub-Total			**42**			
PCI Bus Interface						
	AD(31::00)	32		I/O	PCI Address and Data are multiplexed on the same pins.	
152–149, 146–143		Bits 0–7				
139, 136–134, 131–129, 126		Bits 8–15				
106, 105, 102, 101, 99, 98, 95, 94		Bits 16–23				
86, 83, 80, 77, 75, 72, 70, 69		Bits 24–31				
	C/BE(3::0)#	4		I/O	PCI Bus Command and Byte Enables.	
89, 107, 125, 140		C/BE(3)#, C/BE(2)#, C/BE(1)#, C/BE(0)#				
124	PAR	1		I/O	Even parity from the system. Parity is generated across the AD(31::00) and the C/BE(3::0)# signals.	
110	FRAME#	1		I/O	Frame is driven by the current master to indicate the beginning and duration of an access.	
112	TRDY#	1		I/O	Target Ready indicates the target agent's ability to complete the current data phase of the transaction.	
111	IRDY#	1		I/O	Indicator Ready indicates the initiating agent's ability to complete the current data phase of the transaction.	
117	STOP#	1		I/O	Stop indicates the current target is requesting the master to stop the current transaction.	
115	DEVSEL#	1		I/O	Device Select, when actively driven, indicates the driving device has decoded its address as the target of the current access.	
92	IDSEL	1		I	Initialization Device Select is used as a chip select during Configuration Read and Write transactions.	
120	PERR#	1		I/O	Parity Error is for reporting of parity errors during all PCI transactions, except a Special Cycle.	
59	INTA#	1		O	Interrupt Request signals the Host that the controller needs attention.	
121	SERR#	1		I/O	System Error is for reporting address parity errors, data parity errors on the Special Cycle command, or any other system error where the result is catastrophic.	

Table 8: Controller Pin Description

Pin #	Pin Name	Bits	I_{ol} (I_{oh}) ma	I/O	Description	PU/PD
67	REQ#	1		O	Request indicates to the arbiter that the controller wants to use the bus. This is a point-to-point signal. Every Master has its own REQ#.	
66	GNT#	1		I	Grant indicates to the controller that access to the bus has been granted. This is a point-to-point signal. Every Master has its own GNT#.	
93	CLK	1		I	Clock provides timing for all transactions on the PCI, and is an input to every PCI device. All other PCI signals, except RST# and IRQA#, are sampled on the rising edge of CLK.	
62	RST#	1		I	Reset is used to bring PCI-specific registers, sequencers, and signals to a benign state.	
97	PCIVIO	1		I	Special power pin for defining and driving the PCI Signaling Rail. This pin must be connected to the VIO of the PCI Bus.	
Sub-Total			**51**			
LSSD Interface						
19	+Test A/C Clock	1		I	Multiplexed A Clock and System Clock. Pulled high via internal resister during normal operation.	PU
20	+Test B/C Clock	1		I	Multiplexed B Clock and System Clock. Pulled high via internal resister during normal operation	PU
22	-Clock Isolate	1		I	Clock Isolate pin is used to select between the System clock and the Test clocks. Pulled high via internal resister during normal operation	PU
25	-Test Mode	1		I	Used to put the chip into Test mode. Pulled high via internal resister during normal operation	PU
29	- Driver Inhibit	1		I	This pin is used to tri-state all module drivers. Pulled high via internal resister during normal operation.	PU
28	+Test RAM Clock	1		I	Test RAM Clock is used to test internal RAM. Pulled high via internal resister during normal operation	PU
52	+Power Off	1		I	If set to b'1', the controller is in Static IDD Test mode. This pin must be pulled low via a resister on the card during normal operation.	None
21	+Test B/C ClockG	1		I	Multiplexed B Clock and System Clock. Pulled high via internal resister during normal operation	PU
Sub-Total			**8**			
CHIP Monitoring Interface						
	Chip Monitoring	17	4 (-)	O	Used to debug and monitor the chip.	None
45, 44, 42–39, 37–30, 17, 13, 9		Debug Bus (0) to Debug Bus (16).				
Sub-Total			**17**			
Total Pins			**136**			

Timing Diagrams

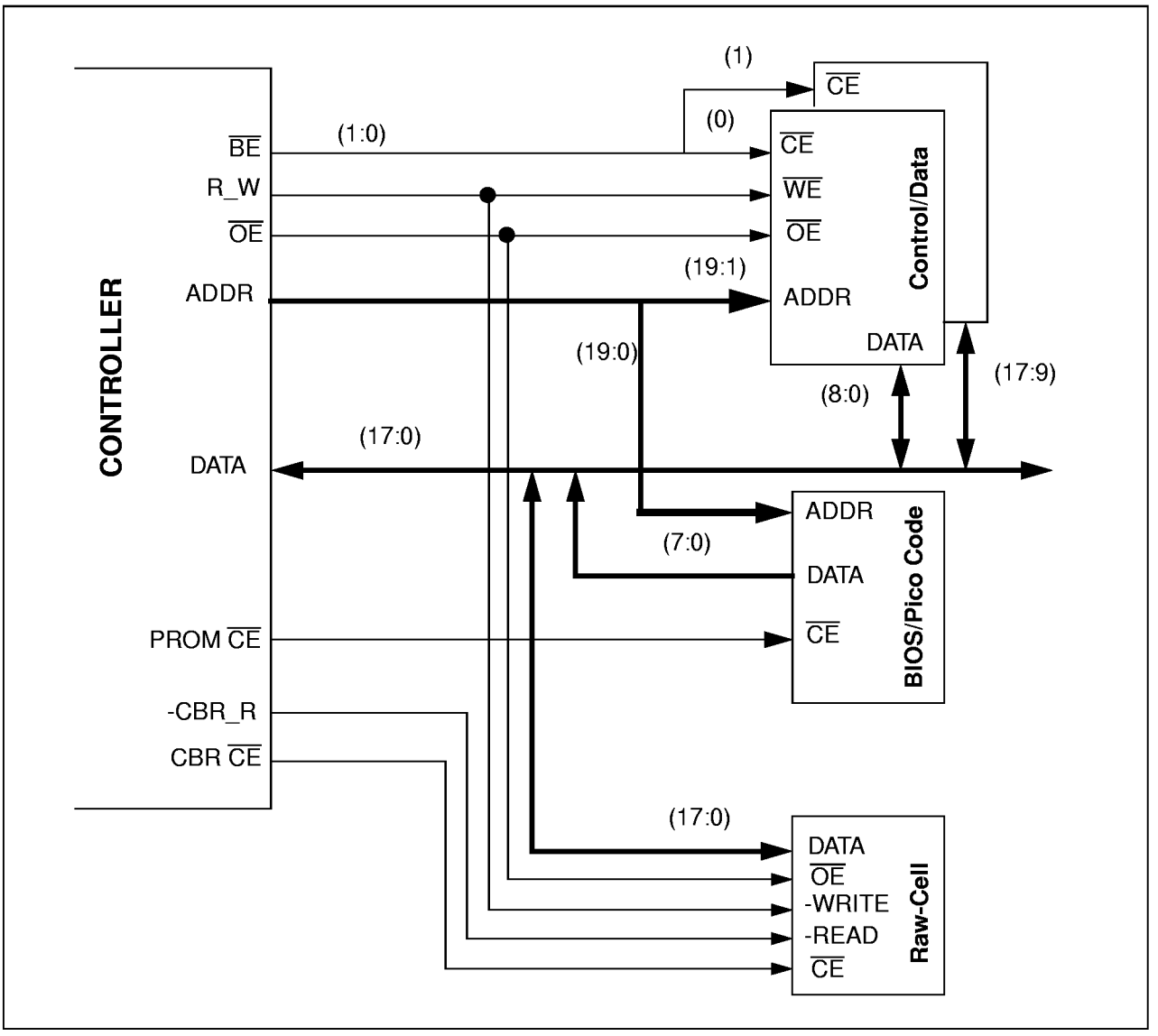
External Memory Interface

The External Memory Interface allows attachment of up to three external memory devices:

- Control/Data SRAM
- BIOS/Pico Code PROM
- Raw Cell Device.

The External Memory Interface uses a 16-MHz clock, derived from the 32-MHz clock input. Thus, the bus cycles on the External Memory Interface are 62.5 ns. Each device access, assuming the number of PROM wait states is zero, is completely contained within one of these bus cycles. The sequence of these accesses depends upon requests from many independent processes and is controlled by the arbitration function of the controller. As a result, the time sequence of these accesses is not predictable. Therefore, devices attaching to the External Memory Interface must be prepared to respond on every bus cycle. For Raw Cell Devices, this readiness to respond is limited to intervals when either TCBR_RDY or RCBR_CELL is asserted.

Figure 17. External Memory Interface Connections



The Control/Data SRAM is required for all adapter configurations using this chip. The BIOS/Pico Code PROM and Raw Cell device are optional.

The Control/Data SRAM is assumed to be two physically distinct 9-bit-wide SRAMs, with a separate Write Enable for each module. By attaching or not attaching upper order address bits, different sizes of SRAMs can be accommodated.

The BIOS/Pico Code PROM, which is a byte-wide device, is attached to only the low-order data byte of the External Memory Interface. Since it is assumed to be an 8-KB device, 13 address bits, including the low-order Bit 0 of the Address bus, are attached to it. A separate Chip Enable is provided to select this device. This PROM is in the PCI Memory Address space. A Page register, located in the PCI I/O address space, is used to provide addressing of 128 pages (8 KB each) of PROM space. This PROM can contain the BIOS code and/or the ATM 25Mbps PCI Controller Pico Code. The system software can read the Pico Code from this PROM and then write the code into the Controller's Pico Code RAMS. Using this Read/Write technique, the controller can be IPLed from the PROM. BIOS code can also be contained in this PROM.

See Figure 17 for connection details of all three possible devices attached to the External Memory Interface.

The remainder of this subchapter describes the timings for the External Memory Interface for each of the three devices that can be attached to it.

The External Memory Interface works for all Control/Data SRAMs, with access times faster than or equal to 20 ns. The following timing diagram shows the timings for Control/Data SRAM accesses.

The PROM Read cycle is shown in Figure 19 on page 40. The PROM cycle time is a minimum of 150 ns and is speed matched to the 62.5-ns Control/Data bus via controller internal state machines. The EXT_MEM_CNTL register controls the number of Wait states that are inserted.

The External Memory Interface is also used to communicate with an external Constant Bit Rate (CBR) traffic device for transmitting and receiving Raw-Cells. The Raw-Cell device is assumed to be a 16-bit-wide FIFO. This Break Out Interface uses the 16-bit data and 2-bit parity buses of the External Memory Interface to Transmit and Receive Raw-Cell data. The -CBR_SEL signal is used to select the Raw-Cell device to Transmit and Receive on the data bus. Table 8 on page 33 for a list of the Raw-Cell interface signals.

Figure 20 on page 41 shows the Transmit handshaking when a Raw-Cell device wants the controller to send a cell. This figure shows the overall handshaking utilized for Raw Cell transmission and is not intended to portray the actual cycles when data are transferred. The handshaking sequence begins with the External Raw-Cell device activating the +TCBR_AVAIL signal. This signal must only be activated when a completed, Raw-Cell Interface Transmit Data Structure is available from the External Raw-Cell device. The controller acknowledges this request by activating the +TCBR_RDY signal. The External Raw-Cell device must then deactivate the +TCBR_AVAIL signal. +TCBR_RDY remains active for the entire cell transfer and becomes inactive in the bus cycle following the last transfer for the Raw-Cell Interface Transmit Data Structure. The controller performs one transfer from this data structure each time the -CBR_SEL signal is active, and the -CBR_R and -OE are inactive to indicate a Read operation.

Figure 23 on page 42 shows the timing diagram and handshaking sequence for the reception of cell data across the Raw Cell Interface. This figure shows the overall handshaking utilized for Raw Cell transmission and is not intended to portray the actual cycles when data are transferred. It is assumed that the receiving Raw-Cell device is always able to receive cell data in the Raw-Cell Interface Receive Data Structure format.

Figure 18. Timing Diagram for Control/Data RAM accesses

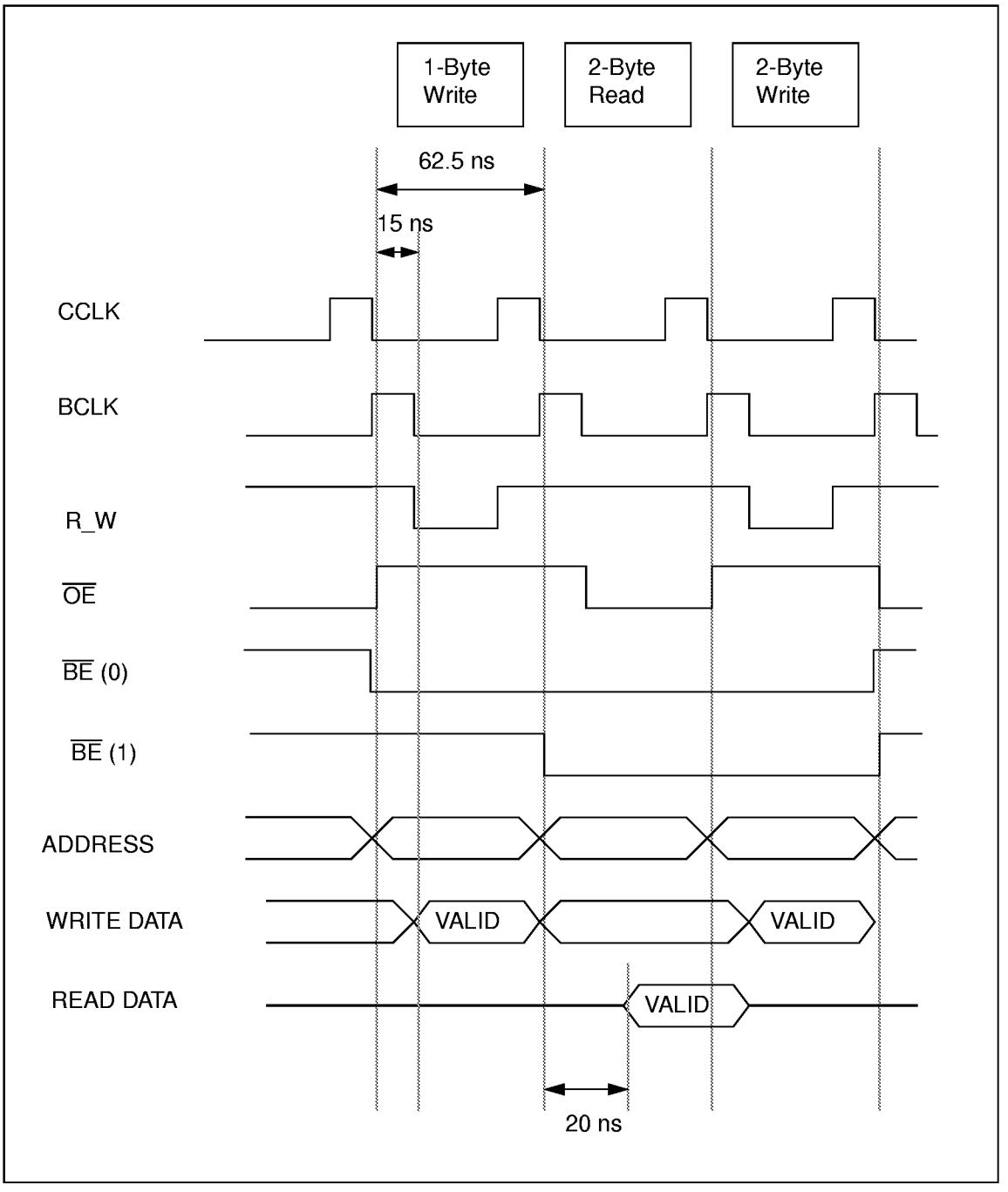
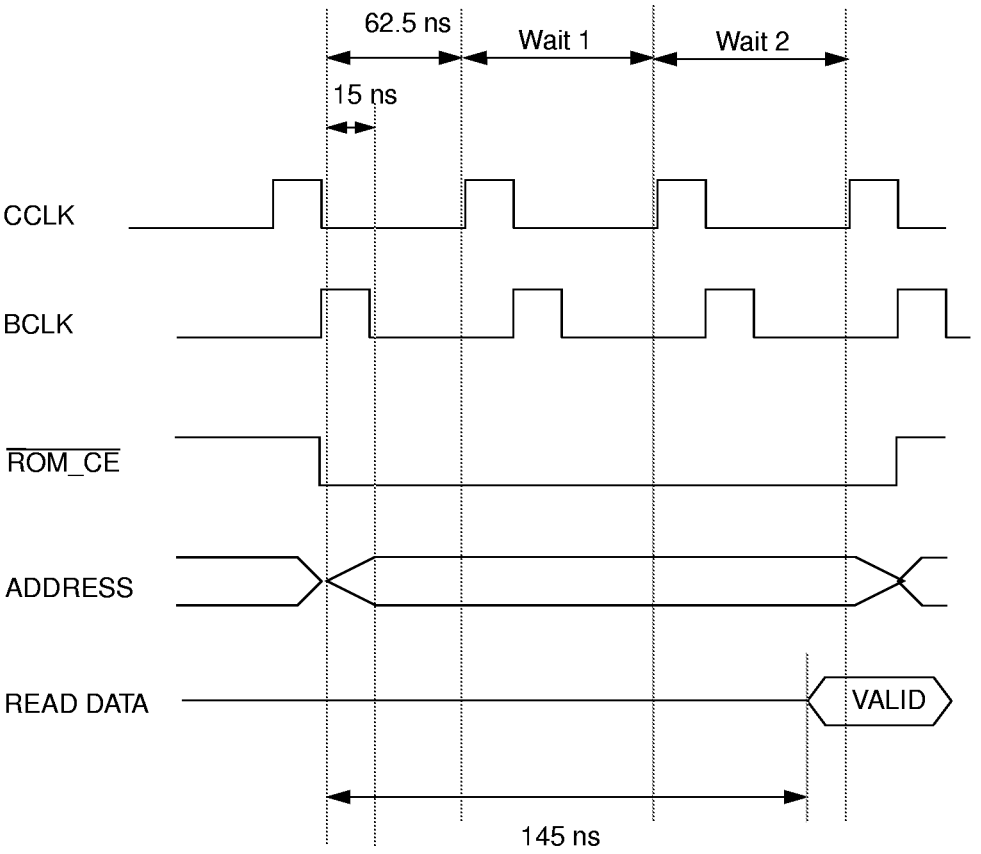


Figure 19. Timing Diagram for BIOS/Pico Code ROM Accesses



Note: The above timing can be achieved by programming two Wait states in EXT_MEM_CNTL register; for example, EXT_MEM_CNTL(1:0) = 10.

Figure 22 shows the CBR_Interrupt Timing. System interrupts can be generated using the CBR_Interrupt signal. This signal, when pulled to a low level by the attached CBR device, sets SISR Bit 16 to a b'1', which results in a system interrupt, assuming the associated MISR bit is set to **1**. Once the interrupt is set in the SISR, the system software can clear the interrupt by reading the SISR. The CBR_Interrupt must be held low for a minimum of 100 ns. If the CBR_Interrupt is held active after the system software clears the interrupt, another interrupt is generated.

Figure 20. Controller Raw-Cell Interface Transmit

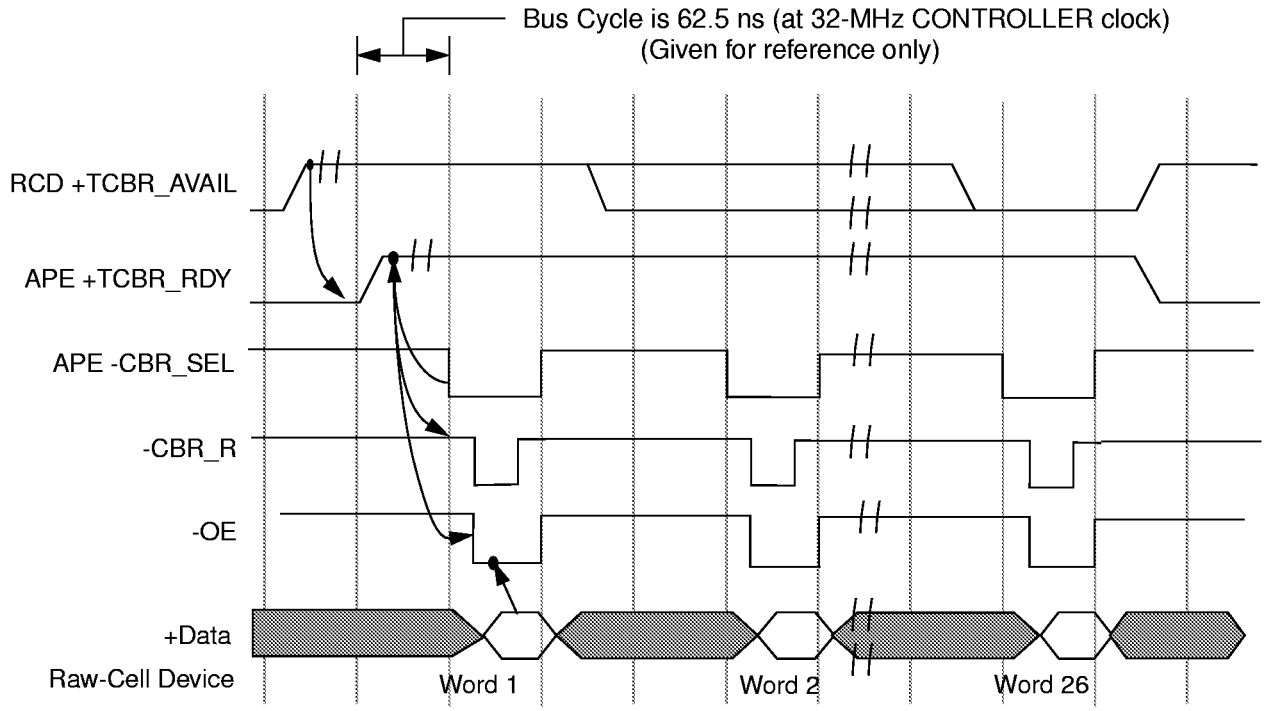


Figure 21. Controller Raw-Cell Interface Receive

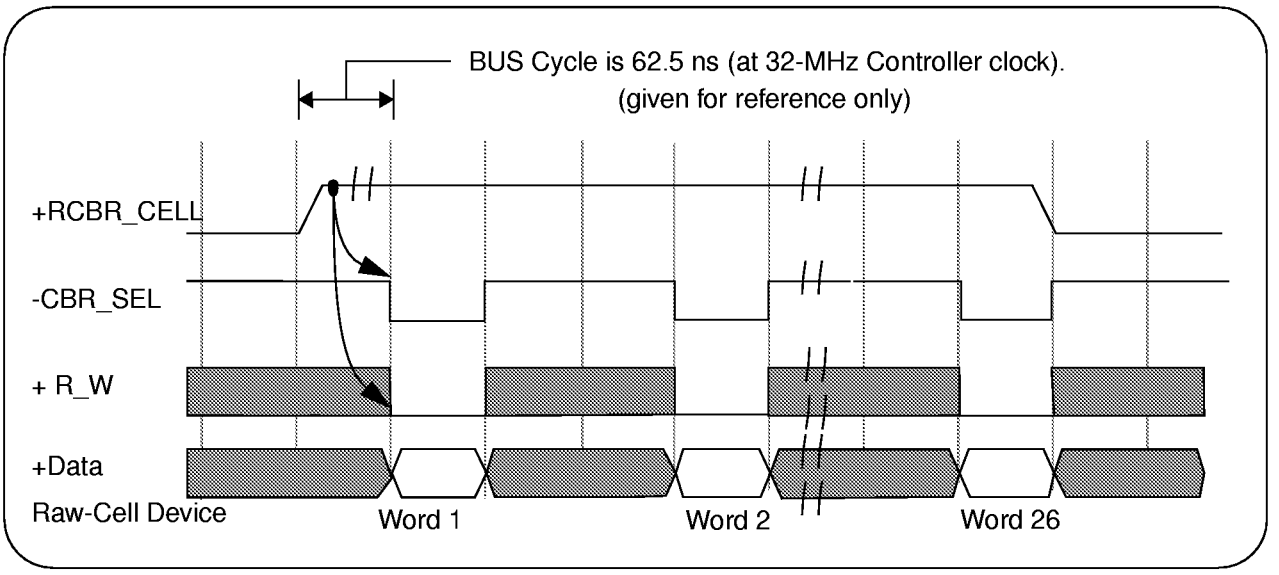


Figure 22. CBR_ Interrupt Timing

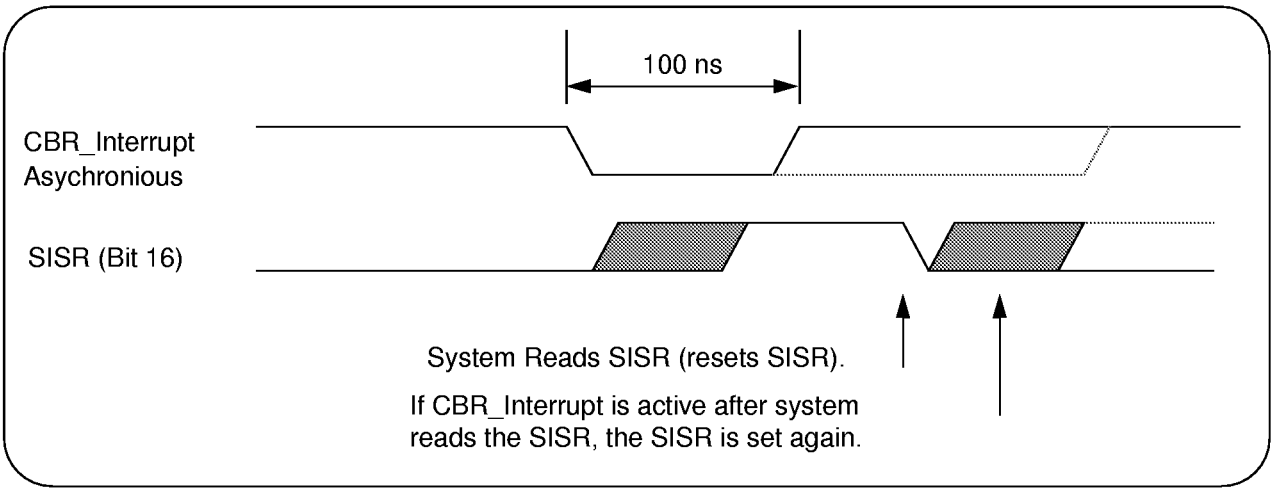


Figure 23 shows the timing diagram and handshaking sequence for the Raw-Cell Interface when the controller receives the beginning of a new cell before the previous cell was completed. Under these circumstances, there is no way to reconstruct the cell. The External Raw-Cell device should recognize that 26 Write cycles occurred during the +RCBR_CELL window. Any exceptions should result in the External Raw-Cell Device taking appropriate action.

Figure 23. Controller Raw-Cell Interface Receive (Abort)

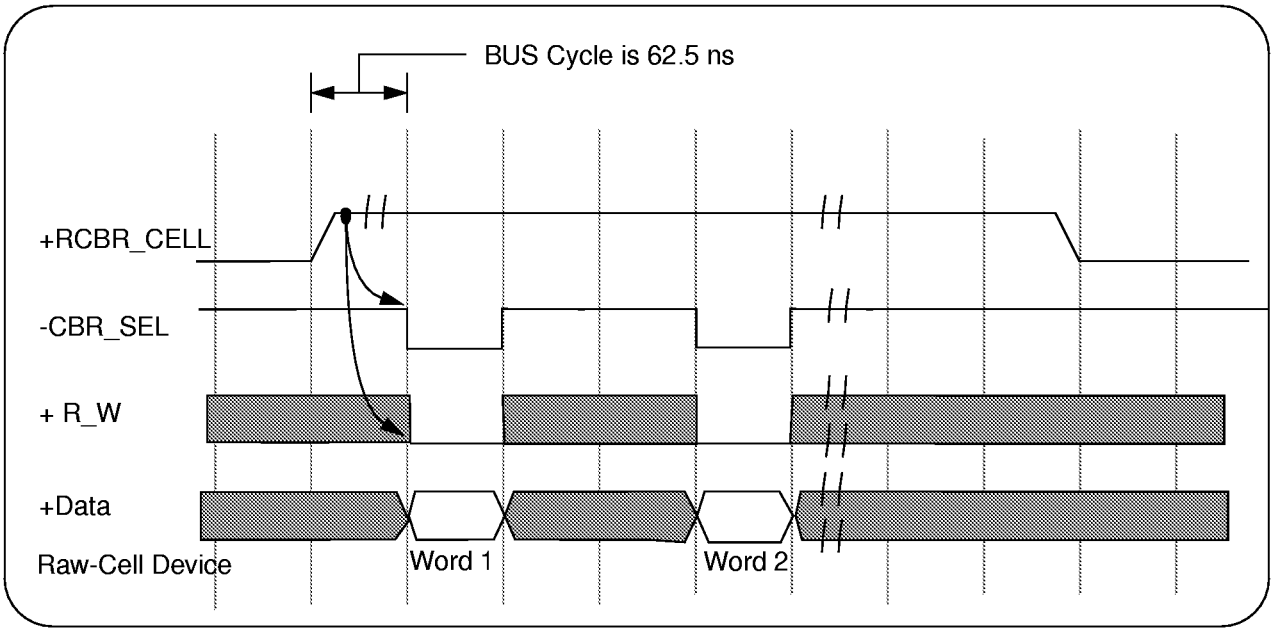
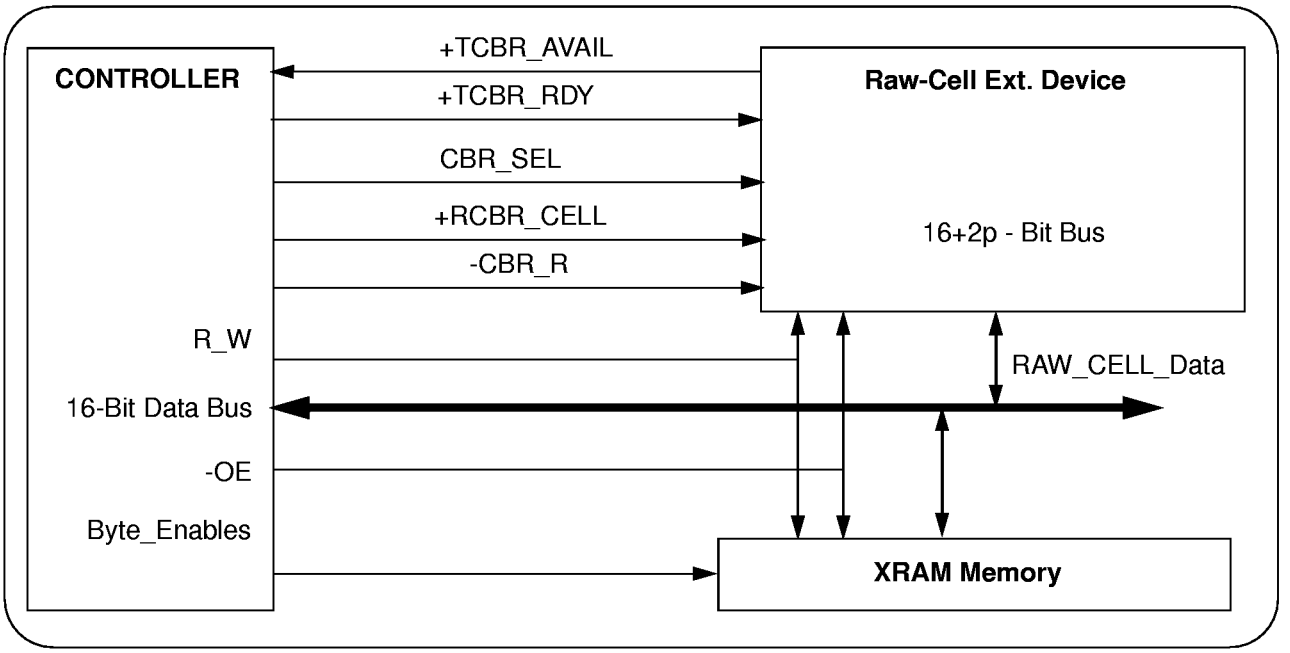


Figure 24 shows an example how the Raw-Cell Interface can be implemented.

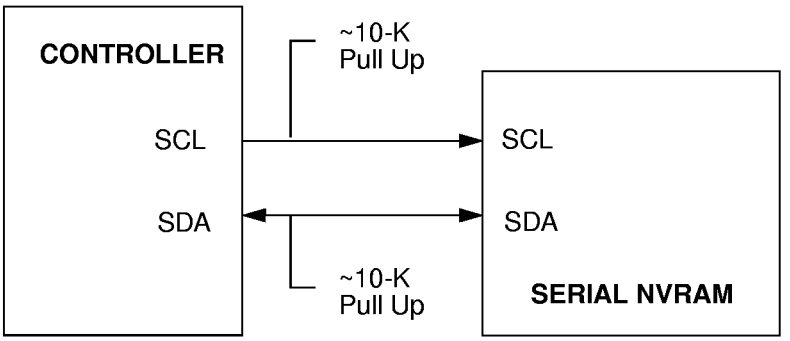
Figure 24. Raw-Cell Interface Example



Serial NVRAM Interface

The serial NVRAM contains the Configuration register values for the controller. The default configuration values are loaded during initialization or reset time. The controller has an internal state machine, which provides the necessary control to interface to the NVRAM. A handshake is provided to indicate when data is available in the controller during Read operations.

Figure 25. Serial NVRAM Attachment



The serial NVRAM Interface software protocol is compatible with XICOR X24C01 or an equivalent XICOR module. Please refer to the specifications of the XICOR serial EEPROMs for timing diagrams and software protocol.

Physical Layer Interface

The Transmit and Receive timing interfaces to the IBM ATM 25Mbps PMD chip are described in the following sections. All the timing numbers apply to the IBM ATM 25Mbps PMD operating in 16-Mbps mode.

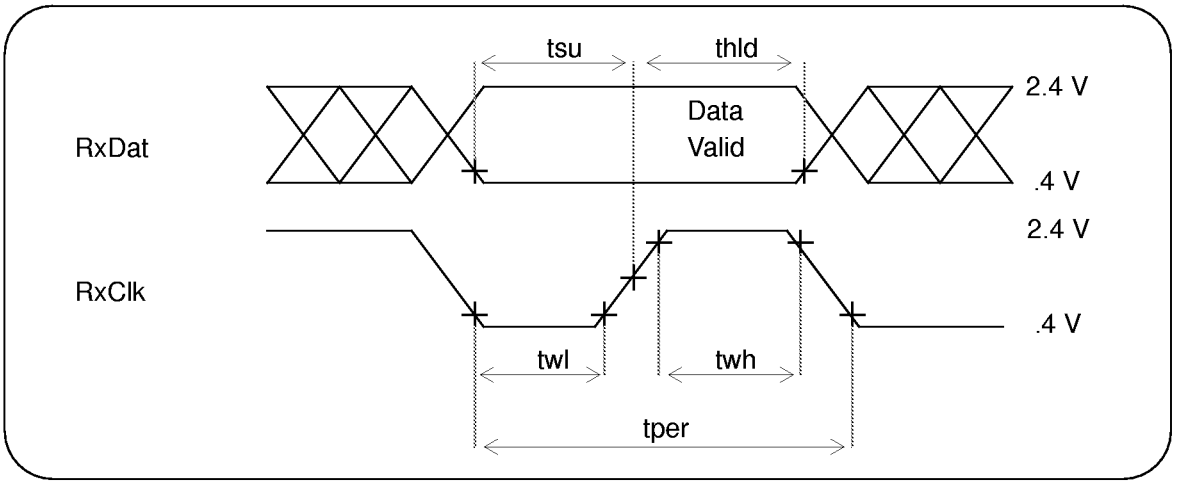
Receive Timing

The timing details for RxClk (Received Clock) and RxDat (Received Data) are shown in Figure 9 and Figure 26.

Table 9: IBM ATM 25Mbps PMD Receive Timing Parameters

Symbol	Parameter	Min	Typical	Max	Units
tsu	Set-Up Time	8			ns
thld	Hold Time	8			ns
twh	Pulse Duration, RxClk High	12			ns
twl	Pulse Duration, RxClk Low	12			ns
tper	Period of RxClk		31.25		ns

Figure 26. IBM ATM 25Mbps PMD Receive Timing Diagram



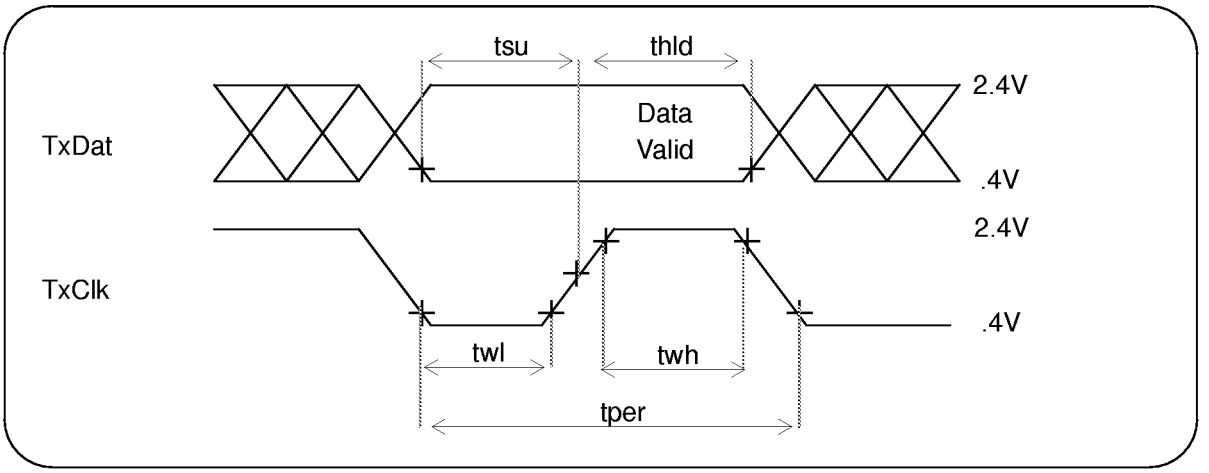
Transmit Timing

The timing details for TxClk (Transmit Clock) and TxDat (Transmit Data) are shown in Figure 10 and in Figure 27 on page 45.

Table 10: IBM ATM 25Mbps PMD Transmit Timing Parameters

Symbol	Parameter	Min	Typical	Max	Units
tsu	Set-Up Time	4			ns
thld	Hold Time	4			ns
twh	Pulse Duration, TxClk High	8			ns
twl	Pulse Duration, TxClk Low	8			ns
tper	Period of TxClk		31.25		ns

Figure 27. IBM ATM 25Mbps PMD Transmit Timing Diagram



PCI System Bus Interface and Timing

The controller supports both Bus Master and Bus Slave operations. All 32/16 Bit I/O Bus Slave cycles are initiated by the Host. These operations are used by the Host to write control information into the controller control RAM and controller internal registers. The Host can also read the control structures and registers using this interface. When the controller chip requires either a data or control structure transfer, it requests a Bus Master cycle.

For a complete description of these operations, refer to the *PCI Local Bus Specification Production,* Revision 2.0.

PCI Basic Read Transaction

Figure 28 on page 46 illustrates a basic PCI Read operation. The Read starts with an address phase which occurs when FRAME# is asserted for the first time and occurs on Clock 2. During the address phase, AD(31::00) contains a valid address and C/BE(3::0)# contains a valid bus command. The first clock of the first data phase is Clock 3. During the data phase, C/BE# indicates which byte lanes are involved in the current data phase. A data phase may consist of a data transfer and wait cycles. The C/BE# output buffers must remain enabled (for both Reads and Writes) from the first clock of the data phase through the end of the transaction. This ensures C/BE# is not left floating for long intervals.

The first data phase on a Read transaction requires a turnaround-cycle (enforced by the target via TRDY#). In this case, the address is valid on Clock 2, and then the master stops driving AD. The earliest the target can provide valid data is Clock 4. The target must drive the AD lines following the turnaround cycle when DEVSEL# is asserted. Once enabled, the output buffers must stay enabled through the end of the transaction. (This ensures AD is not left floating for long intervals.)

A data phase completes when data are transferred, which occurs when both IRDY# and TRDY# are asserted on the same clock edge. (TRDY# cannot be driven until DEVSEL# is asserted.). When either is deasserted, a wait cycle is inserted and no data is transferred. As noted in the diagram, data are successfully transferred on Clocks 4, 6, and 8; and wait cycles are inserted on Clocks 3, 5, and 7. The first data phase completes in the minimum time for a Read transaction. The second data phase is extended on Clock 5, because TRDY# is deasserted. The last data phase is extended because IRDY# was deasserted on Clock 7.

The Master knows, at Clock 7, that the next data phase is the last. However, because the Master is not ready to complete the last transfer (IRDY# is deasserted on Clock 7), FRAME# stays asserted. Only when IRDY# is asserted can FRAME# be deasserted, which occurs on Clock 8.

Figure 28. Basic Read Operation

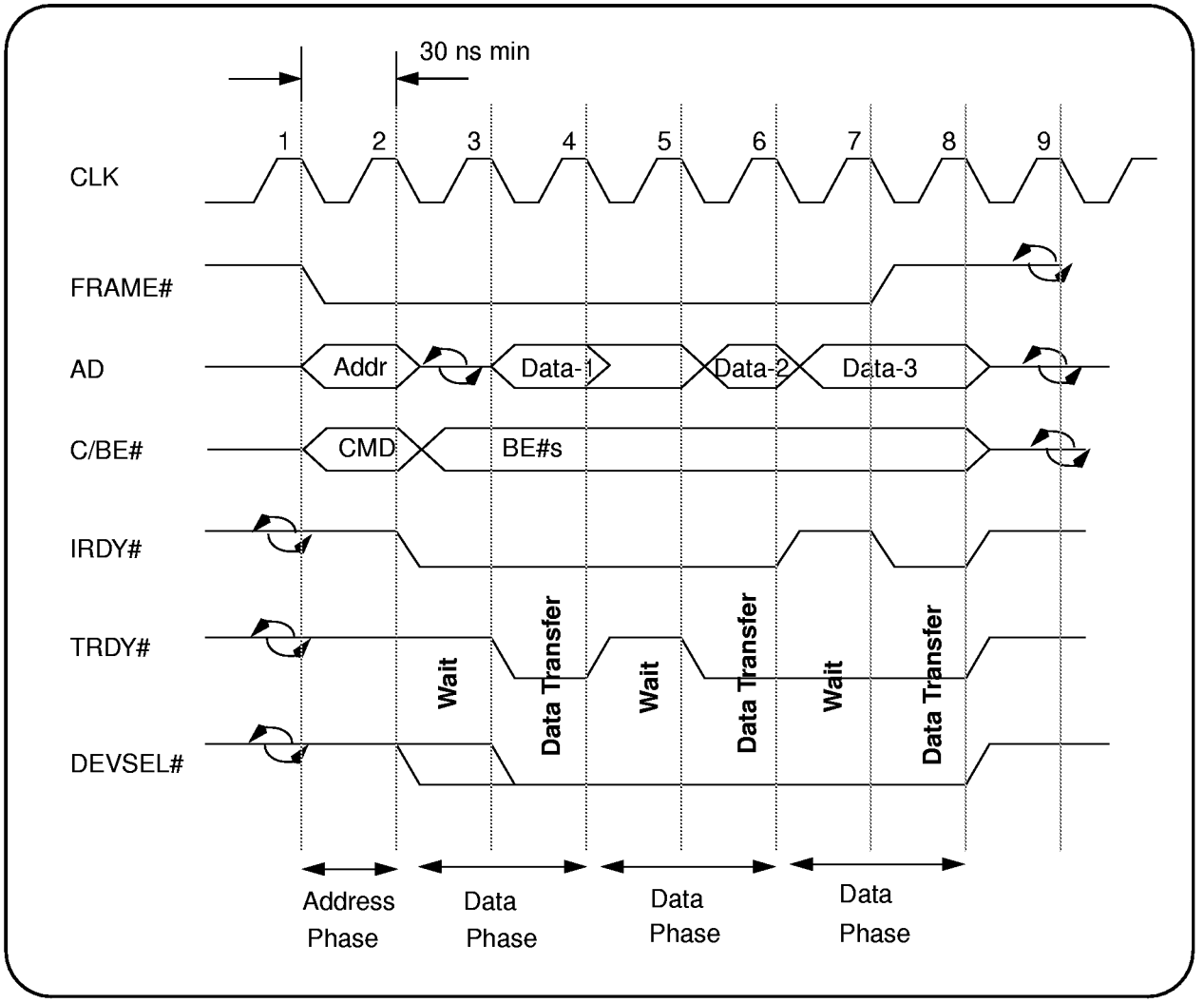
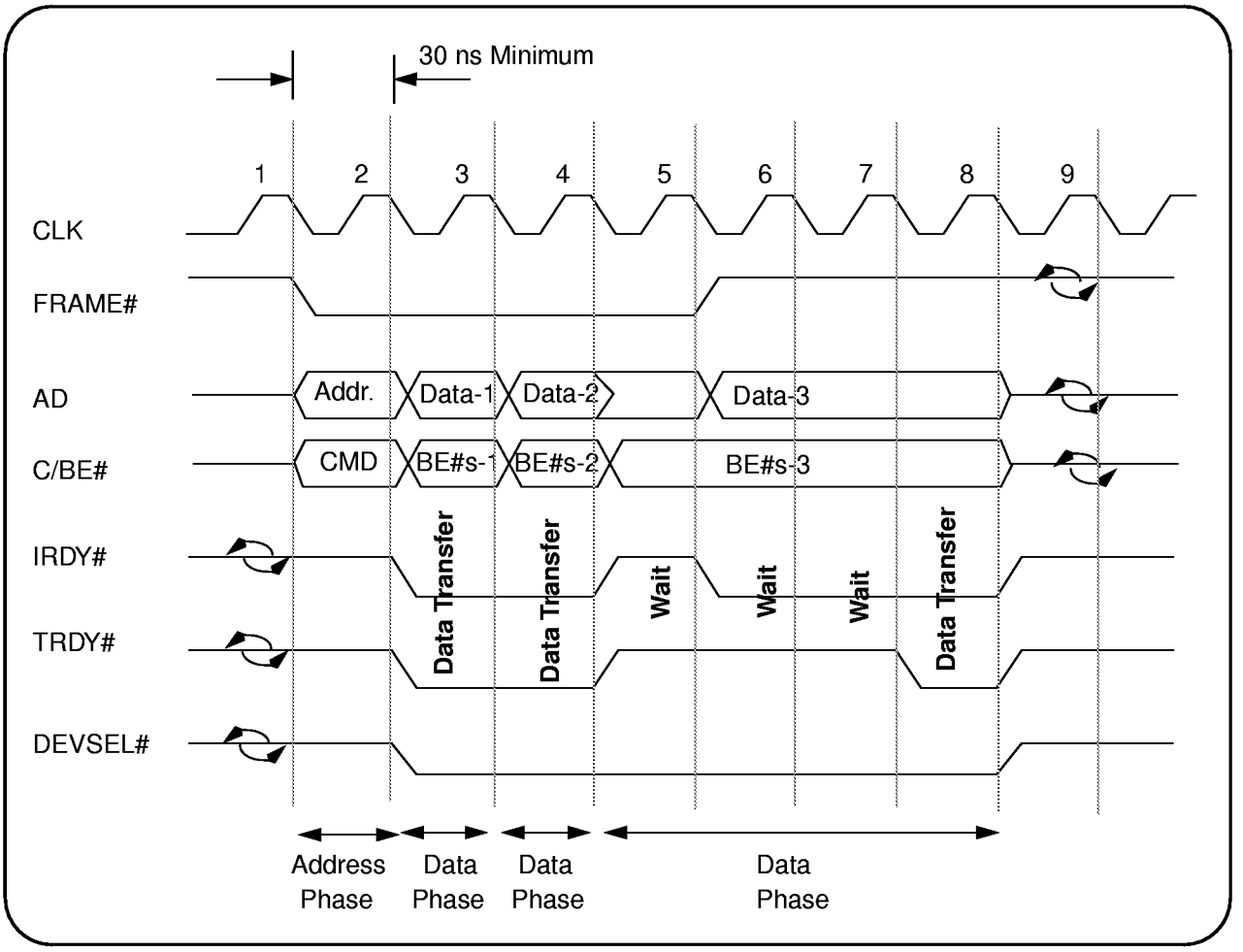


Figure 29. Basic Write Cycle



PCI Basic Write Operation

Figure 29 shows a basic PCI Write transaction. This transaction starts when FRAME# is asserted for the first time, which occurs on Clock 2. A Write transaction is similar to a Read transaction, except no turnaround cycle is required following the address phase because the master provides both address and data. Data phases work the same for both Read and Write transactions.

The first and second data phases complete with zero wait cycles. However, the third data phase has three wait cycles inserted by the target. Notice both agents insert a wait cycle on Clock 5. IRDY# must be asserted when FRAME# is deasserted, indicating the last data phase. The data transfer was delayed by the master on Clock 5 because IRDY# was deasserted. Although this allowed the Master to delay data, it did not allow the Byte Enables to be delayed. The last data phase is signaled by the Master on Clock 6, but does not complete until Clock 8.

Technology and Packaging

The controller is designed in a 0.6-µm CMOS VLSI process and is manufactured by IBM Microelectronics Division. The package offering for the controller is a semi-custom, 208-pin QFP package.

Temperature Ratings

Maximum operating temperature is Tj(max.)=70°C while the minimum operating temperature, Tj(min.), is 0°C. Maximum storage temperature is 150°C; while the minimum storage temperature is –55°C.

Recommended Operating Conditions

Table 11: Supply Voltage and Temperature

Symbol	Parameter	Commercial			Units
		Min.	Nom.	Max.	
V_{CC}	Supply Voltage	3.0	3.3	3.6	V
T_A	Operating Free Air Temperature	0	35	70	C

Electrical Characteristics

The following table gives the absolute ratings for various electrical characteristics.

Table 12: Electrical Characteristics

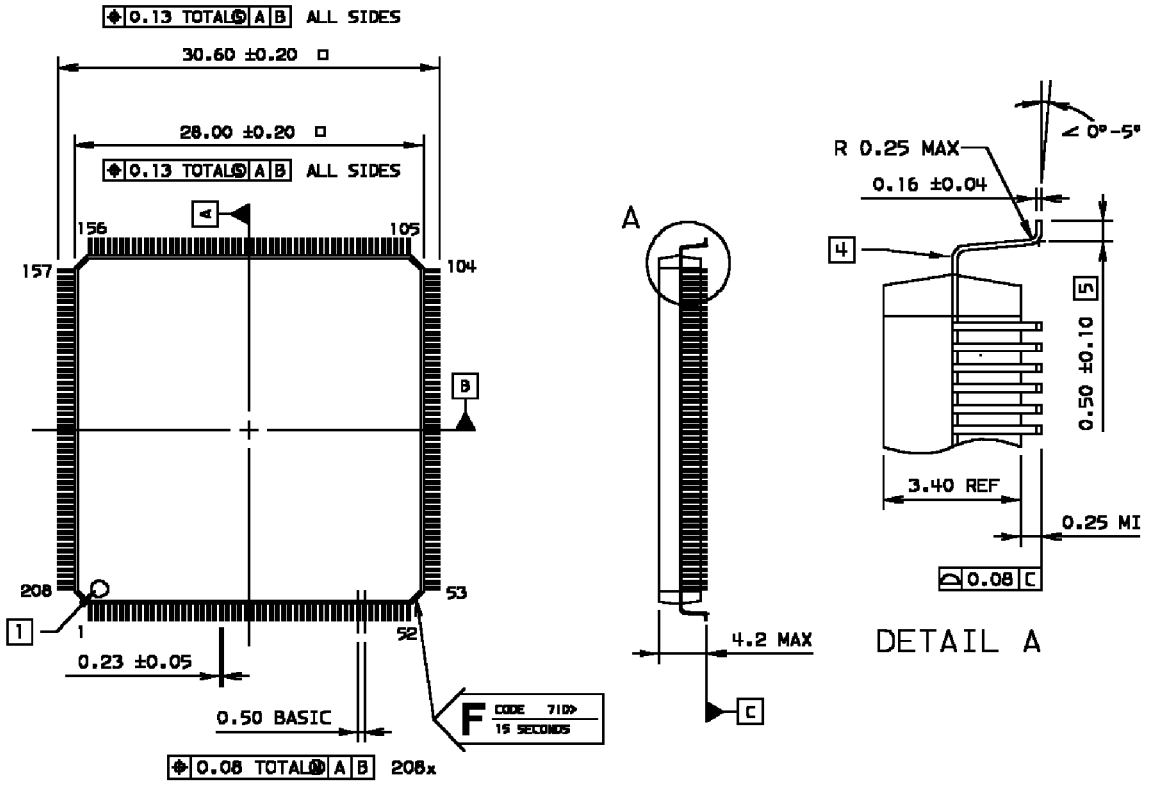
Symbol	Parameter	Conditions	Temperature Range	Min.	Typ.	Max.	Units
V_{IH}	High Level Input Voltage		0°C to 70°C	TBD		TBD	V
V_{IL}	Low Level Input Voltage		0°C to 70°C	TBD		TBD	V
V_{OH}	High Level Output Voltage	V_{CC}=Min, I_{OL}=tbd	0°C to 70°C	TBD			V
V_{OL}	Low Level Output Voltage	V_{CC}=Min, I_{OL}=tbd	0°C to 70°C			TBD	V
I_{OZH}	High Level Off State Output Current	V_{CC}=Max, V_O=V_{CC}(Max)	0°C to 70°C			TBD	µA
I_{OZL}	Low Level Off State Output Current	V_{CC}=Max, V_O=GND	0°C to 70°C			TBD	µA
I_I	Maximum Input Current	V_{CC}=Max, V_I=V_{CC}(Max)	0°C to 70°C			TBD	µA
I_{IH}	High Level Input Current	V_{CC}=Max, V_I=V_{CC}(Max)	0°C to 70°C			TBD	µA
I_{IL}	Low Level Input Current	V_{CC}=Nom, V_O=GND	0°C to 70°C			TBD	µA
I_{OS}	Output Short Circuit Current	V_{CC}=Nom, V_O=GND	0°C to 70°C	TBD		TBD	mA
I_{CC}	Supply Current	V_{CC}=Max	0°C to 70°C			TBD	mA
C_I	Input Capacitance	V_{CC}=Nom, V_I=tbd	0°C to 70°C			TBD	pF
ESD	Electro Static Discharge		0°C to 70°C	-2000		2000	V
PD	Power Dissipation		0°C to 70°C		1000		mW

ElectroMagnetic Compatibility

Physical Design Considerations

Figure 30. Package diagrams

208 Lead QFP



1. VISUAL INDEX AT PIN 1.
2. LEAD FINISH TO BE Sn/Pb ALLOY. MAXIMUM Sn TO BE 90%. THICKNESS TO BE 5.0 - 20.0 MICROMETERS.
3. MARKING: IBM OR SUPPLIER P/N. DATE CODE, AND MFG'S LOGO.
4. FLASH MAY EXTEND TO INNER VERTICAL SECTION OF LEAD.
5. FOOT LENGTH (0.5mm) IS THE HORIZONTAL PROJECTION OF THE DISTANCE FROM THE LEAD TIP TO THE INTER-SECTION OF THE CENTERLINES OF THE LEAD SEGMENTS.
6. DATUMS A & B ARE LOCATED AT THE CENTER OF THE SPACE BETWEEN THE CENTER LEADS AND IS MEASURED AT THE POINT WHERE THE LEADS EXIT THE PLASTIC BODY.
7. APPLICABLE SPECIFICATIONS FOR FINISHED COMPONENTS:
   - A- 68X5655 - MOISTURE SENSITIVE
   - B- 6231587 - GENERAL MECHANICAL
   - C- 23F0325 - PACKAGE FOR SHIPMENT
   - D 2413138 - FLAME APPLICATION PT. FOR FLAMMABILITY SPEC

Acronyms

Table 13: Acronym List

Acronym	Definition
AAL	ATM Adaptation Layer
ANSI	American National Standards Institute
ATM	Asynchronous Transfer Mode
B-ISDN	Broadband ISDN
BECN	Backward Explicit Congestion Notification
BIP	Bit Interleaved Parity
BOM	Beginning Of Message
BTS	Burst Tolerance Specifier
CBR	Constant Bit Rate
CCITT	International Telegraph and Telephone Consultative Committee (ITU)
CDV	Cell Delay Variation
CErr	Cell Error
CHM	Congestion Handling Mode
CLP	Cell Loss Priority
COM	Continuation Of Message
CPCS	Common Part Convergence Sublayer
CRC	Cyclic Redundancy Check
CRD	Control Rate Divider
CS	Convergence Sublayer
CSN	Card Select Number
DMA	Direct Memory Access
EOM	End Of Message
FIFO	First-In/First-Out Queue
FSC	F Symbol Counter
FRAQ	Frequency Acquisition
FRTO	Frame Time Out
GFC	Generic Flow Control
HEC	Header Error Control
INTR	Interrupt Enable
ISC	Invalid Symbol Count
ISDN	Integrated Digital Services Network
ITU-TS	International Telecommunications Union-Telecommunications Standard
LAN	Local Area Network
LC	Logical Channel
LCI	Logical Channel Identifier
LI	Length Indication
LME	Layer Management Entity
LSB	Least Significant Bit
LSS	Least Significant Symbol

Table 13: Acronym List (Continued)

Acronym	Definition
MIS	Missed Service
MISR	Mask Interrupt Status Register
MSB	Most Significant Bit
MSN	Monitoring Sequence Number
MSS	Most Significant Symbol
NNI	Network Node Interface
OAM	Operations and Maintenance
PDU	Protocol Data Unit
PM	Performance Management
PSR	Peak to Sustainable Rate
PT	Payload Type
QOS	Quality Of Service
RCV	Receive
RFL	Receive Free List
ROGB	Receive OAM Cell Buffer
RRL	Receive Ready List
SAP	System Access Port
SDU	Service Data Unit
SErr	Symbol Error
SHC	Software/Hardware Control
SISR	System Interrupt Status Register
SN	Sequence Number
SOC	Start of Cell
SOF	Start of Frame
SRFL	System Receive Free List
SRTS	Sync Remote Time Stamp
SSM	Single Segment Message
SS–SOC	Scrambler Sync – Start of Cell
TCA	Threshold Crossing Alert
TCB	Transmit Cell Buffer
TCL	Transmit Complete List
TCSEL	Transmit Clock Select
TMQ	Traffic Management Queue
TM	Time Marker
TRQ	Transmit Ready Queue
TUC	Total User Cells
TX	Transmit
UIC	User Information Cell
UNI	User-Network Interface
VBR	Variable Bit Rate

Table 13: Acronym List (Continued)

Acronym	Definition
VC	Virtual Channel
VCC	Virtual Channel Connection
VCI	Virtual Channel Identifier
VP	Virtual Path
VPC	Virtual Path Connection
VPI	Virtual Path Identifier
XMIT	Transmit