Order this document by
MC92501UM/A

Advance Information

MC92501

ATM Cell Processor
User's Manual

MOTOROLA

Preliminary

Preliminary

Section 1
Introduction

Section 2
Functional Description

Section 3
System Operation

Section 4
MC92501 External Interfaces

Section 5
MC92501 Data Path Operation

Section 6
MC92501 Protocol Processing Support

Section 7
MC92501 Programming Model

Section 8
Test Operation

Section 9
MC92501 Product Specifications

Appendix A
UPC/NPC Design

Appendix B
Maintenance Slot Calculations

Appendix C
External Memory Calculations

Appendix D
Tutorial

Appendix E

MC92501 Applications

Appendix F
BSDL Code

Appendix G
References

List of Figures

List of Figures

List of Tables

1 Introduction

1.1 ATM Networks

The increa

sing demand for broadband data transmission has led to the development of international protocols issued by the American National Standards Institute (ANSI), International Telecommunications Union-Telecommunications (ITU-T), and the European Telecommunication Standards Institute (ETSI). These standards and the recommendations issued by the ATM Forum, Bellcore, UTOPIA, and others are provided to ensure consistent, reliable, and uninterrupted data transmission world-wide.

The basic unit of transfer defined by these protocols is called a cell. Each cell is composed of 53 bytes that includes a 5-byte header and 48 bytes of data. The header portion of the cell includes several sections of identifying information used by the cell processor to route and transmit the cell, including the type of data being transmitted and the origin and destination. As with many other means of electronic transfer, the cells of information must be combined with other cells (for efficiency of transfer) and transmitted over a long distance. During transfer, the cells are evaluated and, if required, separated from the transmission stream and rerouted, similar to commuters who share transportation routes during parts of their journeys and then transfer to alternate routes to complete their journeys.

Actual data transmission is performed asynchronously because the distance over which the transfer occurs prevent the use of the same clock signal at both ends and even within the transfer network. Therefore, the transfer method has been defined as the Asynchronous Transfer Mode (ATM). Each network consists of user end stations that transmit and receive the 53-byte data cells using virtual connections. The virtual connections are implemented through physical links and switching elements that interconnect them. The specific combination of physical links that implements a virtual connection is chosen when the connection is established. For a given physical link, each connection is assigned a unique connection identifier. The connection identifier is placed in the header of each cell by the transmitting equipment and is used by the receiving equipment to route the cell to the next physical link on the connection path. All cells belonging to a specific virtual connection follow the identical path from the transmitting end station through the switching elements to the receiving end station. **Figure 1-1** shows a typical ATM network.

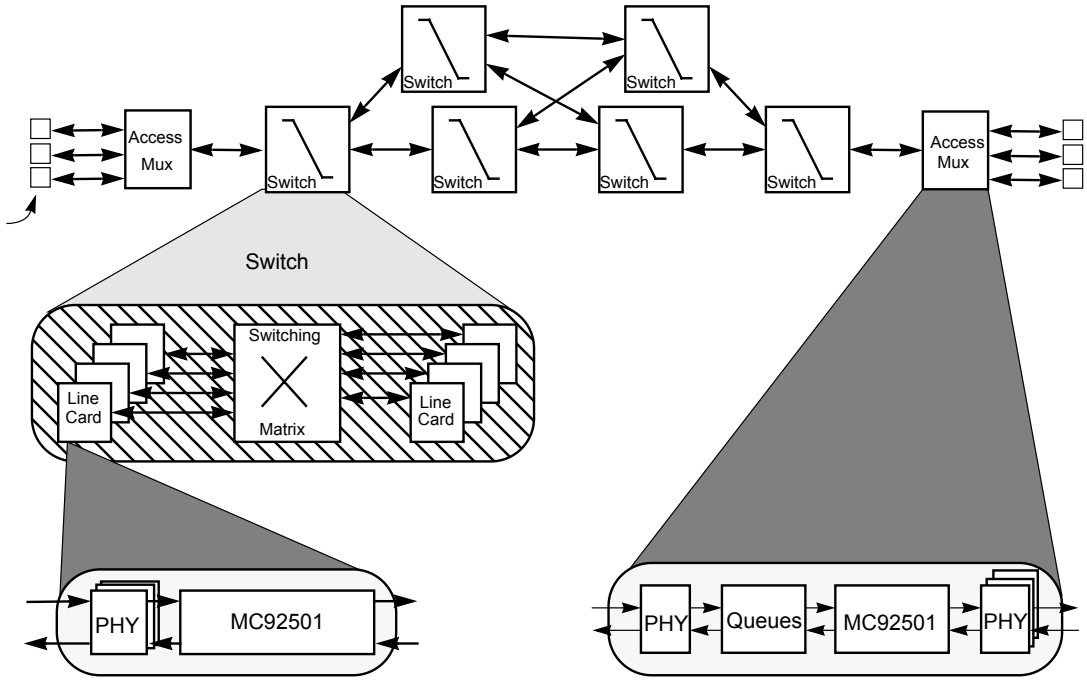


Figure 1-1. Typical ATM Network Structure

As described above, an ATM network is composed primarily of switching elements. Each switching element handles multiple physical links. For an individual ATM cell, a combination of these switching elements route the cell from its source link to its destination link according to a pre-arranged route for the connection to which the cell belongs.

Within the ATM network core, the switches route multiple links to multiple links. A typical ATM core switch consists of a switch matrix and some line cards, one card for each physical link (or group of links). The switch matrix handles the actual routing of the cells. The line cards interface between the physical signal lines and the switch matrix. A line card extracts incoming cells from the arriving bit stream and converts outgoing cells into a bit stream for transmission. The line card also performs ATM layer functions.

At the edges of the ATM network, access multiplexers route a single link to multiple links. An access multiplexer is typically connected to one Physical (PHY) layer device on the network side and to multiple PHY layer devices on the subscriber side.

1.2 MC92501 ATM Cell Processing Device

As shown in **Figure 1-1**, the MC92501 ATM cell processor can be used both in the line cards used by the switching systems in the ATM network core and in the access multiplexer. The primary function of the MC92501 in either application is to provide ATM-layer cell processing and routing functions.

1.2.1 ATM Network Line Card

In a typical line card the MC92501 serves as an interface between a physical layer device and an ATM switch matrix, as shown in **Figure 1-2**. The MC92501 uses a fast External Memory for storing the ATM virtual connection information for the cells it processes. The microprocessor is used for configuration, control and status monitoring of the MC92501 and is responsible for initializing and maintaining the External Memory. The MC92501 is the master of the External Memory bus. At regular intervals the MC92501 allows the microprocessor to access the External Memory for updating and maintenance.

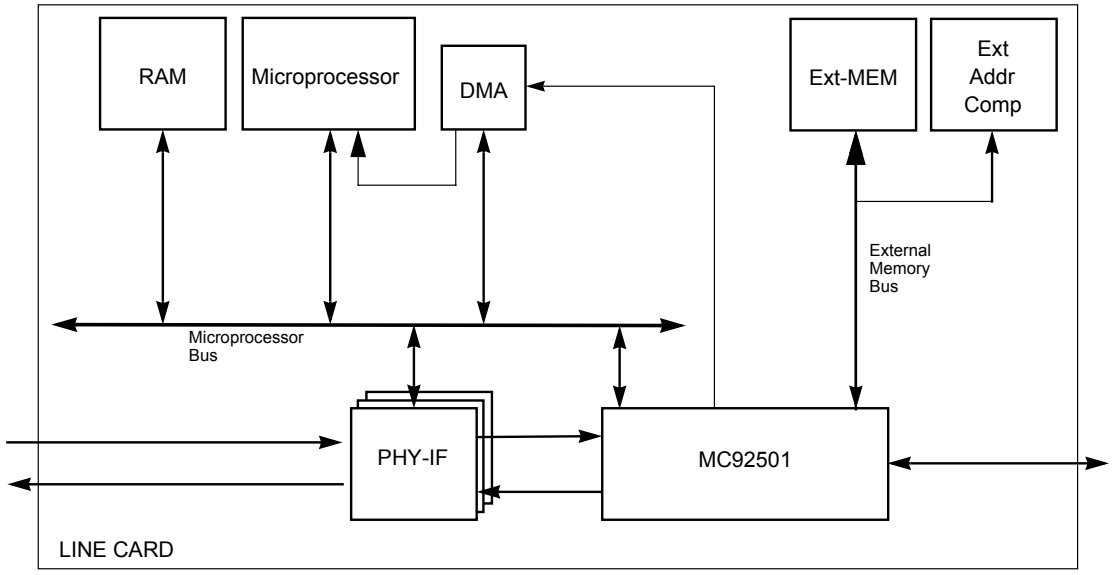


Figure 1-2. Typical ATM Line Card Application

The physical interface (PHY-IF) implements the physical layer functions of the Broadband ISDN (B-ISDN) Protocol Reference Model. This includes the physical medium dependent functions required to transport ATM cells between the ATM user and the ATM switch (i.e., a User Network Interface or UNI) or between two ATM switches (i.e., a Network Node Interface or NNI). The cells are transferred between the physical interface and the MC92501 using the UTOPIA standard. The MC92501 implements ATM-layer functions required to transfer cells to and from the switch over virtual connections. These functions include usage enforcement, address translation, and Operation and Maintenance (OAM) processing.

1.2.2 ATM Network Access Multiplexer

Figure 1-3 shows two example configurations for ATM network access multiplexers using an MC92501 ATM cell processor. In both designs, the MC92501 interfaces the PHY layer devices and the queuing point with the MC92501 switch interface connected to the queues.

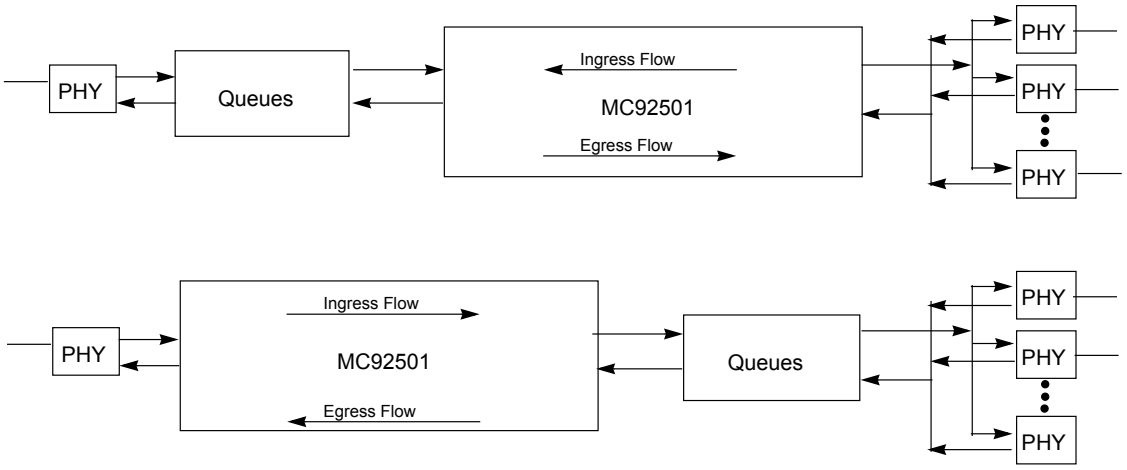


Figure 1-3. MC92501 Network Access Multiplexer Design Examples

1.3 MC92501 Features

- Implements ATM Layer functions for Broadband ISDN according to ANSI Recommendations, ATM Forum Specifications and Bellcore recommendations.
- Provides a throughput capacity of up to 155 Mbps in each direction
- Processes ATM cells from a SONET STS-3c, SONET STS-1, DS3 PLCP, or any other physical link running at up to 155 Mbps
- Optionally supports multiple (up to sixteen) physical links
- Optionally configured as a User Network Interface (UNI) or Network Node Interface (NNI) on a per-link basis
- Operates in conjunction with an External Memory (up to 16 MB) to provide context management for up to 64 K Virtual Connections
- Provides explicit bank select signals to support up to four banks of External Memory
- Provides per-connection cell counters with the ability to maintain multiple copies of the counter tables and dynamically switch between them
- Provides per-link cell counters in both directions
- Provides per-connection Usage Parameter Control (UPC) or Network Parameter Control (NPC) using a Leaky Bucket design with up to 4 buckets per connection.

- Supports packet based UPC using three methods: PPD, EPD and limited EPD
- Supports selective discard on CLP = 1 and CLP = 0 + 1 on selected connections
- Provides support for Operation and Maintenance (OAM) Continuity Check function for all connections
- Supports Virtual Path (VP) and Virtual Channel (VC) level Alarm Surveillance on all connections using an internal scan process to generate and insert OAM cells
- Supports OAM Fault Management Loopback test on all connections
- Supports bidirectional OAM Performance Monitoring on all connections
- Supports Relative Rate ABR on all connections
- Supports changing RM cells priority by marking the Ingress switch parameters.
- Supports CLP Transparency.
- Provides a slave microprocessor interface including a 32-bit data bus.
- Provides indirect access to its External Memory.
- Provides byte-swapping on cell payloads to and from the microprocessor bus in order to support both big-endian and little-endian buses
- Supports cell insertion into the cell streams using direct access registers which may be written by the microprocessor or by a DMA device
- Supports copying cells from the cell streams using direct access registers which may be read by the microprocessor or by a DMA device
- Glueless connection to the MC68360 (QUICC).
- Enables masking of overhead ECI and MTTS fields before use.
- Supports multicast operation
- 256-pin OMPAC package
- Uses 3.3 V power supply
- Input pins are 5 V tolerant

1.3.1 Ingress

The Ingress section (Ingress refers to cells being transferred from the physical interface to the switch):

- Interfaces to one or more physical interface chips via an 8-bit wide, parity-protected receive data bus using either UTOPIA level 1 or UTOPIA level 2 standard interface.
- Decouples PHY timing from switch timing using independent clocks and a FIFO in the physical interface
- Performs Ingress cell discrimination based on pre-assigned ATM cell header values
- Provides either a restricted address table lookup scheme for Ingress address compression or support for an external address compression mechanism
- Reads Virtual Connection related UPC/NPC, OAM and switch context parameters through a 32-bit wide interface to an External Memory
- Provides per-connection usage count
- Provides per-connection option to copy cells to the microprocessor
- Provides per-connection UPC/NPC policing including detection/counting of violating cells
- Supports packet based UPC using three methods: PPD, EPD and limited EPD
- Supports OAM Continuity Check, Alarm Surveillance, Loopback test and Performance monitoring on all connections
- Supports Relative Rate ABR on all connections.
- Supports changing RM cells priority by marking the ingress switch parameters.
- Supports CLP Transparency.
- Supports insertion of cells into the Ingress cell flow
- Optionally performs VPI/VCI translation
- Forwards the received ATM cells to the switch using a UTOPIA-style interface, optionally adding associated internal switch context parameters
- Delay of 3–5 cell times from the PHY to the switch

1.3.2 Egress

The Egress section (Egress refers to cells being transferred from the switch to the physical interface):

- Receives ATM cells and associated switch context parameters (including congestion notification) from the switch using a UTOPIA-style interface
- Provides optional multicast identifier to connection identifier translation
- Reads Virtual Connection related UPC/NPC, OAM and Address Translation parameters through a 32-bit wide interface to an External Memory
- Provides per-connection usage count
- Provides per-connection option to copy cells to the microprocessor
- Provides per-connection UPC/NPC policing including detection/counting of violating cells
- Supports packet based UPC using three methods: PPD, EPD and limited EPD
- Supports OAM Continuity Check, Alarm Surveillance, Loopback and Performance Monitoring test on all connections
- Supports Relative Rate ABR on all connections.
- Supports CLP Transparency
- Supports insertion of cells into the Egress cell flow
- Performs VPI/VCI translation
- Transfers ATM cells to one or more physical interfaces via an 8-bit wide, parity-protected transmit data bus using the UTOPIA standard
- Decouples PHY timing from switch timing using independent clocks and a FIFO in the physical interface
- Delay of 3–5 cell times from the switch to the PHY

1.4 The MC92500 Versus the MC92501

The MC92501 is an updated version of the MC92500. Motorola's ATM strategy is to update products in accordance with the latest technology standards while maintaining *backward compatibility*[1]. The MC92501 adds the following features to those provided by the MC92500:

- OAM Performance monitoring on all connections
- Egress UPC
- UPC discard triggers AAL5 packet discard (implementing both PPD and EPD)
- Selective discard CLP = 1 (or CLP = 0 + 1) flow on selected connections
- ABR RR-marking and EFCI-marking according to TM-4.0
- Change ABR RM cells priority
- Support for CLP transparency
- Indirect External Memory access
- Improved host interface
- Enable masking overhead Egress Connection Identifier (ECI) and Multicast Translation Table Section (MTTS) fields before use
- UTOPIA level 2 PHY interface
- Egress GFC transparency for ECI on header mode (IHAF = 1)

1.5 Notation Conventions

The following notation conventions are used in this document:

- Signal representations:
  - Active high signals (Logic One) use the signal name (e.g., A0).
  - Active low signals (Logic Zero) use the signal name with an overbar (e.g., $\overline{\text{WE}}$ or $\overline{\text{OE}}$).
- Numeric representations:
  - Hexadecimal values start with a "$" sign (e.g., $0FF0 or $80).
  - Decimal values have no symbol attached to the number (e.g., 10 or 34).
  - Binary values start with the letter "b" (e.g., b1010 or b0011).

1. The only exception is the Boundary Scan Register bit definition due to the addition of pins.

2

Functional Description

2.1 Introduction

The MC92501 implements Broadband ISDN (B-ISDN) User Network Interface/Network Node Interface (UNI/NNI) Asynchronous Transfer Mode (ATM) Layer functions and provides context management for up to 64 K Virtual Connections (VCs), either Virtual Channel Connections (VCCs) or Virtual Path Connections (VPCs). ATM cells belonging to a particular VCC on a logical link have the same unique VPI/VCI value in the cell header. Similarly, cells with the same VPC on the same logical link share a unique VPI.

2.2 System Functional Description

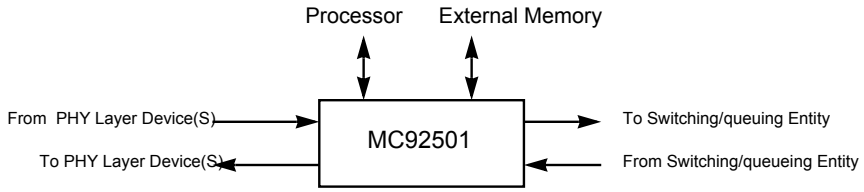


Figure 2-1. ATMC Schematic Interface

The Asynchronous Transmission Mode Cell processor (ATMC) is always placed between the PHY layer device(s) and the switching/queuing entity. The PHY device links serial transmissions operating at up to 155.52 Mbps with the byte-based interface to the MC92501 via internal FIFO cell buffers. The microprocessor initializes and provides real-time control of the data-flow chips (PHY and MC92501) using slave accesses. The MC92501 operates in conjunction with External Memory that provides one context entry for each active connection. The entry consists of two types of context parameters:

- **Static**—These parameters are loaded into the context memory when the VC is established and are valid for that connection duration. Static parameters include traffic descriptors, OAM flags, and parameters used by the ATM switch.
- **Dynamic**—These context parameters (which include cell counters, UPC/NPC fields and OAM parameters) may be modified while cells belonging to that particular connection are processed by the MC92501.

The microprocessor also accesses the External Memory from time to time through the MC92501 to collect traffic statistics and update the OAM parameters.

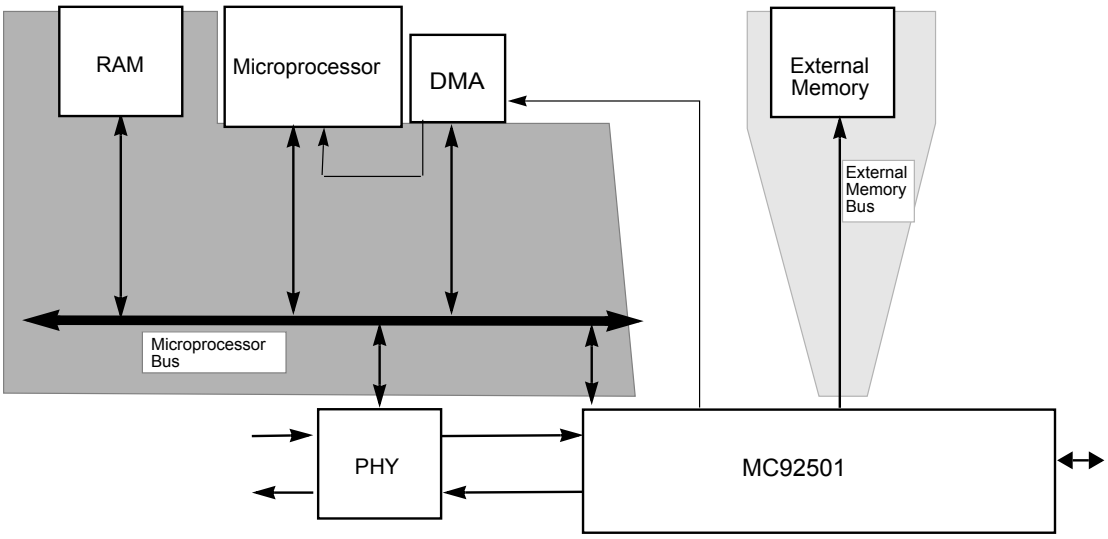


Figure 2-2. ATMC System During Cell Processing

During normal cell processing, the MC92501 has exclusive access to the External Memory (as shown in **Figure 2-2**). The context entries for the cells being processed are read and the updated dynamic parameters are written back. The MC92501 is responsible for the coherency of the External Memory during this time. At user-programmable intervals the MC92501 provides a *Maintenance Slot* for the microprocessor during which no cell processing is done and the microprocessor may access the External Memory (see **Figure 2-3**). The break in cell processing is possible because of the difference between the MC92501 cell-processing rate and the line rate.

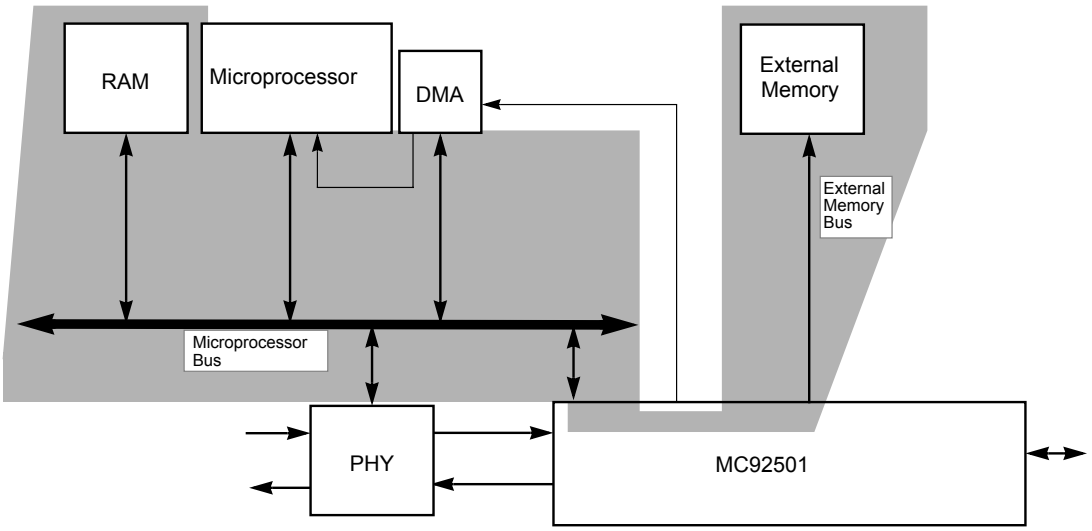


Figure 2-3. ATMC System During Maintenance Operation

The Maintenance Slot can be used by the microprocessor for one or more of the following tasks:

- Connection setup and tear down
- Statistics collection
- Updating OAM parameters of active connections.

The microprocessor is responsible for guaranteeing the coherence of the External Memory at the end of each Maintenance Slot.

2.3 MC92501 Functional Description

The primary functions performed by the MC92501 depend upon its placement in the ATM system, that is, whether it is being used as a cell receiver (Ingress Cell Flow) or a cell transmitter (Egress Cell Flow). In either operating mode, the MC92501 interfaces as a slave to the system microprocessor and can be accessed for in-circuit testing.

2.3.1 Ingress Cell Flow

In the Ingress direction, the MC92501 receives cells from the FIFO in the PHY. Cell discrimination is performed and uses pre-defined header field values to recognize unassigned and invalid cells. Unassigned and invalid cell slots may be used to insert OAM and messaging cells into the Ingress cell flow. Cell rate decoupling is accomplished by discarding unassigned cells.

For VCCs, the 28-bit VPI/VCI address space (32-bit Link/VPI/VCI if multiple physical links are supported) needs to be compressed into a 16-bit Ingress Connection Identifier (ICI). The MC92501 provides two methods for performing VCC address compression to generate the ICI:

- Table lookup based on reduced addressing
- External address compression

For VPCs, the VPI field is used for a lookup into the VP Table to obtain the ICI. The ICI is a pointer used to access the context parameters for the current Ingress cell from the external context memory. Included in these parameters are cell counters, UPC/NPC traffic descriptor, OAM parameters and switch parameters.

The UPC/NPC mechanism counts the arriving cells and, using a flexible arrangement of traffic enforcement algorithms, admits cells that do not violate the traffic characteristics established for that connection. Violating cells are tallied and may be tagged or discarded (i.e., removed from the cell flow).

The OAM parameters are used to control when and how OAM cells are processed and to indicate if the current user cell belongs to a connection selected for a Performance Monitoring test. If the Ingress cell belongs to such a connection, the OAM table in External Memory contains the relevant parameters.

Subsequent to the context processing, the Ingress cells are transferred to the switch. Optionally, the associated switch context parameters may be added to the cell before the header and/or placed in the VPI/VCI fields of the header.

2.3.2 Egress Cell Flow

In the Egress direction, the MC92501 receives cells from the switch along with any associated parameters. One parameter is the Egress Connection Identifier (ECI), which is used for direct lookup into the context table to obtain the VPI/VCI, cell counters, and OAM flags. If multicast translation is enabled, the Multicast Identifier (MI) is received from the switch instead of the ECI, and the ECI is obtained from a lookup in the Multicast Translation Table. If enabled, the UPC function is executed. Cells are subject to processing as indicated by the OAM flags. If the Egress cell belongs to a connection that has been selected for a Performance Monitoring test, the OAM Table in External Memory contains the relevant parameters.

The Egress cell header is generated by inserting the VPI/VCI-field obtained from the Address Translation Table in the (GFC)/VPI/VCI position and modifying the PTI-field, if and when indicated by the switch or in case of an OAM cell. The cell is then forwarded to the PHY queue. Cell rate decoupling is performed in the Egress direction (i.e., unassigned cells are optionally generated if no cells are available from the switch).

2.3.3 Other Functions

A general 32-bit slave system interface is provided for configuration, control, status monitoring, and insertion and extraction of cells. This interface provides for direct register access to the MC92501. The MC92501 also includes a standard JTAG Boundary Scan Architecture Test Access Port (TAP) for testing.

2.4 MC92501 Block Diagram

Figure 2-4 contains a block diagram of the MC92501. The individual blocks are described in this section.

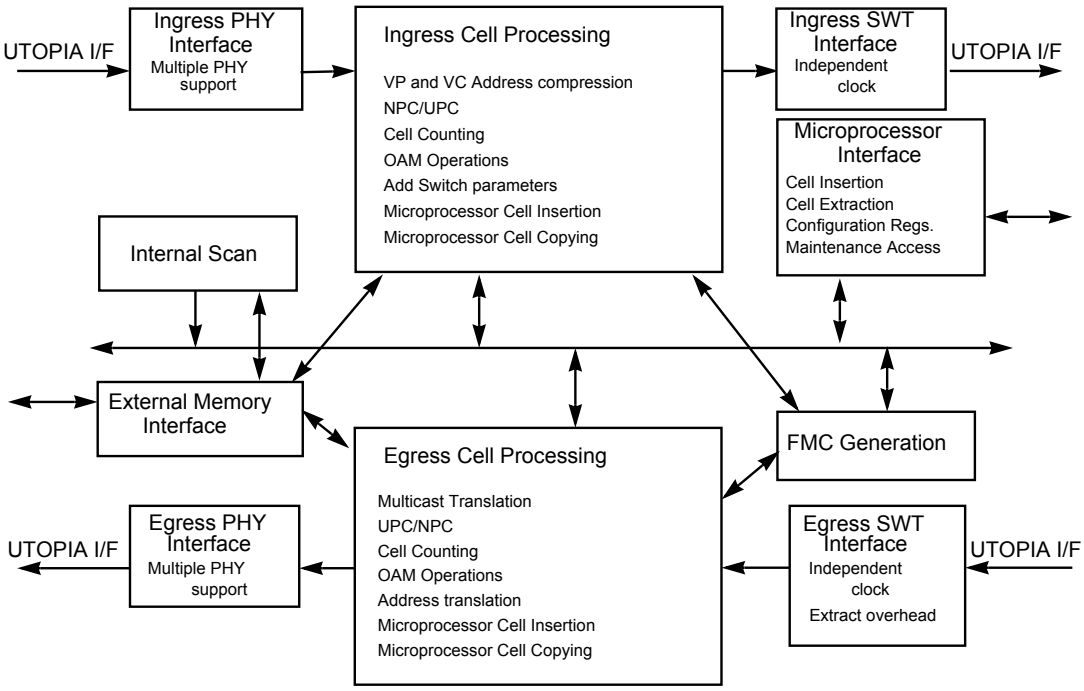


Figure 2-4. MC92501 Block Diagram

2.4.1 Ingress PHY Interface (IPHI)

The Ingress PHY Interface (IPHI) block receives cells on a byte basis from the ATM PHY layer using the UTOPIA standard interface. It assembles the cells and synchronizes their arrival to the MC92501 cell processing slots. Unassigned and invalid cells (see **Table 5-1** and **Table 5-2**) are removed to provide cell rate decoupling. Also, the MC92501 can process cells at a higher rate than the PHY provides them, so there are always some "holes" in the cell flow. These can be used either for cell insertion or for maintenance accesses (used by the microprocessor to maintain the External Memory).

2.4.2 Ingress Cell Processing Unit (IPU)

The Ingress Cell Processing Unit (IPU) operates at a rate of one cell per cell processing slot. The cell may have arrived from the IPHI block or may be inserted from the Microprocessor Interface or Internal Scan blocks. The Ingress OAM function may also insert PM Forward Monitoring cells into the Ingress cell flow. A cell may be inserted when an unused cell slot is available, subject to pacing by a simple Leaky Bucket algorithm.

The IPU performs the following tasks:

- The IPU performs address compression on cells that arrived from the IPHI block in order to associate the cell with Context Table records in the MC92501 External Memory. The address compression function detects inactive cells (cells with no corresponding records in the Context Table).
- UPC/NPC is performed on a connection basis, or optionally on arbitrary groups of connections. The UPC/NPC function may detect violating cells as dictated by the selected UPC/NPC algorithm. Violating cells are normally tagged or removed from the cell flow, but an option exists to perform the UPC/NPC algorithm for statistical purposes only without modifying or removing the cells.
- OAM processing is performed where appropriate. The Ingress OAM function records OAM Alarm cells (AIS/RDI). OAM processing for user cells involved in a Performance Monitoring block test involves computing the Bit-Interleaved Parity (BIP) and updating the Total User Cells (TUC) count. For OAM cells the processing may include overwriting the values of specific fields and checking or generating the CRC-10 field.
- Switch-specific overhead information is read from the context entry and added to the cell before it is sent on to the switch interface block. Address translation may optionally be performed at this point.
- The IPU removes any OAM cell that has reached its end-point from the cell flow.
- Certain cells may be copied to the MPIF for transfer to the microprocessor.

2.4.3 Ingress Switch Interface (ISWI)

The ISWI block contains a cell FIFO. Cells are received from the IPU. When a full cell has been transferred, the overhead information needed by the switch (as programmed by the user) is extracted from the internal data structure along with the ATM header and payload of the cell. This information is transferred to the switch at the rate of one byte per clock cycle.

2.4.4 Egress Switch Interface (ESWI)

The ESWI block contains a cell FIFO. Data is received from the switch at the rate of one byte per clock cycle. The data structure received from the switch includes overhead routing information in addition to the ATM cell. When a full cell has been transferred, it is transformed into an internal data structure and presented to the EPU for processing.

2.4.5 Egress Cell Processing Unit (EPU)

The Egress Cell Processing Unit (EPU) operates at the rate of one cell per cell processing slot. The cell may have arrived from the CSL switch interface block or may have been inserted from the Microprocessor Interface or Internal Scan blocks. The Egress OAM function may also insert PM Forward Monitoring cells into the Egress cell flow. The cell insertion is paced by a simple Leaky Bucket algorithm.

The EPU performs the following tasks:

- The first stage of the Egress cell processing is performing multicast translation, if needed.

- Optionally, the EPU performs UPC/NPC.
- Where appropriate, the EPU performs OAM processing. The Egress OAM function records OAM Alarm cells. OAM processing for user cells involved in a Performance Monitoring block test is limited to computing the Bit-Interleaved Parity (BIP) and updating the Total User Cells (TUC) count. For OAM cells the processing may include overwriting the values of specific fields and checking or generating the CRC-10 field.
- Address translation is performed to replace the address fields of the ATM cell header with the address of the outgoing link.
- The EPU removes any OAM cell that has reached its end point from the cell flow.
- Certain cells may be copied to the MPIF for transfer to the microprocessor.

2.4.6 Egress PHY Interface (EPHI)

The Egress PHY Interface (EPHI) block takes the processed cells from the EPU, disassembles them into bytes, and transfers them to the physical layer using the UTOPIA standard interface. Unassigned cells may be inserted to provide cell rate decoupling.

2.4.7 External Memory Interface (EMIF)

The External Memory Interface (EMIF) block performs address generation for the MC92501 accesses to the External Memory. It also drives the memory control signals.

2.4.8 Microprocessor Interface (MPIF)

The Microprocessor Interface (MPIF) block provides for configuration of the MC92501, the transfer of cells between the microprocessor and the MC92501, and the maintenance of the External Memory. A standard 32-bit slave interface is provided for easy connection to the microprocessor bus. Output signals are provided that can serve as request signals for up to three DMA devices to improve system performance. The Cell Extraction Queue is used to store cells that are directed to the processor. Cells in this queue are transferred first to an internal cell buffer. Then they may be read by the processor. Cells to be inserted in the Ingress or Egress flows are transferred from the processor memory to an internal cell buffer.

2.4.9 Internal Scan (ISCAN)

The Internal Scan (ISCAN) block scans the External Memory for connections on which Alarm Indication Signal (AIS), Remote Defect Indicator (RDI), or Continuity Check (CC) OAM cells must be inserted. When such a connection is found, the cell is generated and added to the insertion queue for the cell flow in the appropriate direction.

2.4.10 FMC Generation

The Forward Monitoring Cell (FMC) Generation block keeps track of the connections on which FMCs are pending during the course of a Performance Monitoring block test and maintains a priority among them. When a hole in the cell flow is available, this block requests the insertion of an FMC on the highest-priority connection.

3

System Operation

3.1 MC92501 Modes of Operation

The MC92501 has two basic modes of operation:

- Setup Mode
- Operate Mode

These modes are described in detail in the following sections.

3.1.1 Setup Mode

The MC92501 enters Setup Mode after reset. This mode is used to configure the MC92501 and to initialize the External Memory. Since the MC92501 does not use the External Memory in this mode, the microprocessor and/or DMA device have unrestricted access. In addition, those registers identified as configuration registers (see **Section 7.2.6** Configuration Registers) may be written only when the MC92501 is in Setup Mode. A write access to any of these registers in Operate Mode is forbidden.

When the MC92501 is in Setup Mode, the Receive PHY Interface is disabled by keeping the RXENB output signal deasserted. The Egress Switch Interface is disabled by keeping STXCLAV deasserted.Unassigned cells (if and when enabled—see **Section 7.2.6.4** Egress PHY Configuration Register (EPHCR)) are provided to the PHY layer. No cells are provided to the Ingress Switch Interface. Cell insertion is not allowed in Setup Mode.

NOTE: Once the MC92501 has switched from Setup Mode to Operate Mode, it does not return to Setup Mode until a hardware or software reset is performed.

3.1.2 Operate Mode

Before switching to Operate Mode, the *Maintenance Period Length (MPL)* field of the Maintenance Control Register (MACTLR) should be defined with a non-zero value. Once this field has been defined, a write access to the Enter Operate Mode Register (EOMR) transfers the MC92501 to Operate Mode within 1–3 cell slots.

NOTE: While in Operate Mode, the MC92501 processes the cells received from the PHY layer and the switch interface block. The configuration registers may be read, but not written, when the MC92501 is in Operate Mode.

3.1.3 Reset

The MC92501 can be reset in either of two ways:

- Hardware reset by asserting the ATMC Power-Up Reset ($\overline{\text{ARST}}$) pin
- Software reset by writing to the Software Reset Register (SRR).

In either case all the internal registers are loaded with their default values. The reset process requires 200 ACLK cycles or 200 MCLK cycles, whichever is greater, after the deassertion of ARST or the write to the SRR. During this time, writing to the MC92501 registers is not allowed. At the conclusion of reset, the MC92501 is in Setup Mode. When a write access to the SRR is performed, the MC92501 begins the reset process only after writing the results of the current cell processing to External Memory. In this way the External Memory remains consistent, and it is not necessary to re-initialize the External Memory when performing a software reset of the MC92501.

During reset, the $\overline{\text{MREQ0}}$, $\overline{\text{MREQ1}}$, and $\overline{\text{MREQ2}}$ signals are configured as $\overline{\text{MCIREQ}}$, MCOREQ, and $\overline{\text{EMMREQ}}$ DMA requests, respectively. The following describes the status of the DMA request signals during and after reset:

- $\overline{\text{EMMREQ}}$—Asserted (pulled low) during and after reset until either a non-zero value is written to the NMR field of the MPCTLR or the MC92501 enters Operate Mode. See **Section 7.2.5.1** Microprocessor Control Register (MPCTLR).
- $\overline{\text{MCIREQ}}$—Deasserted (pulled high) during and after reset. It is asserted upon entering Operate Mode.
- MCOREQ—Deasserted (pulled high) during and after reset until a cell is available for extraction.

3.2 Data Path Clock Configuration

In the MC92501 design, the PHY data path interfaces operate using the same clock. The UTOPIA standard requires the ATM layer to provide the interface clocks, RXCLK and TxClk. Since the PHY interfaces use ACLK, the clock provided to the MC92501, connect the same clock to the RXCLK and TXCLK pins of the PHY component.Therefore, these clock signals of the UTOPIA interface are not explicitly provided by the MC92501.The switch interfaces are independently clocked by the clock signals connected to the SRXCLK and STXCLK pins.This configuration is shown in **Figure 3-1**.

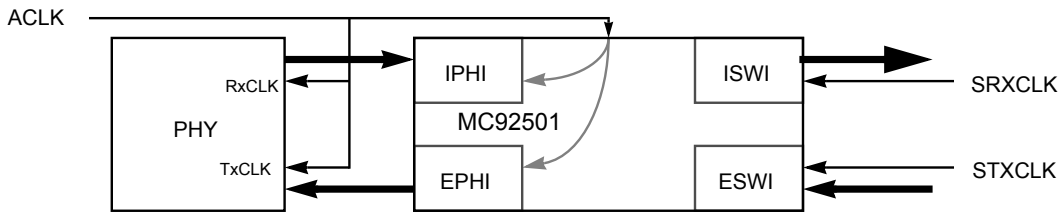


Figure 3-1. MC92501 Clock Configuration

3.3 External Memory Maintenance

The Flags Table in the MC92501 External Memory should be read and reset by the microprocessor on a regular basis. Current OAM standards require response times to failure conditions to be on the order of one second or less. Therefore, the entire Flags Table should be covered at least once per second.

The Counter Tables in the MC92501 External Memory should also be read (and optionally cleared) on a regular basis to prevent a loss of information. The counters are 32 bits wide. This provides more than 3 hours before the counters wrap around even on a connection that uses the entire bandwidth. The pointers to the Counter Tables are control registers, rather than configuration registers, so that the location of the Counter Tables can be changed during normal operation. This allows the entire array of counters to be "sampled" at a given moment and read as the system bandwidth allows.

NOTE: All of the connection cell counters wrap around to zero at their maximum value.

There are 2 methods according to which the microprocessor can access the MC92501 External Memory without interfering with the MC92501 cell processing. According to the first method the microprocessor accesses the ATMC on regular maintenance slots. According to the second method the microprocessor accesses the ATMC indirectly. The first method is described in sections **Section 3.3.1** Maintenance Slots, **Section 3.3.2** Maintenance Slot Parameters, and **Section 3.3.3** Maintenance Slot Structure. The second method is described in **Section 3.3.4** Indirect Memory Access.

3.3.1 Maintenance Slots

the MC92501 provides Maintenance Slots, during which no cell processing is performed, on a regular basis. Interrupting the cell processing is permissible because the MC92501 processes cells at a higher rate than they are transferred at the PHY layer. These breaks is not visible at the PHY-layer interfaces because the delay is absorbed by the MC92501's interface FIFOs.

The maintenance slots provided by the MC92501 can be used for:

- Reading and clearing the Flags Table
- Reading and clearing the Counter Tables
- Updating context parameters (e.g., Send_AIS, Send_RDI)
- Setup and tear down of connections involves reading and writing any or all of External Memory tables

The MC92501 provides output signals to inform the microprocessor and/or DMA device of an upcoming Maintenance Slot. **Figure 3-2** illustrates a typical architecture using a DMA device.

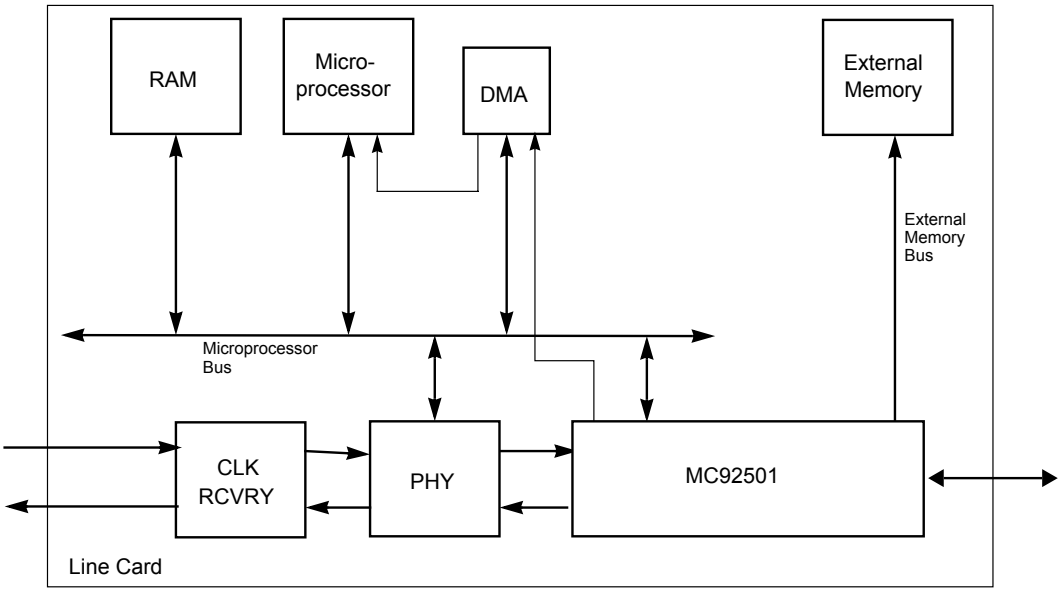


Figure 3-2. ATM Line Card Architecture with DMA Device

3.3.2 Maintenance Slot Parameters

The duration of a Maintenance Slot is the same as a cell slot (i.e., one cell processing time). A cell processing time is defined as the period of ACLK multiplied by 64 (the number of clocks used to process each cell). The rate at which maintenance slots are provided is programmable in terms of the *Maintenance Period Length (MPL)* (the number of cell slots between maintenance slots). See **Section 7.2.5.2** Maintenance Control Register (MACTLR). The maximum number of cell slots the MC92501 uses to process the cells it receives from the PHY layer is determined by the link rate. An STS-3c link provides up to 353,208 cells per second. On the other hand, the number of available MC92501 cell processing slots per second is the inverse of the cell processing time, or the ACLK frequency divided by 64. For example, if the ACLK frequency is 25 MHz, there are 390,625 cell slots per second. The difference between the total number of cell slots and the number used for processing arriving cells provides free cell slots that are usually used as maintenance slots. However, if the cell insertion rate must be higher than the number of holes in the link rate cell flow, additional free cell slots may be used for processing inserted cells. It is imperative that the ACLK frequency be high enough (relative to the link rate) to provide sufficient free slots for both of these purposes. The division of the free slots among the two purposes is determined by the value of the *Maintenance Period Length (MPL)* field. In our example of ACLK running at 25 MHz, there are 390,625 – 353,208 = 37,417 free slots per second. If the MPL is programmed as 19 (every 20th slot is a Maintenance Slot), there are 390,625 / 20 = 19,531 maintenance slots provided per second, leaving 37,417 – 19,531 = 17,886 empty slots per second to be used for cell insertion. Alternatively, if the MPL is programmed as 63, there are 390,625 / 64 = 6103 maintenance slots provided per second, leaving 37,417 – 6103 = 31,314 empty slots per second to be used for cell insertion.

NOTE: **Appendix B** provides a table of results of the Maintenance Slot calculations for specific frequencies, as well as an explanation of the equations used for the calculations.

3.3.3 Maintenance Slot Structure

As mentioned above, the MC92501 provides three signals to assist the microprocessor in using the maintenance slots. Each of these DMA signals can be programmed to function as the External Memory Maintenance Request (EMMREQ) and as such may be connected to a DMA device as a request signal. EMMREQ assertion is synchronized to MCLK. The input to the synchronization is asserted a programmable number of clock cycles before the start of a Maintenance Slot as defined by the *Maintenance Slot DMA Request (MSDR)* field of the Maintenance Configuration Register (MACONR). EMMREQ is deasserted after a programmable number of maintenance accesses as defined by the *Deassertion of Maintenance Request (DMR)* field of the Microprocessor Control Register (MPCTLR) have been performed or at the end of the Maintenance Slot, whichever occurs first.

Alternatively, if no DMA device is being used, the *Maintenance Slot Enable Interrupt Enable (MSEE)* interrupt may be enabled. This interrupt is used to notify the microprocessor that a Maintenance Slot is imminent and is asserted a programmable number of clock cycles before the start of a maintenance slot as defined by the *Maintenance Slot Interrupt Request (MSIR)* field of the Maintenance Configuration Register (MACONR). Upon receiving the interrupt, the microprocessor could poll the *Cycle Mode Interrupt Enable (CME)* bit of the Interrupt Register (IR) to determine when the Maintenance Slot has actually begun.

The number or type of accesses to the External Memory during a Maintenance Slot is clearly variable depending on the processor clock speed as well as the bus architecture. The effective duration of a Maintenance Slot is (63 * ACLK period – 4 * MCLK period) due to internal synchronizations. The MC92501 reclaims the bus at the end of the Maintenance Slot. If a maintenance access extends beyond the end of the effective Maintenance Slot, it is abruptly terminated, and the *Maintenance Access Error Interrupt Enable (MNAEE)* bit of the Interrupt Register (IR) is set to report the error. In this case the interrupt is equivalent to a bus error since invalid data may have been written to the External Memory. MNAEE is also set if a maintenance access is performed outside of a Maintenance Slot, with the exception described below.

If the $\overline{\text{MDTACK}}n$ signal is used for maintenance accesses (i.e., the *DTACK Drive during External Memory Accesses (DDEM)* field of the Microprocessor Configuration Register (MPCONR) is non-zero), a maintenance access may start before the beginning of the Maintenance Slot once either EMMREQ has been asserted or the MSEE interrupt bit has been set. In this case the read/write of the External Memory is delayed until the beginning of the Maintenance Slot, and DTACK is asserted once the read/write has been performed.

NOTE: When updating the External Memory, care should be taken that at the end of a Maintenance Slot there are no inconsistencies in the External Memory.

A typical Maintenance Slot used by a DMA device is shown in **Figure 3-3**. The first part of the slot is used for bus arbitration. Once the DMA device becomes the bus master, one channel performs four read accesses. Then another channel performs four write accesses.

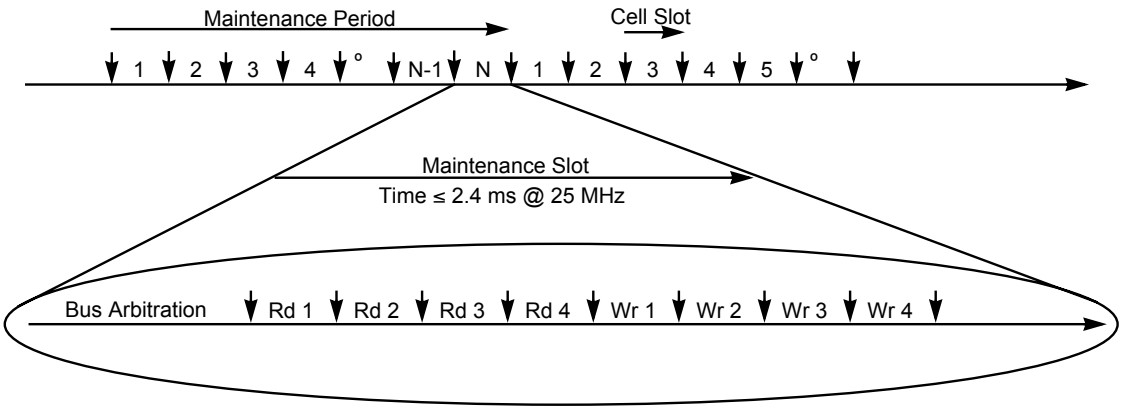


Figure 3-3. Example of Maintenance Slot Usage

3.3.4 Indirect Memory Access

The MC92501 enables the processor to access External Memory during Operate Mode. The indirect access takes place at least once in every cell processing slot. The Indirect access is not performed during maintenance. Indirect External Memory access is performed using 2 registers: the Indirect External Memory Access Address Register (IAAR**)** and the Indirect External Memory Access Data Register (IADR**)**.

3.3.4.1 Write Access

In order to write to the External Memory, the processor should poll the *Indirect External Memory Access Busy (IAB)* bit in the IAAR register to verify if it may write the IAAR register and the IADR register. It then writes the data into the IADR register, and the address, size and direction = 0 into the *Indirect External Memory Access Address (IAA)* field and the *Indirect External Memory Access Size (IAW)* bit *and the Indirect External Memory Access DIR (IAD)* bit of the IAAR register. Writing to the IAAR register triggers the MC92501 to wait for a dedicated clock, and write the data to External Memory using the given address and data. Once the MC92501 finishes writing, it clears the *Indirect External Memory Access Busy (IAB)* bit in the IAAR register.

3.3.4.2 Read Access

In order to read from the External Memory, the processor should poll the *Indirect External Memory Access Busy (IAB)* bit in the IAAR register to verify if it may write the IAAR register. It then writes the address, size and direction = 1 into the *Indirect External Memory Access Address (IAA)* field, the *Indirect External Memory Access Size (IAW)* bit and the *Indirect External Memory Access DIR (IAD)* bit of the IAAR register. Writing to the IAAR register triggers the MC92501 to wait for a dedicated clock, and read the data from

External Memory using the given address and write the data into the IADR register. Once the data was written into the IADR register, the MC92501 clears the *Indirect External Memory Access Busy (IAB)* bit in the IAAR register. The Processor then may read the data from the IADR register. The address space covered by this interface includes all the non-destructive External Memory access and external address compression device.

NOTE: An indirect write access to an External Memory space that can be written by the MC92501 is not recommended. For example, do not perform an indirect write access to a flag table record of an active connection; use the maintenance cell slot for this purpose.

Table 3-1 summarizes the indirect access fields:

Table 3-1. Indirect Access fields

Indirect External Memory Access DIR (IAD)	Indirect External Memory Access Size (IAW)	Least Significant Bit of Indirect External Memory Access Address (IAA)	Data Order (DO)	Function
0	0	x	x	Write IADR[31:00] to External Memory word bits [31:00]
0	1	0	0	Write IADR[31:16] to External Memory word [31:16]
0	1	0	1	Write IADR[15:00] to External Memory word [15:00]
0	1	1	0	Write IADR[15:00] to External Memory word [15:00]
0	1	1	1	Write IADR[31:16] to External Memory word [31:16]
1	x	x	x	Read External Memory word bits [31:00] to IADR[31:00]

3.4 Cell Insertion And Extraction

The transfer of ATM cells between the MC92501 and the microprocessor uses two arrays of registers, the Cell Insertion Registers for transfers to the MC92501 and the Cell Extraction Registers for transfers from the MC92501. The structure of the inserted and extracted cells is provided in the tables in the following sections. Detailed definitions of the cell structures are provided in **Section 7.4.1** Inserted Cell and **Section 7.4.2** Extracted Cell. The payload of the cell, contained in registers 4–15 of each array, is considered a collection of bytes. Therefore, the order of the bytes within the 32-bit registers are arranged differently if low-endian ordering is being used on the microprocessor bus. The overhead information, on the other hand, consists of 32-bit fields and is not affected by the byte order of the bus. The byte-swapping of the payload is enabled by setting the *Data Order (DO)* bit in the Microprocessor Configuration Register (MPCONR).

3.4.1 Cell Insertion

The microprocessor inserts cells into the Ingress or Egress cell flows indirectly by using the Cell Insertion Registers. This set of sixteen registers holds one cell together with the overhead information necessary to insert it properly in one of the cell flows. The registers may be written in any order. If similar cells are to be inserted one after the other, registers whose value does not change need not be written.

The Cell Insertion Registers are mapped into two distinct address spaces (see **Section 7.2.1** Cell Insertion Registers (CIR0–CIR15)). The address spaces differ in the number of registers that can be accessed and in the identity of the trigger register. Writing the trigger register causes the cell to be inserted, so this must always be the last register to be written. Once the trigger register has been written, the CIRs may not be written again with the next cell until the *Cell Insertion Queue Empty (CIQE)* bit in the Interrupt Register (IR) has been set by the MC92501. In the cell insertion address space the trigger register is CIR15. In the alternative cell insertion address space the trigger register is ACIR1. The alternative address space is useful for inserting cells whose header and payload are generated by the MC92501 and do not have to be inserted by the user. For example, an AIS cell can be inserted by writing to only ACIR0 and ACIR1.

Additionally, the MC92501 provides support for transferring cells using a DMA device without need for a processor interrupt. Each of the DMA request output signals can be programmed to function as MCIREQ which is asserted in Operate Mode whenever the Cell Insertion Register array is available to be written. Writing to CIR14 in the cell insertion address space or to ACIR0 in the alternative cell insertion address space causes MCIREQ to be deasserted until the array is available again. Writing to CIR15 or ACIR1 also deasserts MCIREQ. See **Section 4.5.2.2** Cell Insertion with DMA Support for details. Both of the Cell Insertion Register address spaces contain a number of “don’t care” bits in their addresses (see **Figure 7-2**). This effectively maps the registers repeatedly in this address space and allows the transfer of a large buffer of data using a single buffer descriptor in the DMA device.

CIR0	Cell Descriptor				ACIR0
CIR1	Connection Descriptor				ACIR1
CIR2	Unused				
CIR3	ATM cell header (VPI,VCI,PTI,CLP)				
CIR4	ATM octet 6	ATM octet 7	ATM octet 8	ATM octet 9	
CIR5	ATM octet 10	ATM octet 11	ATM octet 12	ATM octet 13	
CIR6	ATM octet 14	ATM octet 15	ATM octet 16	ATM octet 17	
CIR7	ATM octet 18	ATM octet 19	ATM octet 20	ATM octet 21	
CIR8	ATM octet 22	ATM octet 23	ATM octet 24	ATM octet 25	
CIR9	ATM octet 26	ATM octet 27	ATM octet 28	ATM octet 29	
CIR10	ATM octet 30	ATM octet 31	ATM octet 32	ATM octet 33	
CIR11	ATM octet 34	ATM octet 35	ATM octet 36	ATM octet 37	
CIR12	ATM octet 38	ATM octet 39	ATM octet 40	ATM octet 41	
CIR13	ATM octet 42	ATM octet 43	ATM octet 44	ATM octet 45	
CIR14	ATM octet 46	ATM octet 47	ATM octet 48	ATM octet 49	
CIR15	ATM octet 50	ATM octet 51	ATM octet 52	ATM octet 53	

Figure 3-4. Inserted Cell Structure (Motorola-style byte order)

CIR0	Cell Descriptor				ACIR0
CIR1	Connection Descriptor				ACIR1
CIR2	Unused				
CIR3	ATM cell header (VPI,VCI,PTI,CLP)				
CIR4	ATM octet 9	ATM octet 8	ATM octet 7	ATM octet 6	
CIR5	ATM octet 13	ATM octet 12	ATM octet 11	ATM octet 10	
CIR6	ATM octet 17	ATM octet 16	ATM octet 15	ATM octet 14	
CIR7	ATM octet 21	ATM octet 20	ATM octet 19	ATM octet 18	
CIR8	ATM octet 25	ATM octet 24	ATM octet 23	ATM octet 22	
CIR9	ATM octet 29	ATM octet 28	ATM octet 27	ATM octet 26	
CIR10	ATM octet 33	ATM octet 32	ATM octet 31	ATM octet 30	
CIR11	ATM octet 37	ATM octet 36	ATM octet 35	ATM octet 34	
CIR12	ATM octet 41	ATM octet 40	ATM octet 39	ATM octet 38	
CIR13	ATM octet 45	ATM octet 44	ATM octet 43	ATM octet 42	
CIR14	ATM octet 49	ATM octet 48	ATM octet 47	ATM octet 46	
CIR15	ATM octet 53	ATM octet 52	ATM octet 51	ATM octet 50	

Figure 3-5. Inserted Cell Structure (Intel-style byte order)

3.4.2 Cell Extraction

Cell extraction uses a combination of a Cell Extraction Queue, which is a 16-cell FIFO, and a set of Cell Extraction Registers. The following paragraphs include a detailed description of these components used in cell extraction.

3.4.2.1 Cell Extraction Queue

Cells that are copied from the Ingress or Egress cell flows are added to the Cell Extraction Queue to await their transfer to the microprocessor. The queue is necessary because both the Ingress and Egress Processing Units may copy cells during the same cell processing slot for a number of consecutive cell slots, while the microprocessor cannot read the cells at that rate.

The Cell Extraction Queue consists of a 16-cell FIFO as shown in **Figure 3-6**. There are two programmable thresholds defined in the Microprocessor Control Register (MPCTLR). The *Cell Extraction Queue Interrupt Threshold (CEQI)* bit of the Interrupt Register (IR) is asserted when the Cell Extraction Queue is filled to the Interrupt Threshold. The purpose of the Interrupt Threshold is to allow the microprocessor to wait until a number of cells have accumulated and then to read the cells in a burst. The Low Priority Threshold defines a “danger” situation where the Cell Extraction Queue is almost full. The *Cell Extraction Queue Low Priority Threshold (CEQL)* bit of the Interrupt Register (IR) is asserted at this point. When the Cell Extraction Queue is full beyond the Low Priority Threshold, the MC92501 places only high-priority cells on the queue. Low-priority cells that are copied from the Ingress or Egress cell flows are discarded. If the Cell Extraction Queue is completely full, the *Cell Extraction Queue Full (CEQF)* bit of the Interrupt Register (IR) is asserted, and no cells are added to the queue.

The microprocessor interface filters the cells to be placed in the Cell Extraction Queue, as illustrated in **Figure 3-6**. When a cell is to be copied, the Cell Indication is checked. If the *Extraction Reason (RSN)* field indicates Copy_All or Copy_OAM, the cell is filtered according to the *Indication Cell Name (ICN)* field using the Cell Extraction Queue Filtering Register 1 (CEQFR1) and, if the queue is above the low-priority threshold, the Cell Extraction Queue Priority Register 1 (CEQPR1). Otherwise, the cell is filtered using Cell Extraction Queue Priority Register 0 (CEQPR0) and, if above the low-priority threshold, Cell Extraction Queue Priority Register 0 (CEQPR0). Additionally, if the *GFC Reason (GFCR)* bit is set, the corresponding bit of CEQFR0 and CEQPR0 is used.

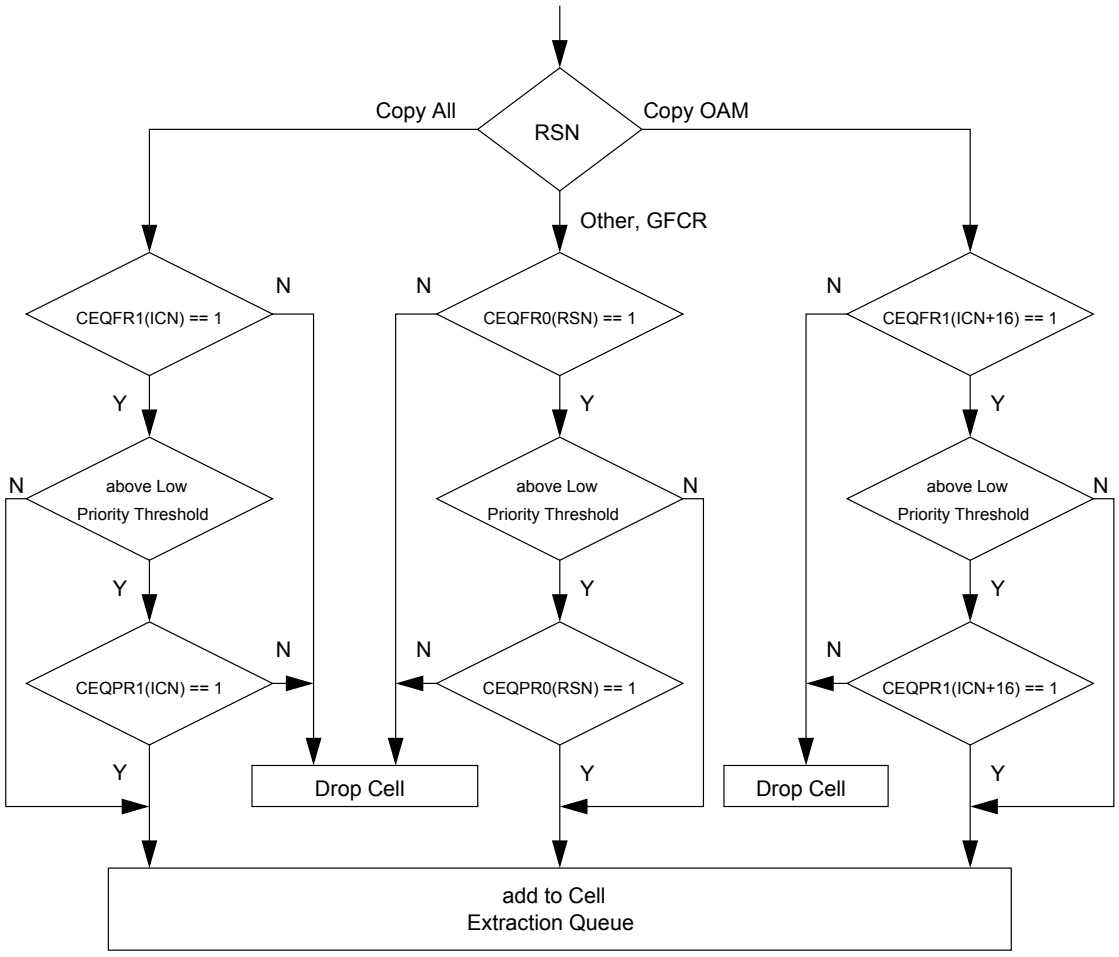


Figure 3-6. Cell Extraction Queue Filtering

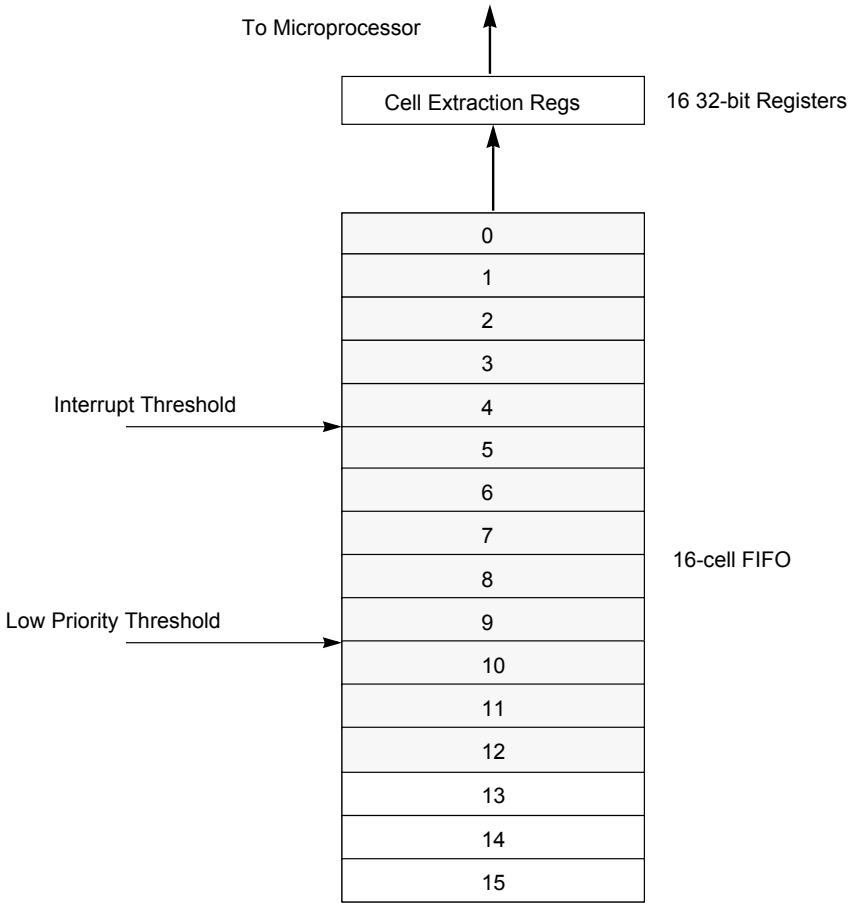


Figure 3-7. Cell Extraction Queue

3.4.2.2 Cell Extraction Registers

Cells that are copied from the Ingress or Egress cell flows are transferred to the microprocessor by using the Cell Extraction Registers. This set of sixteen registers holds one cell together with the overhead information that describes its source and cell type. The registers may be read in any order. Registers whose contents are not of interest in the specific cell need not be read at all.

The Cell Extraction Registers are mapped into a single address space. See **Section 7.2.2** Cell Extraction Registers (CER0–CER15). Reading the trigger register, CER15, frees the array for the MC92501 to write the next cell, so this must always be the last register to be read. Once the trigger register has been read, the CERs should not be read again for the next cell until the *Cell Extraction Queue Ready Interrupt Enable (CEQRE)* bit in the Interrupt Register has been set by the MC92501. Additionally, the MC92501 provides support for transferring cells using a DMA device without need for a processor interrupt. Each of the DMA request lines can be programmed to function as MCOREQ

which is asserted whenever the Cell Extraction Register array contains a cell to be read. Reading from CER14 causes MCOREQ to be deasserted until the next cell is ready. Reading from CER15 also deasserts MCOREQ. See **Section 4.5.2.1** Cell Extraction with DMA Support for details. The MC92501 is designed to support "back to back" cell extraction; if the next cell is ready when CER14 of the current cell is read, MCOREQ is not deasserted, thus allowing continuous DMA operation. The Cell Extraction Register address space contains a number of "don't care" bits in its addresses (see **Figure 7-3**). This effectively maps the registers repeatedly in this address space and allows the transfer of a large buffer of data using a single buffer descriptor in the DMA device.

CER0	Cell Indication			
CER1	Connection Indication			
CER2	Timestamp			
CER3	ATM cell header (VPI,VCI,PTI,CLP)			
CER4	ATM octet 6	ATM octet 7	ATM octet 8	ATM octet 9
CER5	ATM octet 10	ATM octet 11	ATM octet 12	ATM octet 13
CER6	ATM octet 14	ATM octet 15	ATM octet 16	ATM octet 17
CER7	ATM octet 18	ATM octet 19	ATM octet 20	ATM octet 21
CER8	ATM octet 22	ATM octet 23	ATM octet 24	ATM octet 25
CER9	ATM octet 26	ATM octet 27	ATM octet 28	ATM octet 29
CER10	ATM octet 30	ATM octet 31	ATM octet 32	ATM octet 33
CER11	ATM octet 34	ATM octet 35	ATM octet 36	ATM octet 37
CER12	ATM octet 38	ATM octet 39	ATM octet 40	ATM octet 41
CER13	ATM octet 42	ATM octet 43	ATM octet 44	ATM octet 45
CER14	ATM octet 46	ATM octet 47	ATM octet 48	ATM octet 49
CER15	ATM octet 50	ATM octet 51	ATM octet 52	ATM octet 53

Figure 3-8. Extracted Cell Structure (Motorola-style byte order)

CER0	Cell Indication			
CER1	Connection Indication			
CER2	Timestamp			
CER3	ATM cell header (VPI,VCI,PTI,CLP)			
CER4	ATM octet 9	ATM octet 8	ATM octet 7	ATM octet 6
CER5	ATM octet 13	ATM octet 12	ATM octet 11	ATM octet 10
CER6	ATM octet 17	ATM octet 16	ATM octet 15	ATM octet 14
CER7	ATM octet 21	ATM octet 20	ATM octet 19	ATM octet 18
CER8	ATM octet 25	ATM octet 24	ATM octet 23	ATM octet 22
CER9	ATM octet 29	ATM octet 28	ATM octet 27	ATM octet 26
CER10	ATM octet 33	ATM octet 32	ATM octet 31	ATM octet 30
CER11	ATM octet 37	ATM octet 36	ATM octet 35	ATM octet 34
CER12	ATM octet 41	ATM octet 40	ATM octet 39	ATM octet 38
CER13	ATM octet 45	ATM octet 44	ATM octet 43	ATM octet 42
CER14	ATM octet 49	ATM octet 48	ATM octet 47	ATM octet 46
CER15	ATM octet 53	ATM octet 52	ATM octet 51	ATM octet 50

Figure 3-9. Extracted Cell Structure (Intel-style byte order)

4

MC92501 External Interfaces

4.1 Introduction

The MC92501 provides the following four external interfaces:

- PHY Interface
- Switch Interface
- External Memory interface
- Microprocessor interface

The following section provide detailed descriptions of each of these interfaces.

4.2 PHY Interface

The PHY Receive and Transmit Interfaces can be programmed to be either UTOPIA Level 1 standard compliant [12] or UTOPIA level 2 compliant [13] by programming the *Ingress UTOPIA Mode (IUM)* bit and the *Egress UTOPIA Mode (EUM)* bit. All of the interface signals are synchronous to the MC92501 cell processing clock (ACLK). Output signals from the MC92501 are updated following the rising edge of ACLK, and input signals to the MC92501 are sampled at the rising edge of ACLK.

4.2.1 UTOPIA Level 1 Receive PHY Interface (Ingress)

The MC92501 Receive PHY Interface supports the UTOPIA Level 1 receive interface with both octet-level and cell-level handshakes. The *Ingress PHY Operation Mode (IPOM)* bit of the Ingress PHY Configuration Register (IPHCR) determines whether the interface operates in the octet-based mode or the cell-based mode. In addition to the UTOPIA signals, the RXPHYID bus is provided for multiple PHY support. If a single PHY is used, the RXPHYID[3:0] signals should all be tied to GND. The signals involved in the receive interface are shown in **Figure 4-1**.

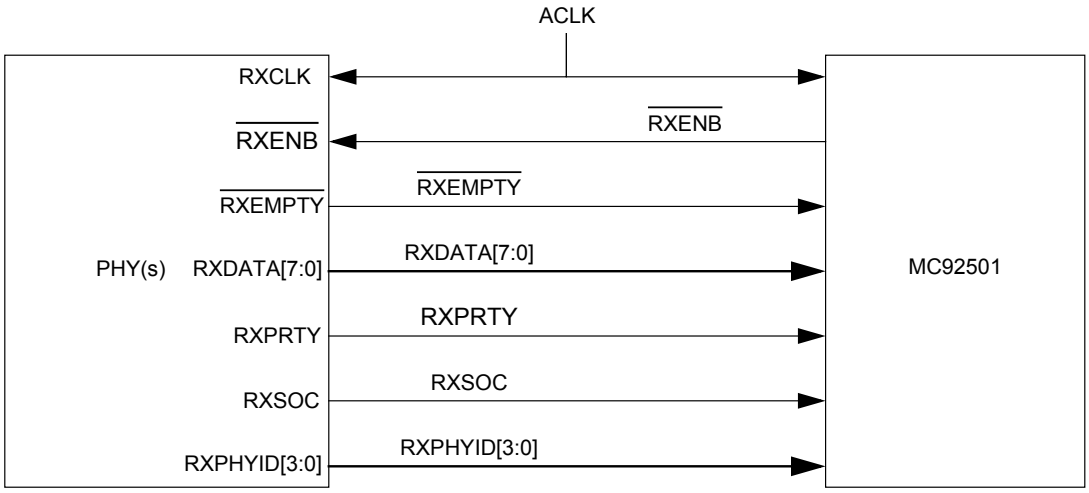


Figure 4-1. MC92501 Receive PHY Interface

The MC92501 requests data transfers in the receive direction by asserting RXENB. Since the MC92501 processes entire cells, it requests data on a cell basis and deasserts RXENB only at the end of a cell transfer if it cannot receive another cell. When the PHY detects RXENB deasserted, the values that it drives on RXEMPTY, RXDATA, RXPRTY, RXSOC, and RXPHYID during the following clock cycle are "don't care". **Figure 4-2** shows the deassertion of RXENB at the end of a cell when the MC92501 is unable to receive another cell. RXENB is deasserted after the second-last octet of the cell has been received and is not asserted again until the MC92501 is able to receive another entire cell.

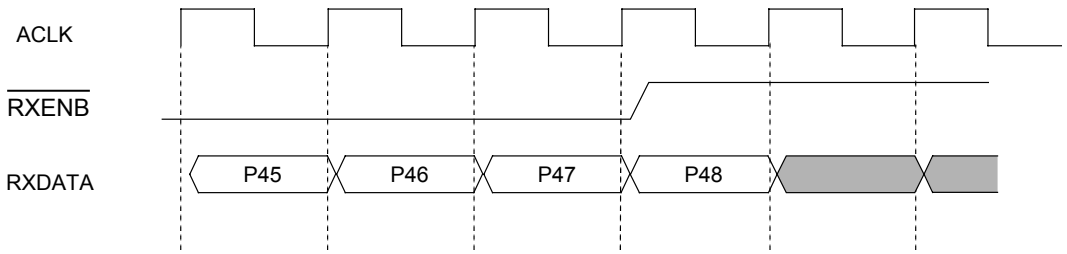


Figure 4-2. MC92501 Receive Timing—End of Cell

The PHY asserts RXSOC together with the first octet of each cell in order to synchronize the MC92501 to the beginning of a new cell. When RXSOC is valid and asserted, the value on RXPHYID is stored as the link number from which the cell is arriving. RXPRTY contains the odd parity over RXDATA. It is not shown in the timing diagrams since its timing is identical to that of RXDATA. For a discussion of parity checking at the Ingress Receive Interface, see Section 5.2.1 Interface to Physical Layer—Cell Assembly.

4.2.1.1 Octet-Based Receive Interface

When using the octet-based receive interface, the RXEMPTY signal is used to indicate that valid data is being presented during the current clock cycle. RXEMPTY must be valid whenever RXENB was asserted on the previous clock cycle. The MC92501 samples RXEMPTY, RXDATA, RXPRTY, RXSOC, and RXPHYID at the rising edge of ACLK if RXENB was already asserted at the previous rising edge. If RXEMPTY is deasserted, the values sampled from RXDATA, RXPRTY, and RXSOC are valid. Otherwise, these values are discarded. **Figure 4-3** shows the start of a cell when using the octet-based interface.

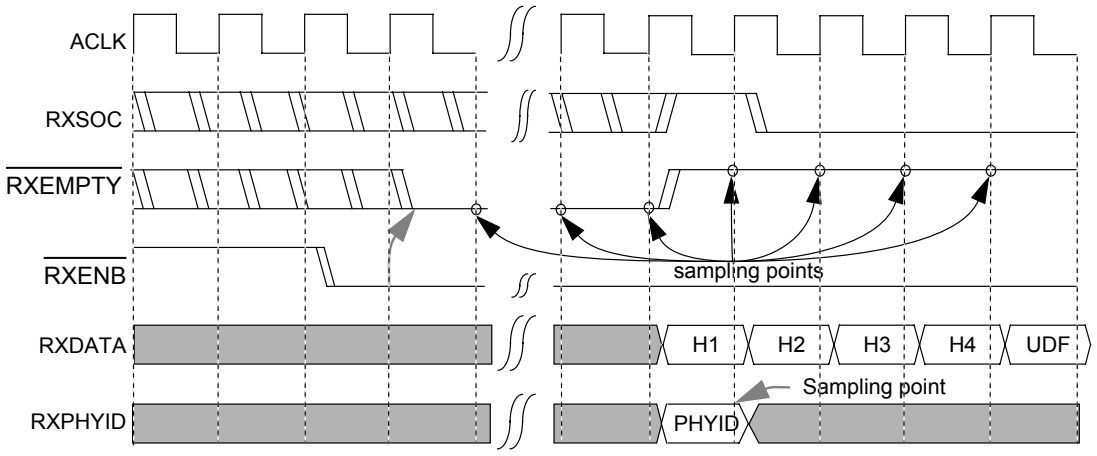


Figure 4-3. Start of Cell—MC92501 Empty (Octet-Based Interface)

Figure 4-4 shows the end of a cell when the MC92501 is able to receive another cell, but the PHY has no more data to transfer. RXEMPTY is asserted after the last byte of valid data has been transferred.

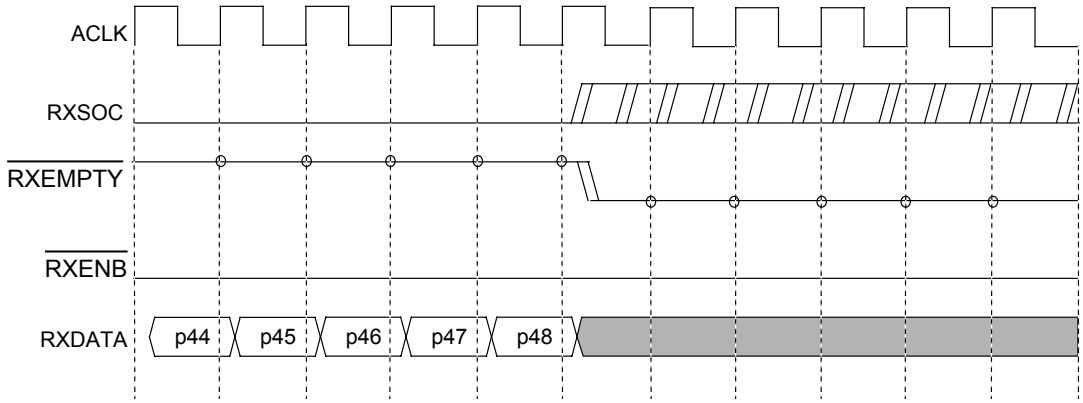


Figure 4-4. End of Cell—PHY Empty (Octet-Based Interface)

Figure 4-5 shows the reception of back-to-back cells. RXSOC is asserted together with the first octet of the new cell. The new value of RXPHYID is sampled at this point.

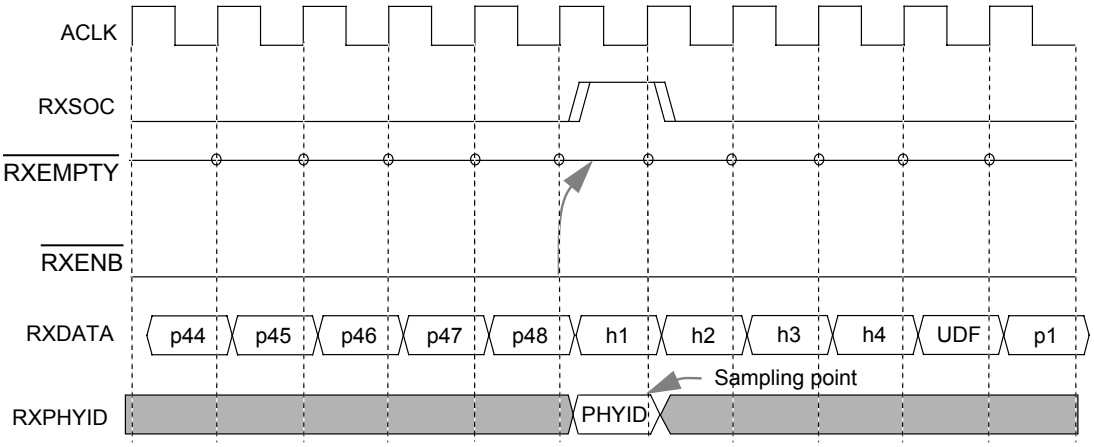


Figure 4-5. Back-to-Back Cell Input (Octet-Based Interface)

Figure 4-6 shows an example where the PHY runs out of valid data in the middle of the cell and uses RXEMPTY to regulate the transfer of data. When the MC92501 detects RXEMPTY asserted, the value on RXDATA is discarded, and is not included in the received cell.

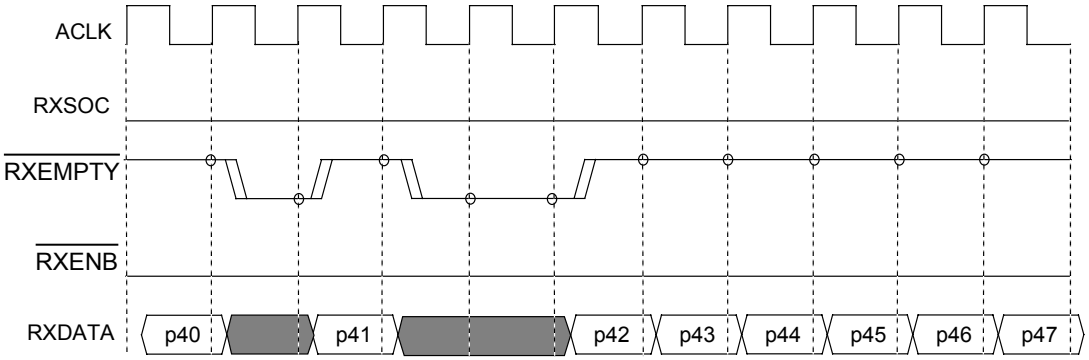


Figure 4-6. Response to $\overline{\text{RXEMPTY}}$ Assertion (Octet-Based Interface)

4.2.1.2 Cell-Based Receive Interface

If a cell-based UTOPIA receive interface is used, RXEMPTY is connected to a Receive Cell Available (RxClav[1]) pin on the PHY device. In this case, the deassertion of RXEMPTY means that the PHY has an entire cell available to be transferred in consecutive clock cycles. Therefore, once RXEMPTY is sampled deasserted by the MC92501, it is ignored until the entire cell has been transferred, and the PHY may not interrupt the transfer of data in the middle of a cell. The start of a cell is shown in **Figure 4-7**. RXEMPTY is required to be valid from the clock cycle following the assertion of RXENB until it has been sampled deasserted.

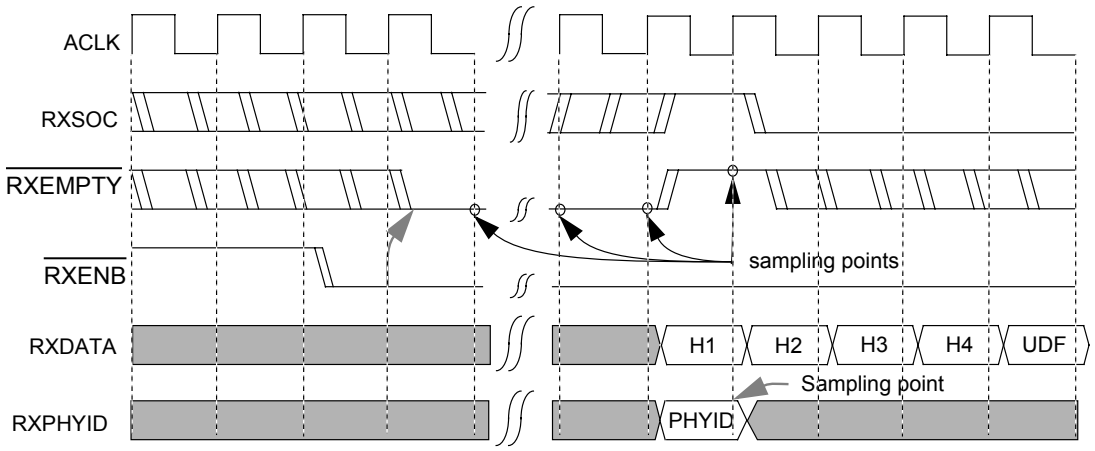


Figure 4-7. Start of Cell—MC92501 Empty (Cell-Based Interface)

Figure 4-8 shows the end of a cell when the MC92501 is able to receive another cell, but the PHY does not have an entire cell available. RXEMPTY is sampled starting after the last octet of the cell has been transferred.

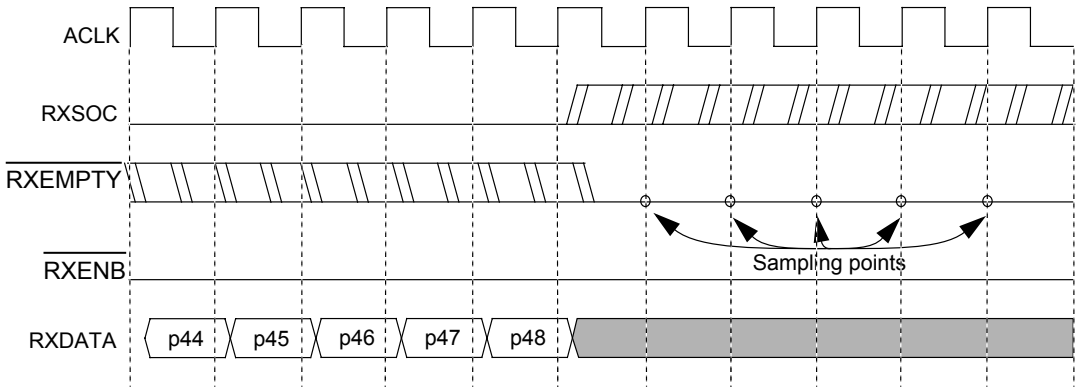


Figure 4-8. End of Cell—PHY Empty (Cell-Based Interface)

1.UTOPIA name.

Figure 4-9 shows the reception of back-to-back cells. RXEMPTY is sampled starting after the last octet of the cell has been transferred. Since it is deasserted, the next cell transfer begins immediately. RXSOC is asserted together with the first octet of the new cell. The new value of RXPHYID is sampled at this point.

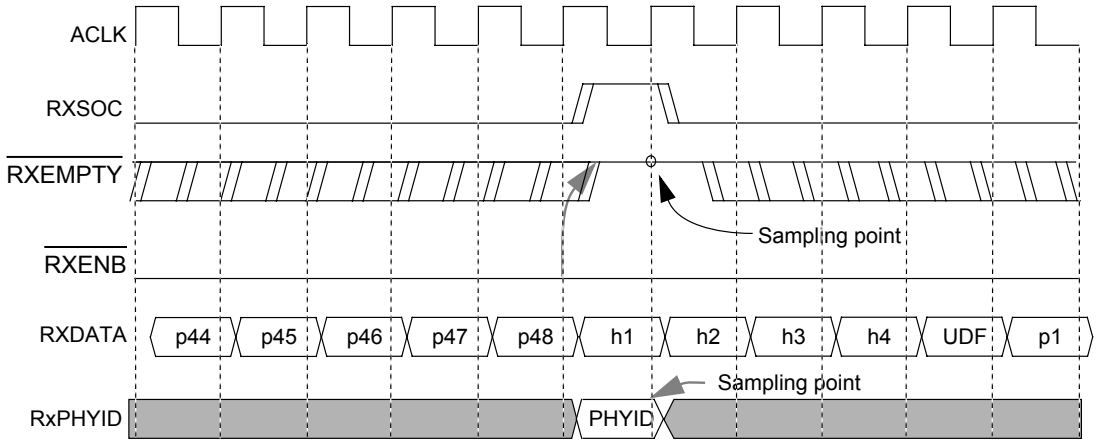


Figure 4-9. Back to Back Cell Input (Cell-Based Interface)

4.2.2 UTOPIA Level 1 Transmit PHY Interface (Egress)

The MC92501 Transmit PHY Interface supports the UTOPIA Level 1 transmit interface with both octet-level and cell-level handshakes. The *Egress PHY Operation Mode (EPOM)* bit of the Egress PHY Configuration Register (EPHCR) determines whether the interface operates in the octet-based mode or the cell-based mode. In addition to the UTOPIA signals, the TXPHYID bus, as well as the TXPHYIDV signal, are provided for multiple PHY support. If a single PHY is used, all of these signals may remain unconnected. The function of TXCCLR is discussed below. The signals involved in the transmit interface are shown in **Figure 4-10**.

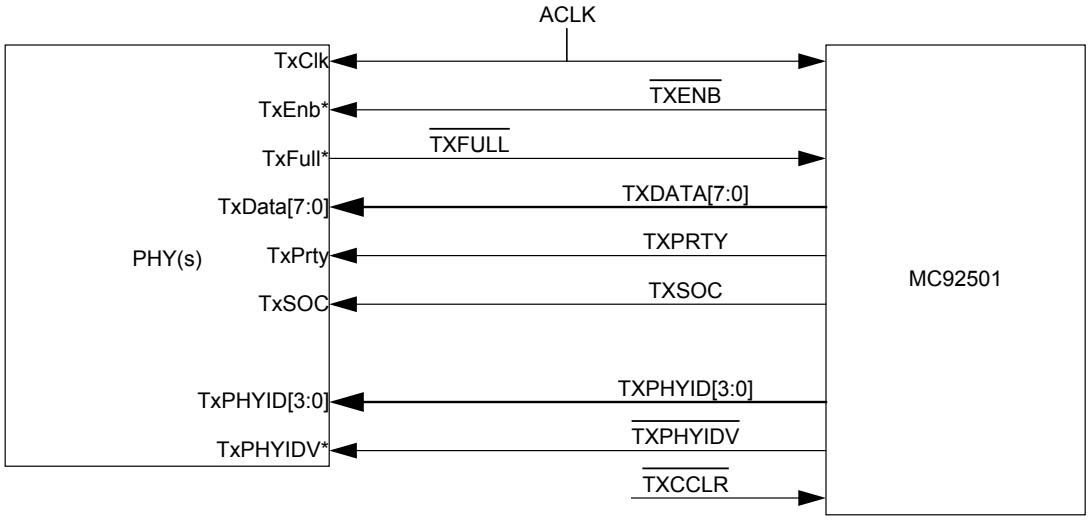


Figure 4-10. MC92501 Transmit PHY Interface

The $\overline{\text{TXENB}}$ output signal is used by the MC92501 to indicate that it is driving valid data on TXSOC, TXDATA, and TXPRTY in the current clock cycle. When TXENB is asserted, the PHY should sample the data values. When TXENB is deasserted, the data values are invalid. **Figure 4-11** shows the MC92501 using TXENB at the end of a cell to indicate that it has no more valid data to transmit. One clock later, it is ready to transmit the next cell and asserts TXENB.

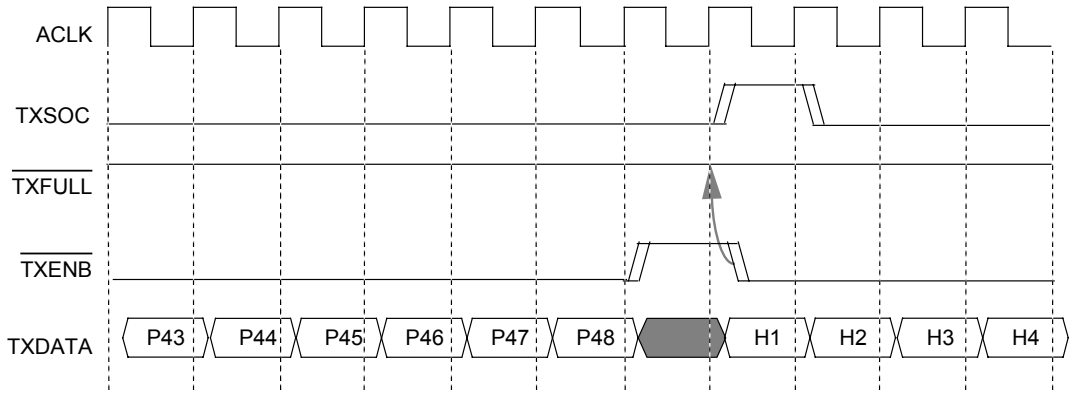


Figure 4-11. MC92501 Transmit Timing—MC92501 is Empty

The MC92501 asserts TXSOC during the first byte of a cell to synchronize the PHY. TXPRTY contains the odd parity over TXDATA. It is not shown in the timing diagrams since its timing is identical to that of TXDATA. Depending on the value of the *PHY ID Control (PHIDC)* bit of the MC92501 General Configuration Register (GCR), the TXPHYID pins are connected to one of two internal buses, TXCPHYID (current cell) or TXNPHYID (next cell). The TXCPHYID bus contains the number of the link to which the current cell belongs. It is valid for the entire duration of the cell transfer. This bus can be used to produce device enable signals so that only the addressed PHY device actually samples the transmitted cell. The TXNPHYID bus contains the number of the link to which the next cell belongs. It is provided in advance so that the TXFULL signal can be driven by the correct PHY device (the one that receives the next cell) at the end of the current cell transfer. The TXPHYIDV signal is asserted when TXNPHYID is valid. TXNPHYID may be provided as early as the second octet of the cell if the MC92501 has the next cell ready for transmission. On the other hand, it may be provided as late as one clock cycle before the MC92501 intends to transmit the new cell. In either case TXNPHYID remains valid until TXENB is asserted to transmit the first octet of the new cell. Thus, the PHY can extend the valid time of TXNPHYID (and delay the next cell) by asserting TXFULL. The fifth octet of each cell (the User-Defined Field into which the PHY must insert the HEC) is transmitted as zero by the MC92501. The timing of the PHYID signals is provided in **Figure 4-12**.

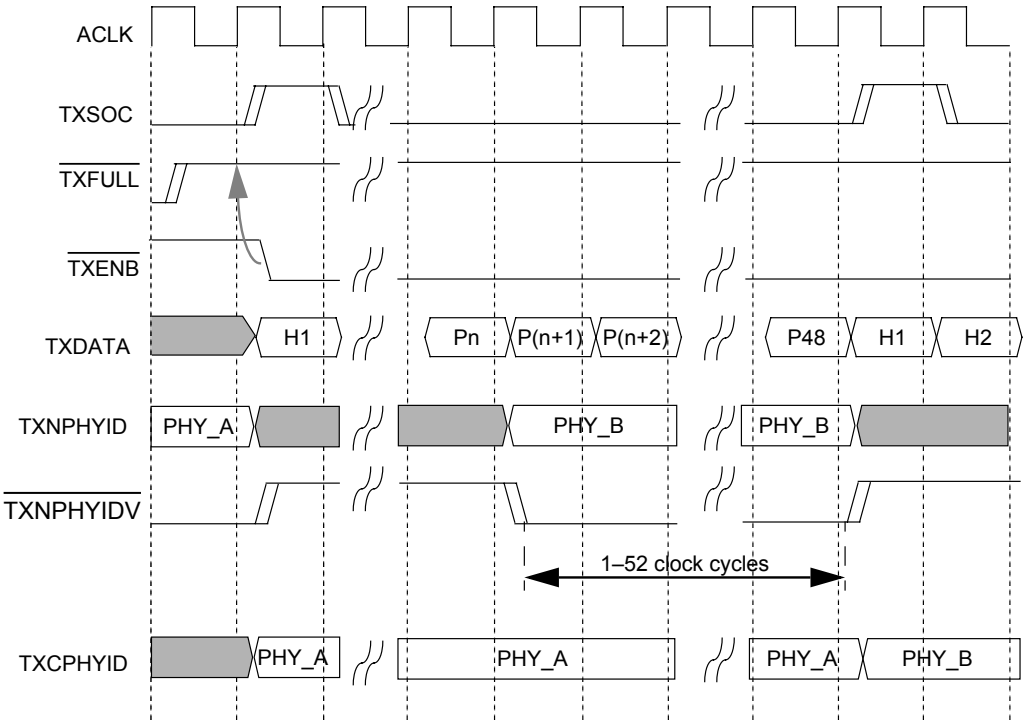


Figure 4-12. Timing of PHYID Signals

4.2.2.1 Octet-Based Transmit Interface

When using the octet-based transmit interface, the PHY uses the TXFULL signal to regulate the data flow on an octet basis. When TXFULL is deasserted and the MC92501 has a cell to transmit, TXENB is asserted, and the data is transferred. If the PHY cannot receive more data, it asserts TXFULL. The MC92501 checks TXFULL at every rising edge of ACLK. If TXFULL is asserted, TXENB is deasserted until the deassertion of TXFULL. **Figure 4-13** shows the situation where the MC92501 has a cell waiting to be transmitted. As soon as TXFULL is deasserted, the MC92501 asserts TXENB and begins to transmit the cell.

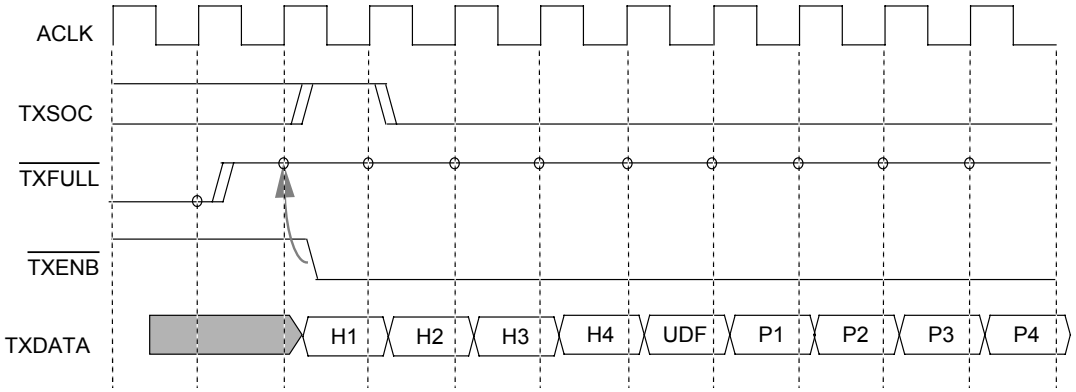


Figure 4-13. Start of Cell (Octet-Based Interface)

Figure 4-14 shows the transmission of back-to-back cells. TXSOC is asserted together with the first octet of the new cell.

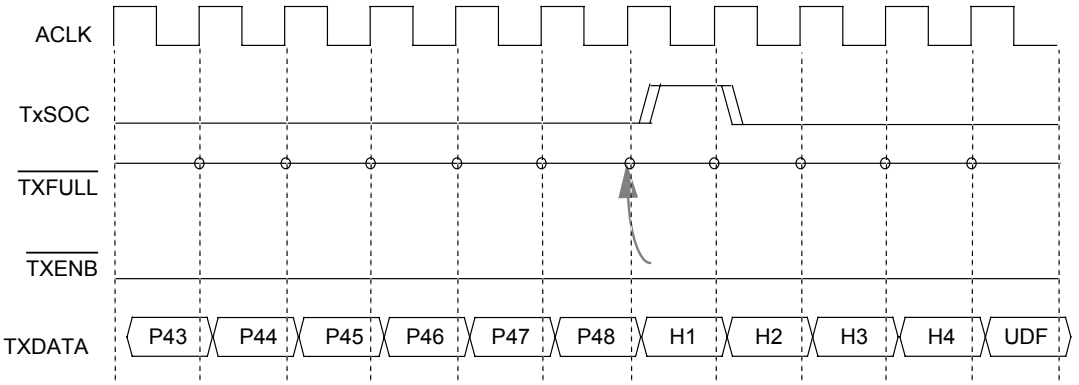


Figure 4-14. Back-to-Back Cell Output (Octet-Based Interface)

Figure 4-15 shows an example where, in the middle of a cell, the PHY is unable to accept more data and uses TXFULL to regulate the transfer of data. In response, the MC92501 deasserts TXENB and does not update the data.

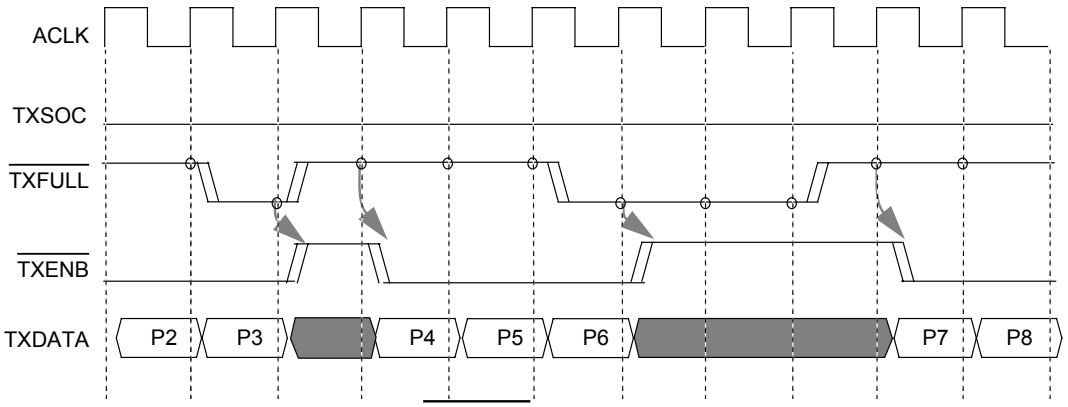


Figure 4-15. Response to TXFULL Assertion (Octet-Based Interface)

4.2.2.2 Cell-Based Transmit Interface

If a cell-based UTOPIA interface is used, TXFULL is connected to a Transmit Cell Available (TxClav[1]) pin on the PHY device. In this case the deassertion of TXFULL means that the PHY layer can accept an entire cell in consecutive clock cycles. Therefore, once the MC92501 begins to transmit a cell, TXFULL is ignored until the entire cell has been transferred, and the PHY cannot interrupt the transfer of data in the middle of a cell. The start of a cell is shown in **Figure 4-16**. TXFULL is required to be valid until TXENB is asserted.

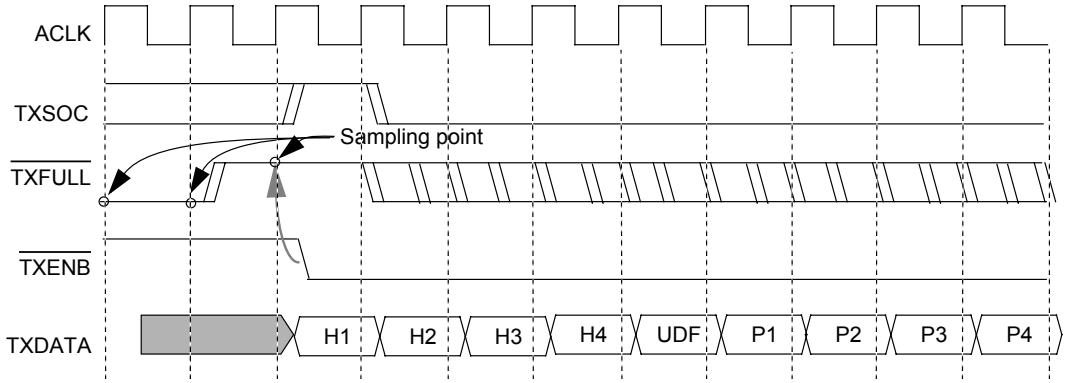


Figure 4-16. Start of Cell (Cell-Based Interface)

Figure 4-17 shows the transmission of back-to-back cells. TXFULL is sampled starting with the last octet of the cell. Since it is deasserted, the next cell transfer begins immediately. TXSOC is asserted together with the first octet of the new cell.

1.UTOPIA name

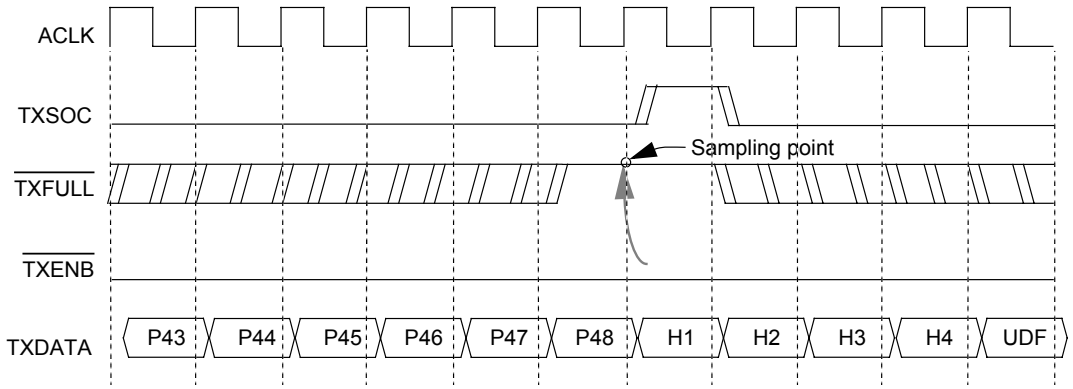


Figure 4-17. Back-to-Back Cell Output (Cell-Based Interface)

4.2.2.3 Clearing a Transmitted Cell

The TXCCLR input signal is used to clear (discard) the cell that is currently being transmitted. This feature is provided for implementations with multiple PHY devices. If the addressed PHY malfunctions, TXCCLR may be asserted to discard the cell so that it does not block cells that are addressed to other PHY devices. **Figure 4-18** shows the timing of the TXCCLR signal. When the MC92501 samples TXCCLR asserted at a rising edge of ACLK, TXENB is deasserted at the next rising edge of ACLK, and the current cell is discarded. TXENB remains deasserted for three clock cycles. On the fourth rising edge of ACLK after the assertion of TXCCLR was detected, transmission may continue with the next cell. Assuming that another cell is waiting for transmission and that TXFULL has not been asserted, TXENB is asserted, TXSOC is asserted, and the first octet of the next cell is provided on TXDATA.

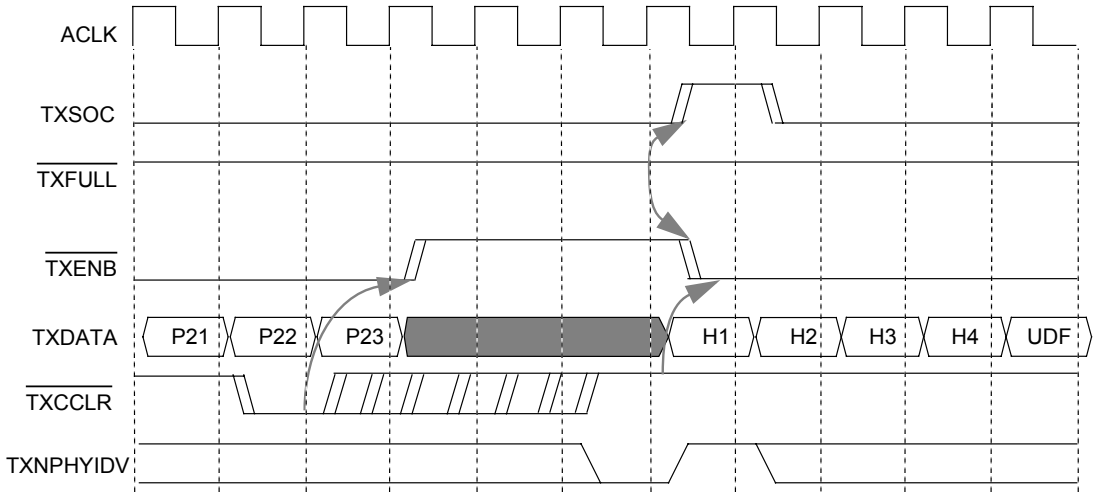


Figure 4-18. Response to TXCCLR Assertion

The value of TXCCLR is ignored for the three clock cycles after its assertion is detected. However, if it remains asserted for more than four clock cycles, the next cell may be discarded also, as shown in **Figure 4-19**.

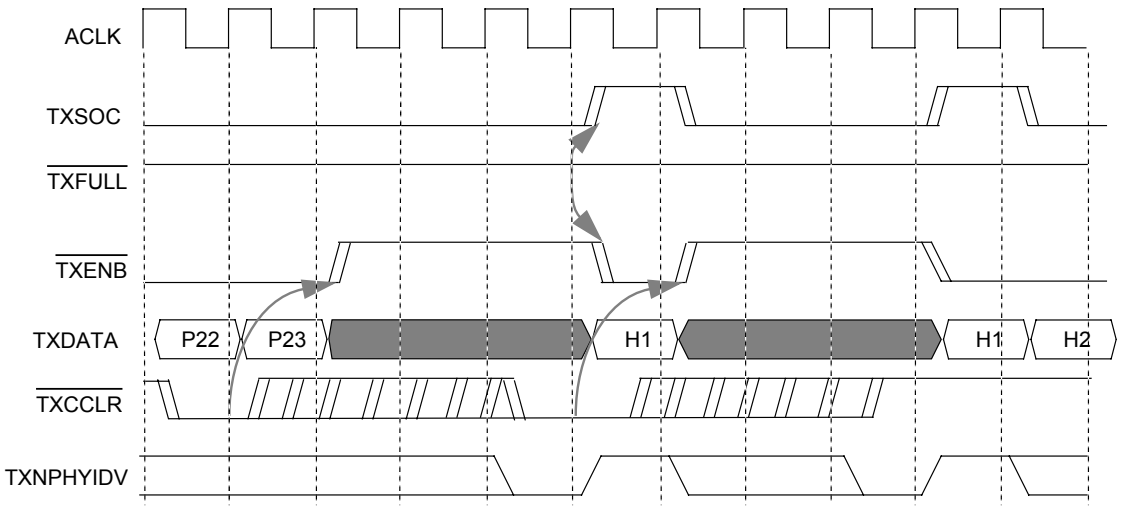


Figure 4-19. Consecutive Cell Discards

4.2.3 UTOPIA Level 2 Receive PHY Interface (Ingress)

The MC92501 Receive PHY Interface supports the UTOPIA Level 2 receive interface with 1 RxClav signal. In this mode the MC92501 operates in cell-based mode only and the The *Ingress PHY Operation Mode (IPOM)* bit of the Ingress PHY Configuration Register (IPHCR) should be programmed accordingly. The signals involved in the receive interface are shown in **Figure 4-20**.

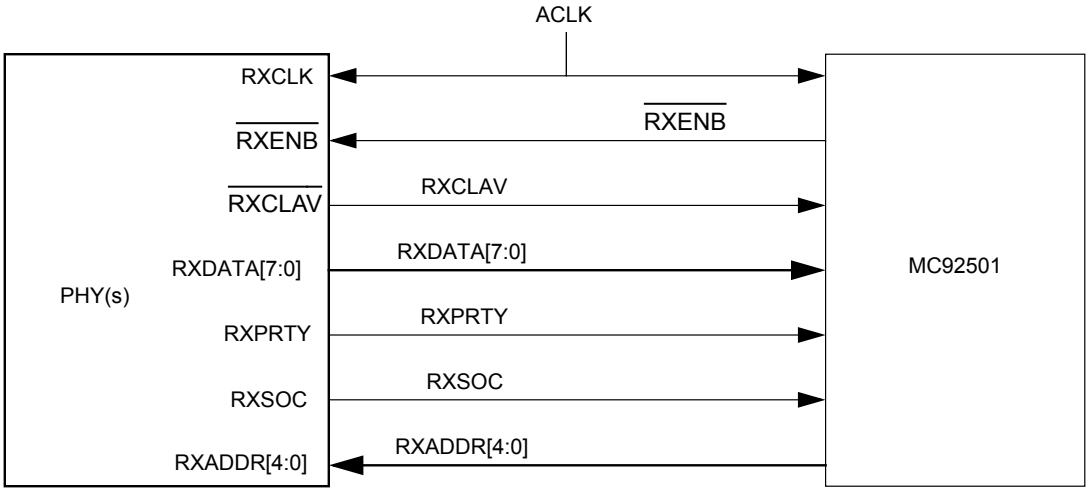


Figure 4-20. MC92501 Receive PHY Interface

In this mode, the *Receive PHY ID 0–3/ Receive Address 0–3 (RXPHYID0–RXPHYID3/ RXADDR0–RXADDR3)* signals function as RXADDR0–RXADDR3.

NOTE: RXADDR4 is a new functional signal in Revision B. It was not used in Revision A of the MC92501.

In order to poll a PHY, the MC92501 assigns the polled PHY address on the RXADDR[4:0] lines on the first clock. The polled PHY drives the RXCLAV signal on the second clock. On that clock the MC92501 assign 1F hex to the RXADDR[4:0] lines and samples the RXCLAV. Whenever there is no cell traffic the MC92501 polls all the PHY's whose corresponding ingress link register *Link Enable (LE)* bit is set in a cyclic descending order. The first PHY that indicates that it is ready is then selected by the MC92501. The MC92501 assigns its address on the RXADDR[4:0] bus with the RXENB signal deasserted on the first clock and on the second clock assign 1F On the RXADDR[4:0] lines and asserts the RXENB signal. One clock later the MC92501 outputs the first byte of the cell on the RXDATA[7:0] lines and continues outputting the cells bytes on subsequent clocks contiguously. See **Figure 4-21**.

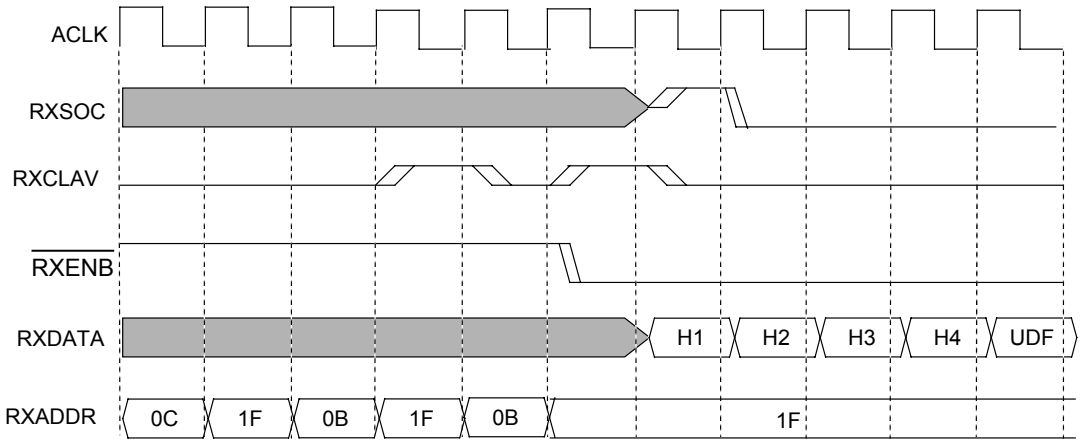


Figure 4-21. Poll PHYs, Select PHY, and Start Reading a Cell

When a cell transfer is in progress, the MC92501 polls the PHYs from the PHY that comes next to the current selected PHY in a descending cyclic order. Once the MC92501 detects a PHY that is ready, it stops the polling process and waits for the current cell transfer to end. It then de-selects current cell, selects the chosen cell, and starts transferring the cell data. See **Figure 4-22**.

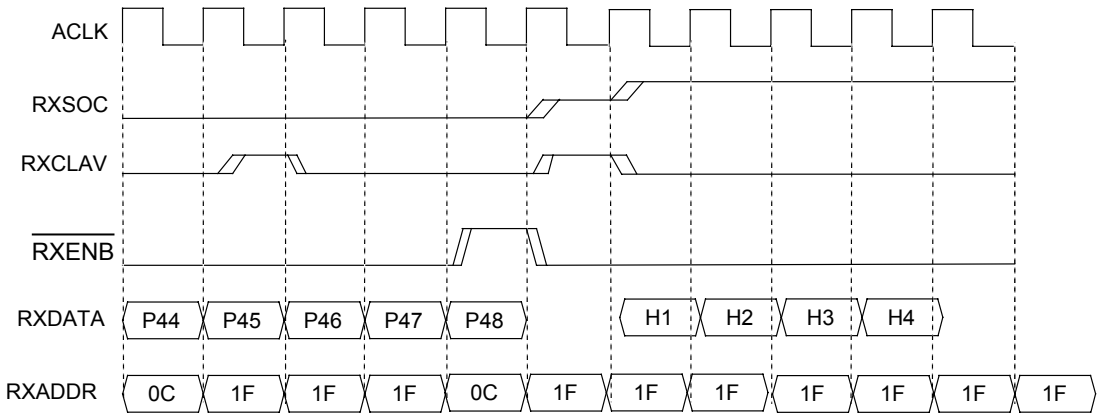


Figure 4-22. The MC92501 Polls the PHYs While Transferring a Cell and Starts Transferring Data from the Selected Cell

The MC92501 does not poll the current selected cell in the middle of its cell transfer since RXCLAV of its PHY may not indicate the availability of the next cell. If no PHY is ready then the MC92501 deselects current cell, polls the current PHY and if it is ready it selects it. See **Figure 4-23**.

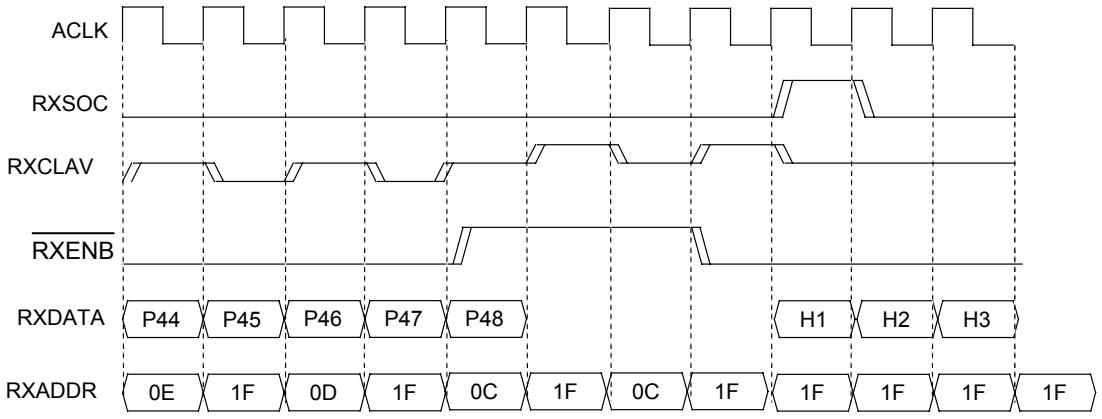


Figure 4-23. The MC92501 Reads Cells from the Same PHY

4.2.4 UTOPIA Level 2 Transmit PHY Interface (Egress)

The MC92501 Transmit PHY Interface supports the UTOPIA Level 2 transmit interface with 1 TXCLAV signal. In this mode the MC92501 operates in cell-based mode only and the The *Egress PHY Operation Mode (EPOM)* bit of the Egress PHY Configuration Register (EPHCR) should be programmed accordingly. The signals involved in the receive interface are shown in **Figure 4-24**.

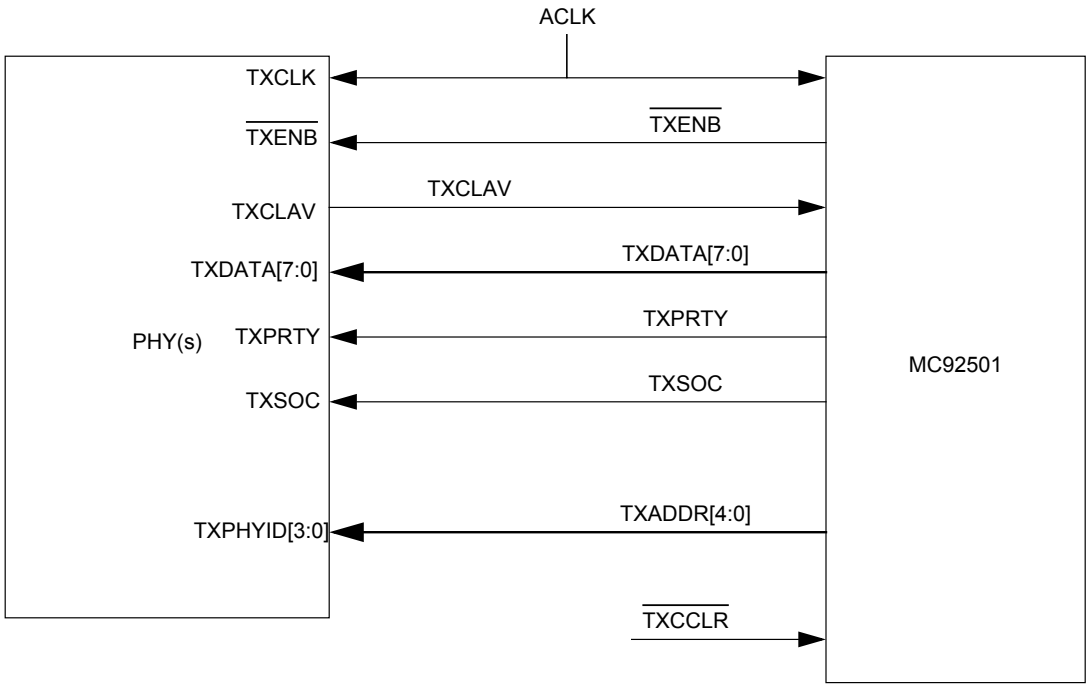


Figure 4-24. MC92501 Transmit PHY Interface

In this mode the *Transmit PHY ID 0–3 / Transmit Address 0–3 (TXPHYID0–TXPHYID3/ TXADDR0–TXADDR3)* signals and the *Transmit Next PHY ID Valid/Transmit Address 4 (TXPHYIDV/TXADDR4)* signal are used as TXADDR0–TXADDR4. The MC92501 polls the link of the cell in its Egress PHY Interface FIFO and when enabled it outputs the cell to the link PHY. The MC92501 performs address polling on all other links as well in order to enable external logic to monitor the PHY's status. **Figure 4-25** illustrates the case where the MC92501is currently outputting a cell to PHY #7. It polls all the PHY's in a cyclic manner and stops 4 clocks prior to the end of the cell with PHY #7 being the last polled PHY. Since the next cell in the TPH belongs to PHY #c, then this PHY is polled 2 clocks prior to the end of cell and is found to be ready. The MC92501 selects PHY #c and starts outputting the next cell to PHY #c.

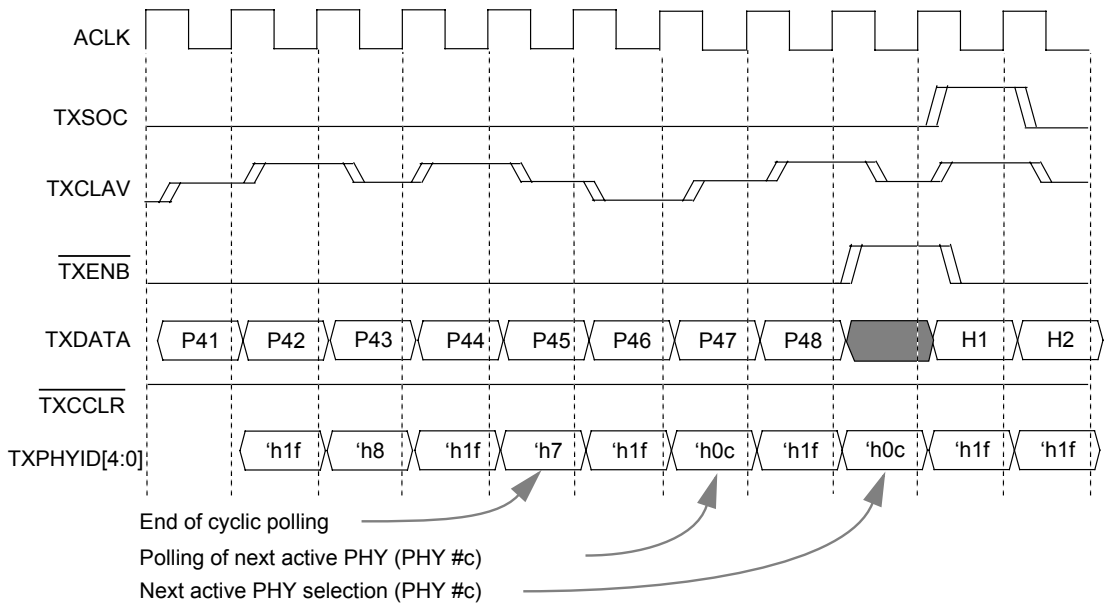


Figure 4-25. ATMC CFB Transmit Timing—End of Cell Output and PHY Switching

Figure 4-26 illustrates an application on which external logic monitors RXCLAV signal while the MC92501 is polling the PHY devices. Based on this information the external logic can input cells to the MC92501 Egress flow.

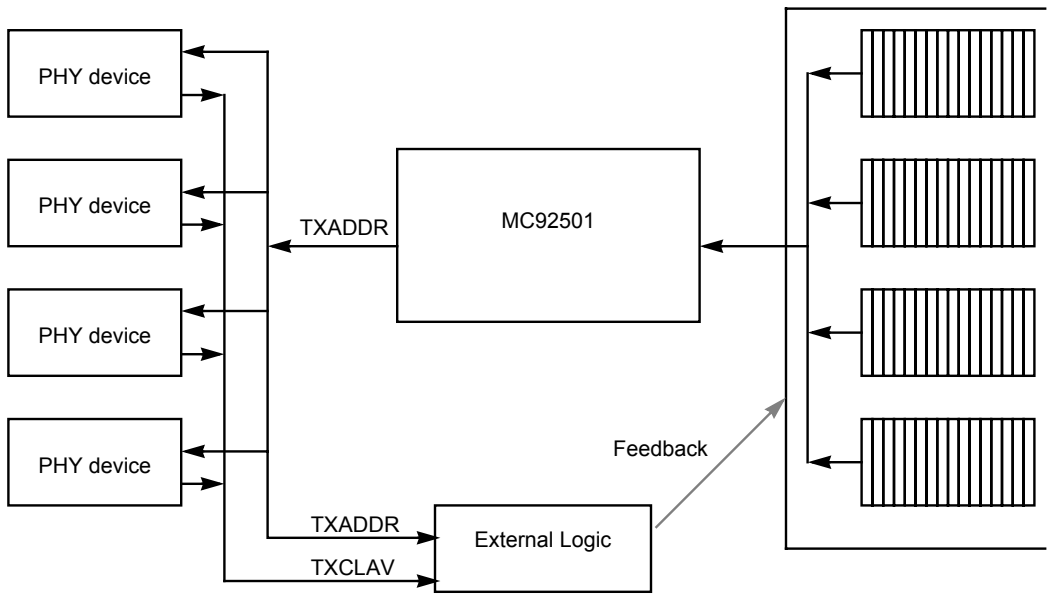


Figure 4-26. ATMC CFB End of Cell Output and PHY Switching

4.3 Switch Interface

The Switch Receive and Transmit Interfaces are similar to the UTOPIA standard interface [12], with the Switch playing the role of the ATM layer and the MC92501 playing the role of the PHY layer. All of the receive interface signals are synchronous to SRXCLK, and all of the transmit interface signals are synchronous to STXCLK. Output signals from the MC92501 are updated following the rising edge of the interface clock, and input signals to the MC92501 are sampled at the rising edge of the interface clock. The flow of data is controlled by enable signals in both directions. The general rule is that an enable signal in the direction of the data flow refers to data during the current cycle and an enable signal in the direction opposite to the data flow refers to data at the end of the next cycle.

4.3.1 Receive Interface (Ingress)

The signals involved in the MC92501-Switch receive interface are shown in **Figure 4-27**.

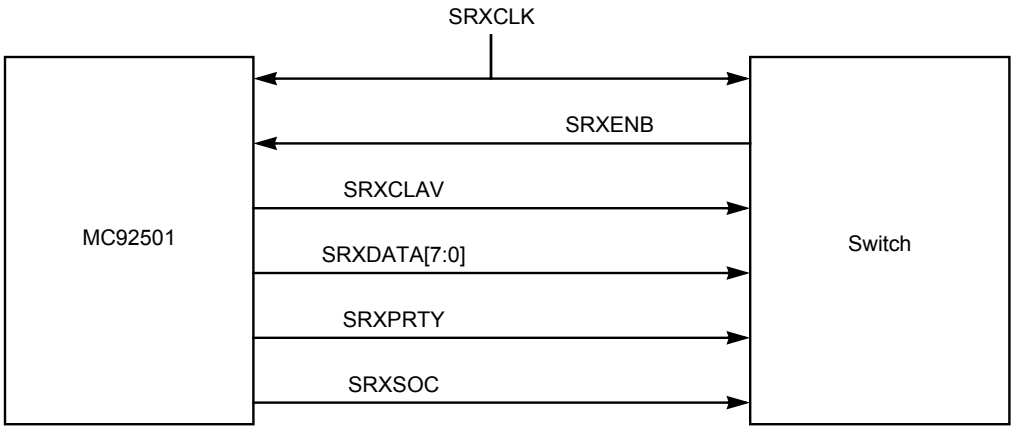


Figure 4-27. MC92501 Switch Receive Interface

The switch requests data transfers in the receive direction by asserting $\overline{\text{SRXENB}}$. The MC92501 responds by supplying valid data (if available) during the next clock cycle. If SRXENB is detected as deasserted at the rising edge of SRXCLK, then SRXDATA, SRXPRTY, and SRXSOC are not updated as shown in **Figure 4-27**. In this way the switch can throttle the flow of data. If the *Ingress Switch SRXDATA Driver Control (ISSDC)* bit of the Ingress Switch Interface Configuration Register (ISWCR) is reset, then SRXDATA, SRXPRTY, and SRXSOC are not driven when $\overline{\text{SRXENB}}$ is deasserted as shown in **Figure 4-28**.

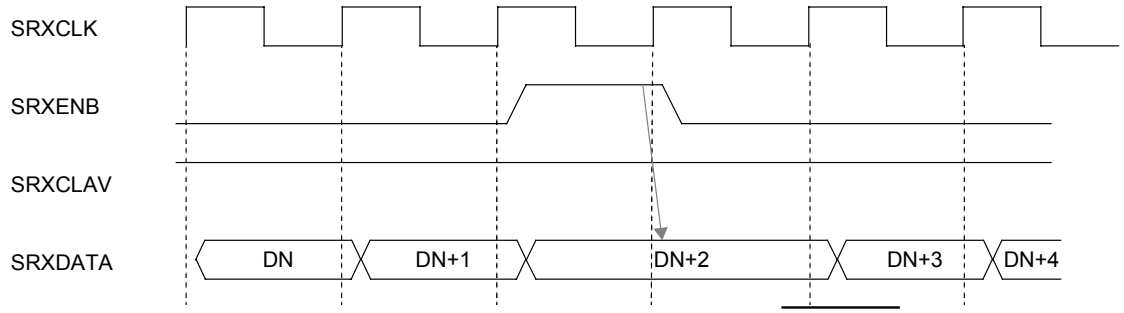


Figure 4-28. Receive Timing without Tri-state—Effect of SRXENB Deassertion

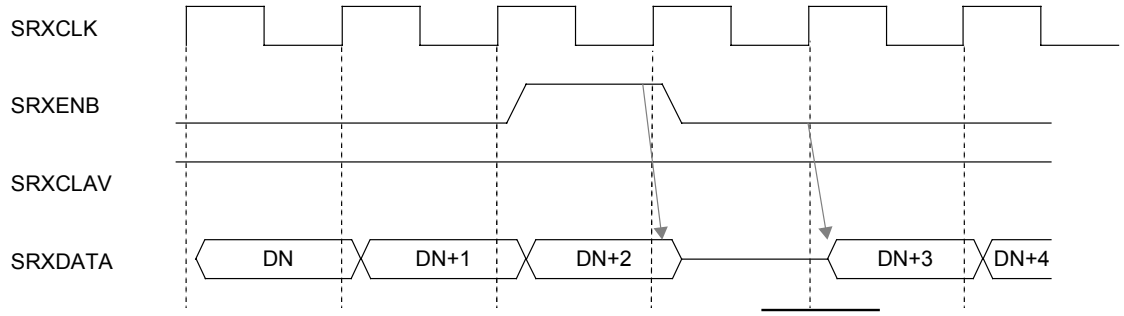


Figure 4-29. Receive Timing with Tri-state—Effect of SRXENB Deassertion

Figure 4-30 shows the beginning of a cell transfer. Upon detecting SRXCLAV asserted, the switch should assert SRXENB within a cell transfer slot (the number of SRXCLK cycles required to transfer one cell to the switch). Otherwise, the MC92501 may be unable to accept data from the PHY, and cells may be lost. The MC92501 asserts SRXSOC together with the first octet of each cell in order to synchronize the switch to the beginning of a new cell. For example, if 57 octets are transferred, SRXSOC is asserted on the transfer of the first overhead byte and not on the first header byte.

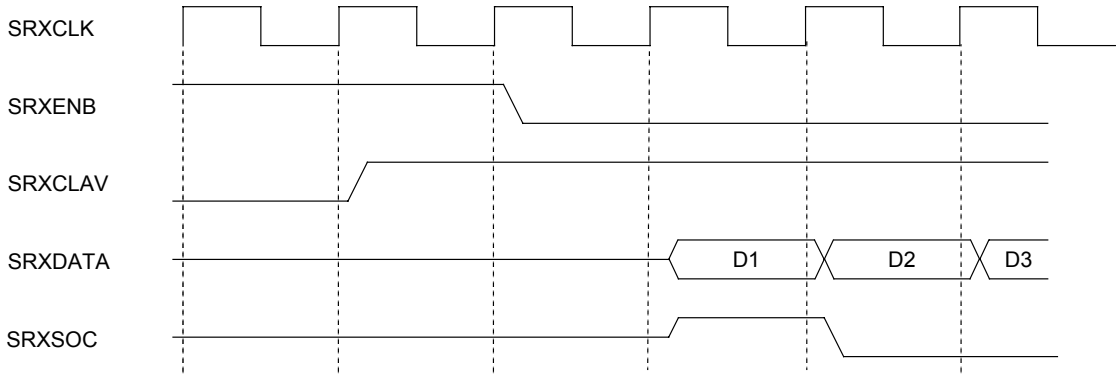


Figure 4-30. Receive Timing—Start of Cell

The MC92501 deasserts SRXCLAV at the end of a cell transfer if it does not have another cell to provide as shown in **Figure 4-31**. If there is another cell available, SRXCLAV is not deasserted and the cells can be transferred back-to-back as shown in **Figure 4-32**.

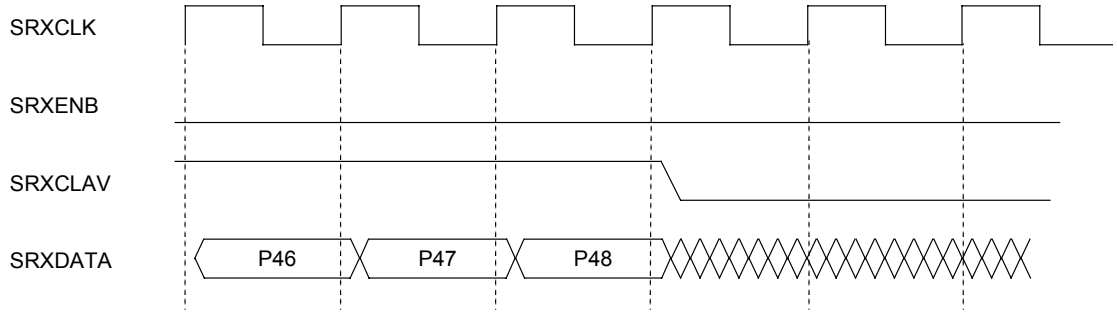


Figure 4-31. Receive Timing—End of Cell

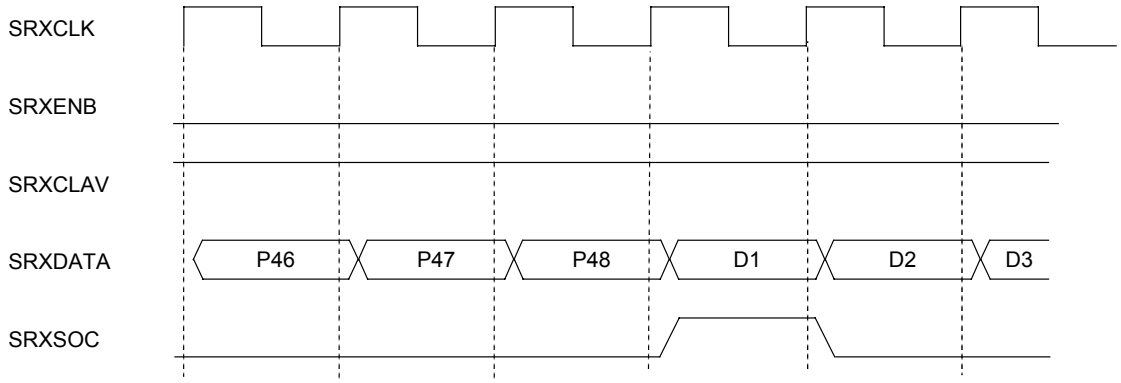


Figure 4-32. Receive Timing—Back-to-Back Cell Transfers

SRXPRTY contains the parity over SRXDATA. It is not shown in the timing diagrams since its timing is identical to that of SRXDATA. The type of parity (even/odd) is determined by the *Ingress Switch Parity Mode (ISPM)* bit defined in Section 7.2.6.5 Ingress Switch Interface Configuration Register (ISWCR).

4.3.2 Transmit Interface (Egress)

The signals involved in the MC92501-Switch transmit interface are shown in **Figure 4-33**.

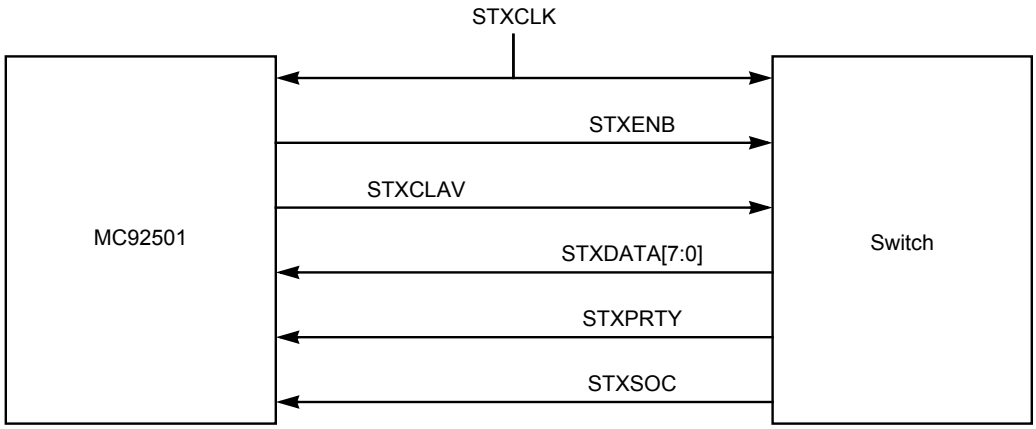


Figure 4-33. MC92501 Switch Transmit Interface

Figure 4-34 shows the relationships among the transmit interface signals. When the switch has data to transfer and STXCLAV is asserted, the switch asserts STXENB and drives STXDATA with an octet of valid data. Once STXCLAV is asserted, the MC92501 must accept an entire cell, and the switch may continue to transfer one byte per clock cycle until the end of the cell. The switch asserts STXSOC together with the first octet of each cell to synchronize the MC92501 to the start of the cell.

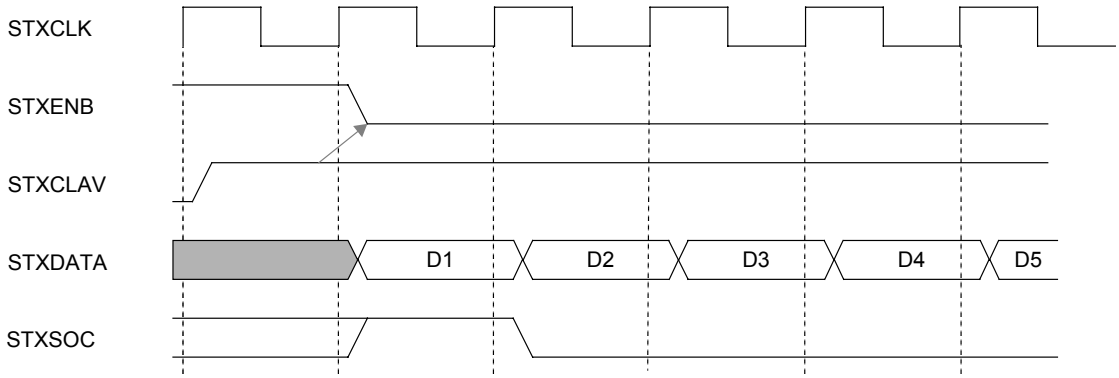


Figure 4-34. Transmit Timing—Start of Cell

When the MC92501 is full, it deasserts STXCLAV at least four clock cycles before the end of the cell as shown in **Figure 4-35**. When the switch detects the deassertion of STXCLAV, it transfers the remainder of the cell and then deasserts STXENB. The switch may assert STXENB only after detecting the assertion of STXCLAV. Asserting STXENB otherwise is an error and is ignored by the MC92501. The MC92501 samples STXDATA, STXPRTY, and STXSOC on each rising edge of STXCLK during which STXENB is asserted.

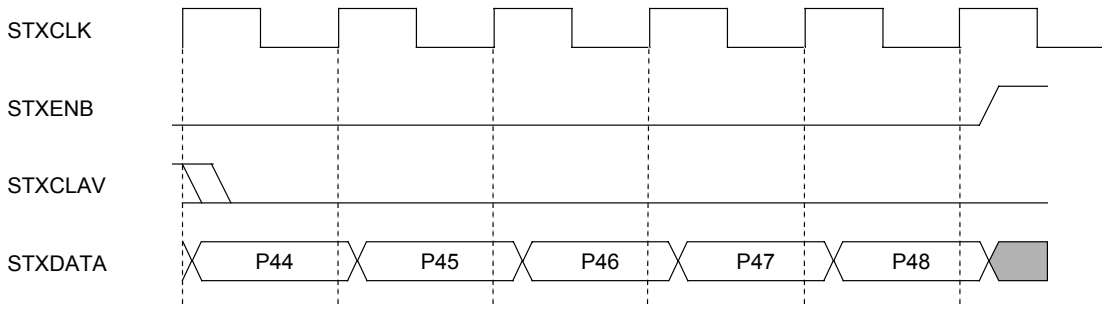


Figure 4-35. Transmit Timing (End of Cell)

STXPRTY contains the parity over STXDATA. It is not shown in the timing diagrams since its timing is identical to that of STXDATA. The type of parity (even/odd) is determined by the *Egress Switch Parity Control (ESPC)* bit defined in **Section 7.2.6.6** Egress Switch Interface Configuration Register (ESWCR). Parity checking is controlled by the *Egress Switch Parity Enable (ESPR)* and *Egress Payload Parity Enable (EPLP)* bits defined in Section 7.2.6.6 Egress Switch Interface Configuration Register (ESWCR).

4.4 External Memory Interface

The MC92501 is the sole master of the External Memory (EM) interface. The microprocessor can access the EM during maintenance slots by using the External Memory address space. These maintenance accesses are described later in this section. Due to the bandwidth needed from the EM and the requirement of one access per ACLK cycle, a fast static RAM with an access time of 20 ns is chosen for 25 MHz operation. **Figure 4-36** shows a recommended design for 4 MB of EM. By using 4M × 1 memory devices, the entire EM address space (16 MB) can be implemented. It should be noted that if more than four loads are placed on a single bank select signal, the signal must be buffered and 15 ns memory devices should be used.

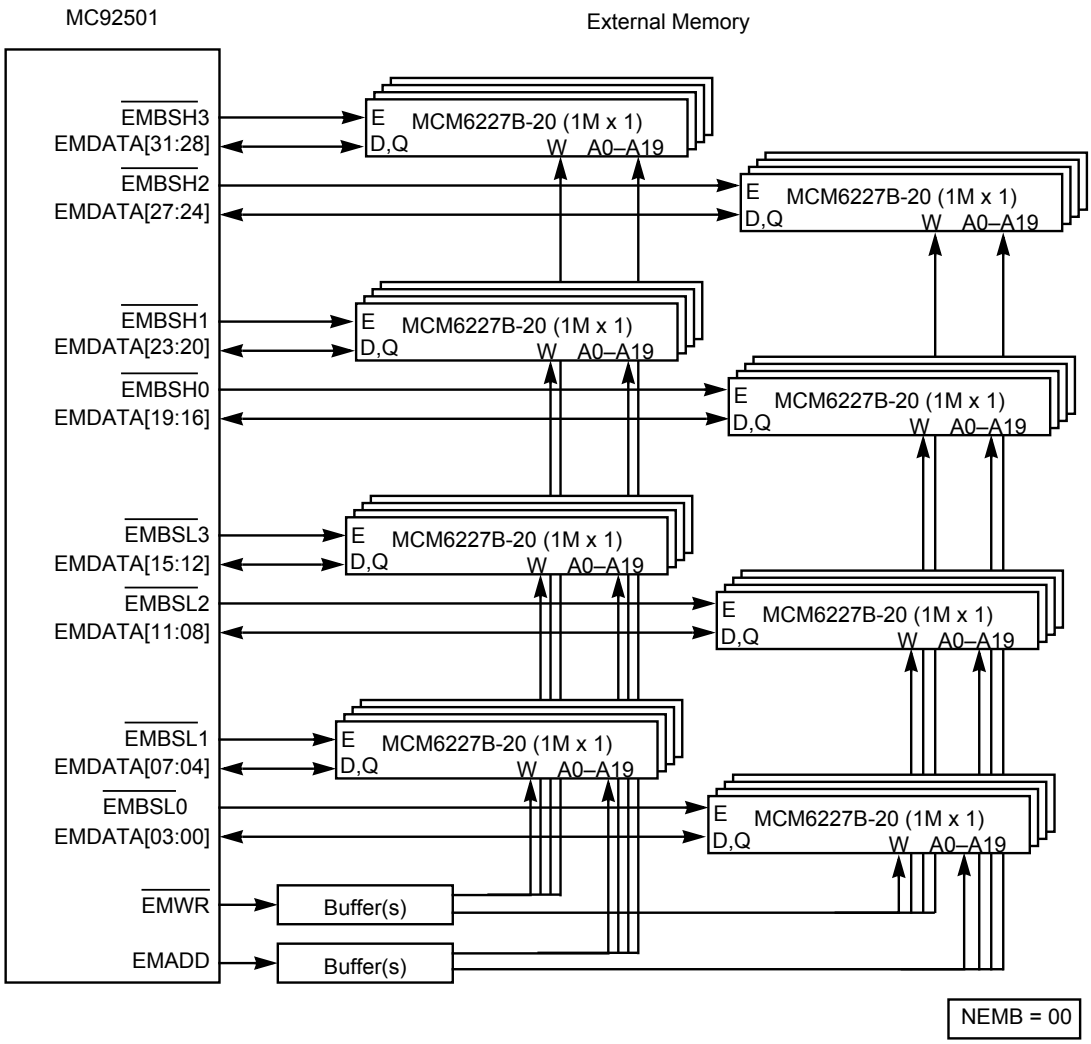


Figure 4-36. External Memory Configuration Providing 4 MB in 1 Bank

4.4.1 EM Bank Select Signals

The MC92501 provides eight bank select signals for the EM. The eight signals are organized as four pairs of word select signals, providing the option to divide the EM into as many as four banks. The operation of the bank select signals is controlled by the External Memory Interface Configuration Register (EMICR). The *Number of External Memory Banks (NEMB)* field provides three options: one bank, two banks, and four banks.

If the memory is organized as one bank:

- All $\overline{\text{EMBSL}n}$ signals are asserted when the low word of the EM is accessed. (16- or 32-bit access)
- All $\overline{\text{EMBSH}n}$ signals are asserted when the high word of the EM is accessed. (16- or 32-bit access)

Each group of four signals can be used to divide the load if the bank is built in bit slices. Such a configuration is shown in **Figure 4-36**.

If two banks are used:

- $\overline{\text{EMBSL0}}$and $\overline{\text{EMBSL2}}$ are asserted when the low word of the EM Bank 0 is accessed. (16- or 32-bit access)
- EMBSL1 and $\overline{\text{EMBSL3}}$ are asserted when the low word of the EM Bank 1 is accessed. (16- or 32-bit access)
- EMBSH0 and $\overline{\text{EMBSH2}}$ are asserted when the high word of the EM Bank 0 is accessed. (16- or 32-bit access)
- EMBSH1 and $\overline{\text{EMBSH3}}$ are asserted when the high word of the EM Bank 1 is accessed. (16- or 32-bit access)

Each group of two signals can be used to divide the load if the banks are built in bit slices. If four banks are used:

- $\overline{\text{EMBSL0}}$ is asserted when the low word of the EM Bank 0 is accessed. (16- or 32-bit access)
- EMBSL1 is asserted when the low word of the EM Bank 1 is accessed. (16- or 32-bit access)
- EMBSL2 is asserted when the low word of the EM Bank 2 is accessed. (16- or 32-bit access)
- EMBSL3 is asserted when the low word of the EM Bank 3 is accessed. (16- or 32-bit access)
- EMBSH0 is asserted when the high word of the EM Bank 0 is accessed. (16- or 32-bit access)
- EMBSH1 is asserted when the high word of the EM Bank 1 is accessed. (16- or 32-bit access)
- EMBSH2 is asserted when the high word of the EM Bank 2 is accessed. (16- or 32-bit access)
- EMBSH3 is asserted when the high word of the EM Bank 3 is accessed. (16- or 32-bit access)

This configuration is shown in **Figure 4-37**.

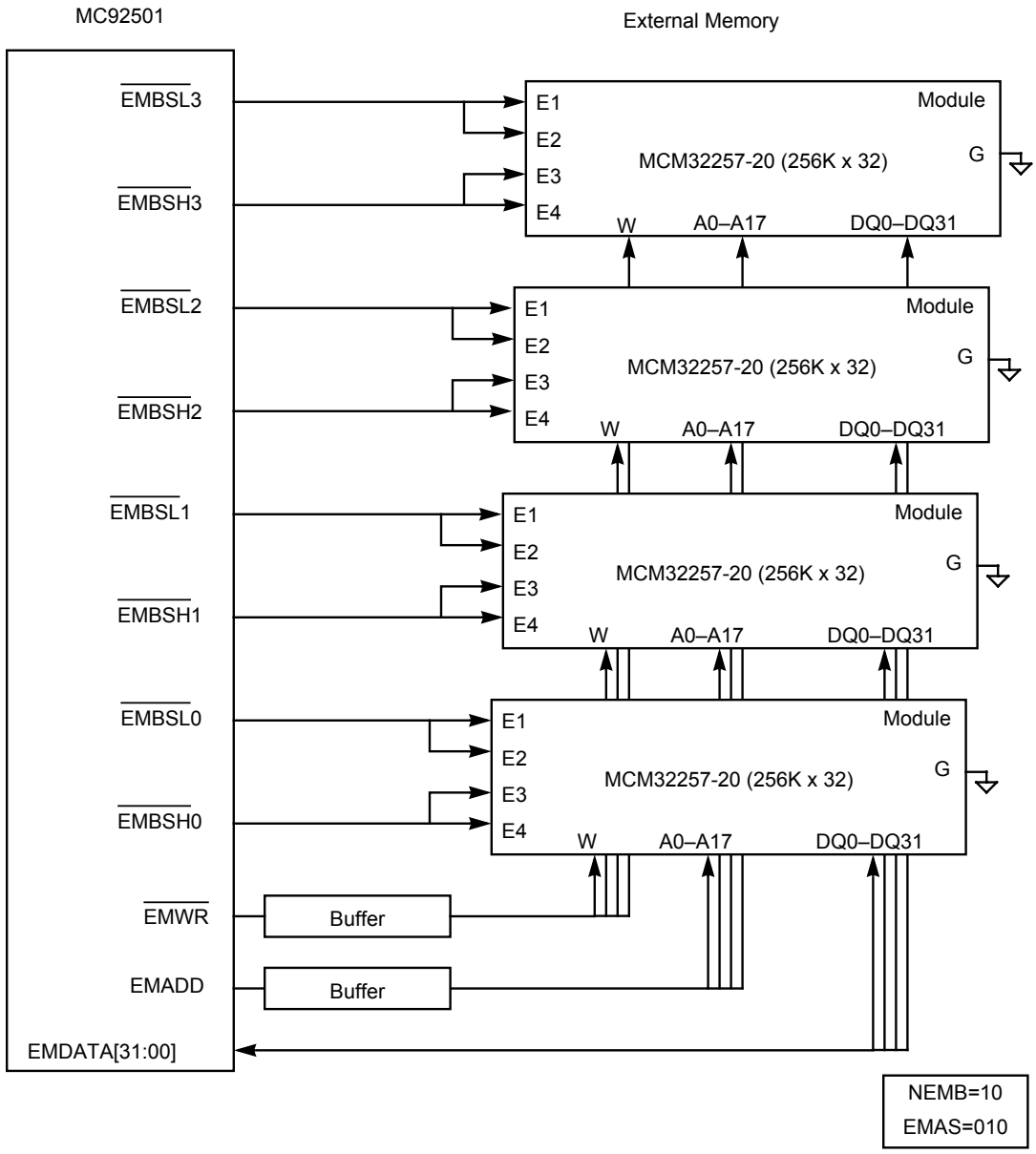


Figure 4-37. EM Configuration Providing 4 MB Using 4 Banks

The MC92501 determines which bank is addressed according to the one or two MSBs of the active address. The active address is the portion of EMADD that is actually in use, as determined by the *External Memory Address Space (EMAS)* field defined in Section 7.2.6.34 External Memory Interface Configuration Register (EMICR). This is illustrated in **Table 4-1** and **Table 4-1**. The interpretation of the MSBs of the active address is presented in **Table 4-3**.

Table 4-1. EM Address Bits Used for Bank Select (2 Banks)

EMAS	EM Address Bits															
000	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
001		22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
010			21	20	19	18	17	16	15	14	13	12	11	10	9	8
011				20	19	18	17	16	15	14	13	12	11	10	9	8
100					19	18	17	16	15	14	13	12	11	10	9	8
101						18	17	16	15	14	13	12	11	10	9	8

Table 4-2. EM Address Bits Used for Bank Select (4 Banks)

EMAS	EM Address Bits															
000	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
001		22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
010			21	20	19	18	17	16	15	14	13	12	11	10	9	8
011				20	19	18	17	16	15	14	13	12	11	10	9	8
100					19	18	17	16	15	14	13	12	11	10	9	8
101						18	17	16	15	14	13	12	11	10	9	8

Table 4-3. Active EM Banks Selected by the MSBs of the Active Address

NEMB	MSB = 00	MSB = 01	MSB = 10	MSB = 11
00	0,1,2,3			
01	0,2		1,3	
10	0	1	2	3

4.4.2 EM Interface Timing For Normal Access

The MC92501 provides great flexibility in the design of the External Memory by making the timing of the data and control signals programmable in units of 1/8 of the ACLK period for normal (non-maintenance) accesses. The various options are described in Section 7.2.6.33 External Memory Timing Configuration Register (EMTCR). At reset, the variables are loaded with default values that provide the correct timing for typical configurations. The timing diagrams in this section assume the default values.

4.4.2.1 Normal Read Cycle

The MC92501 uses a read cycle to read data from one of the tables residing in the External Memory. The timing illustrated in **Figure 4-38** uses the default timing values for a read cycle where the bank select signals are asserted at 2/8 past the ACLK rising edge and deasserted at the rising edge. EMDATA is sampled at the rising edge of ACLK.

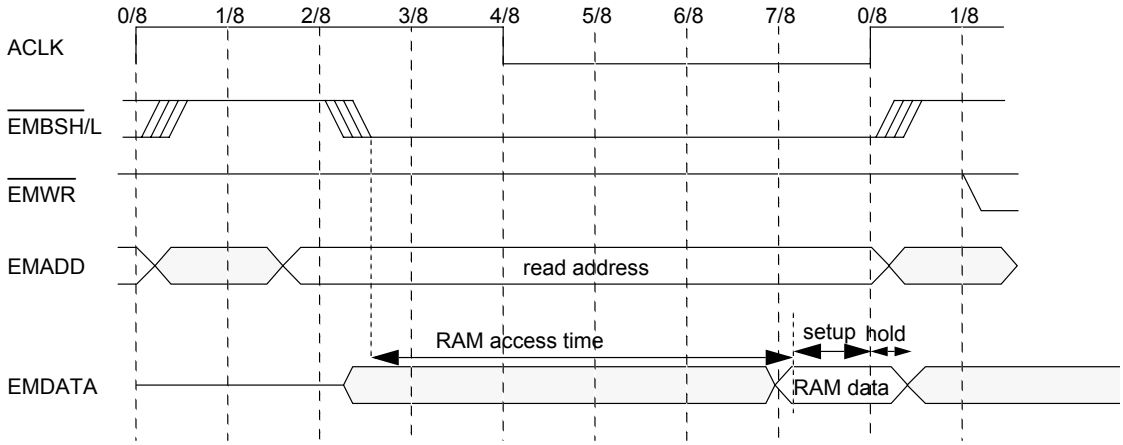


Figure 4-38. External Memory Normal Read Cycle

4.4.2.2 Normal Write Cycle

The MC92501 uses a write cycle to write data to one of the tables residing in the External Memory. The timing illustrated in **Figure 4-39** uses the default timing values for a write cycle. The bank select signals are asserted at 2/8 past the ACLK rising edge and deasserted at 6/8 past the rising edge.

is asserted at 1/8 past the ACLK rising edge and deasserted at 6/8 past the rising edge. EMDATA is driven from 2/8 past the ACLK rising edge until the next rising edge.

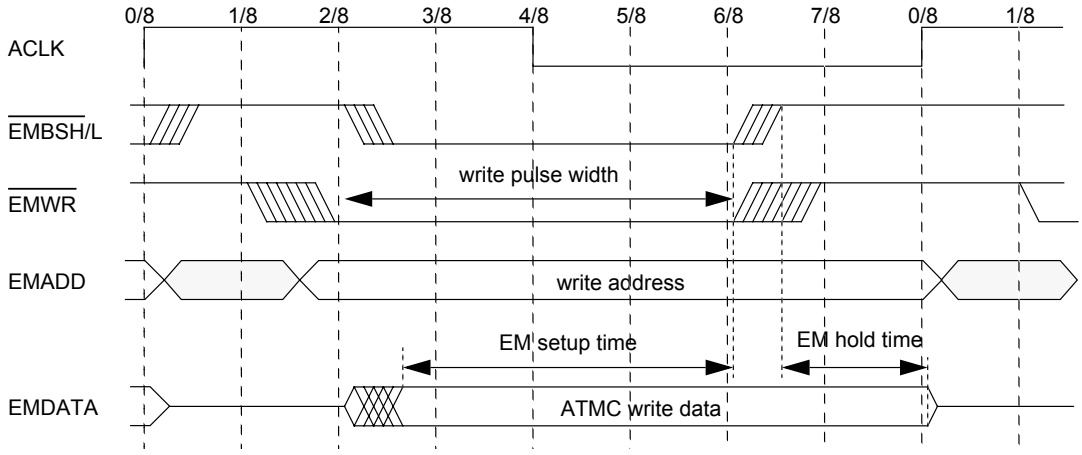


Figure 4-39. External Memory Normal Write Cycle

4.4.3 EM Interface Timing for Maintenance Access

During maintenance cell slots, the MC92501 does not use the External Memory for cell processing. Instead, the microprocessor interface is used to access the External Memory through the External Memory address space (see **Figure 7-1**). The MC92501 connects the Microprocessor address and data buses to the External Memory address and data buses internally. The MC92501 also drives the External Memory control signals according to the type of access being performed by the microprocessor. The extent of a maintenance cell slot is 64 ACLK cycles. The number of accesses that the microprocessor can perform during this time depends on the ratio of the frequencies of MCLK and ACLK.

There are three types of External Memory accesses that the microprocesor can perform : read, write, and read/clear. A read/clear access is a read access using the “destructive” address space. When a read/clear access is performed, the MC92501 automatically writes back zeros to the location that the microprocessor read. This is accomplished during the “dead” time between accesses, thereby reducing by 50% the amount of time required to read and clear the cell counters, for example. When the microprocessor is performing External Memory accesses, the External Memory interface becomes a synchronous interface using MCLK. This avoids having to synchronize signals between the Microprocessor and External Memory interfaces.

Figure 4-40 shows a maintenance read access. EMDATA is sampled at least three MCLK phases after the assertion of the bank select signals. This timing is much more relaxed than a normal access, so it should not constrain the design of the External Memory..

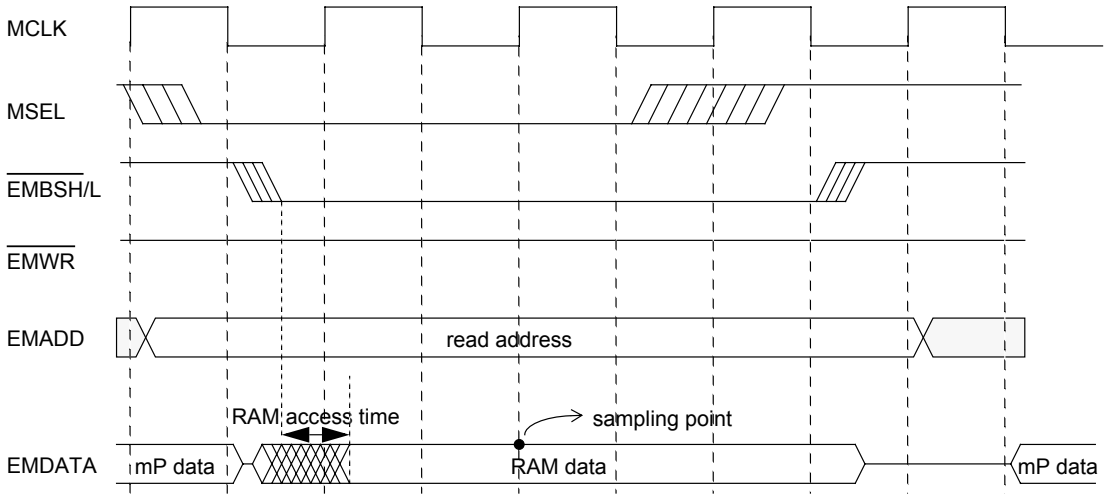


Figure 4-40. External Memory Maintenance Read Access

Figure 4-41 shows a maintenance write access. The bank select signals are asserted for at least one MCLK cycle. This timing is much more relaxed than a normal access, so it should not constrain the design of the External Memory.

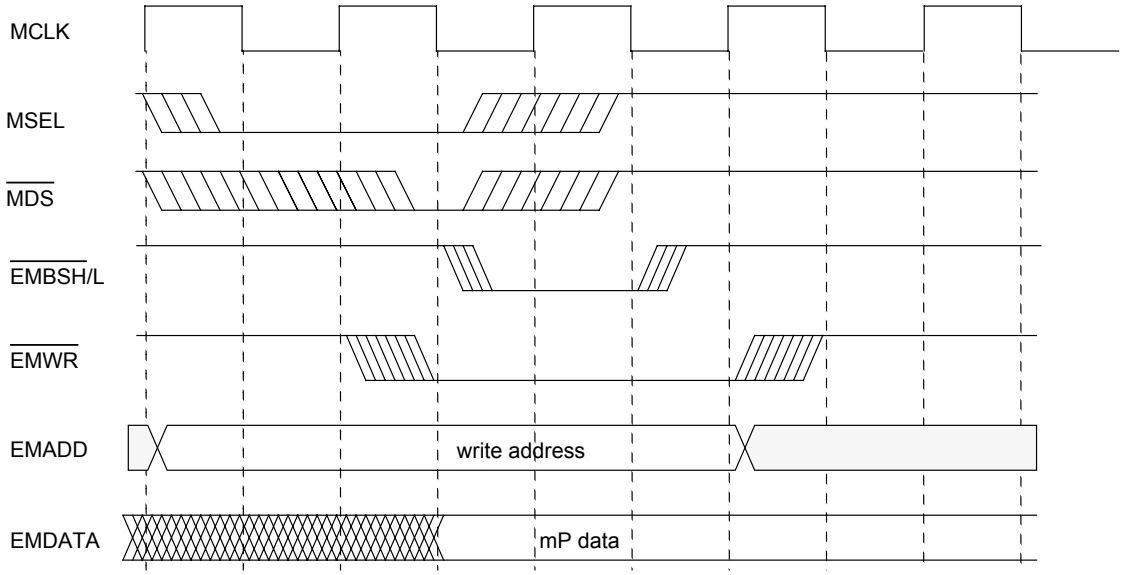


Figure 4-41. External Memory Maintenance Write Access

Figure 4-42 shows a maintenance read/clear access. The write of zero to the External Memory extends for three MCLK phases. The entire read/clear sequence requires four MCLK cycles, which is the minimum length of the read access on the Microprocessor Interface. Thus, the clear does not extend the read access from the system point of view.

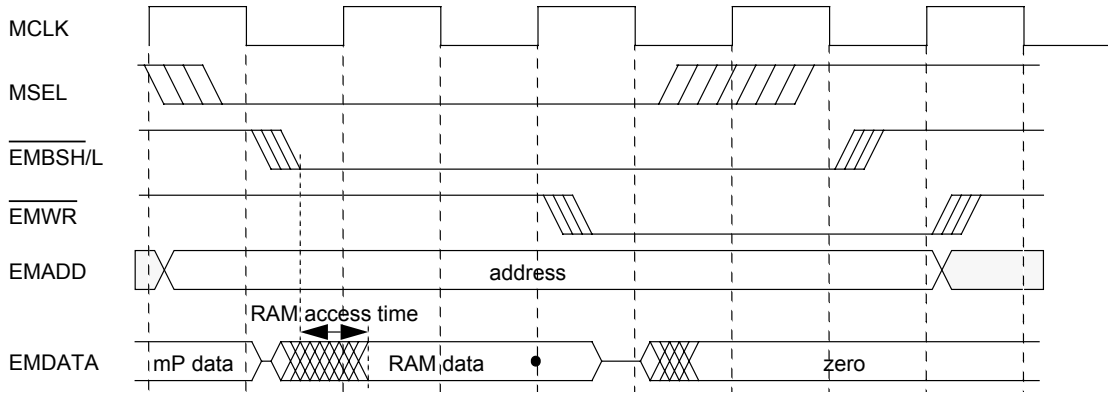


Figure 4-42. External Memory Maintenance Read/Clear Access

NOTE: When performing a read/clear maintenance access, at least one wait state is required for proper operation even if the duration of an MCLK phase is longer than T_d.

4.4.4 External Address Compression Device Access

Normal (non-maintenance) accesses to the External Address Compression (EAC) Device have the same timing as normal External Memory accesses, except that EACEN is asserted instead of the EM bank select signals. The MC92501 always drives EMADD with all 1s when accessing the EAC Device. Maintenance accesses to the EAC Device have the same timing as External Memory maintenance accesses, except that the EACEN signal is asserted instead of the EM bank select signals. The value provided on MADD is driven on EMADD, so 24 bits (16 MB) of External Address Compression Device address space are available to the microprocessor. **Figure 4-43** shows a possible implementation of the EAC circuitry. Note that the MC92501 sees the EAC as one write register and one read register. The microprocessor is free to use the same data path with other addresses to control and/or program the address compression device during maintenance slots.

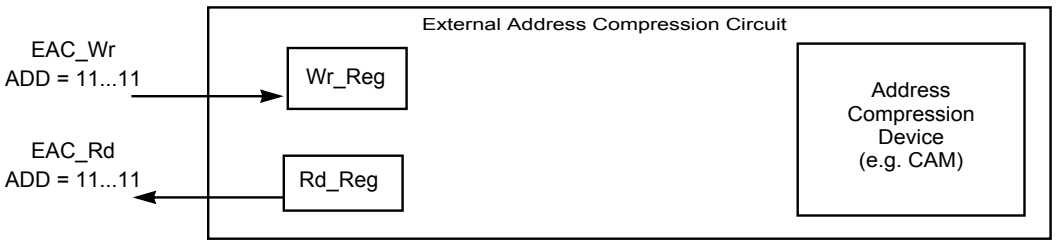


Figure 4-43. Example Implementation of External Address Compression

4.5 Microprocessor Interface

The Microprocessor Interface (MPIF) block provides for configuration of the MC92501 in addition to the transfer of cells between the microprocessor and the MC92501. A standard 32-bit slave interface is provided for easy connection to the microprocessor bus. This section describes the Read and Write operations on the processor port.

4.5.1 Processor Read and Write Operations

MADD and MWR are sampled by the MC92501 on the first MCLK falling edge at which MSEL is detected asserted, but they are also used at the assertion of MSEL. Therefore, both MADD and MWR have to be stable from a Setup time before MSEL is asserted until a Hold time after the MCLK falling edge. During a read access, MWR should remain deasserted for the entire access since the MDATA bus is driven by the MC92501 whenever MSEL is asserted and MWR is deasserted (Read).

Data Acknowledge pins (MDTACK0 and MDTACK1) are provided for designs that do not use wait state generators. In the Following explanation MDTACK*n* is referred to instead of both MDTACK0 and MDTACK1. During a write access, MDTACK*n* is asserted by the MC92501 to indicate that the data has been sampled. Once MDTACK*n* has been detected asserted at a falling edge of MCLK, MDATA is allowed to become invalid a hold time after the next falling edge. During a read access, MDTACK*n* is asserted by the MC92501 to indicate that the data is valid. Once MDTACK*n* has been detected asserted at a falling edge of MCLK, MDATA may be sampled at the next falling edge. MDTACK*n* is a tri-state output signal. It is driven by the MC92501 from the assertion of MSEL until the falling edge of MCLK following the deassertion of MSEL. Active pull-up is provided by deasserting MDTACK*n* at the deassertion of MSEL and continuing to drive it until the falling edge of MCLK.

NOTE: The standard timing of MDTACK*n* for read accesses takes into account typical values of capacitance on MDATA and typical setup times for data input pins. If these values cannot be met, the assertion of MDTACK*n* may be delayed by using the fields of the Microprocessor Configuration Register (MPCONR). The designer should verify that the data setup times are met when relying on MDTACK*n*.

There are 2 MDTACK signals instead of 1 in order to enable glueless interface to systems in which there are 2 MDTACK signals and their combination conveys the bus width of the slave. The MDTACK1 signal is driven only when the MDTACK0 signal is driven and when the *Mdtack Drive Control (MDC)* is set. Refer to for details.

NOTE: See Section 9.3 Electrical and Physical Characteristics for detailed timing information.

4.5.1.1 Processor Read Operations

There are three basic types of microprocessor read operations which are described in the following paragraphs: general register reads, cell extraction register reads, and maintenance read accesses.

4.5.1.1.1 General Register Read

When reading an MC92501 general register, the read operation must be extended long enough to allow for synchronization to the cell processing clock (ACLK). This is accomplished by adding wait states. The MC92501 asserts MDTACK*n* when the data is valid. The assertion of MDTACK*n* may be delayed for one or two ACLK periods by using the *DTACK Drive during General Register Accesses (DDGR)* field of the Microprocessor Configuration Register (MPCONR). **Figure 4-44** describes a register read operation. The MC92501 samples MADD, MWR, and MSEL on the MCLK falling edge. When MDATA is valid, the MC92501 drives MDTACK*n* low (asserted). The MC92501 continues to drive MDATA and assert MDTACK*n* as long as MSEL is asserted and MWR is deasserted.

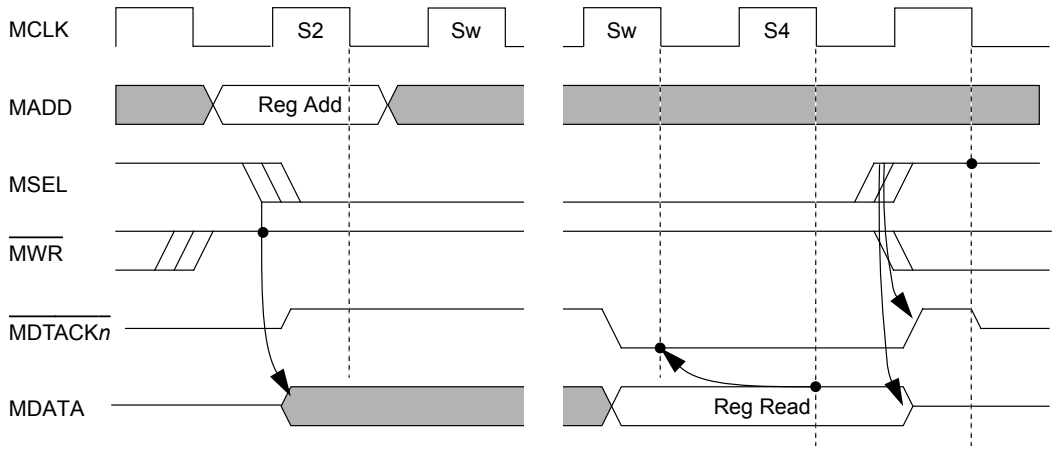


Figure 4-44. MC92501 Register Read Timing with DTACK

Figure 4-45 describes a register read operation that does not use MDTACK*n*. In this case the microprocessor must insert a predetermined number of wait states before sampling the data. MDATA is valid once T_R has elapsed since the falling edge of MCLK following the assertion of MSEL. The value of T_R depends on the period of ACLK. Note that this calculation defines the minimum number of wait states, but more may be added. The MC92501 continues to drive MDATA as long as MSEL is asserted and MWR is deasserted.

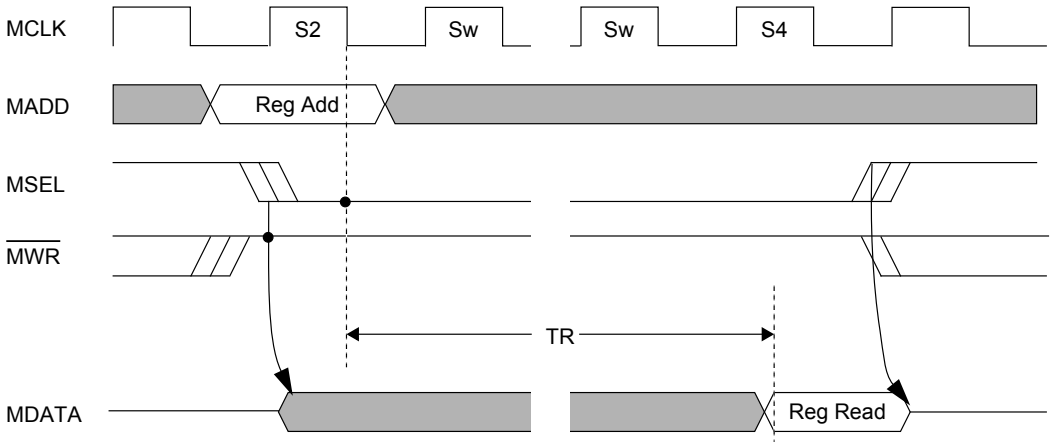


Figure 4-45. MC92501 Register Read Timing without DTACK

4.5.1.1.2 Cell Extraction Register Read

To improve performance, the MC92501 Cell Extraction Registers receive special treatment. **Figure 4-46** describes a Cell Extraction Register read operation. The MC92501 drives the register value on the MDATA pins after detecting MSEL asserted, MWR deasserted, and MADD containing a CER address on the MCLK falling edge. The value on the MDATA pins is stable after propagation delays from both the assertion of MSEL and the MCLK falling edge. When using the MC68360 QUICC or a similar processor, the timing of MDATA is sufficient that it may be sampled at the next MCLK falling edge, so no wait states are necessary. The MC92501 continues to drive MDATA as long as MSEL is asserted and MWR is deasserted, so the processor is free to insert one or more wait states on its own initiative. To allow a zero-wait-state access even when using the MDTACK*n* signal, the MC92501 asserts MDTACK*n* asynchronously as a direct result of the assertion of MSEL with MADD containing a CER address. Thus, if MSEL is asserted early enough, the assertion of MDTACK*n* can be detected at the S2 falling edge, and no wait states are added. The assertion of MDTACK*n* may be delayed for one or two MCLK periods by using the *DTACK Drive during Cell Extraction Register Accesses (DDCE)* field of the Microprocessor Configuration Register (MPCONR).

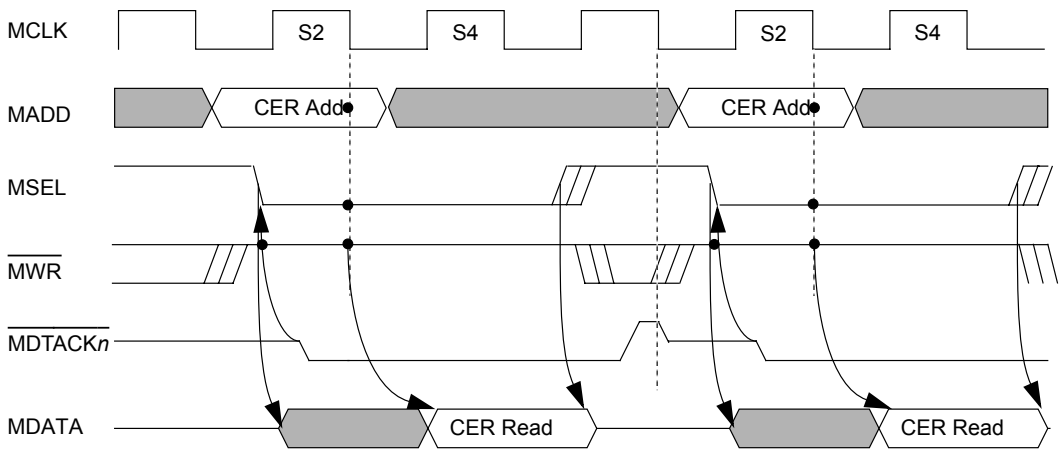


Figure 4-46. Cell Extraction Register Read Timing

4.5.1.1.3 Maintenance Read Access

The processor uses a Maintenance Read Access to read from the External Memory or EAC Device through the MC92501 Microprocessor Interface, as shown in **Figure 4-47**. The MC92501 samples MADD at the falling edge of MCLK where MSEL is detected asserted, reads the External Memory using the address provided, and drives the data on MDATA. MDATA is valid a delay time (T_d) after the next rising edge of MCLK. When using the MC68360 QUICC or similar processor, the MDATA may be sampled at the third MCLK falling edge from the assertion of MSEL, so one wait state is necessary. The MC92501 continues to drive MDATA as long as MSEL is asserted and MWR is deasserted, so the processor is free to insert more wait states. If additional wait states are inserted on a Maintenance Read (not read/clear) Access, the External Memory read cycle is extended and the EM data continues to be sampled. This type of access can used for an EAC Device that cannot meet the 20 ns access time requirement.

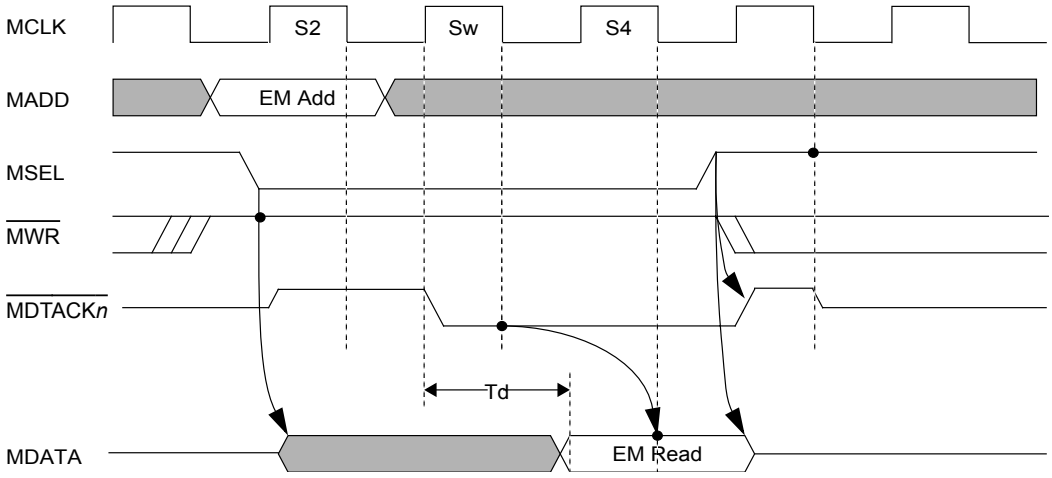


Figure 4-47. Maintenance Read Access Timing

In order to allow a one-wait-state access when using the $\overline{\text{MDTACK}n}$ signal, the MC92501 asserts MDTACK*n* at the rising edge of MCLK following the detection of the assertion of MSEL with MWR deasserted and MADD containing an External address at the S2 falling edge. Thus, the assertion of MDTACK*n* can be detected at the next falling edge, and one wait state is added. The assertion of MDTACK*n* may be delayed for one or two MCLK periods by using the *DTACK Drive during External Memory Accesses (DDEM)* field of the Microprocessor Configuration Register (MPCONR).

NOTE: When performing a read/clear maintenance access, at least one wait state is required for proper operation even if the duration of an MCLK phase is longer than T_d.

4.5.1.2 Processor Write Operations

As with the processor read operation, there are three basic types of microprocessor write operations which are described in the following paragraphs: general register writes, cell insertion register writes, and maintenance write accesses.

4.5.1.2.1 General Register Write

When writing to an MC92501 general register, the write operation must be extended long enough to allow for synchronization to the cell processing clock (ACLK). This is accomplished by adding wait states. The MC92501 asserts MDTACK*n* when the register has been written. At this point, the write operation may be completed.

Figure 4-48 describes a register write operation. The MC92501 starts the write operation when MSEL is asserted by sampling MADD and MWR. MDS should be asserted when MDATA is valid. When the register has been written, the MC92501 drives MDTACK*n* low (asserted). The MC92501 continues to drive MDTACK*n* as long as MSEL is asserted.

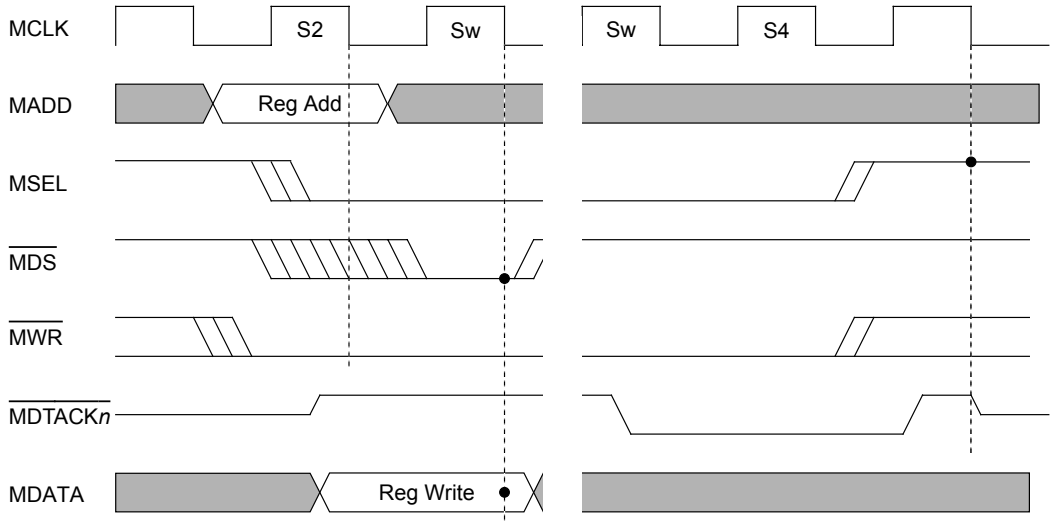


Figure 4-48. MC92501 Register Write Timing with DTACK

Figure 4-49 describes a register write operation that does not use MDTACK*n*. In this case, the microprocessor must insert a predetermined number of wait states before deasserting MSEL. MSEL must remain asserted for time T_W . The value of T_W depends on the period of ACLK.

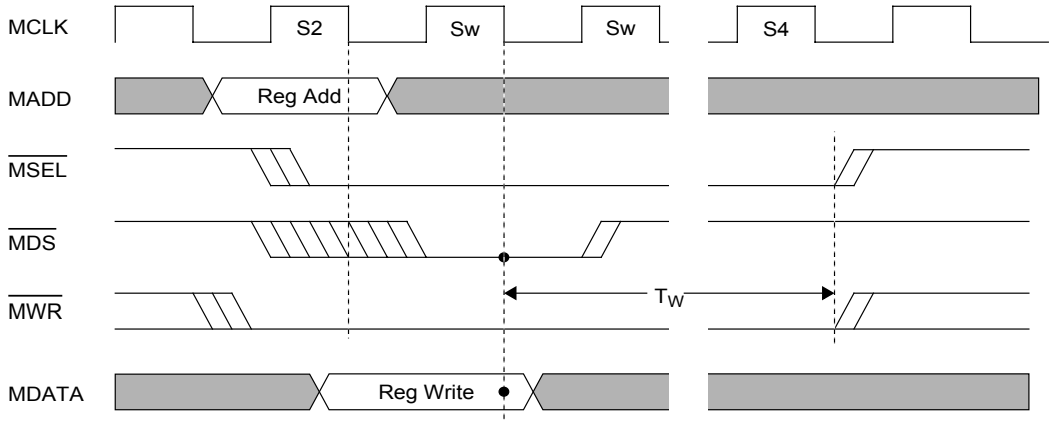


Figure 4-49. MC92501 Register Write Timing without DTACK

4.5.1.2.2 Cell Insertion Register Write

In order to improve performance, the MC92501 Cell Insertion Registers receive special treatment. **Figure 4-50** describes a Cell Insertion Register write operation. The MC92501 starts the write operation by sampling MADD, MWR, and MSEL on the MCLK falling edge. The MC92501 samples MDATA on the next MCLK falling edge since MDS is asserted. Note that MDS is ignored until one cycle after MSEL has been sampled asserted, so MDS may be asserted together with MSEL without affecting the timing of the write operation.

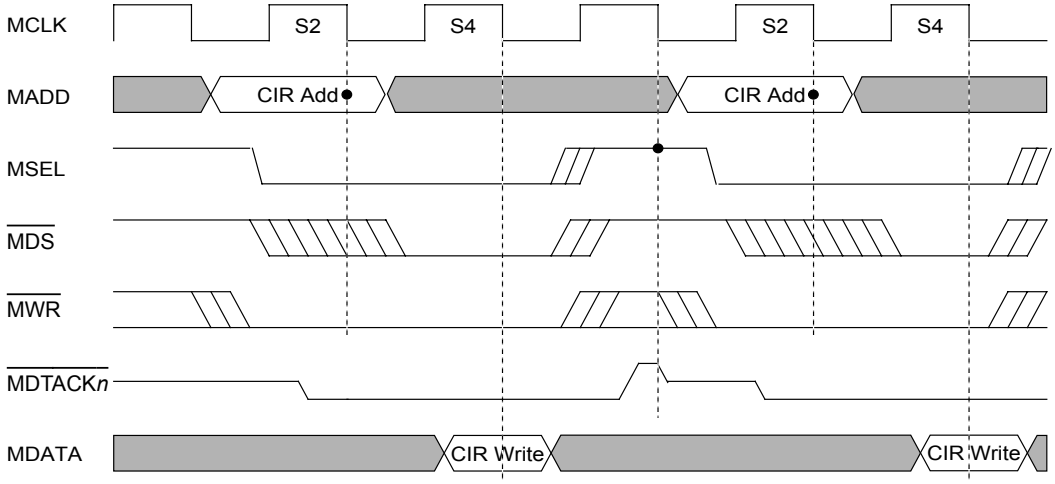


Figure 4-50. Cell Insertion Register Write Timing

In order to allow a zero-wait-state access even when using the MDTACK*n* signal, the MC92501 asserts MDTACK*n* asynchronously as a direct result of the assertion of MSEL with MADD containing a CIR address. Thus, if MSEL is asserted early enough, the assertion of MDTACK*n* can be detected at the S2 falling edge, and no wait states are added. The assertion of MDTACK*n* may be delayed until the assertion of MDS by setting the *Drive DTACK at Data Strobe (DDDS)* bit of the Microprocessor Configuration Register (MPCONR).

Figure 4-51 describes a Cell Insertion Register write operation where the assertion of MDS is delayed. The duration of this write operation is four clock cycles. The MC92501 starts the write operation by sampling MADD, MWR, and MSEL on the MCLK falling edge. The MC92501 samples MDATA on the MCLK falling edge at which MDS is asserted—in this case MDS is asserted two MCLK cycles following MSEL, and the MC92501 samples the register value one cycle later than in **Figure 4-50**. The timing of MDTACK*n* assumes that DDDS is set.

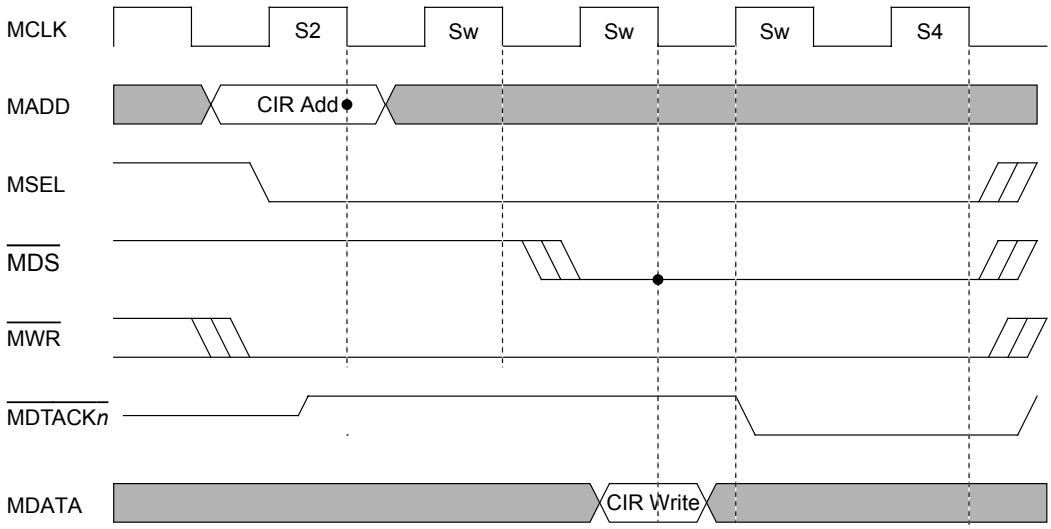


Figure 4-51. Cell Insertion Register Write Timing with Delayed Data Strobe

4.5.1.2.3 Maintenance Write Access

The processor uses a Maintenance Write Access to write to the External Memory or External Address Compression Device through the MC92501 Microprocessor Interface. The timing of a Maintenance Write Access is identical to the timing of a Cell Insertion Register Write which is described in the preceding section. If additional wait states are inserted on a Maintenance Write Access, the External Memory write cycle is extended. This type of access could be used for an External Address Compression Device that cannot meet the 20 ns access time requirement. In a Maintenance Write Access to the External Memory, the MSWH and MSWL signals are used with the same timing as MADD. The values on these signals are used to drive the EMBSH*n* and EMBSL*n* signals to provide separate select signals for the high and low words of the External Memory.

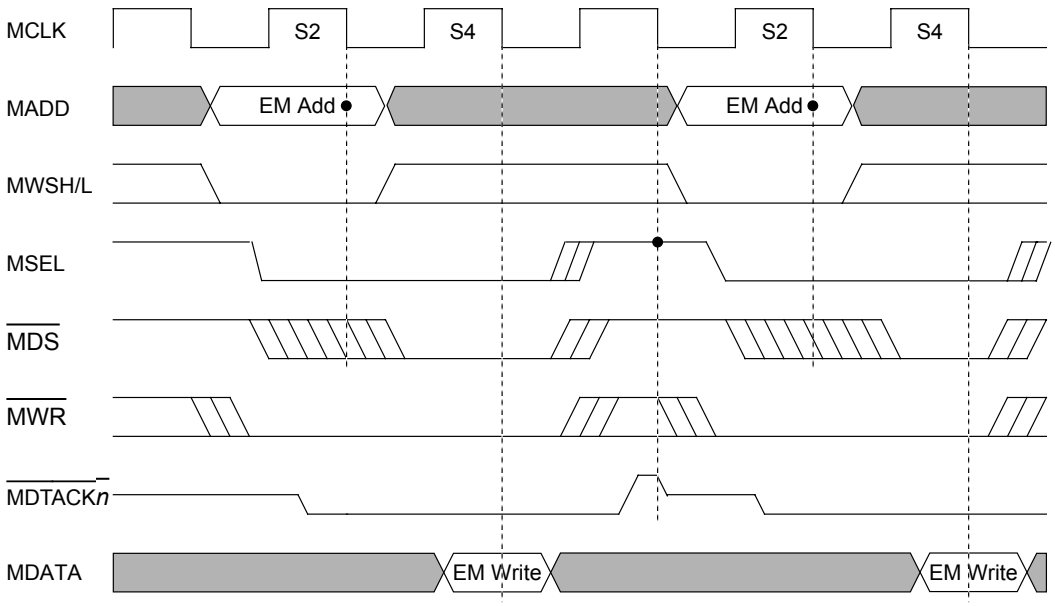


Figure 4-52. Maintenance Write Access Timing

4.5.2 DMA Device Support

The MC92501 provides support for transferring cells using a DMA device without the need for a processor interrupt. It maintains 3 output signals which can serve as request lines for an external DMA device: MREQ0, MREQ1 and MREQ2. Each of these signals can be programmed for one of the following dma requests: MCIREQ, MCOREQ and EMMREQ. See Section 7.2.6.1 Microprocessor Configuration Register (MPCONR) for details. In the following sections the MCIREQ, MCOREQ and EMMREQ are referred to as signals instead of as internal dma requests for the simplicity of the explanation only.

4.5.2.1 Cell Extraction with DMA Support

The output signal $\overline{\text{MCOREQ}}$ is provided by the MC92501 to be used as a request signal to a DMA device. MCOREQ is asserted at a falling edge of MCLK whenever the Cell Extraction Register array contains a cell to be read. MCOREQ is deasserted following the deassertion of MSEL after CER14 has been read, as shown in **Figure 4-53**.

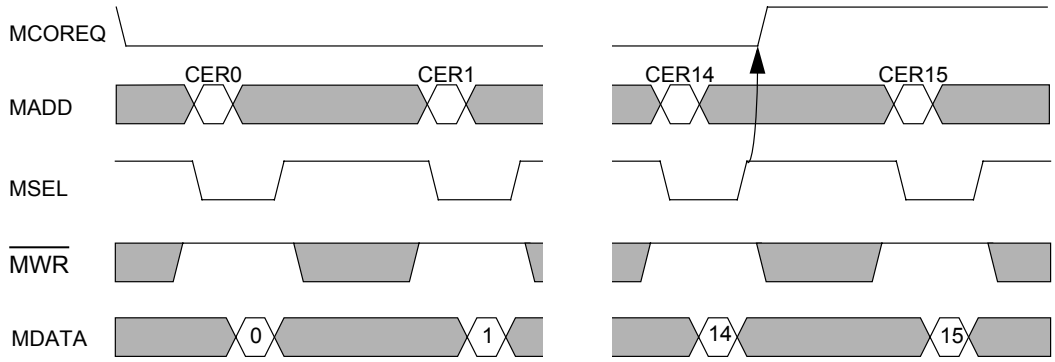


Figure 4-53. DMA Device Support on Cell Extraction Register

If CER15 is read before CER14, $\overline{\text{MCOREQ}}$ is deasserted after CER15 has been read. Note that the MC92501 supports "back to back" cell extraction; if there is another cell waiting to be read, MCOREQ is not deasserted in order to indicate that the next cell is also ready. **Figure 4-54** shows the deassertion of MCOREQ at the falling edge of MCLK following the deassertion of MSEL. A clock-by-clock timing diagram of a Cell Extraction Register read access can be found in **Figure 4-46**.

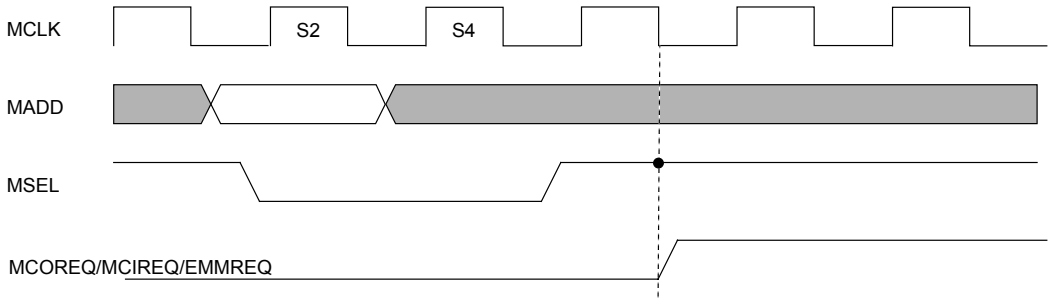


Figure 4-54. DMA Request Deassertion

4.5.2.2 Cell Insertion with DMA Support

The output signal MCIREQ is asserted at a falling edge of MCLK whenever the Cell Insertion Register array is available for writing. The deassertion of MCIREQ depends on the method used to write to the Cell Insertion Registers as explained below. If the cell is written using the cell insertion address space, MCIREQ is deasserted following the deassertion of MSEL after CIR14 has been written, as shown in **Figure 4-55**. If CIR15 is written before CIR14, MCIREQ is deasserted after CIR15 has been written.

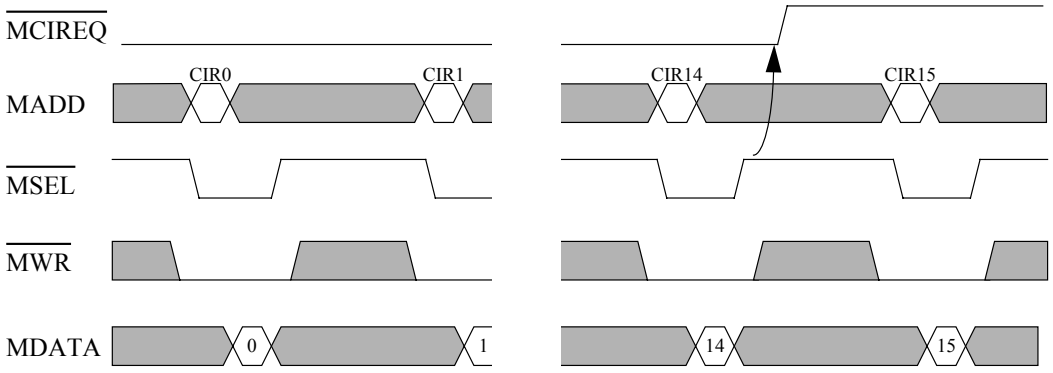


Figure 4-55. DMA Device Support on Cell Insertion Register

If the cell is written using the alternative cell insertion address space, MCIREQ is deasserted following the deassertion of MSEL after ACIR0 has been written. If ACIR1 is written before ACIR0, MCIREQ is deasserted after ACIR1 has been written. MCIREQ is always deasserted at the falling edge of MCLK following the deassertion of MSEL. This timing is identical to that of MCOREQ and is illustrated in **Figure 4-54**. Clock-by-clock timing diagrams of a Cell Insertion Register write access can be found in **Figure 4-50** and **Figure 4-51**.

4.5.2.3 Maintenance Accesses with DMA Support

The output signal EMMREQ is asserted during each Maintenance Slot. It may be used as a request signal to a DMA device that performs the maintenance accesses. The assertion of EMMREQ occurs at a falling edge of MCLK a programmable amount of time before the beginning of each Maintenance Slot. EMMREQ is deasserted at the falling edge of MCLK following the deassertion of MSEL after a programmable number of maintenance accesses have been performed. This timing is identical to that of MCOREQ/MCIREQ and is illustrated in **Figure 4-54**. If the programmed number of maintenance accesses are not performed, EMMREQ is deasserted at a falling edge of MCLK following the end of the Maintenance Slot. See **Section 3.3.3** Maintenance Slot Structure for more information.

5

MC92501 Data Path Operation

5.1 Introduction

The MC92501 provides two internal data paths:

- Ingress Data Path (data is received from the network)
- Egress Data Path (data is transmitted to the network)

The two data paths are described in detail in the following paragraphs.

5.2 Ingress Data Path Operation

The Ingress data path includes the following steps:

1. Cell assembly from physical layer
2. Address compression
3. Context Table lookup
4. Cell counting
5. UPC/NPC processing
6. Cell insertion
7. OAM processing
8. Appending switch overhead information and address translation
9. Transfer to switch

The cell flow through these steps is shown in **Figure 5-1**. Each step is described in the subsections below. During the processing, the decision can be made to remove a cell from the cell flow based on the connection parameters or the OAM processing, among other reasons. Such a cell may be copied to the Cell Extraction Queue.

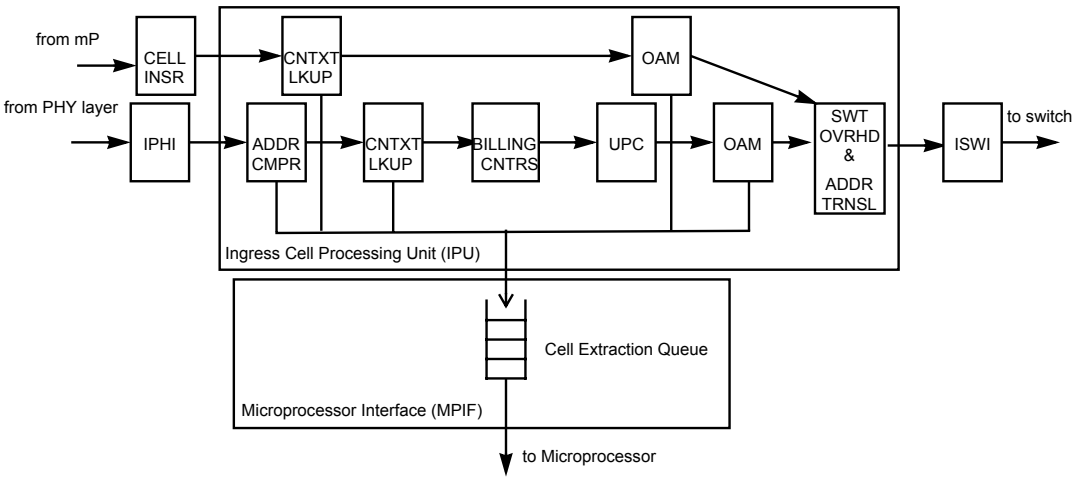


Figure 5-1. Ingress Data Path

5.2.1 Interface to Physical Layer—Cell Assembly

The Ingress Physical-Layer Interface (IPHI) block receives cell data from the physical layer using the UTOPIA standard interface. The bytes are assembled into cells since the MC92501 processing is on a cell basis. The cells are held in a FIFO and are read one per cell slot by the Cell Processing block. The input signal RXSOC is used to delineate the beginning of a cell. The 53 bytes of each cell are counted by the MC92501. If RXSOC is not asserted at the proper time, a protocol error is reported by asserting the *Ingress PHY Protocol Handshake Error (IPHE)* bit in the Interrupt Register (IR).

The input data pins are parity protected as described in Section 4.2.1 UTOPIA Level 1 Receive PHY Interface (Ingress). Parity checking by the MC92501 is optional and is enabled by the *Ingress PHY Parity Enable (IPPR)* bit of the Ingress PHY Configuration Register (IPHCR). When parity checking is enabled, the MC92501 expects RXPRTY to contain odd parity over RXDATA. If a parity error is detected on any of the four octets of the ATM header, the cell is discarded at the PHY interface. A parity error detected on the HEC byte does not cause the cell to be discarded. If a parity error is detected on an octet of the cell payload, the treatment depends on the *Ingress Payload Parity Enable (IPLP)* bit of the Ingress PHY Configuration Register (IPHCR). If IPLP is reset, the payload parity error is ignored, and the cell is processed normally. If IPLP is set, the cell is removed from the cell flow and copied to the Cell Extraction Queue. When any (header or payload) parity error is detected, the error is reported by asserting the *Ingress PHY Parity Error (IPPE)* bit of the Interrupt Register (IR). The reason for treating parity errors on the payload of the cell differently is that some applications can correct single or multiple bit errors using higher-level protocols. For these applications discarding the cell because of a parity error in the payload would be harmful. A parity error in the header, however, can lead to miss-routing of the cell, so the error is not ignored.

The HEC received from the physical layer is not checked by the MC92501, and it is discarded from the cell. Since the MC92501 processes cells at a higher rate than they are received from the physical layer, the IPHI block cannot assemble a cell during every cell processing slot. When no complete cell is available, the IPHI block informs the Ingress Cell Processing block (IPU), and a hole is inserted in the cell flow through the MC92501. The IPHI block checks the cell header and recognizes the “unassigned” and “invalid” header values defined in **Table 5-1** and **Table 5-2**. Unassigned cells are treated as holes in the MC92501 cell flow. Invalid cells may optionally (see Section 7.2.6.3 Ingress PHY Configuration Register (IPHCR)) be treated as holes or be copied to the processor for counting.

Table 5-1. Pre-assigned Header Values at the UNI

Use	GFC	VPI	VCI	PTI	CLP
Unassigned cell	XXXX	00000000	00000000 00000000	XXX	0
Invalid pattern at the ATM layer	XXXX	00000000	00000000 00000000	XXX	1

Note: X = don’t care bit

Table 5-2. Pre-assigned Header Values at the NNI

Use	VPI	VCI	PTI	CLP
Unassigned cell	0000 00000000	00000000 00000000	XXX	0
Invalid pattern at the ATM layer	0000 00000000	00000000 00000000	XXX	1

Note: X = don't care bit

5.2.2 Address Compression

Ingress address compression maps the address field(s) in the received cell header into a pointer to the Context Table entry that relates to the cell's virtual connection. The MC92501 supports two types of switching service, Virtual Path (VP) and Virtual Channel (VC). For VP switching, the address is the Virtual Path Identifier (VPI) field of the cell header. For VC switching, the address consists of both the VPI and the Virtual Channel Identifier (VCI) fields of the cell header. When the MC92501 supports multiple PHY devices, the mapping of ATM addresses to Context Table entries must be done separately for each PHY layer link. For this purpose, the number of the link from which a cell arrived can be treated as an additional address field. In this case the VP switching address consists of the Link/VPI fields; a VC switching address consists of the Link/VPI/VCI fields. If the link is at a UNI, as indicated by the bits of the UNI Register (UNIR), the four Most Significant Bits of the *VPI Mask (VPM)* field of the Ingress Link Register should be set to 0, since the actual VPI consists of only 8 bits. The MC92501 also optionally (see Section 7.2.6.10 Ingress Processing Configuration Register (IPCR)) checks that the received GFC bits are all 0; if not, the cell is copied to the Cell Extraction Queue.

NOTE: Cells that are inactive (i.e., for which no valid connection is found during address compression) are removed from the cell flow and copied to the Cell Extraction Queue.

5.2.2.1 Address Compression Options

The MC92501 supports two methods for performing address compression:

1. Table lookup using restricted address spaces
2. External address compression

The choice of method is user programmable per link by using the *Address Compression Method* (*ACM)* field in the Ingress Link Registers (ILNK0–ILNK15). **Table 5-3** summarizes the available options.

Table 5-3. Address Compression Options

ACM	VP Lookup	VC Lookup	Ext Addr Comp
00	3	3	
01	3		
10			3
11	3		3

When ACM = 00, a two-stage lookup method is used. First, the VP Table is checked. If VP switching is performed, the VP Table contains the ICI. Otherwise, the VP Table contains a pointer to the VC Table which contains the ICI.

The case in which ACM = 01 is intended for applications in which only VP switching is performed. In this case, a one-stage lookup consults only the VP Table. When the MSB of the ACM is set, the address compression is done externally. In this case, if the LSB is also set (11), the VP Table is consulted in addition to the external compression. If the VP Table contains a valid ICI, this value is used instead of the external result. More details are provided in the following sections.

5.2.2.2 Table Lookup

When some of the bits of the VPI and/or VCI are not allocated, the address range can be reduced enough to make a table lookup scheme practical. The method described here provides a great deal of flexibility in that the number and location of the allocated bits of the VPI may be specified per physical link and the number and location of the allocated bits of the VCI may be specified per VPI. Following the description of the table lookup method, a number of optional variations on the scheme for the purpose of reducing the External Memory requirements are presented. An overview of the table lookup scheme is provided in **Figure 5-2**. A detailed description follows.

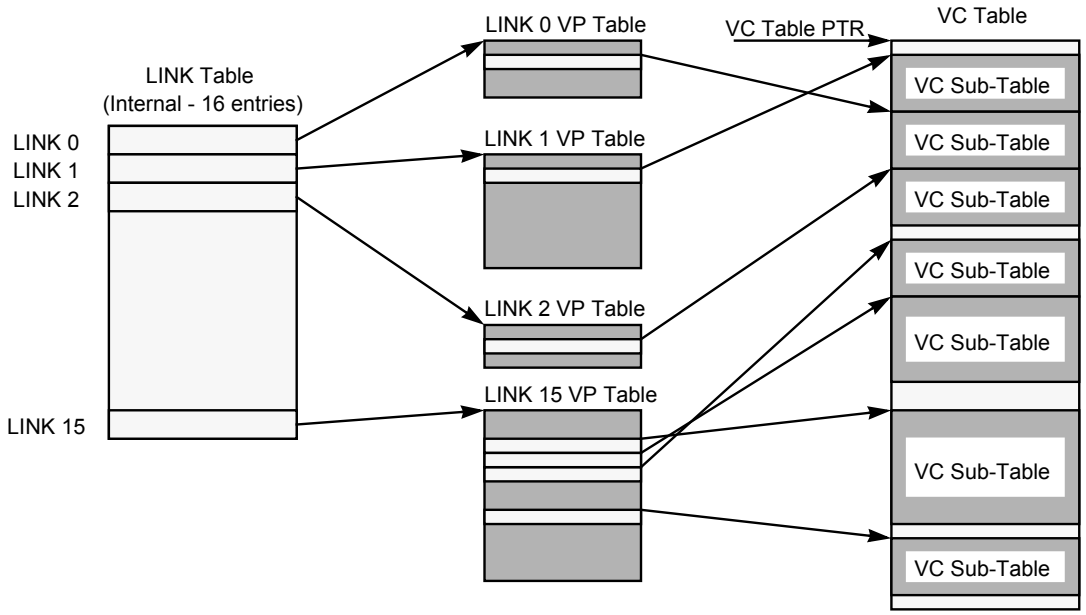


Figure 5-2. Address Compression Tables

5.2.2.2.1 Link Table

The internal Link Table, which is implemented by 16 registers (see Section 7.2.5.12 Ingress Link Registers (ILNK0–ILNK15), defines which physical links are valid and how to perform the VP Table lookup for cells arriving from each of the physical links. The logical structure of the Link Table for the table lookup scheme is shown in **Figure 5-3**. The Ingress Link Register addressed by the link number from which the current cell arrived is read as the first step of the address compression, and the *Link Enable (LE)* bit is checked. If LE is reset, the cell came from an inactive link, and it is removed from the cell flow and copied to the Cell Extraction Queue. If LE is set, address compression continues.

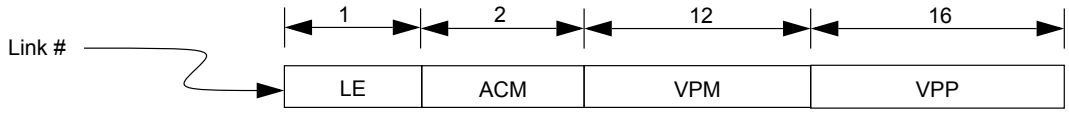


Figure 5-3. Link Table Logical Structure

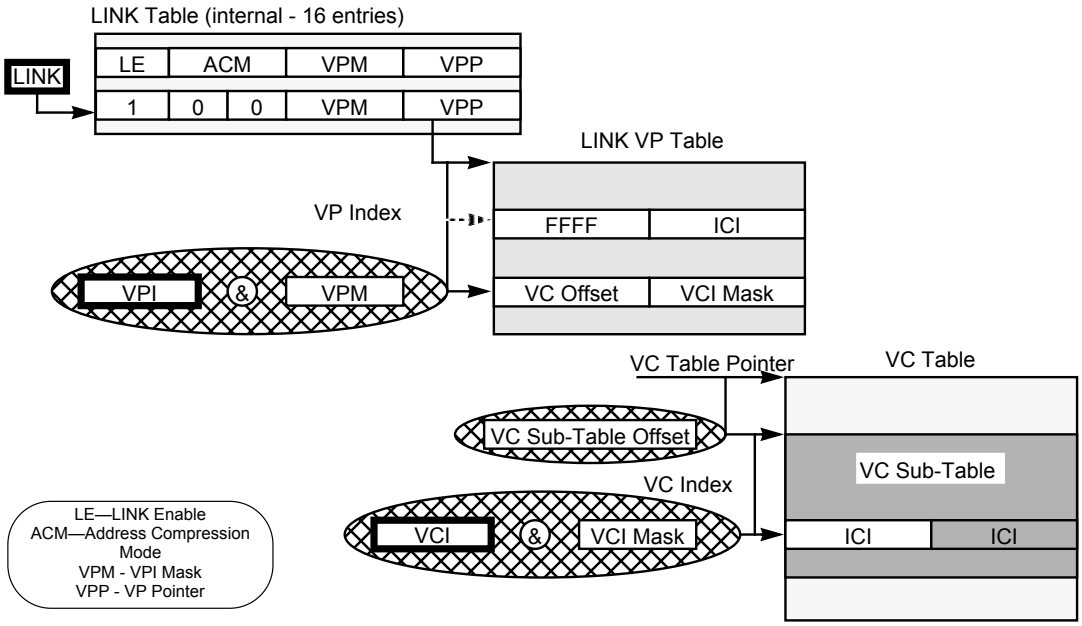


Figure 5-4. Full Table Lookup Scheme

The External Memory address of the VP Table entry is determined as illustrated in **Figure 5-4**. Each link has its own VP table whose location is derived from the *VP Pointer (VPP)* field of the Ingress Link Register by shifting it eight bits to the left. The *VPI Mask (VPM)* field of the Ingress Link Register is used to choose the allocated bits of the VPI which are then aligned to the right to form the VP Index. For instance, if the VPM is 805 H, bits 11, 2, and 0 of the VPI form the three-bit index to the record of the VP connection

within the VP sub-table as shown in **Figure 5-5**. The MC92501 also optionally (see **Section 7.2.6.10** Ingress Processing Configuration Register (IPCR)) checks that all un-allocated bits of the VPI are 0; if not, the cell is removed from the cell flow and copied to the Cell Extraction Queue. The size of each link's VP table should correspond to the number of bits that are set in the link's VPM. The actual External Memory address is computed by adding the VP Index to the link's VPP. Some examples of these calculations can be found in **Table 5-4**.

Table 5-4. VP Table Address Calculations (VC Lookup Enabled)

VPP[1]	Table Size	VPM	VPI	VP Index[2]	EM Address of VP Table Entry
2400 H	32 records	037 H	019 H[3]	9 H	2400 << 8 = 240000 9 << 2 = 000024 240024
2400 H	8 records	805 H	801 H	5 H	2400 << 8 = 240000 5 << 2 = 000014 240014
2403 H	256 records	0FF H	086 H	86 H	2403 << 8 = 240300 86 << 2 = 000218 240518

Notes:
1. The VPP is in units of 256 bytes, i.e. it is shifted 8 bits to the left to produce the actual EM address of the VP Table.
2. Since the record size is 1 long word, the VP Index is shifted 2 bits to the left.
3. If un-allocated bits are checked, the address compression would fail for this cell. It would be removed from the cell flow and copied to the Cell Extraction Queue.

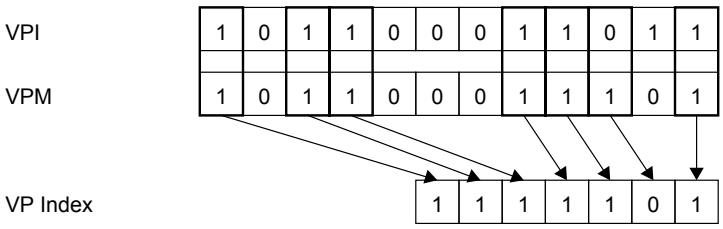


Figure 5-5. VP Index Derivation

5.2.2.2.2 VP Table

If VP switching is performed on the VP connection, the VP Table entry appears as shown in **Figure 5-6**. The reserved value of all 1s in the VC Sub-Table Offset field is used to indicate that VP switching is performed. The ICI points to the context entry for the VP connection unless it contains the reserved value of all 1s, in which case the connection is not active. If VC switching is performed, the VP Table entry defines how to perform the VC Table lookup, and it appears as shown in **Figure 5-7**.

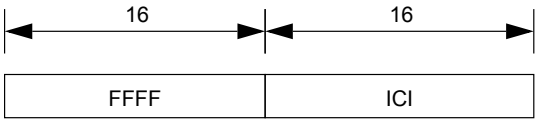


Figure 5-6. VP Table Record Logical Structure for VP Switching

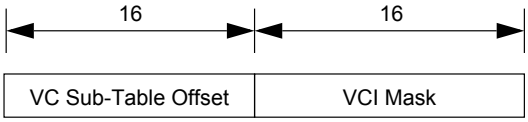


Figure 5-7. VP Table Record Logical Structure as Pointer to VC Table

The External Memory address of the VC Table record is determined as illustrated in **Figure 5-4**. The VC Table Pointer as defined in the VC Table Pointer Register (VCTP) is the base address of the VC Table. Each VPI has its own VC sub-table which is offset from the base address by the *VC Sub-Table Offset* field of the VP Table entry which is in units of long words. The *VCI Mask* field of the VP Table entry is used to choose the allocated bits of the VCI which are then aligned to the right to form the VC Index. For instance, if the VCI Mask is 1805 H, bits 12, 11, 2, and 0 of the VCI form the four-bit index to the record of the VC connection within the VC sub-table. The MC92501 also optionally (see **Section 7.2.6.10** Ingress Processing Configuration Register (IPCR)) checks that all un-allocated bits of the VCI are 0; if not, the cell is removed from the cell flow and copied to the Cell Extraction Queue. The size of each VPI's VC sub-table should correspond to the number of bits that are set in the VPI's VCI Mask as shown in **Table 4-3**. The actual External Memory address is computed by adding the VC Table Pointer, the VPI's VC Sub-Table Offset, and the VC index. Some examples of these calculations can be found in **Table 5-5**.

Table 5-5. VC Table Address Calculations

VC Table Pointer[1]	VC Sub-Table Offset[2]	Table Size	VCI Mask	VCI	VC Index[3]	EM Address of VC Table Entry
8400	100	32 records	0037	0019	9	8400 << 8 = 840000 100 << 2 = 000400 9 << 1 = 000012 840412
8403	201	8 records	0805	0801	5	8403 << 8 = 840300 201 << 2 = 000804 5 << 1 = 00000A 840B0E

Notes:
1. The VC Table Pointer is in units of 256 bytes, i.e. it is shifted 8 bits to the left to produce the actual EM address.
2. The VC Sub-Table Offset is in units of long words, i.e. it is shifted 2 bits to the left.
3. Since the record size is 2 bytes, the VP Index is shifted 1 bit to the left.
4. If un-allocated bits are checked, the address compression would fail for this cell. It would be removed from the cell flow and copied to the Cell Extraction Queue.

5.2.2.2.3 VC Table

The VC Table entry appears as shown in **Figure 5-8**. The Ingress Connection Identifier (ICI) points to a valid VCC entry in the Context Table unless it contains the reserved value of all 1's, in which case the connection is not active. Cells belonging to inactive connections are removed from the cell flow and copied to the Cell Extraction Queue.

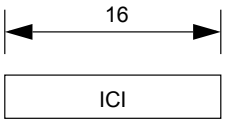


Figure 5-8. VC Table Record Logical Structure

5.2.2.2.4 VC Lookup Disable

If the ACM field in the Ingress Link Register is 01, the VC Table lookup is skipped, and no VC sub-tables exist for this link. In this case whenever the ICI field in the VP Table is all 1's, no Context Table entry exists for the cell, which is removed from the cell flow and copied to the Cell Extraction Queue as an inactive cell. Additionally, the VP sub-table for this link is condensed by 50% since all the entries are of the form of **Figure 5-6** and two entries are placed in each 32-bit word as shown in **Figure 5-9**. This option should be used only if all the connections on this link require VP switching.

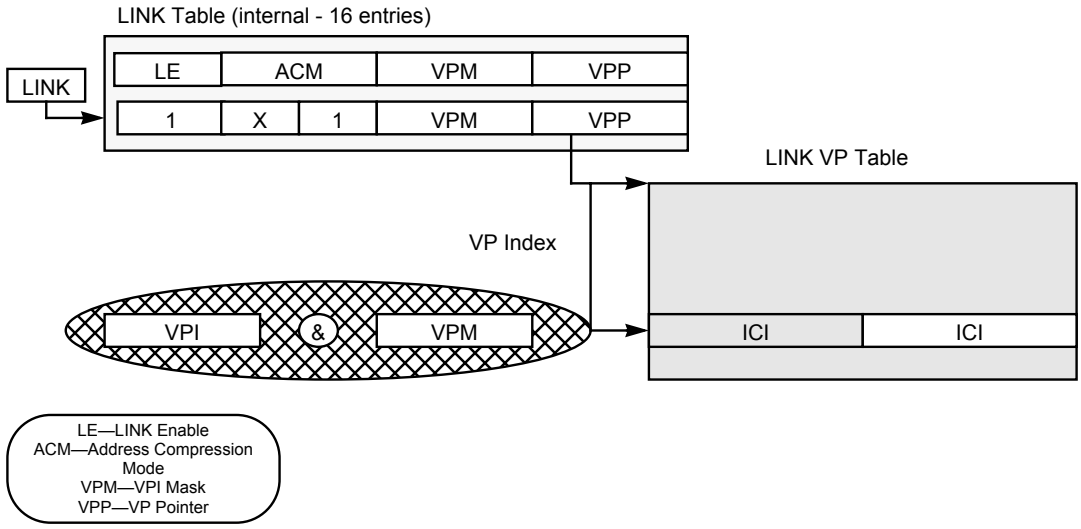


Figure 5-9. Address Compression with VC Lookup Disable

Table 5-6 shows an example of the calculation of the VP Table Address when there is no VC Table Lookup. In this case the VP Index is shifted one bit to the right before adding it to the VPP since each long word contains two records.

Table 5-6. VP Table Address Calculations (VC Lookup Enabled)

VPP[1]	Table Size	VPM	VPI	VP Index[2]	EM Address of VP Table Entry
2400 H	32 records	037 H	019 H	9 H	2400 << 8 = 240000 9 << 1 = 000012 240012

Notes: 1. The VPP is in units of 256 bytes, i.e. it is shifted 8 bits to the left to produce the actual EM address of the VP Table.
2. Since the record size is 2 bytes, the VP Index is shifted 1 bits to the left.

5.2.2.3 External Address Compression

The external address compression method allows the user total flexibility in performing the Ingress address compression. This method is chosen by setting the MSB of the ACM field in the Ingress Link Register. In this case the LSB of the ACM field indicates whether VP Lookup is to be performed as a first stage before providing the address for external compression. In either case the Ingress Link Register addressed by the link number from which the current cell arrived is read as the first step of the address compression, and the *Link Enable (LE)* bit is checked. If LE is reset, the cell came from an inactive link, and it is removed from the cell flow and copied to the Cell Extraction Queue. If LE is set, address compression continues.

5.2.2.3.1 External Address Compression with VP Lookup

When using the external address compression method with VP Lookup, VP compression is performed identically to the Table Lookup method with VC Lookup Disable (see **Figure 5-9**). In addition, the Link/VPI/VCI combination is written to the External Address Compression Device. If the VP Lookup was successful (ICI not all 1s), the ICI read from the VP Table is used.

If the VP Lookup failed, the results of the external compression are used. A specified number of ACLK periods, N_{EAC} (≥ 40), after the EAC write access, the MC92501 performs a read operation from the EAC device and receives the user's response as illustrated in **Figure 5-10**. The logical structure of the response consists of a Valid bit and an ICI as shown in **Figure 5-11**. (The precise data structures of the external address compression accesses are defined in Section 5.2.2.3 External Address Compression. Details of the External Address Compression Device interface are provided in Section 5.2.2.3 External Address Compression.)

If the *Valid (V)* bit is set, the Ingress Connection Identifier (ICI) points to a valid entry in the Context Parameters Table unless it contains the reserved value of all 1s. If the Valid bit is reset, or if the ICI is all 1s, there is no Context Parameters Table entry for the cell, and it is removed from the cell flow and copied to the Cell Extraction Queue as an inactive cell.

Figure 5-10. External Address Compression Events

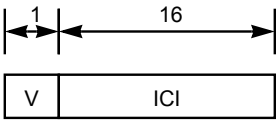


Figure 5-11. External Address Compression Response Logical Structure

5.2.2.3.2 VP Lookup Disable

If the ACM field in the Ingress Link Register is 10, the VP Table lookup is skipped entirely, and no VP table exists for this link.

5.2.3 Ingress Context Table Lookup

Once the Ingress Connection Identifier (ICI) of the cell is known, the context parameters can be read from the Context Parameters Table. The structure of the Context Parameters Table is presented in Section 7.3.3 Context Parameters Table.

5.2.4 Cell Counting

If the processed cell was received from the physical layer (not inserted internally), one of the connection cell counters from the Ingress Billing Counters Table is incremented, unless the table does not exist—see Section 7.2.6.10 Ingress Processing Configuration Register (IPCR). One of the link cell counters from the Ingress Link Counters Table is also incremented if the table exists—see Section 7.2.6.14 General Configuration Register (GCR). The appropriate counter is chosen based on the CLP bit and whether the cell is an OAM cell.

5.2.5 UPC/NPC

The MC92501 performs the UPC/NPC function for the ingress flow if the *UPC Flow (UPCF)* bit in the ATMC CFB Configuration Register (ACR) is reset. See **Section 6.2 UPC/NPC Support**.

5.2.6 Ingress Cell Insertion

The MC92501 makes use of the holes in the cell flow provided by the IPHI block (whether due to the difference between the cell processing and arrival rates or the reception of unassigned or invalid cells) to insert cells into the Ingress cell flow. The cell insertion rate is paced by a single Leaky Bucket to ensure that the switch is not flooded with inserted cells beyond its capacity.

The parameters of the Leaky Bucket are determined by the Ingress Insertion Leaky Bucket Register (IILB). The bucket contents value is incremented by the *Ingress Average Insertion Period (IAIP)* when a cell is inserted in the Ingress cell flow and is decremented by one in each cell slot. The MC92501 inserts a cell in an available hole only if the bucket contents value contained in the Ingress Insertion Bucket Fill Register (IIBF) is smaller than the *Ingress Insertion Bucket Limit (IIBL)*.

NOTE: The IIBL field is zero after reset and must be written with a non-zero value to enable cell insertion.

The IAIP consists of a 12-bit integer part and a 4-bit fractional part which provide for values as large as 4095 cell processing times (equivalent to inserted cells being 0.024% of the cell flow) with a precision of 1/16 of a cell processing time. At the typical value of 100 cell processing times (1% of the cell flow), the precision is 0.0006 percentage point. Note that programming the IAIP with a value of zero provides unlimited cell insertion. The 16-bit IIBL provides for a burst of 16 cells when using the maximum value of IAIP and proportionately larger bursts when using smaller values of IAIP.

For example, if the inserted cells may be 1.3% of the cell flow and the switch can handle a burst of up to 10 cells, the IAIP is 1/.013 = 76.92 which may be rounded to 76 15/16, or $04C.F. The IIBL is defined to be one IAIP value less than the required bucket size. Therefore, in our example the IIBL should be 10 – 1 = 9 times the IAIP value: 76.94 × 9 = 692, or $02B4.

The types of cells that can be inserted in the Ingress cell flow are:

- OAM cells generated internally by the MC92501 including:
  - AIS cells
  - RDI cells
  - Continuity Check cells
  - PM Forward Monitoring cells
- OAM cells generated by the microprocessor
- Other cells generated by the microprocessor

The various types of cells that can be inserted in the Ingress cell flow are classified by their insertion priority and held in separate queues. The insertion priorities are (from highest to lowest):

1. PM Forward Monitoring cells generated internally by the MC92501
2. Cells from the microprocessor
3. AIS, RDI, and CC cells generated internally by the MC92501

The data structure of inserted cells from the microprocessor is provided in **Section 7.4.1** Inserted Cell. Note that inserted cells have their connection identifier explicitly available, so they do not undergo address compression. Inserted cells are not presented to the UPC/NPC mechanism, nor are they counted in the connection counters.

5.2.7 Ingress Copy/Remove/OAM Processing

If the *Ingress Copy All (ICA) cells* bit is set, the cell is added to the Cell Extraction Queue to be transferred to the microprocessor. If the *Ingress Remove All (IRA) cells* bit is set, the cell is removed from the cell flow after undergoing UPC/NPC and OAM processing. The option exists to copy and/or remove those cells whose VCI is identified as “reserved”. See Section 7.2.6.28 Ingress VCI Copy Register (IVCR) and Section 7.2.6.30 Ingress VCI Remove Register (IVRR) for details. This option is enabled on a connection basis by the *Ingress VCR/VRR Registers Enable (IVRE)* bit of the Ingress Parameters word. There is also an option to copy those cells whose PTI value is 110 or 111 to the Cell Extraction Queue. This option is controlled by the *Ingress PTI 6 Copy (IP6C)* and *Ingress PTI 7 Copy (IF7C)* bits of the Ingress Parameters word.

The CRC-10 field of received OAM cells is checked. If an error is detected in this field, or if the cell is an Illegal OAM cell (see Section 6.3.4.1 Illegal OAM Cells), no OAM processing is performed on the cell. The cell is removed from the cell flow and copied to the Cell Extraction Queue. All OAM cells are further classified by the OAM Cell Type and OAM Function Type fields (see **Figure 6-5** and Table 6-3). The OAM cells that receive special processing are:

- Cells received from the physical layer:
  - AIS (see **Section 6.3.5.1.1 VP/VC AIS)**
  - RDI (see **Section 6.3.5.1.2 VP/VC RDI**)
  - Loopback (see **Section 6.3.5.3 VPC/VCC Loopback Cells)**
  - Continuity Check (see **Section 6.3.5.1.3 Continuity Check)**
  - Forward Monitoring (see **Section 6.3.7 Performance Monitoring)**
  - Backward Reporting (see **Section 6.3.7 Performance Monitoring)**
  - If this is a segment/connection termination point (see **Section 6.3.4 General OAM**) of the OAM flow, the segment/end-to-end OAM cell is removed from the cell flow.
- Inserted cells:
  - Forward Monitoring from the processor (see **Section 6.3.7 Performance Monitoring)**
  - Forward Monitoring internally generated (see **Section 6.3.7 Performance Monitoring)**

When a user data cell is processed, the traffic bits (see **Section 6.3.5.1.3 Continuity Check**) are set. User data cells that belong to an active PM block test are processed as described in **Section 6.3.7 Performance Monitoring**.

5.2.8 Switch Overhead Information

The MC92501 optionally performs address translation on the Ingress cell flow. The new address fields are taken from the Ingress Translation Address word of the Context Parameter Table in the External Memory. The *Ingress Address Translation VPI Enable (IAPE)* and *Ingress Address Translation VCI Enable (IACE)* fields of the Ingress Processing Configuration Register (IPCR) determine exactly which fields of the ATM cell header are overwritten. See Section 7.2.6.10 Ingress Processing Configuration Register (IPCR) for more details.

The source of the switch overhead information provided by the MC92501 is the Context Parameters Table entry for the connection. This information is appended to the cell as shown in the following tables. The overhead bytes are transferred before the cell, most-significant byte first (left-to-right in the tables). The number of long words of switch parameters (0, 1, or 2) is indicated by the *Ingress Switch Parameter Control (SPC)* field defined in Section 7.2.6.13 ATMC CFB Configuration Register (ACR). If any of the switch parameters words is not provided, the values of the corresponding overhead bytes added to the cell are undefined. The data structure transferred to the switch is controlled by two fields of the Ingress Switch Interface Configuration Register (ISWCR), *Ingress Switch Number Of Bytes (ISNB)* and *Ingress Switch HEC Field (ISHF)*.

Figure 5-12 shows the octets transferred to the switch as a function of ISNB when no HEC octet is inserted (ISHF=00).

ISNB	Transmitted Octets (shaded)																
	Switch Parameters 2				Switch Parameters 1				Switch Parameters 0				ATM Header				Payload
0000																	
1111																	
1110																	
1101																	
1100																	
1011																	
1010																	
1001																	
1000																	
0111																	
0110																	
0101																	
0100																	

Figure 5-12. Data Structure with No HEC Octet (ISHF=00)

Figure 5-13 shows the octets transferred to the switch as a function of ISNB when the HEC octet is inserted and presented as 00000000 (ISHF=10). In this case the only purpose of the HEC octet is to provide for compatibility with switches that expect the HEC octet in their cell structure.

NOTE: The MSB of Switch Parameters 2 is unused even if 64 bytes are transferred to the switch.

ISNB	Transmitted Octets (shaded)																	
	Switch Parameters 2				Switch Parameters 1				Switch Parameters 0				ATM Header					Payload
0000																	0	
1111																	0	
1110																	0	
1101																	0	
1100																	0	
1011																	0	
1010																	0	
1001																	0	
1000																	0	
0111																	0	
0110																	0	
0101																	0	

Figure 5-13. Data Structure with HEC = 0 (ISHF = 10)

Figure 5-14 shows the octets transferred to the switch as a function of ISNB when the HEC octet is taken from the switch parameters (ISHF = 11) and all three switch parameter long words exist in the External Memory (SPC = 11). In this case the MSB of Switch Parameters 2 is placed in the HEC octet.

ISNB	Transmitted Octets (shaded)																	
	Switch Parameters 2				Switch Parameters 1				Switch Parameters 0				ATM Header					Payload
0000	X																X	
1111	X																X	
1110	X																X	
1101	X																X	
1100	X																X	
1011	X																X	
1010	X																X	
1001	X																X	
1000	X																X	
0111	X																X	
0110	X																X	
0101	X																X	

Figure 5-14. Data Structure with HEC Octet from Switch Parameters 2

Figure 5-15 shows the octets transferred to the switch as a function of ISNB when the HEC octet is taken from the switch parameters (ISHF = 11) and two switch parameter long words exist in the External Memory (SPC = 10). In this case the MSB of Switch Parameters 1 is placed in the HEC octet. Note that this byte is also transmitted in the overhead of the cell if the switch data structure contains at least 61 bytes.

ISNB	Transmitted Octets (shaded)																	
					Switch Parameters 1				Switch Parameters 0				ATM Header					Payload
0000					X												X	
1111					X												X	
1110					X												X	
1101					X												X	
1100					X												X	
1011					X												X	
1010					X												X	
1001					X												X	
1000					X												X	
0111					X												X	
0110					X												X	
0101					X												X	

Figure 5-15. Data Structure with HEC Octet from Switch Parameters 1

Figure 5-16 shows the octets transferred to the switch as a function of ISNB when the HEC octet is taken from the switch parameters (ISHF = 11) and only one switch parameter long word exists in the External Memory (SPC = 01). In this case the MSB of Switch Parameters 0 is placed in the HEC octet. Note that this byte is also transmitted in the overhead of the cell if the switch data structure contains at least 57 bytes.

ISNB	Transmitted Octets (shaded)																	
									Switch Parameters 0				ATM Header					Payload
0000									X								X	
1111									X								X	
1110									X								X	
1101									X								X	
1100									X								X	
1011									X								X	
1010									X								X	
1001									X								X	
1000									X								X	
0111									X								X	
0110									X								X	
0101									X								X	

Figure 5-16. Data Structure with HEC Octet from Switch Parameters 0

5.2.9 Transfer to Switch

The Ingress Switch Interface (ISWI) block receives cells from the Cell Processing block, queues them, and transfers the data structure to the switch. The switch interface signals are identical to the UTOPIA Level 1 Receive Interface with the MC92501 playing the role of the PHY layer and the switch playing the role of the ATM layer. The switch interface signals are clocked by an independent clock signal, SRXCLK. The switch is required to accept cells from the MC92501 when they are presented on the interface with a delay of up to one cell slot for synchronization. Note that the cells may be presented at a higher rate than they are received from the PHY layer due to cell insertion. The switch must be capable of receiving the cells at a sustained rate of one cell per cell slot. Otherwise, the cells may back up in the MC92501, processing is halted, and cells are not accepted from the PHY layer. Although the maximum sustained rate is one cell per cell slot, the rate can be limited by the insertion pacing mechanism described in **Section 5.2.6** Ingress Cell Insertion.

5.3 Egress Data Path Operation

The Egress data path includes the following steps:

1. Transfer from switch
2. Multicast identifier translation (if necessary)
3. Cell insertion
4. Context Table lookup
5. UPC
6. OAM processing
7. Address translation
8. Cell Counting
9. Transmission to the physical layer

The cell flow through these steps is shown in **Figure 5-17**. Each step is described in the subsections below. During the processing, the decision can be made to remove a cell from the cell flow for any of several reasons. Such a cell may be copied to the Cell Extraction Queue.

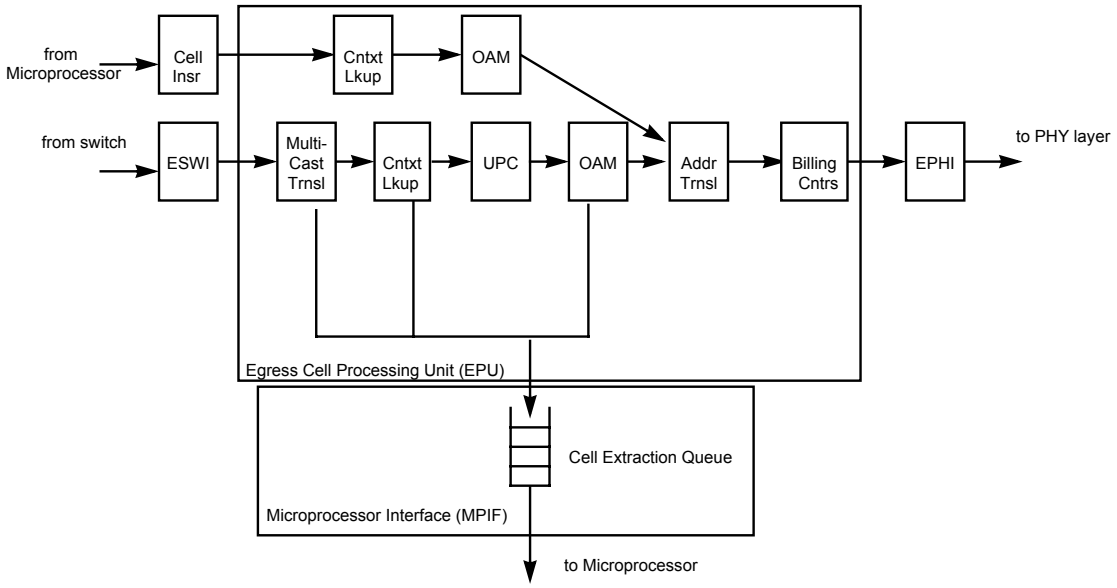


Figure 5-17. Egress Data Path

5.3.1 Transfer from Switch

The Egress Switch Interface (ESWI) block contains a cell FIFO. Data is received from the switch at the rate of 1 byte per clock cycle. The data structure received from the switch includes overhead routing information in addition to the ATM cell. When a full cell has been transferred, it is transformed into an internal data structure and presented to the Egress Cell Processing block. The switch interface signals are identical to the UTOPIA Level 1 Transmit Interface with the MC92501 playing the role of the PHY layer and the switch playing the role of the ATM layer. The switch interface signals are clocked by an independent clock signal, STXCLK. The input signal STXSOC is used to delineate the beginning of a cell.

The input data pins are parity protected. If a parity error is detected on the input pins, the error is reported by asserting the *Egress Switch Parity Error (ESPE)* bit of the Interrupt Register. If the parity error occurs on a byte containing any of the overhead fields used by the MC92501 or on a byte of the cell header, the cell is discarded. If the parity error occurs on a payload byte, the cell is optionally discarded. If a protocol error is detected on the input pins, the current cell is discarded and the error is reported by asserting the *Egress Switch Protocol Handshake Error (ESHE)* bit of the Interrupt Register.

The ESWI block contains a small cell FIFO to assemble the bytes received from the switch and synchronize the cells to the cell processing time of the Egress Cell Processing block. Th FIFO size is programmed to be either 4 or 6 cells using the *Egress Switch FIFO Control (ESFC)* bit (see Section 7.2.6.6 Egress Switch Interface Configuration Register (ESWCR) for more information). The FIFO is read by the Cell Processing block at a rate limited by the PHY layer and by cell insertion. When the ESWI FIFO is full, the MC92501 refuses to accept a cell from the switch by deasserting STXCLAV.

The number of bytes in the cell data structure received from the switch is programmable as described in **Section 7.2.6.6** Egress Switch Interface Configuration Register (ESWCR). The *Egress Switch Number Of Bytes (ESNB)* field defines the number of bytes received from the switch. Note that the HEC octet that is provided when the *Egress Switch HEC Field (ESHF)* bit is set is considered overhead information and is not stored as part of the cell header.

The bytes are provided by the switch in the following order:

1. Overhead bytes (number determined by ESNB)
2. ATM Cell Header (4 bytes; PTI, CLP valid; VCI valid if VP switching)
3. HEC octet (provided only if ESHF is set)—This octet may be used for overhead information since no HEC value is stored in the internal data structure
4. ATM cell payload (48 bytes)

The fields contained in the overhead bytes are:

- Egress Connection Identifier (ECI) / Multicast Identifier (MI) (see Section 5.3.2 Multicast Identifier Translation)
- Multicast bit (M) (see Section 5.3.2 Multicast Identifier Translation)
- Explicit Forward Congestion Indication (EFCI) (see Section 5.3.7 Address Translation)
- Multicast Translation Table Section (MTTS) (see Section 5.3.2 Multicast Identifier Translation)

The location of these fields in the overhead, header and HEC bytes is programmed using fields described in **Section 7.2.6.6** Egress Switch Interface Configuration Register (ESWCR) and **Section 7.2.6.7 Egress Switch Overhead Information Register (ESOIR0)**. This mechanism is illustrated in the following figures.

Figure 5-18 shows the extraction of the ECI field from the switch cell data structure. Each of the two bytes of the ECI may be taken from any of the non-payload bytes of the structure using the ***Identifier Most-Significant Byte (IMSB)*** and ***Identifier Least-Significant Byte (ISLB)*** fields of the Egress Switch Interface Configuration Register (ESWCR).

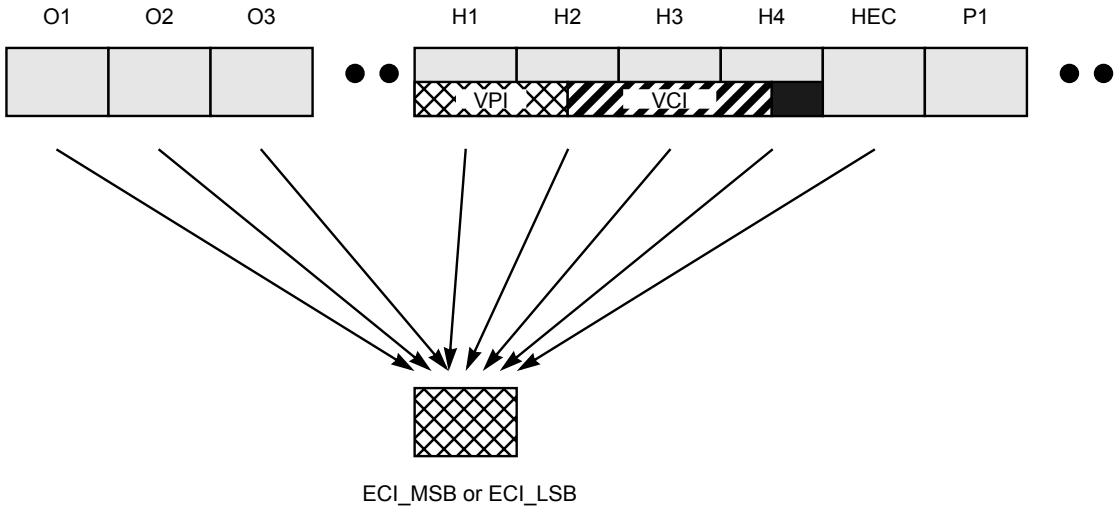


Figure 5-18. ECI Extraction from Switch Cell Data Structure

Figure 5-19 shows the extraction of the EFCI or M bits from the switch cell data structure. Each of these bits may be taken from any bit of any non-payload byte of the structure. The location is specified in two stages. The byte is specified using the ***EFCI Byte Location (EFBY)*** or ***M Byte Location (MBY)*** fields, and the bit location within the byte is specified using the ***EFCI Bit Location (EFBI)*** or ***M Bit Location*** *(MBI)* fields of the Egress Switch Overhead Information Register (ESOIR0).

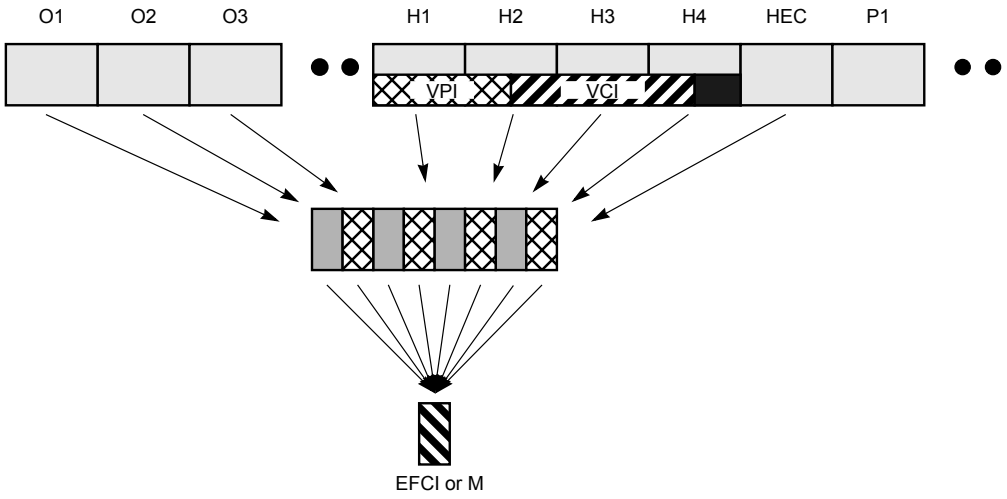


Figure 5-19. EFCI and M Extraction from Switch Cell Data Structure

Figure 5-20 shows the extraction of the MTTS field from the switch cell data structure. This field may be taken from any of the five possible bit alignments within any non-payload byte of the structure. The location is specified in two stages. The byte is specified using the ***MTTS Byte Location (MTBY)*** field, and the bit alignment within the byte is specified using the ***MTTS Bit location (MTBI)*** field of the Egress Switch Overhead Information Register (ESOIR0).

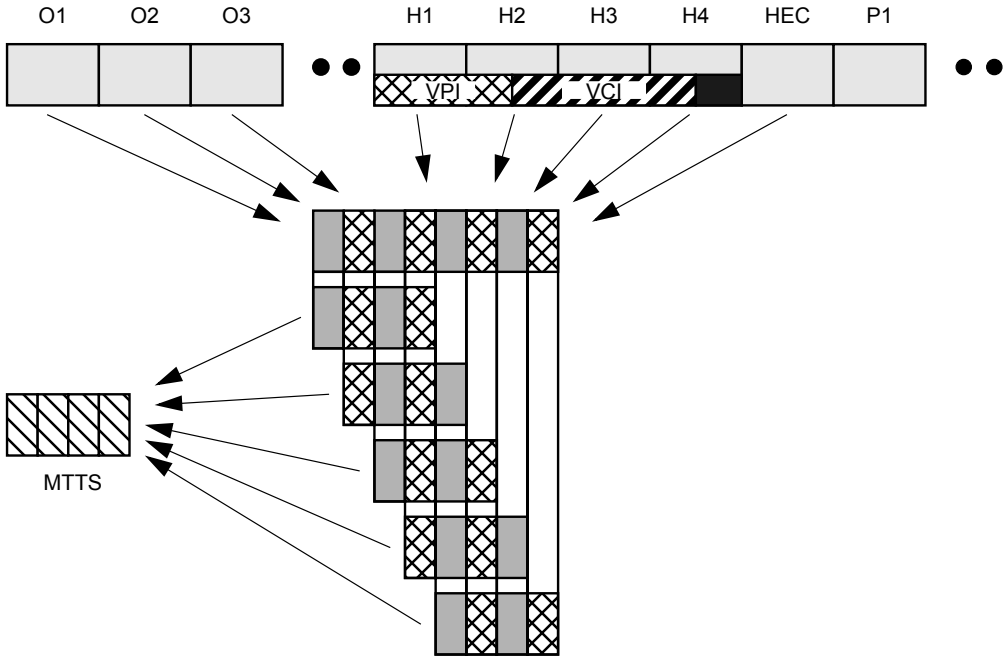


Figure 5-20. MTTS Extraction from Switch Cell Data Structure

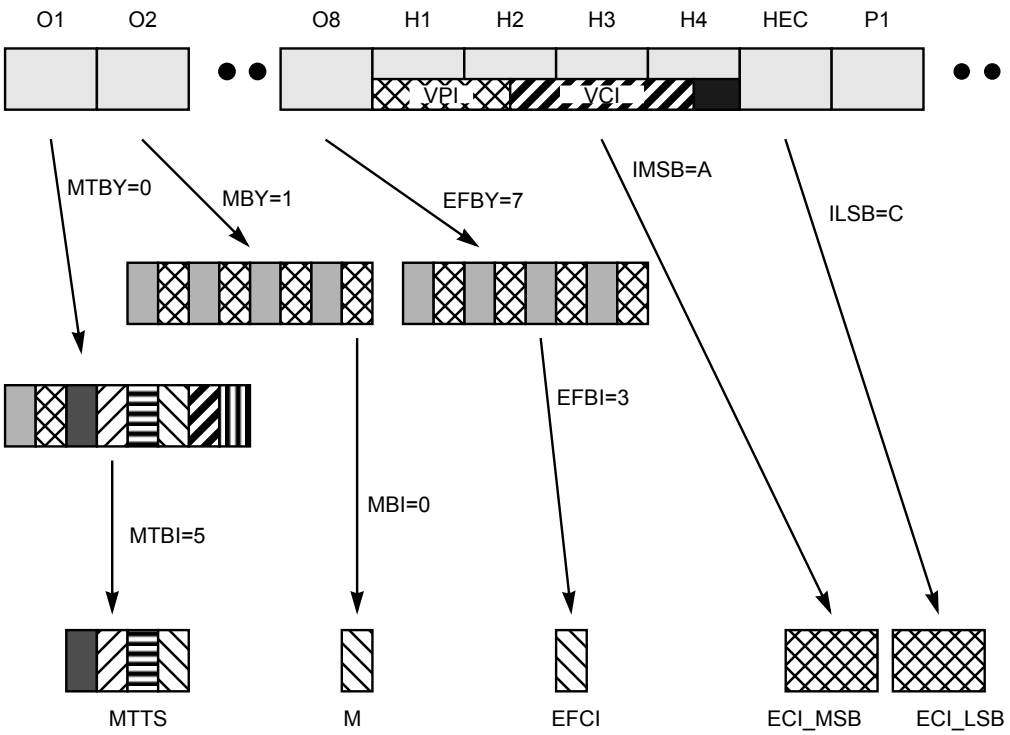


Figure 5-21. Overhead Extraction Example

Figure 5-21 shows an example configuration and the resulting overhead extraction. The ECI can be extracted from the header by setting the *Identifier in Header Address Fields (IHAF)* bit. On this case The header VPI field size can be programmed to either 12 bits or 8 bits using the *VPI Size in ECI on Header (VPS) mode* bit of the Egress Switch Interface Configuration Register (ESWCR). Once the valid fields have been retrieved, the remaining overhead bytes received from the switch are discarded since they are of no use to the MC92501 and are not transferred to the PHY layer. The Egress switch interface block provides overhead routing information to the Cell Processing block, as follows:

- The size of the ECI/MI field used by the Egress cell processing block can be programmed by writing to the *Egress Cell Processing Block ECI size (ECES)* field in the Egress Overhead Manipulation Register (EGOMR).
- The size of the MTTS field used by the Egress cell processing block can be programmed by writing to the *Egress Cell Processing Block MTTS Size (ECTS)* field in the Egress Overhead Manipulation Register (EGOMR).
- The M bit used by the Egress cell processing block can be either the M bit which was extracted from the cell's overhead, or the logical not of the M bit which was extracted from the cell's overhead, or 1 or 0 by programming the *Egress Cell Processing Block M bit Source (ECMS)* field in the Egress Overhead Manipulation Register (EGOMR).

NOTE: The M, MTTS, and MI fields are used for Multicast Identifier Translation, described in the next section. If Multicast Identifier Translation is not performed, the ECI field contains the Egress Connection Identifier of the connection to which the cell belongs. An ECI value of all 1s is invalid and causes the cell to be removed from the cell flow and copied to the Cell Extraction Queue.

5.3.2 Multicast Identifier Translation

Multicasting involves copying a cell that arrived at the switch and transmitting it on multiple physical links. In the general case the ECI of the connection to which the cell belongs are different on each link. If the switch can provide the correct ECI to each ATMC device, the multicast operation is transparent to the MC92501. However, if the switch can not provide separate ECIs for each link, a common multicast identifier may be provided to all of the ATMC devices. Each MC92501 translates the multicast identifier into the ECI for its physical link. Multicast Identifier Translation is enabled globally by the *Multicast Translation Table Control (MLTC)* field of the Egress Multicast Configuration Register (EMCR). By setting the Multicast (M) bit in the overhead information provided with each cell, the switch informs the MC92501 of the necessity of performing multicast identifier translation on the cell. If this bit is set, the overhead information contains a Multicast Translation Table Section (MTTS) and a Multicast Identifier (MI). The MTTS field is effectively concatenated to the left of the less significant portion of the MI to obtain the index to the Multicast Translation Table, as shown in **Figure 5-22**.

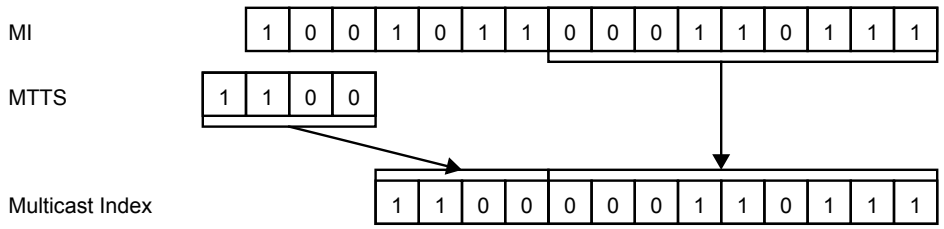


Figure 5-22. Sectionized Multicast Derivation Example

The ECI is found by reading from the Multicast Translation Table using this index as shown in **Figure 5-23**. This real ECI is used for all further processing. The number of bits from the MI to be used is programmable (see Section 7.2.6.12 Egress Multicast Configuration Register (EMCR)) as shown in **Table 5-7**. For example, if 10 bits of the MI are used when the MI is $1234 and the MTTS is $F, the resulting index is $3E34. A possible application of the MTTS field is an MC92501 supporting multiple physical links. If the link number is provided in the MTTS field, a separate lookup is performed for each physical link. If MLTC is reset, the M bit is ignored, and the ECI is taken from the switch overhead information. If the ECI is all 1s, the cell is removed from the cell flow and copied to the Cell Extraction Queue.

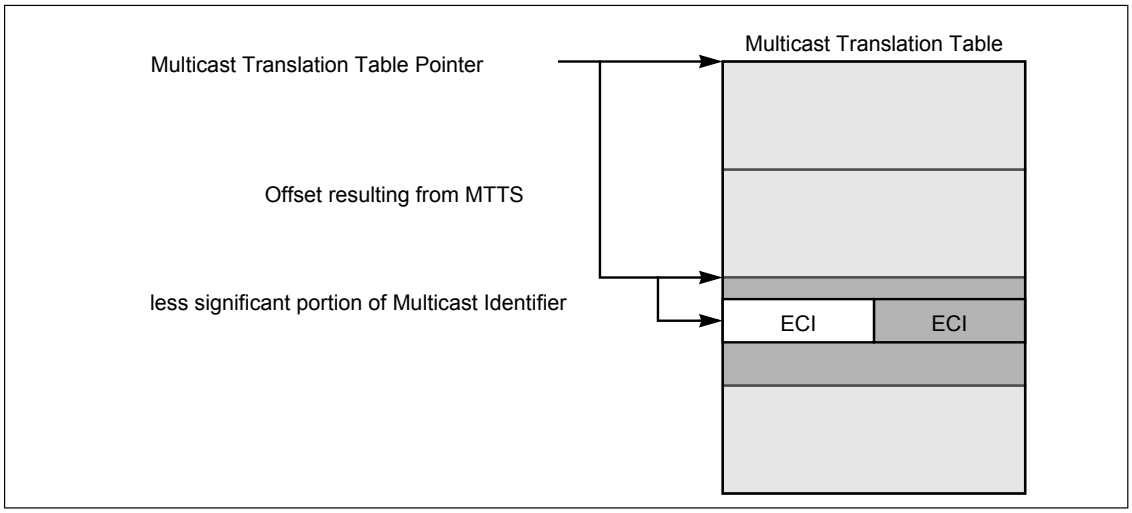


Figure 5-23. Multicast Translation

Table 5-7. Number of MI Bits Used for Sectioned Multicast Translation Table

EMIC	MI bits used
000	9
001	10
010	11
011	12
100	13
101	14
110	15
111	16

5.3.3 Egress Cell Insertion

In order to insert cells into the Egress cell flow, the MC92501 creates holes in the cell flow received from the switch interface block by not taking a cell from the FIFO. Inserting many cells in a short period of time may overload the switch's queueing capability. Therefore, the cell insertion rate is regulated by a Leaky Bucket.

The parameters of the Leaky Bucket are determined by the Egress Insertion Leaky Bucket Register (EILB) described in **Section 7.2.5.9 Egress Insertion Leaky Bucket Register (EILB)**. The bucket contents value is incremented by the *Egress Average Insertion Period (EAIP)* when a cell is inserted in the Egress cell flow and is decremented by one in each cell slot. The MC92501 inserts a cell in the cell flow only if the bucket contents value contained in the Egress Insertion Bucket Fill Register (EIBF) is smaller than the *Egress Insertion Bucket Limit (EIBL)*.

NOTE: The EIBL field is zero after reset and must be written with a non-zero value to enable cell insertion.

The Leaky Bucket parameters are defined as in the Ingress Processing Unit (see Section 5.2.6 Ingress Cell Insertion). The types of cells that can be inserted in the Egress cell flow are:

- OAM cells generated internally by the MC92501 including:
  - AIS cells
  - RDI cells
  - Continuity Check cells
  - PM Forward Monitoring cells
- OAM cells generated by the microprocessor
- Other cells generated by the microprocessor

The various types of cells that can be inserted in the Egress cell flow are classified by their insertion priority and held in separate queues. The insertion priorities are (from highest to lowest):

1. PM Forward Monitoring cells generated internally by the MC92501
2. Cells from the microprocessor
3. AIS, RDI, and CC cells generated internally by the MC92501

The data structure of inserted cells from the microprocessor is provided in **Section 7.4.1** Inserted Cell.

5.3.4 Egress Context Table Lookup

If the MC92501 is supporting multiple physical links, the link number that identifies the physical link to which the cell should be transferred is retrieved from the Connection Address word. In addition to identifying the cell's destination, the link number is also used to check if the physical link is enabled in the Egress Link Enable Register (ELER). If not, the cell is removed from the cell flow and copied to the Cell Extraction Queue. Since the Egress Connection Identifier (ECI) of the cell is explicitly provided, the context parameters can be read from the Context Parameters Table. The structure of the Context Parameters Table is presented in Section 7.3.3 Context Parameters Table.

NOTE: If address translation is not performed (see Section 5.3.7 Address Translation), the link number taken from the source determined by the *Egress Link Number Selection (ELNS)* bit of the Egress Switch Interface Configuration Register (ESWCR) is used instead.

5.3.5 UPC/NPC

The MC92501 performs the UPC/NPC function for the Egress flow if the *UPC Flow (UPCF)* bit in the ATMC CFB Configuration Register (ACR) is set. See **Section 6.2 UPC/NPC Support**.

5.3.6 Egress Copy/Remove/OAM Processing

If the *Egress Copy All (ECA) cells* bit is set, the cell is added to the Cell Extraction Queue to be transferred to the microprocessor. If the *Egress Remove All (ERA) cells* bit is set, the cell is removed from the cell flow after undergoing OAM processing. The option exists to copy and/or remove those cells whose VCI is identified as "reserved". See Section 7.2.6.29 Egress VCI Copy Register (EVCR) and Section 7.2.6.31 Egress VCI Remove Register (EVRR) for details. This option is enabled on a connection basis by the *Egress VCR/VRR Registers Enable (EVRE)* bit of the Egress Parameters word. There is also an option to copy those cells whose PTI value is 110 or 111 to the Cell Extraction Queue. This option is controlled by the *Egress PTI 6 Copy (EP6C)* and *Egress PTI 7 Copy (EP7C)* bits of the Egress Parameters word.

The CRC-10 field of received OAM cells is checked. If an error is detected in this field, or if the cell is an Illegal OAM cell (see Section 6.3.4.1 Illegal OAM Cells), no OAM processing is performed on the cell. The cell is removed from the cell flow and copied to the Cell Extraction Queue. All OAM cells are further classified by the OAM Cell Type and OAM Function Type fields (see **Figure 6-5** and **Table 6-3**). The OAM cells that receive special processing are:

- Cells received from the physical layer include:
  - AIS (see **Section 6.3.5.1.1 VP/VC AIS**)
  - RDI (see **Section 6.3.5.1.2 VP/VC RDI**)
  - Loopback (see **Section 6.3.5.3 VPC/VCC Loopback Cells)**
  - Continuity Check (see **Section 6.3.5.1.3 Continuity Check)**
  - Forward Monitoring (see **Section 6.3.7 Performance Monitoring)**

- Backward Reporting (see **Section 6.3.7 Performance Monitoring)**
- If this is a segment/connection termination point (see **Section 6.3.4 General OAM**) of the OAM flow, the segment/end-to-end OAM cell is removed from the cell flow.

• Inserted cells:
- Forward Monitoring from the processor (see **Section 6.3.7 Performance Monitoring)**
- Forward Monitoring internally generated (see **Section 6.3.7 Performance Monitoring)**

When a user data cell is processed, the traffic bits (see **Section 6.3.5.1.3 Continuity Check**) are set. User data cells that belong to an active PM block test are processed as described in **Section 6.3.7 Performance Monitoring**.

5.3.7 Address Translation

The address fields of the cell header are optionally replaced by the outgoing address of the outgoing link as read from the Egress Translation Address word of the Context Parameter Table. The address translation is controlled by the *Egress Address Translation Disable (EATD)* bit of the Egress Switch Interface Configuration Register (ESWCR). If the outgoing address is provided by the switch interface block, no address translation is performed. If the outgoing address is not provided with the cell, the address is translated using the value read from the Context Parameters Table. If the cell belongs to a VPC, only the VPI field is replaced. If the cell belongs to a VCC, both the VPI and VCI fields are replaced. If the cell belongs to a UNI link, the replacement of the GFC field is controlled by the *Replace GFC Field (RGFC)* bit of the Egress Processing Configuration Register (EPCR). The MC92501 sets the middle bit of the PTI on cells whose received PTI is 000 or 001 when the EFCI bit received from the Egress switch interface block is set.

5.3.8 Cell Counting

For each cell transmitted to the PHY layer, one of the counters from the Egress Billing Counters Table for this connection is incremented, unless the table does not exist - see Section 7.2.6.11 Egress Processing Configuration Register (EPCR). One of the link cell counters from the Egress Link Counters Table is also incremented if the table exists - see Section 7.2.6.14 General Configuration Register (GCR). The appropriate counter is chosen based on the CLP bit and whether the cell is an OAM cell. Cells that are removed from the cell flow are not included in the usage counts. Inserted cells and internally generated cells are included in the usage counts.

5.3.9 Transmission to the Physical Layer

The Egress Physical-Layer Interface (EPHI) block receives cells from the Cell Processing block, queues them, and transmits the cell data to the physical layer using the UTOPIA standard interface. The cells are then stored in a FIFO and are disassembled into bytes for transmission to the physical layer. The size of the FIFO is programmable to either 2 or 4 cells using the *Egress PHY Interface FIFO Control (EPFC)* bit. See Section 7.2.6.4 Egress PHY Configuration Register (EPHCR) for more information.

If the EPHI FIFO is empty, the MC92501 optionally (see **Section 7.2.6.4** Egress PHY Configuration Register (EPHCR)) fills the hole with a cell to provide a continuous cell flow at the physical layer bit rate. The type of cell used to fill the holes is either "Unassigned" (an ATM layer cell) or "Idle" (a physical layer cell) according to the *Egress Generate Idle Cells (EGIC)* bit defined in **Section 7.2.6.4** Egress PHY Configuration Register (EPHCR). See **Table 7-111** and **Table 7-121** for the header values used for unassigned and idle cells. If multiple physical links are supported, the generation of unassigned/idle cells is not supported and should not be enabled.

Since the MC92501 processes cells at a higher rate than they are transmitted to the physical layer, the EPHI block cannot transfer a cell during every cell processing slot. Over time, cells may accumulate in the EPHI FIFO until it is full. When this happens, the MC92501 does not process a cell during the next cell processing slot, allowing the FIFO to drain to the physical layer. TXPRTY is always driven with odd parity over TXDATA, regardless of whether or not parity checking is enabled on the Ingress PHY Interface. The fifth octet of the transmitted cell (the HEC field) is always transmitted as zero, regardless of the value passed to the MC92501 by the switch interface block.

6

MC92501 Protocol Processing Support

6.1 Overview

The MC92501 ATM Cell processor incorporates support for a number of industry protocols to provide maximum functional flexibility. The protocol support is described in detail in the following paragraphs and includes:

- Usage Parameter Control/Network Parameter Control (UPC/NPC) Support, including:
  - Cell based UPC
  - Packet based UPC
  - Partial Packet Discard (PPD)
  - Early Packet Discard (EPD)
  - Limited Early Packet Discard (Limited EPD)
- Operation And Maintenance (OAM) Support, including:
  - Internal Scan
  - General OAM
  - Fault Management
  - Performance Management
  - Performance Monitoring
  - Activation/Deactivation OAM Cells
- Available Bit Rate (ABR) Support
- CLP Transparency Support
- Selective Discard Support
- Multiple PHY Support

6.2 UPC/NPC Support

One of the major advantages of ATM is the ability to distribute the available bandwidth among many connections dynamically. However, this same feature makes congestion in an ATM network difficult to predict. In order to facilitate network management, limits are imposed on connection traffic parameters. Typically, the maximum average bandwidth and burstiness are defined. Even when the usage parameters are defined, a single user not conforming to the agreed-upon parameters can cause congestion that reduces the quality of other user service. Therefore, usage parameters should be enforced at the entrance to the network so that violating users suffer any reduced service quality. This enforcement is called Usage Parameter Control (UPC) at a User-Network Interface (UNI) and Network Parameter Control (NPC) at a Network-Network Interface (NNI). The UPC/NPC function can be activated on either the Ingress flow (by setting the *UPC Flow (UPCF)* bit in the ATMC CFB Revision Register (ARR)—see **Section 7.2.4.4**) or the Egress flow (if the bit is reset).

6.2.1 Cell Based UPC

The MC92501 UPC/NPC algorithm, based on the concept of Leaky Buckets, detects cells that violate the traffic agreement and, optionally, either tags violating cells (i.e., changes CLP-field from 0 to 1) or discards them (removes from the cell flow). As described in **Appendix A**, each connection has a flexible arrangement of Leaky Buckets (0 to 4), Leaky Bucket parameters, and UPC/NPC enforcement algorithms. At connection setup time, a set of bucket characteristics is loaded into the Bucket Table section of context memory to define the expected cell arrival pattern for a particular connection. The UPC/NPC function uses these characteristics to enforce the agreed-upon user traffic requirements.

NOTE: For constant bit rate and variable bit rate connections, constant bucket characteristics are normally defined when the connection is set up. Other types of connections may require dynamic UPC/NPC enforcers which the processor updates bucket characteristics while the connection is active. Such changes must be made cautiously in order to maintain consistency among various enforcer parameters.

All unremoved user data cells are subject to UPC/NPC processing according to the connection parameters. Maintaining counts of discarded or tagged cells per connection in the Policing Counters Table is optional (see **Section 7.2.6.10** Ingress Processing Configuration Register (IPCR) or **Section 7.2.6.11** Egress Processing Configuration Register (EPCR)). An Ingress-only “don’t touch” option is provided to apply the UPC/NPC algorithm for statistical purposes without tagging or discarding the violating cells. This option is controlled by the *UPC/NPC Don’t Touch (UDT)* bit in the connection Ingress Parameters record. Placing the identical bucket pointer in the Common Parameters word of each connection allows the UPC/NPC mechanism to enforce the sum of several connections . This method is likely to be used at the boundary point where many VCCs are combined into a VPC.

6.2.2 Packet Based UPC

ATM Forum TM-4.0 Section 5.8 states that if a network element needs to discard cells, then it is typically more effective to discard them at the packet level rather than the cell level. Adhering to this recommendation, the MC92501 UPC function may perform cell-based discard or packet-based discard. It supports packet discard on VC connection AAL5 packets (not including OAM cells). The MC92501 offers 4 modes of UPC operation on a per connection basis: Cell-based UPC, Partial Packet Discard (PPD), Early Packet Discard (EPD) and Limited Early Packet Discard (Limited EPD). These modes are selected on a per connection basis using the *UPC Operation Mode (UOM)* bit in the Common Extension Parameters Table.

NOTE: A stream of one or more user cells belonging to the same VC connection on which the PTI[0] bit equals 1 on the last cell and PTI[0] bit equals 0 on all the other cells.

6.2.3 Cell-based UPC

This is the default mode. The MC92501 discards cells on a per cell basis.

6.2.4 Partial Packet Discard (PPD)

According to this algorithm, once a cell is discarded, all succeeding cells belonging to that packet are discarded (the only exception is the last cell—see explanation below). The UPC functions include:

- The UPC can be in one of 2 states: discarding and not discarding.
- While the UPC is in the not discarding state, it performs normal cell-based operation with tagging and policing counters update.
- The UPC transits from the not discarding to the discarding state on the first discarded cell.
- While the UPC is in the discarding state, it doesn't update the UPC bucket and it does increment the policing discard counter.
- When the last cell of a packet is received while the MC92501 is in the discarding mode there are 2 options:
  - If all the cells belonging to that packet were discarded then this last cell is discarded.
  - If not all the cells belonging to that packet were discarded then this means that the packet was truncated and this last cell is admitted in order to delineate the corrupted packet from the next packet. There is however one exception—if this last cell is violating cell-based UPC, then it is discarded.

Figure 6-1 shows PPD algorithm usage. The first packet is truncated. The last cell of the first packet is transmitted and thus avoids the concatenation of the corrupted packet to packet #2. Packet #3 is truncated as well. Its last cell is not transmitted because it cannot be admitted by the cell-based UPC. Packet #4 is not transmitted at all.

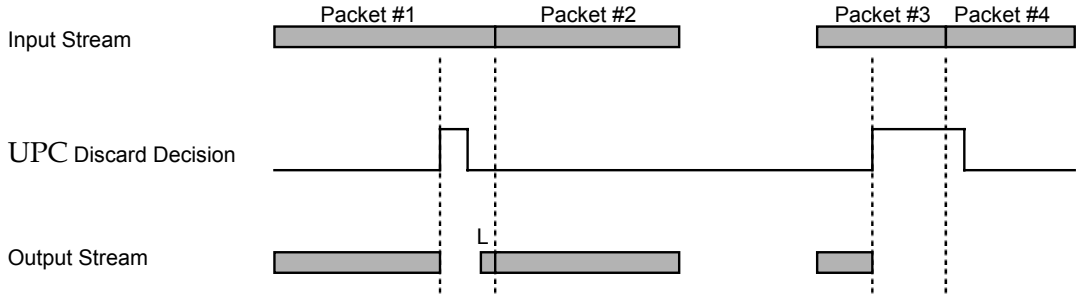


Figure 6-1. PPD Algorithm Example

6.2.5 Early Packet Discard (EPD)

According to this algorithm, the decision to discard a packet occurs only at the beginning of a packet. This means that a packet is either fully discarded or fully passed. EPD operates as follows:

- When the EPD is discarding cells, the buckets are not updated and the policing discard counter is incremented.
- When the EPD decides that a frame should be passed:
  - All tagging buckets continue to work in a cell-based fashion.
  - All discarding buckets perform their calculations as if the limit parameter is infinite and therefore increment the bucket content and do not discard any cells. As a result, their bucket content can be greater than their bucket limit.
  - The MC92501 may increment its police tagging counter.

Figure 6-2 is an example of the EPD algorithm. The first packet cells violate UPC, but due to EPD, this packet is fully passed. The second packet is fully discarded. The third packet cells violate UPC, but this packet is not discarded. Because the fourth packet arrives after a relatively long time, the UPC buckets are drained and the UPC is no longer violated. Therefore, the fourth packet is passed.

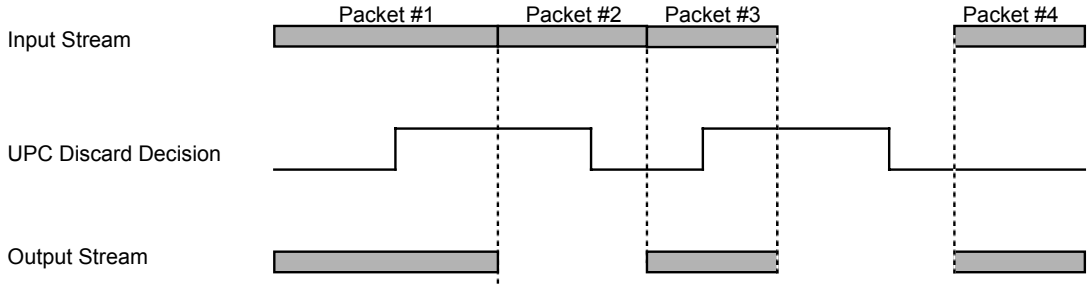


Figure 6-2. EPD Algorithm Example

6.2.6 Limited Early Packet Discard (Limited EPD)

One disadvantage of the EPD algorithm is that once a packet is passed, the decision can not be changed until the last cell of that packet. In the case of big packets, this can cause switch congestion. Using the limited EPD algorithm, a connection can stop passing cells once it reaches a predefined limit. In the MC92501, that limit is reached once the first bucket starts discarding cells. The first bucket should have the same parameters as one of the other buckets except for the bigger limit. **Figure 6-3** shows how the three buckets function together.

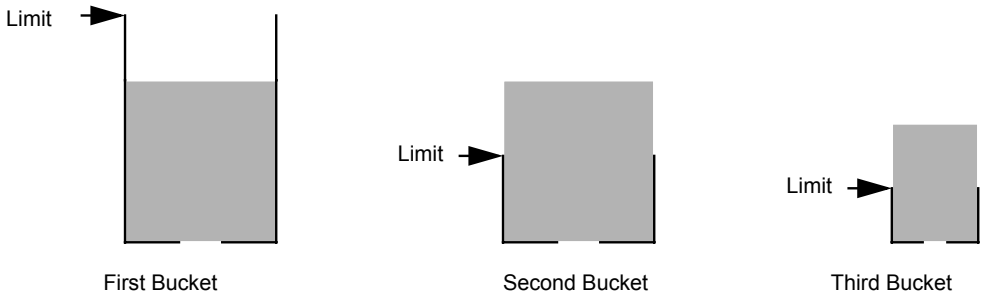


Figure 6-3. Three Bucket Example

The first bucket limits the EPD algorithm.The first and second buckets share the same parameters except for the limit, therefore their bucket content is always the same. Although the second bucket content is higher than its limit, cells are admitted by the EPD algorithm. When the first bucket reaches its limit, then cells are discarded.

Figure 6-4 illustrates the difference between EPD and limited EPD function.

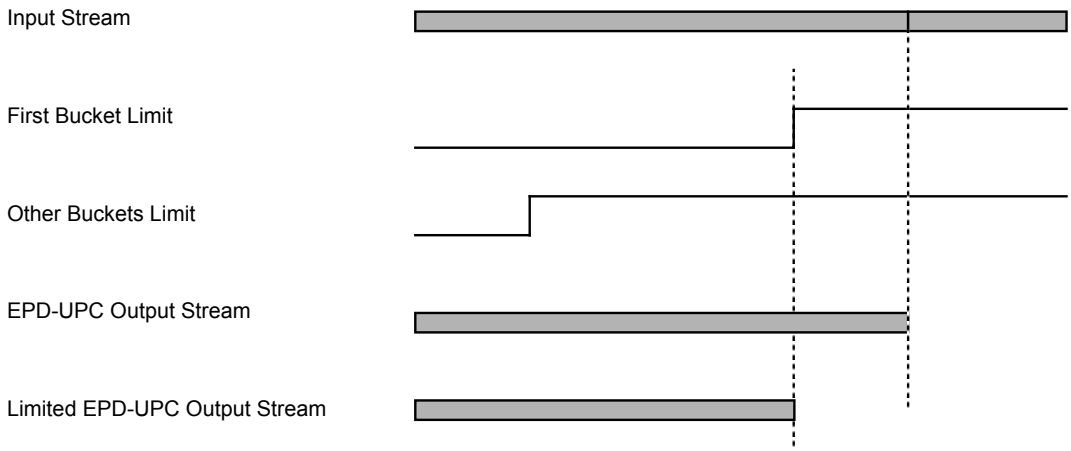


Figure 6-4. Comparison of EPD vs. Limited EPD

6.3 OAM Support

This section describes the ways in which the MC92501 provides support for ATM-level (F4 and F5) Operations and Maintenance (OAM) of its connections. Alarm Surveillance, Continuity Check, Loopback cells, and Performance Management block tests are supported on all connections.

NOTE: The operation of the OAM functions at VP/VC boundaries is discussed in **Appendix D**.

6.3.1 Conventions

The following conventions are used:

- **Copying a cell**—Transferring a cell to the microprocessor.
- **Removing a cell**—Removing a cell from the cell flow. (A cell removed from the Ingress cell flow is not transferred to the switch.)

6.3.2 ATM Layer OAM Definitions

Table 6-1. Pre-assigned Header Values at the UNI

Use	GFC	VPI	VCI	PTI	CLP
Segment OAM F4 flow cell	XXXX	AAAAAAAA	00000000 00000011	0A0	A
End-to-End OAM F4 flow cell	XXXX	AAAAAAAA	00000000 00000100	0A0	A
Segment OAM F5 flow cell	XXXX	AAAAAAAA	AAAAAAAA AAAAAAAA	100	A
End-to-End OAM F5 flow cell	XXXX	AAAAAAAA	AAAAAAAA AAAAAAAA	101	A

Notes: 1. A = available for use by the appropriate ATM layer function
2. X = don't care bit

Table 6-2. Pre-assigned Header Values at the NNI

Use	VPI	VCI	PTI	CLP
Segment OAM F4 flow cell	AAAA AAAAAAAA	00000000 00000011	0A0	A
End-to-end OAM F4 flow cell	AAAA AAAAAAAA	00000000 00000100	0A0	A
Segment OAM F5 flow cell	AAAA AAAAAAAA	AAAAAAAA AAAAAAAA	100	A
End-to-End OAM F5 flow cell	AAAA AAAAAAAA	AAAAAAAA AAAAAAAA	101	A

Note: A = available for use by the appropriate ATM layer function

6.3.2.1 Virtual Path (F4) Flow Mechanism

The F4 flow is recognized by means of preassigned Virtual Channel Identifiers within the Virtual Path. There are two kinds of F4 flows which can exist simultaneously:

- **End-to-end (identified by pre-assigned VCI 4)—**This flow is used for end-to-end VPC operations communications. Cells inserted into this flow can only be removed by the end-points of the Virtual Path.
- **Segment (identified by pre-assigned VCI 3)—**This flow is used for communicating operations information with the bound of one VPC link or multiple interconnected VPC links. The concatenation of VPC links is called a VPC Segment. Cells inserted into this flow may only be removed by the Segment endpoints. Segment endpoints must remove these cells to prevent confusion in adjacent segments.

Cells can be inserted into the flow at any connecting point. However, cells may only be terminated at the F4 end points. The MC92501 recognizes F4 OAM cells on any connection where the EVPC/IVPC bit is set and the header is as shown in **Table 6-1** and **Table 6-2**. Cells with F4 OAM headers received on a connection with EVPC/IVPC = 0 are treated as described in Section 6.3.4.1 Illegal OAM Cells.

6.3.2.2 Virtual Channel (F5) Flow Mechanism

The F5 flow is recognized by means of preassigned Payload Type Identifier (PTI). There are two kinds of F5 flows which can exist simultaneously. These are:

- **End-to-end (identified by PTI 5)—**This flow is used for end-to-end VCC operations communications. Cells inserted into this flow can only be removed by the endpoints of the Virtual Channel.
- **Segment (identified by PTI 4)—**This flow is used for communicating operations information with the bound of one VCC link or multiple interconnected VCC links. The concatenation of VCC links is called a VCC Segment. Segment endpoints must remove these cells to prevent confusion in adjacent segments.

Cells can be inserted into the flow at any connecting point, however, cells may only be terminated at the F5 end points. The MC92501 recognizes F5 OAM cells on any connection where the EVPC/IVPC bit is reset and the header is as shown in **Table 6-1** and **Table 6-2**.

6.3.2.3 OAM Types and Function Types

End-to-end and Segment flows may use all OAM functions unless otherwise noted. The following descriptions do not differentiate between End-to-end and Segment flows. **Table 6-3** defines the values of OAM Cell Type and Function Type recognized by the MC92501. The location of these fields in the OAM cell is shown in **Figure 6-5**.

Table 6-3. OAM Types Explicitly Identified by the MC92501

OAM Cell Type	Code	Function Type	Code
Fault Management	0001	AIS	0000
		RDI	0001
		Continuity Check	0100
		Loopback	1000
Performance Management	0010	Forward Monitoring	0000
		Backward Reporting	0001

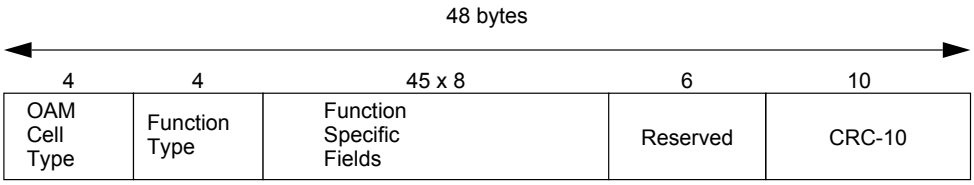


Figure 6-5. OAM Cell Payload Structure

Any unused Function Specific Fields in received OAM cells are ignored by the MC92501. When internally generating OAM cells, the MC92501 fills the unused field with the standard default value of $6A and fills the 6-bit Reserved field with zeroes. When inserting OAM cells with the payload provided by the microprocessor, the MC92501 does not modify the unused and Reserved fields.

6.3.3 Internal Scan

The MC92501 uses the Internal Scan process to generate Alarm Indication Signal (AIS), Remote Defect Indicator (RDI), and Continuity Check (CC) cells and insert them into the cell flows. When the scan process is activated by the microprocessor writing to the Start SCAN Register (SSR), the MC92501 scans the Context Parameters Table records (see Section 7.3.3.5 Egress Parameters and Section 7.3.3.6 Ingress Parameters) starting from the record indicated by the *Context Memory Highest Value (CMHV)* field of the Internal Scan Control Register (ISCR) down to the first record (CI = 0) to see if any of the Send AIS, Send RDI, or Send CC bits are set, in which case the appropriate cell is generated and queued for insertion in the Ingress or Egress insertion queue. When the scan is completed and the generated cells inserted into the cell flow, the *Scan Process Done (SPD)* bit of the Interrupt Register (IR) is set. The types of cells included in the scan are defined by the Internal Scan Control Register (ISCR) (see **Section 7.2.5.11)**. The scan should be activated normally about once per second, in accordance with the OAM standards.

6.3.4 General OAM

At the VC switch (VCX) at which a VPC is terminated, the VCCs are accessible. Context Table records are maintained for each of the VCCs (VPC bit is 0) as well as the reserved VCI values used for the OAM F4 flows (VPC bit is 1) (see **Appendix D** for details). In this way, all OAM features which the MC92501 supports on all connections (Fault Management, Activation/Deactivation) are supported on each VCC individually as well as on the VPC collectively. **Table 6-4** describes the general OAM bits that define the scope of the OAM processing at this node. **Figure 6-6** shows endpoint visibility. **Figure 6-7** illustrates the use of the OAM origin and termination bits at the endpoints of the OAM flows.

Table 6-4. General OAM Bits

Logical Name	Bit Name	Context Parameters Table Word	Stat/ Dyn	Explanation
VPC	EVPC IVPC	Egress/ Ingress Parameters	S	Virtual Path Connection
Segment OAM Termination	ESOT ISOT	Egress/ Ingress Parameters	S	This node is the terminating point of the VCC/VPC segment. All arriving Segment OAM cells should be removed from the cell flow after the OAM processing. Also, segment Loopback OAM cells with the Loopback Location ID = “Endpoint” are looped back here.
Segment OAM Origin	ESOO ISOO	Egress/ Ingress Parameters	S	This node is the originating point of the VCC/ VPC segment.
End-to-end OAM Termination	EEOT IEOT	Egress/ Ingress Parameters	S	This node is the terminating point of the VCC/VPC OAM flow. All arriving OAM cells should be removed from the cell flow after the OAM processing. Also, end-to-end Loopback OAM cells with the Loopback Location ID = “Endpoint” are looped back here.
Copy other OAM cells	ECOT ICOT	Egress/ Ingress Parameters	S	Copy OAM cells whose OAM type is not recognized by the MC92501.
Copy all OAM cells	ECAO ICAO	Egress/ Ingress Parameters	S	Copy all received OAM cells. This bit may be set for monitoring all OAM traffic on a connection.

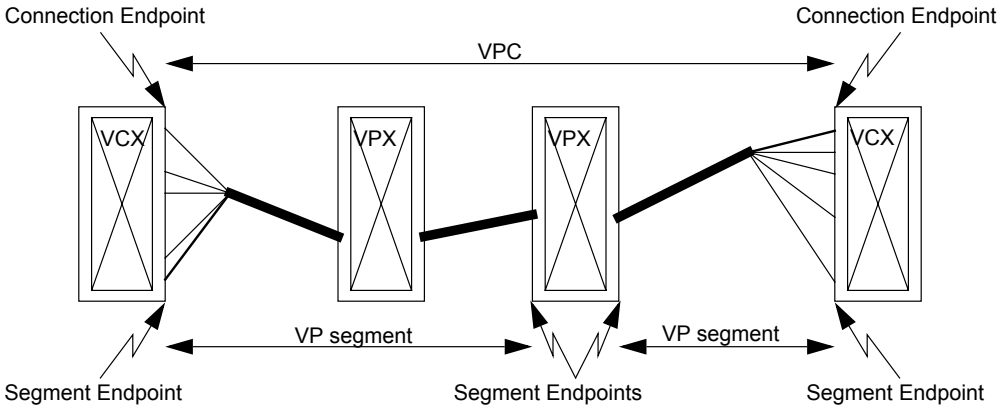


Figure 6-6. Visibility of VCCs at the Endpoints of VPCs

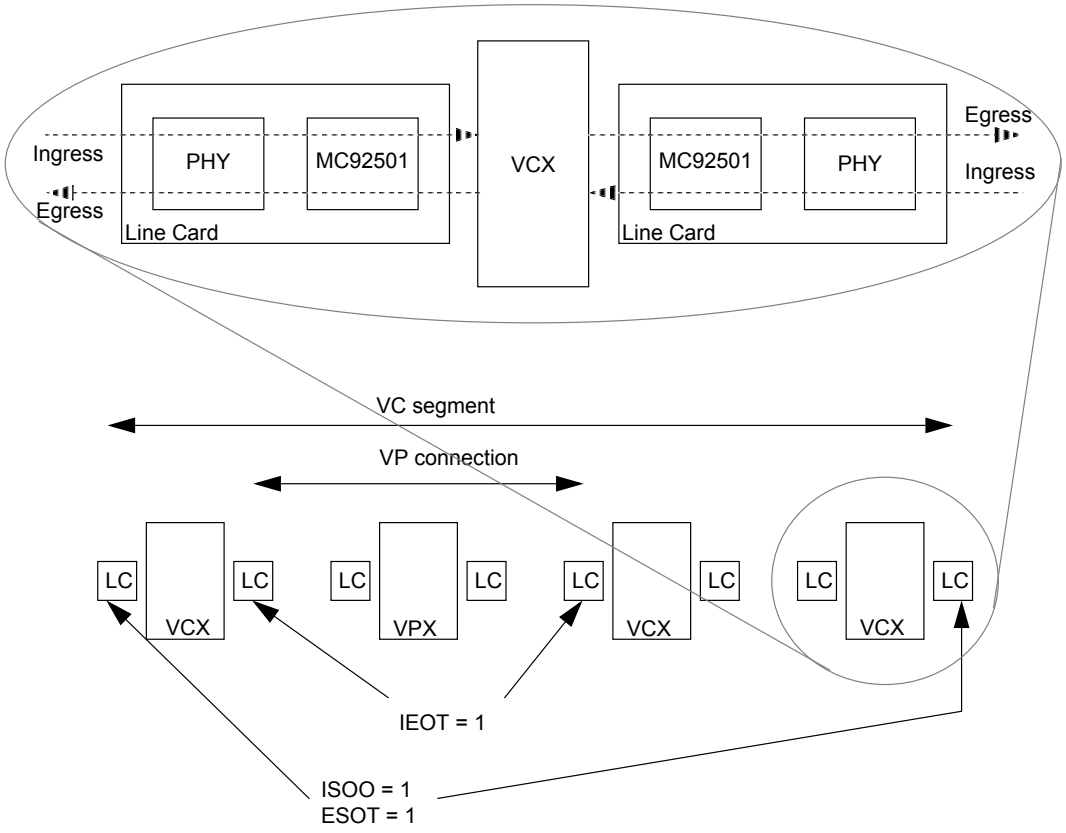


Figure 6-7. Example of VP Connection / VC Segment

6.3.4.1 Illegal OAM Cells

OAM cells that are illegal or outside of their legal scope are copied to the Cell Extraction Queue. This includes a Segment OAM cell received at a segment OAM origin point (ESOO/ISOO = 1) which is not a segment OAM termination (ESOT/ISOT = 0).

6.3.4.2 Other OAM Cells

OAM cells with an OAM Cell Type or OAM Function Type not explicitly handled by the MC92501 (any combination not appearing in Table 6-3) are copied to the Cell Extraction Queue if *Egress Copy Received Other OAM Cells/Ingress Copy Received Other OAM Cells (ECOT/ICOT)* is set.

NOTE: This includes Activation/Deactivation OAM cells.

6.3.5 Fault Management

Fault management includes two areas:

- Alarm surveillance
- Failure localization and testing

6.3.5.1 Alarm Surveillance

The function specific fields of the AIS/RDI Fault Management cell as shown in **Figure 6-8** are:

- **Failure Type**—No values are currently standardized. The default value is $6A . This field is ignored by the MC92501 when processing received AIS/RDI cells.
- **Failure Location**—No values are currently standardized. The default value is $6A in each octet. This field is ignored by the MC92501 when processing received AIS/RDI cells.

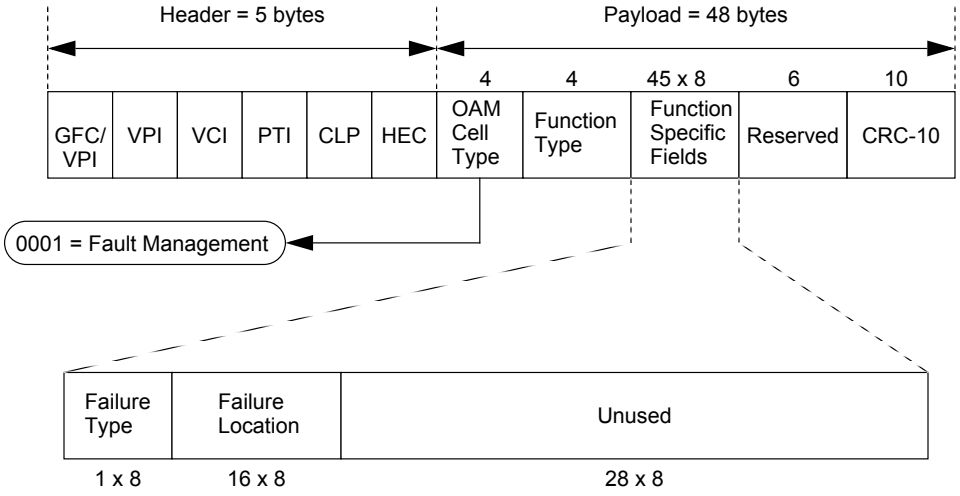


Figure 6-8. AIS/RDI Fault Management Cell

6.3.5.1.1 VP/VC AIS

The MC92501 provides full support for generating Alarm Indication Signal (AIS) cells and for reporting that an AIS cell has been received. To initiate the generation of AIS cells in the Egress cell flow, the microprocessor sets the *Egress Send AIS (ESAI) cell* bit for a specific connection. This bit causes the MC92501 to generate an AIS cell for this connection on each pass of the Internal Scan if the *Egress Enable AIS (EEAI)* bit of the Internal Scan Control Register (ISCR) is set. To initiate the generation of AIS cells in the Ingress cell flow, the microprocessor sets the *Ingress Send AIS (ISAI) cell* bit for a specific connection. This bit causes the MC92501 to generate an AIS cell for this connection on each pass of the Internal Scan if the *Ingress Enable AIS (IEAI)* bit of the Internal Scan Control Register (ISCR) is set. See Section 6.3.3 Internal Scan for more details. In order to meet the Bellcore requirement to insert the first AIS cell in less than 500 ms, the first AIS cell may be inserted directly by the microprocessor using the Cell Insertion Registers, while the succeeding cells are generated by the Internal Scan.

NOTE: If the ERET/IRET bit (see **Table 6-7**) is set by the MC92501 while AIS cells are being generated, a valid cell has been received, and the ESAI/ISAI bit can most likely be reset.

AIS cells that are generated by the Internal Scan are always End-to-End. The Failure Type and Failure Location fields are coded with their default value of $6A in each octet since no values are currently standardized. The remaining 28 octets of the function-specific fields are also coded with $6A since they are unused in AIS cells. AIS cells that have been generated as a result of the ESAI bit are inserted in the Egress cell flow, and those that have been generated as a result of the ISAI bit are inserted in the Ingress cell flow. When an end-to-end AIS cell is received on a connection, the appropriate (Ingress or Egress) Receive AIS bit of that connection is set. Note that segment AIS cells are treated as OAM cells, but not as AIS cells. Therefore, they do not cause the Receive AIS bits to be set, and they may be copied by using the ECAO/ICAO bit, but not by using the ECAS/ICAS bit. All connections should be scanned by the microprocessor for the Receive AIS bits at regular intervals. The default scan rate should be once per second to allow the AIS state to be declared within one second of the AIS cell having been received.

The *Egress Copy Received AIS Cells (ECAS)* bit and/or the *Ingress Copy Received AIS Cells (ICAS)* bit may be set at any point along the connection to indicate that all AIS cells received on the connection should be copied to the microprocessor. If the Failure Type and Failure Location fields are set to their default values, the AIS cells do not contain useful information so the ECAS and ICAS bits would not normally be set. AIS cells are removed from the cell flow at the connection endpoint (see **Table 6-4**).

Table 6-5 describes the External Memory bits that are used for AIS processing.

Table 6-5. AIS Bits

Logical Name	Bit Name	Table	Stat/ Dyn	Used by	Explanation
Send AIS	ESAI ISAI	Context Parameters	S	Internal Scan	insert an AIS cell in the Egress/ Ingress direction
Receive AIS on Egress	ERAS	Flags	D	Egress	AIS cell has been received in the Egress cell flow.
Receive AIS on Ingress	IRAS	Flags	D	Ingress	AIS cell has been received in the Ingress cell flow.
Copy Received AIS Cells	ECAS ICAS	Context Parameters	S	Egress/ Ingress	copy AIS cells to the Cell Extraction Queue

6.3.5.1.2 VP/VC RDI

The MC92501 provides full support for generating Remote Defect Indicator (RDI) cells and for reporting that an RDI cell has been received. To initiate the generation of RDI cells in the Egress cell flow, the microprocessor sets the *Egress Send RDI (ESRD) cell* bit for a specific connection. This bit causes an RDI cell to be generated on this connection on each pass of the Internal Scan if the *Egress Enable RDI (EERD)* bit of the Internal Scan Control Register (ISCR) is set. To initiate the generation of RDI cells in the Ingress cell flow, the microprocessor sets the *Ingress Send RDI (ISRD) cell* bit for a specific connection. This bit causes an RDI cell to be generated on this connection on each pass of the Internal Scan if the *Ingress Enable RDI (IERD)* bit of the Internal Scan Control Register (ISCR) is set. See Section 6.3.3 Internal Scan for more details.

The first RDI cell may be inserted directly to reduce the initial delay, while the succeeding cells are generated by the Internal Scan. Note that if the Receive End-to-end Traffic bits (see **Table 6-7**) are set while RDI cells are being generated, a valid cell has been received, and the ESRD/ISRD bit can most likely be reset. RDI cells that are generated by the Internal Scan are always End-to-End. The Failure Type and Failure Location fields are coded with their default value of $6A in each octet since no values are currently standardized. The remaining 28 octets of the function-specific fields are also coded with $6A since they are unused in RDI cells. RDI cells that have been generated as a result of the ESRD bit are inserted in the Egress cell flow, and those that have been generated as a result of the ISRD bit are inserted in the Ingress cell flow.

When an end-to-end RDI cell is received on a connection, the appropriate (Ingress or Egress) Receive RDI bit of that connection is set. Note that segment RDI cells are treated as OAM cells, but not as RDI cells. Therefore, they do not cause the Receive RDI bits to be set, and they may be copied by using the ECAO/ICAO bit, but not by using the ECRD/ICRD bit.

All connections should be scanned by the microprocessor for the Receive RDI bit at regular intervals. The default scan rate should be once per second which allows the RDI state to be declared within one second of the RDI cell being received. The *Egress Copy Received RDI Cells (ECRD)* bit or the *Ingress Copy Received RDI Cells (ICRD)* bit may be set at any point along the connection to indicate that all RDI cells received on the connection should be copied to the Cell Extraction Queue. If the Failure Type and Failure Location fields are set to their default values, the RDI cells do not contain useful information so the ECRD and ICRD bits would not normally be set. RDI cells are removed from the cell flow at the connection endpoint (see **Table 6-4**). **Table 6-6** describes the External Memory bits that are used for RDI processing.

Table 6-6. RDI Bits

Logical Name	Bit Name	Table	Stat/ Dyn	Used by	Explanation
Send RDI	ESRD ISRD	Context Parameters	S	Internal Scan	Insert an RDI cell in the Egress/Ingress direction
Receive RDI on Egress	ERRD	Flags	D	Egress	RDI cell has been received in the Egress cell flow.
Receive RDI on Ingress	IRRD	Flags	D	Ingress	RDI cell has been received in the Ingress cell flow.
Copy Received RDI Cells	ECRD ICRD	Context Parameters	S	Egress/ Ingress	Copy RDI cells to the Cell Extraction Queue

Figure 6-9 shows the F4 flows for the case where the VP connection end points are logically at the edges of the VCX switches at which a set of VCCs are collected to be a VPC.

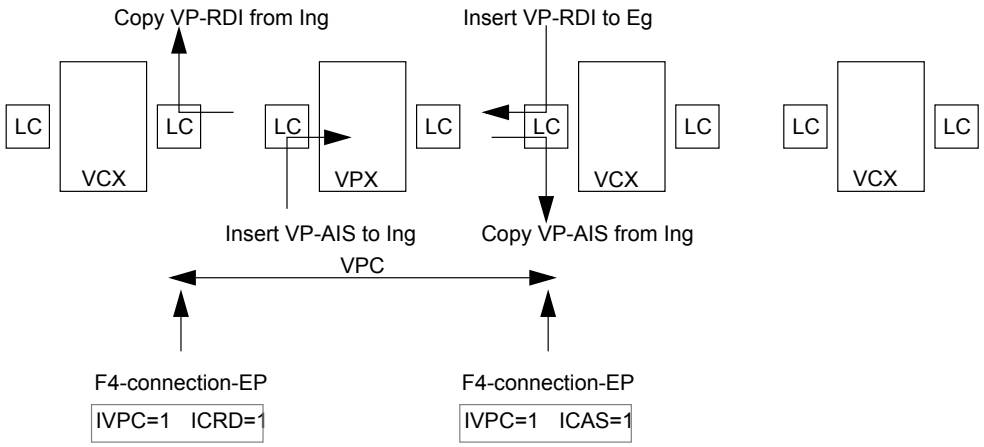


Figure 6-9. F4 AIS/RDI Flows for a VPC Internal to the Network

Figure 6-10 shows the F4 flows for the case where the VP connection is routed to the user end point through a VPX. In this case the connection endpoint must be on the line card closest to the user.

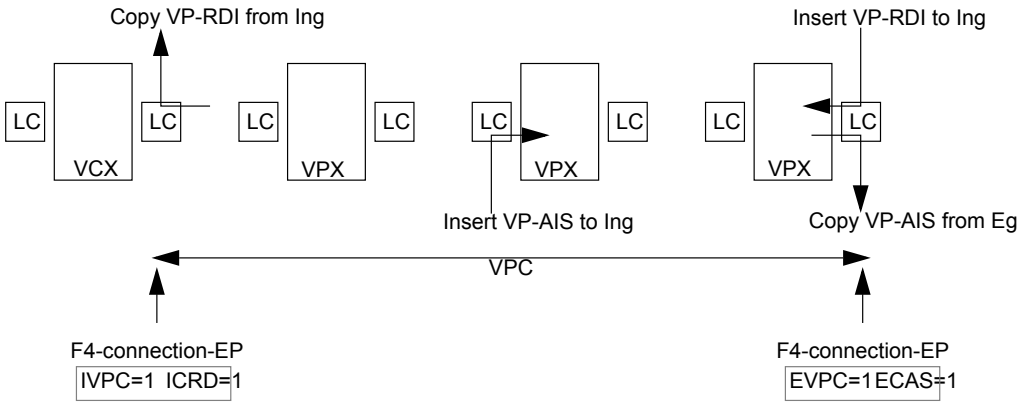


Figure 6-10. F4 AIS/RDI Flows for a VPC that Crosses the UNI

Figure 6-11 shows the F5 flows for a Virtual Channel Connection where one endpoint is on the inside of the switch and the other endpoint is on the outside of the switch.

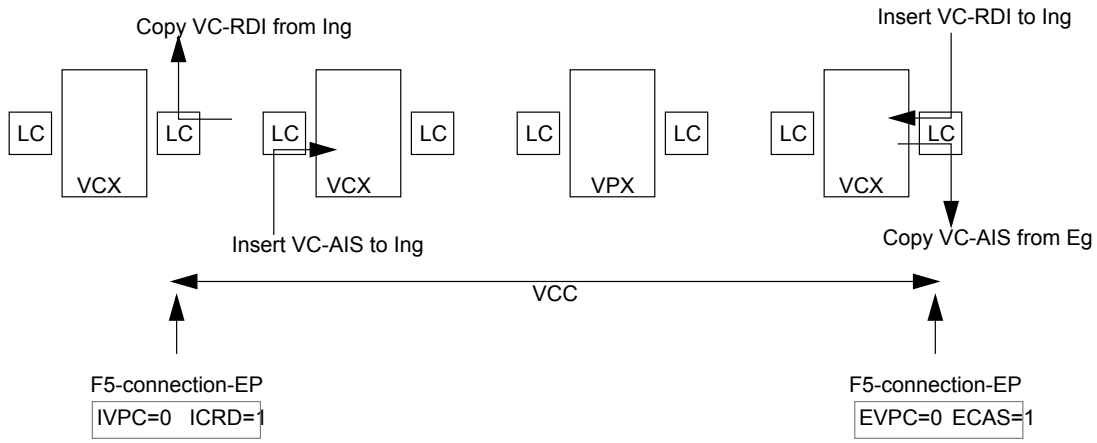


Figure 6-11. F5 AIS/RDI Flows

These examples illustrate where to set the Copy AIS and Copy RDI bits for the given flows. As previously stated, setting these bits to copy the cells is normally not necessary, and the Flag Table may be used to determine that an AIS or RDI cell has been received.

6.3.5.1.3 Continuity Check

The MC92501 provides full support for generating Continuity Check (CC) cells. The MC92501 also provides support for reporting about the connection traffic in order to enable the microprocessor to determine when connections have lost continuity. Both End-to-End and Segment Continuity Checks can be run simultaneously on the same connection. To initiate the generation of CC cells, the microprocessor sets any of the Send CC bits (ESCS, ISCS, ESCE, ISCE) for a specific connection. Each bit causes a CC cell to be generated on this connection during a pass of the Internal Scan if the corresponding bit of the Internal Scan Control Register (ISCR) is set. See Section 6.3.3 Internal Scan for more details.

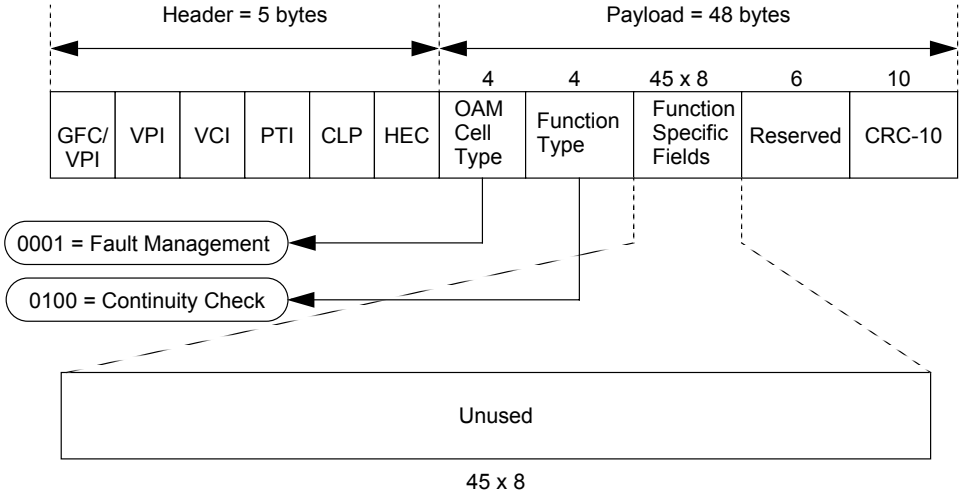


Figure 6-12. CC Fault Management Cell

The Receive Traffic bits are used to record if a valid cell (user information cell or CC cell) has been processed on the connection. On both the Ingress and Egress sides there are two bits to identify cases in which there is continuity on the segment but continuity from the endpoint has been lost. When a user cell or End-to-End CC cell is received, both Receive Traffic bits are set. When a Segment CC cell is received, only the Receive Segment Traffic bit is set.

NOTE: At the termination of a VPC, the user traffic is recorded by setting the Receive Traffic bits of the individual VCCs since the VCCs are visible. In order to check for continuity on the VPC, the microprocessor may logically OR the Receive Traffic bits of all the VCCs belonging to the VPC.

CC cells are removed from the Egress or Ingress flow at the connection/segment endpoint (see **Table 6-4**). **Table 6-7** describes the External Memory bits that are used for Continuity Check processing.

Table 6-7. Continuity Check Bits

Logical Name	Bit Name	Table	Stat/ Dyn	Used by	Explanation
Send CC segment	ESCS ISCS	Context Parameters	S	Internal Scan	Insert a segment CC cell in the Egress/Ingress direction
Send CC end-to-end	ESCE ISCE	Context Parameters	S	Internal Scan	Insert an end-to-end CC cell in the Egress/Ingress direction
Receive segment traffic on Egress	ERST	Flags	D	Egress	User data cell or CC cell received in the Egress cell flow
Receive end-to-end traffic on Egress	ERET	Flags	D	Egress	User data cell or end-to-end CC cell received in the Egress cell flow
Receive segment traffic on Ingress	IRST	Flags	D	Ingress	User data cell or CC cell received in the Ingress cell flow
Receive end-to-end traffic on Ingress	IRET	Flags	D	Ingress	User data cell or end-to-end CC cell received in the Ingress cell flow

6.3.5.2 Failure Localization and Testing

The function specific fields of the OAM Loopback cell are:

- **Loopback Indication**—This field provides a boolean indication as to whether or not the cell has already been looped back. The seven most significant bits are always coded as 0. The least significant bit is 1 before the cell is looped back and 0 after the cell has been looped back.
- **Correlation Tag**—Multiple Loopback cells may have be inserted in the stream. The Correlation Tag provides a means of correlating transmitted Loopback cells with received cells.
- **Loopback Location ID**—This field identifies the point along the connection where the loopback is to occur. Default value is all ones, indicating the end point of the connection or segment.
- **Source ID**—This field identifies the originator of the loopback cell. Default value is all ones.

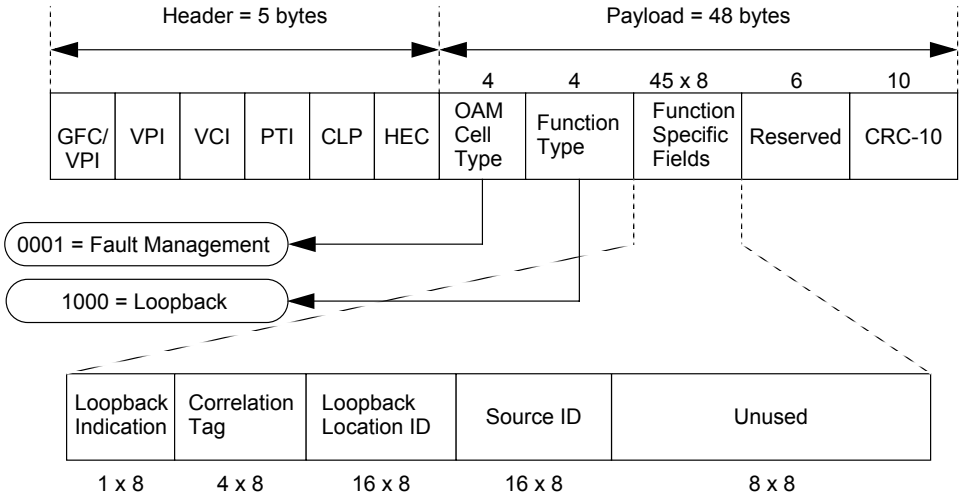


Figure 6-13. OAM Loopback Cell

6.3.5.3 VPC/VCC Loopback Cells

The MC92501 provides support for looping back and copying Loopback OAM cells. Loopback OAM cells are prepared by the microprocessor and inserted through the MC92501 to the specified cell flow. The MC92501 checks the Loopback Location ID of received Loopback OAM cells for a match with the MC92501's programmable Node ID (see **Section 7.2.6.24–Section 7.2.6.27**). It also checks for the "endpoint" Loopback Location ID (all 1s) on both the Ingress and Egress sides and declares a match if the corresponding OAM Termination bit is set for the connection. If the Loopback Location ID matches and the LSB of the Loopback Indication is 1, the cell is removed from the cell flow. The cell is copied to the Cell Extraction Queue so the microprocessor can simply decrement (reset) the Loopback Indication and insert the cell in the opposite direction.

NOTE: An intra-switch loopback can be performed by inserting a loopback cell in the Ingress with a Loopback Location ID that provides a match at the Egress side of the switch.

The MC92501 checks the Source ID of received Loopback OAM cells in the Ingress cell flow for a match with the MC92501's programmable Node ID (see **Section 7.2.6.24–Section 7.2.6.27**). If the Source ID matches and the LSB of the Loopback Indication is 0, the cell is removed from the cell flow and copied to the Cell Extraction Queue. A cell whose Loopback Indication equals 0 and whose Source ID contains the default value of all 1s is copied to the Cell Extraction Queue. (Such a cell is copied at every point along the OAM flow since the MC92501 cannot determine the location of the source.) Loopback cells with a Loopback Indication of 0 are removed at the connection/segment endpoint. **Table 6-8** summarizes the treatment of OAM Loopback cells.

Table 6-8. Processing of OAM Loopback Cells

Loopback Indication	Source ID	Loopback Location ID	Treatment	
			Intermediate Pt	End Pt
1	X	Node ID	Copy and remove	Copy and remove
1	X	All 1s	—	Copy and remove
0	Node ID	X	Copy and remove	Copy and remove
0	All 1s	X	Copy	Copy and remove
0	X	X	—	Remove

Figure 6-14 shows an example where the Loopback Location ID is not an endpoint, so the actions taken do not depend on the flow type (F4/F5).

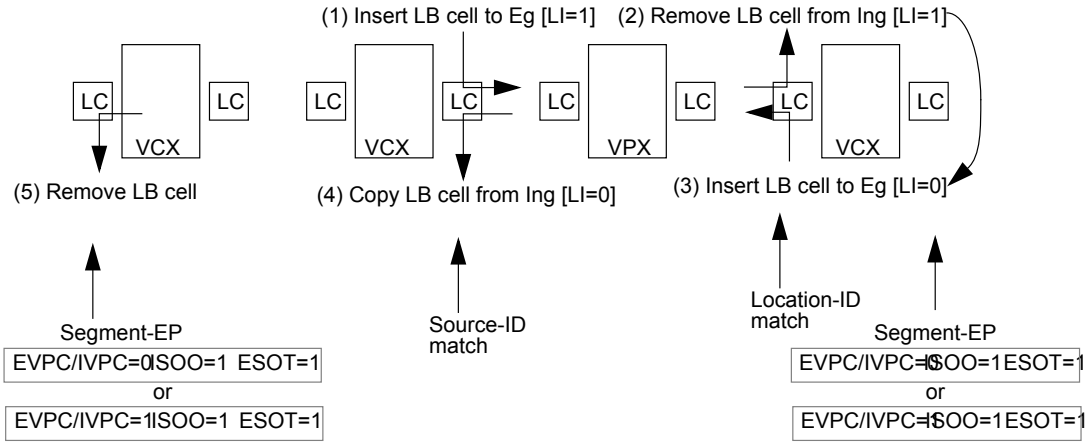


Figure 6-14. Loopback Not at Endpoint

Figure 6-15 shows the flow of an F5 OAM Loopback cell with both the Loopback_Location_ID and the Source ID set to all 1s.

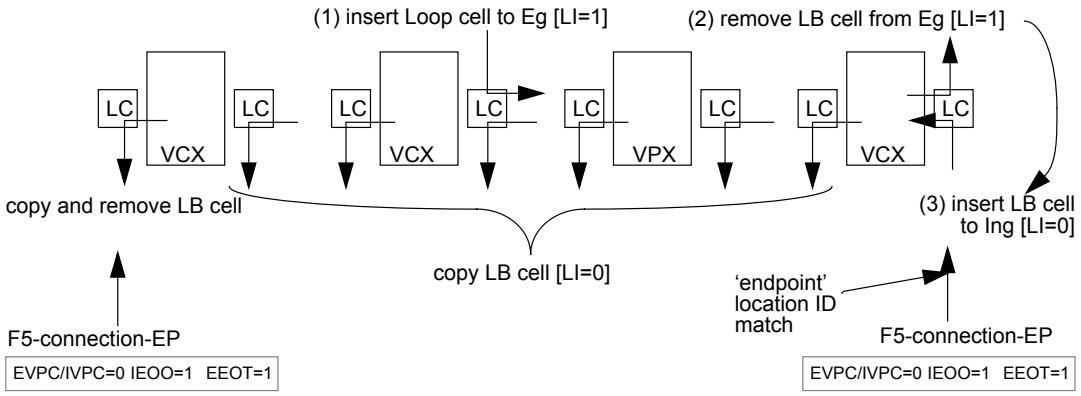


Figure 6-15. Loopback at Endpoint of VCC

6.3.6 Performance Management

The following fields are used for both Forward Monitoring and Backward Reporting cells:

- **Monitoring Cell Sequence Number (MCSN)**—The sequence number of the performance monitoring cell modulo 256.
- **Total User Cell 0+1 Number (TUC_{0+1})**—Indicates the total number of user cells (modulo 65,536) transmitted before the Forward Monitoring cell was inserted.
- **Total User Cell 0 Number (TUC_0)**—Indicates the total number of CLP=0 user cells (modulo 65,536) transmitted before the Forward Monitoring cell was inserted.
- **Time-Stamp (TSTP)**—May be used to represent the time when the cell was inserted.

The following field is used for Forward Monitoring cells:

- **Block Error Detection Code ($BEDC_{0+1}$)**—Even parity over the payload of the block of user cells transmitted since the last Forward Monitoring cell.

The following fields are used for Backward Reporting cells:

- **Total Received Cell Count 0 ($TRCC_0$)**—Indicates the total number of CLP = 0 user cells (modulo 65,536) received before the Forward Monitoring cell was received.
- **Block Error Result (BLER)**—Carries the number of errored parity bits detected by the BEDC of the received Forward Monitoring cell.
- **Total Received Cell Count 0+1 ($TRCC_{0+1}$)**—Indicates the total number of user cells (modulo 65,536) received before the Forward Monitoring cell was received.

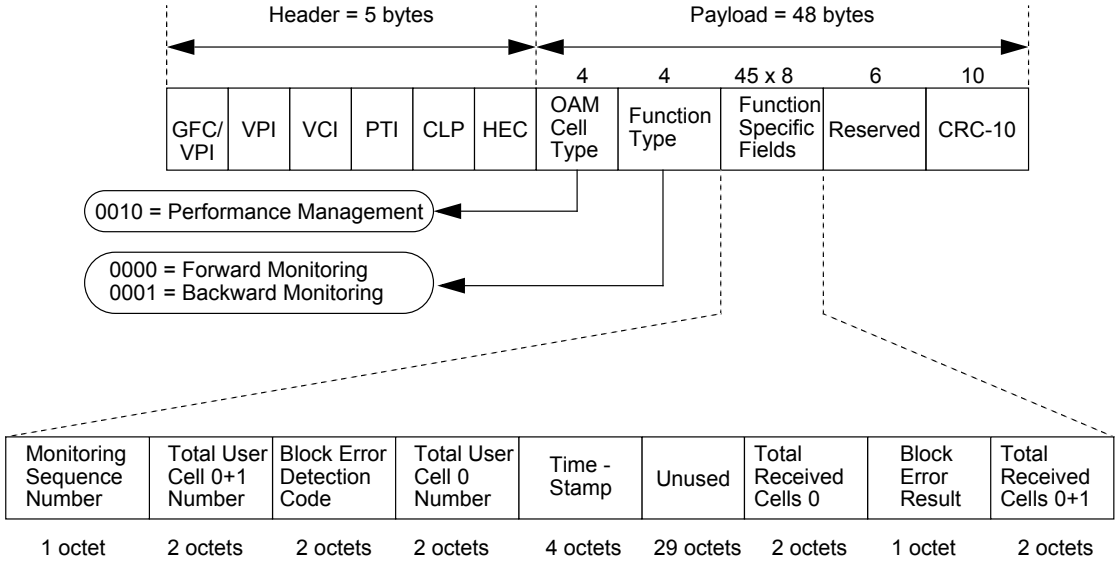


Figure 6-16. Performance Management Cell

6.3.7 Performance Monitoring

Performance monitoring of a connection is accomplished by monitoring blocks of cells sent between end points of connections or segments. Blocks of cells are delimited by Forward Monitoring cells. Each Forward Monitoring cell contains statistics about the immediately preceding block of cells. When an end point receives a Forward Monitoring cell, the statistics that the end point generated locally across the same block are added to produce a Backward Reporting cell which is returned to the opposite end point. The MC92501 supports simultaneous bidirectional block tests on all connections. When running a bidirectional test, Forward Monitoring cells are generated for one direction and checked for the other direction. At the end point of a VPC, where the VCCs are accessible, Performance Monitoring (PM) cell generation is supported on either the individual VCCs or on the VPC as a whole, but not both simultaneously (see **Appendix D**). At a connection end point PM cell generation is supported on the connection or on a segment, but not both simultaneously. In other words an individual user information cell may belong to at most one active PM test. This limitation is imposed due to the potential complexity of multiple PM calculations during the processing of a single cell.

To run a block test, initialize the static bits of the OAM Table at both the originating and terminating points. This defines the type of test to be run. The MCSN, TUC, BEDC, and TUC0 fields of the OAM Table are also initialized, normally to 0. To start the block test, the microprocessor inserts the first Forward Monitoring Cell (FMC) using Format V of the cell descriptor (see Section 7.4.1.1 Cell Descriptor). When this cell is processed at the originating point, the Test_Running bit of the OAM Table entry is set, indicating that the block test has begun. As long as the Test_Running bit is set, all cells involved in the block test (user cells and those with reserved values of VCI or PTI that are not excluded by the

Performance Monitoring Exclusion Register (PMER)) are counted in the TUC_{0+1} field and included in the BEDC calculation. CLP = 0 cells are also counted in the TUC_0 field. Stop the block test at any time by resetting the Test_Running bit.

Each time the value of the TUC_{0+1} field reaches a multiple of the block size, the MC92501 stores the fact that an FMC must be inserted in the same direction. At the next insertion opportunity, an FMC is generated using the current MCSN, TUC, BEDC, and TUC0 fields of the OAM Table entry and inserted into the cell flow. In order to meet the requirement of inserting a Forward Monitoring cell within one-half of the block size on end-to-end block tests, the MC92501 must have enough opportunities to insert the FMCs. The insertion opportunities are limited by the insertion Leaky Bucket parameters (see **Section 5.2.6** Ingress Cell Insertion and **Section 5.3.3** Egress Cell Insertion). Additionally, the insertion opportunities in the Ingress cell flow are limited by the number of empty slots (see Section 3.3.2 Maintenance Slot Parameters).

If an end-to-end FMC is not inserted within one-half of the block size and another user cell is processed, the *FMC Queue End-to-end Overflow (FQEO)* bit of the Interrupt Register is set. This is a warning only, and the FMC is still inserted when the opportunity arises. In the event that any FMC is not inserted for an entire block and another user cell arrives, the *FMC Queue Overflow (FQO)* bit of the Interrupt Register is set. This indicates that there is an FMC missing from the block test because only one FMC is generated to cover two blocks. The CLP bit of internally generated FMCs is taken from the FCLP bit of the OAM Table. This provides the programmability required by Bellcore [see Reference 10, **Appendix G**]. The function-specific fields of generated FMCs are coded when the cell is actually inserted as follows:

- **Monitoring Cell Sequence Number (MCSN)—** The MCSN field of the OAM Table is inserted in this field and then incremented.
- **Total User Cell 0+1 Number (TUC_{0+1})—** The TUC field of the OAM Table is inserted in this field.
- **Block Error Detection Code (BEDC)—** The BEDC field of the OAM Table is inserted in this field and then cleared.
- **Total User Cell 0 Number (TUC_0)—** The TUC0 field of the OAM Table is inserted in this field.
- **Time-Stamp (TSTP)—** This field is coded either with the default value of all 1's or with the current MC92501 cell time, depending on the value of the *FMC Time-Stamp Enable (FTM)* bit of the ATMC CFB Configuration Register (ACR).
- The remaining 34 octets of the function-specific fields are coded with $6A since they are unused in Forward Monitoring cells.

As each FMC arrives at the terminating end point, the Block Error Result is calculated in accordance with ATM standards [see Reference 2, **Appendix G**]. It is placed in the payload of an OAM Fields Template (as described in **Section 7.4.2.1.5**). The TUC and TUC0 fields of the template are taken from the payload of the FMC. The TRCC and TRCC0 fields of the template are taken from the TUC and TUC0 fields of the OAM Table entry. Then the OAM Fields Template is sent to the Cell Extraction Queue. The FMC is removed from the cell flow at the connection/segment OAM termination point.

The microprocessor may use the payload of the OAM Fields Template structure in preparing a cell for insertion (see Section 7.4.1 Inserted Cell for the inserted cell structure options). If one of the "payload generation from OAM fields template" options for the inserted BRC is chosen, the MC92501 generates the payload of a Backward Reporting cell and inserts the BRC in the backward flow.

The function-specific fields of generated BRCs are coded when the cell is actually inserted as follows:

- **Monitoring Cell Sequence Number (MCSN)**—The BMCSN field of the OAM Table is inserted in this field and then incremented.
- **Total User Cell 0+1 Number (TUC_{0+1})**—The TUC field of the OAM Fields Template is inserted in this field.
- **Block Error Detection Code (BEDC)**—This field is coded with $6A.
- **Total User Cell 0 Number (TUC_0)**—The TUC0 field of the OAM Fields Template is inserted in this field.
- **Time-Stamp (TSTP)**—This field is coded either with the default value of all 1's or with the current MC92501 cell time, depending on the value of the *FMC Time-Stamp Enable (FTM)* bit of the ATMC CFB Configuration Register (ACR).
- **Total Received Cell Count 0 ($TRCC_0$)**—The TRCC0 field of the OAM Fields Template is inserted in this field.
- **Block Error Result (BLER)**—The BLER field of the OAM Fields Template is inserted in this field.
- **Total Received Cell Count 0+1 ($TRCC_{0+1}$)**—The TRCC field of the OAM Fields Template is inserted in this field.
- The remaining 29 octets of the function-specific fields are coded with $6A since they are unused in Backward Reporting cells.

The results of the block test are collected at data storage points. The storage point may be the originating point, the terminating point, or any point in between. In support of data storage points, the MC92501 provides options to copy FMCs and BRCs on a connection basis. The Context Parameter Table bits that control these options are listed in **Table 6-9**. BRCs are removed from the cell flow at the originating end point.

Table 6-9. Performance Monitoring Bits

Logical Name	Bit Name	Table	Stat/ Dyn	Used By	Explanation
Copy Segment FMCs	ECSF ICSF	Context Parameters	S	Egress/ Ingress	Copy received Segment FMC to Cell Extraction Queue
Copy End-to-end FMCs	ECEF ICEF	Context Parameters	S	Egress/ Ingress	Copy received End-to-end FMC to Cell Extraction Queue
Copy segment BRCs	ECSB ICSB	Context Parameters	S	Egress/ Ingress	Copy received Segment BRC to Cell Extraction Queue
Copy end-to-end BRCs	ECEB ICEB	Context Parameters	S	Egress/ Ingress	Copy received End-to-end BRC to Cell Extraction Queue

An alternative method for collecting the results of the PM test is to define an OAM entry at an intermediate point. This point is treated like the terminating point described above, except that the FMC is not removed from the cell flow. When an FMC is received, the calculation of the BLER is performed at this point also, and an OAM Fields Template containing the BRC fields is sent to the Cell Extraction Queue. Since Performance Monitoring tests are run on a small fraction of the connections and each test requires considerable storage, a separate OAM Table is defined in External Memory for storing the fields needed to support active PM tests. The Context Parameters Table records contain pointers to the records of the OAM Table. The OAM_PTR is valid only if the *Ingress OAM Pointer is Valid (IPOV)* or *Egress OAM Pointer is Valid (EPOV)* bit is set. **Table 6-10** describes the fields of the OAM Table.

Table 6-10. OAM Table Fields

Logical Name	Field Name	# bits	Stat/ Dyn	Meaning
FMC_Gen	FMCG	1	S	This is the originating point. Generate FMCs when necessary.
Connection_Identifier	ECI/ICI	16	S	Connection identifier of connection on which PM is being performed. See Appendix D for an explanation of the use of this field when VCCs are bundled.
F4_level	F4	1	S	Defines the level of the block test
BT_SEG/E2E	SEG	1	S	Defines the scope of the block test
FMC CLP bit	FCLP	1	S	Is used as the CLP bit of generated FMC cells
Block_Size	BLK	2	S	Encodes the block size (128, 256. 512, 1024)
Test_Running	TR	1	D	Set when an FMC is processed.
MCSN	MCSN	8	D	Monitoring Cell Sequence Number
TUC	TUC	16	D	Total User Cell count
BEDC	BEDC	16	D	Block Error Detection Code (BIP-16)
TUC0	TUC	16	D	Total User Cell (CLP=0) count
BRC MCSN	BMCSN	8	D	Monitoring Cell Sequence Number for Backward Reporting Cells
Last_MCSN	LMCSN	8	D	used by the MC92501 to store the MCSN from the previous FMC
TUC_Difference	TUCD	16	D	Used by the MC92501 to store the difference between the received TUC and the local TUC (TRCC)

Figure 6-17 shows the flow of a block test on a VPC segment where the segment endpoints are outside, thereby including the switch in the test.

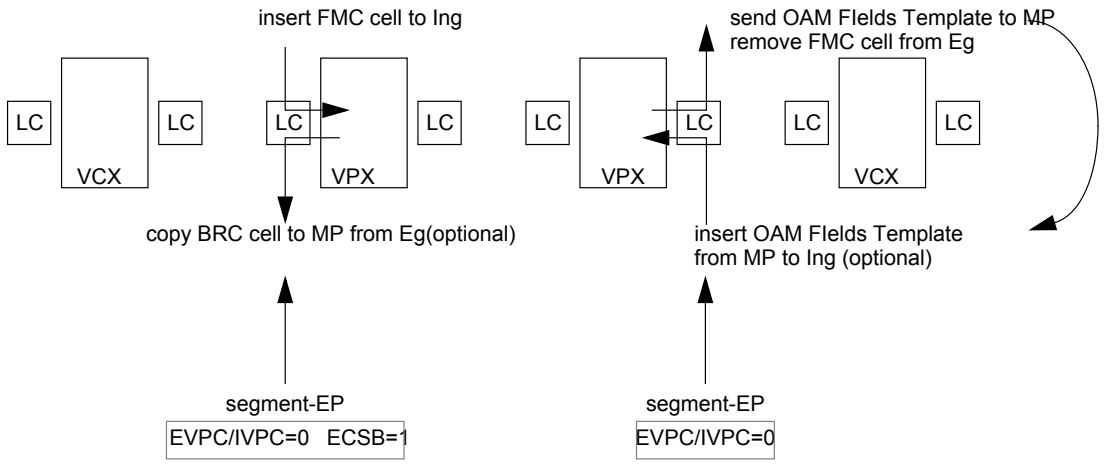


Figure 6-17. Performance Management Block Test on a VPC Segment

Figure 6-18 shows the flow of a block test on a VCC using an intermediate data collection point. The endpoints of the F5 flow have been defined outside the switches.

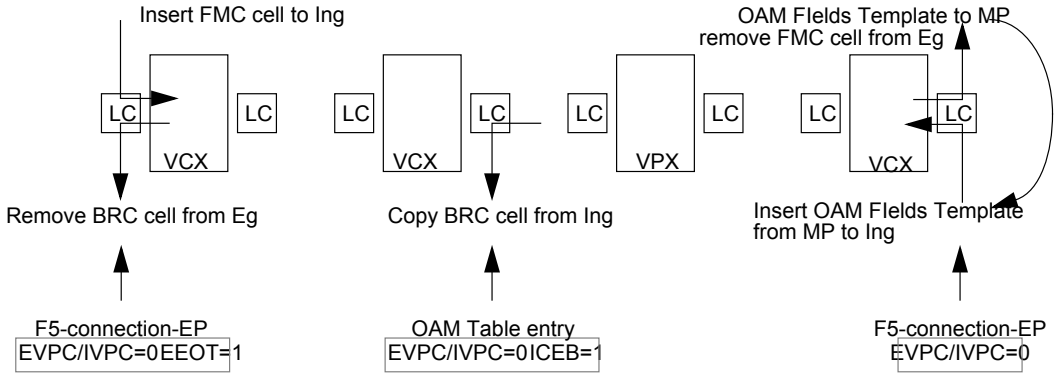


Figure 6-18. Performance Management Block Test on a VCC

Figure 6-19 shows the F4 flows for the case where the VP connection end points are logically at the edges of the VCX switches at which a set of VCCs are collected to be a VPC.

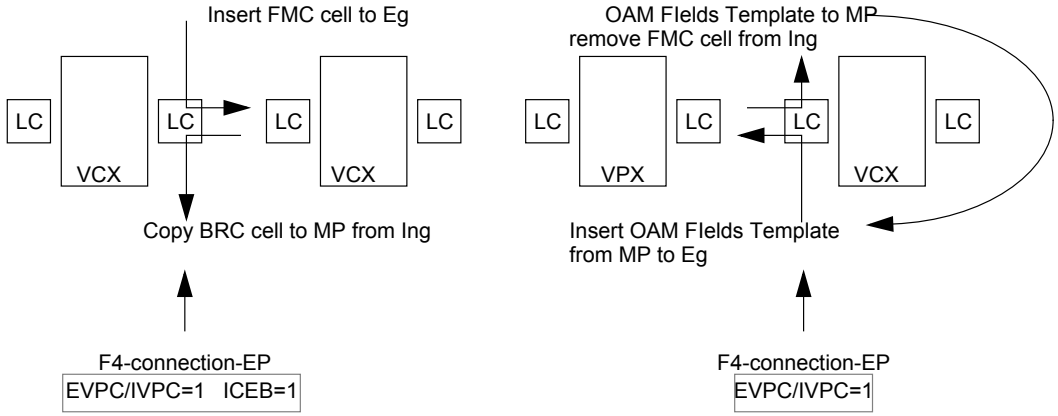


Figure 6-19. F4 PM Block Test on a VPC Internal to the Network

Figure 6-20 shows the F4 flows for the case where the VP connection is routed to the user end point through a VPX. In this case the connection endpoint must be on the LC board closest to the user.

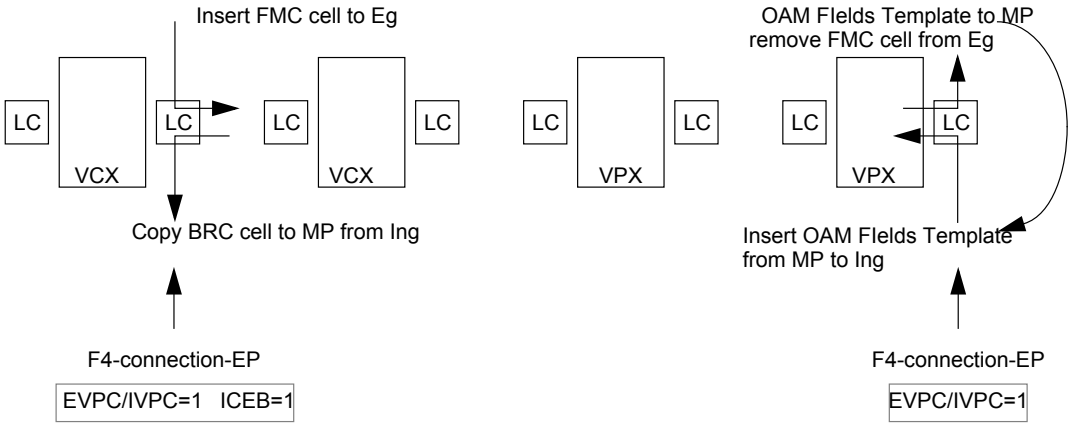


Figure 6-20. F4 PM Block Test on a VPC that Crosses the UNI

6.3.8 Activation/Deactivation OAM Cells

The MC92501 does not provide direct activation or deactivation of OAM features. Instead, Activation/Deactivation OAM cells are generated by the microprocessor. The MC92501 simply inserts these cells into one of the cell flows (Ingress or Egress), as directed. This procedure allows the microprocessor to maintain tight control of the OAM. Received Activation/Deactivation cells may be copied to the microprocessor at any point by setting the appropriate Copy Other OAM bit. Setting the ECOT/ICOT bit

in order to copy Activation/Deactivation cells at points other than the endpoint could be useful for intermediate points that are collecting Performance Monitoring (PM) data and need to know when the PM test has been deactivated. All received Activation/ Deactivation OAM cells are removed from the cell flow at the segment/connection endpoint. Some examples of common Activation/Deactivation flows are shown in the following figures.

Figure 6-21 shows the F4 flows for the case where the VP connection end points are logically at the edges of the VCX switches at which a set of VCCs are collected to be a VPC.

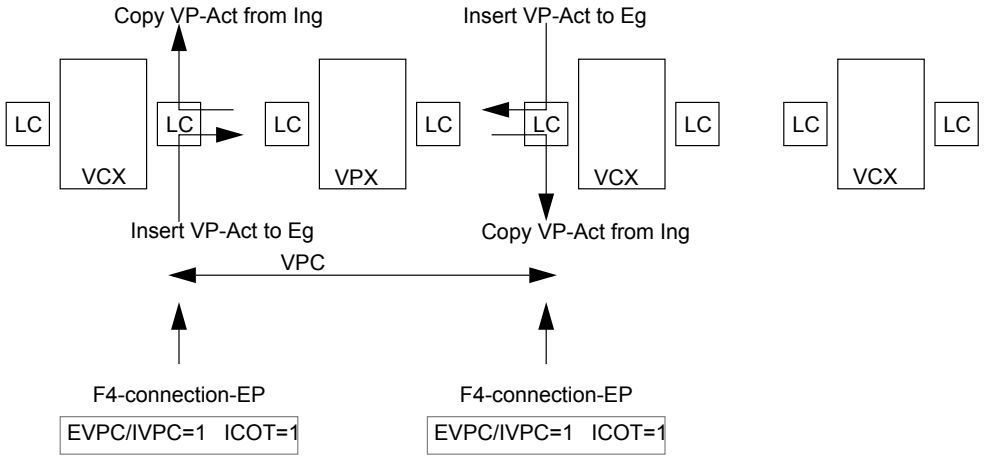


Figure 6-21. F4 OAM Flow for a VPC Internal to the Network

Figure 6-22 shows the F4 flows for the case where the VP connection is routed to the user end point through a VPX. In this case the connection endpoint must be on the LC board closest to the user.

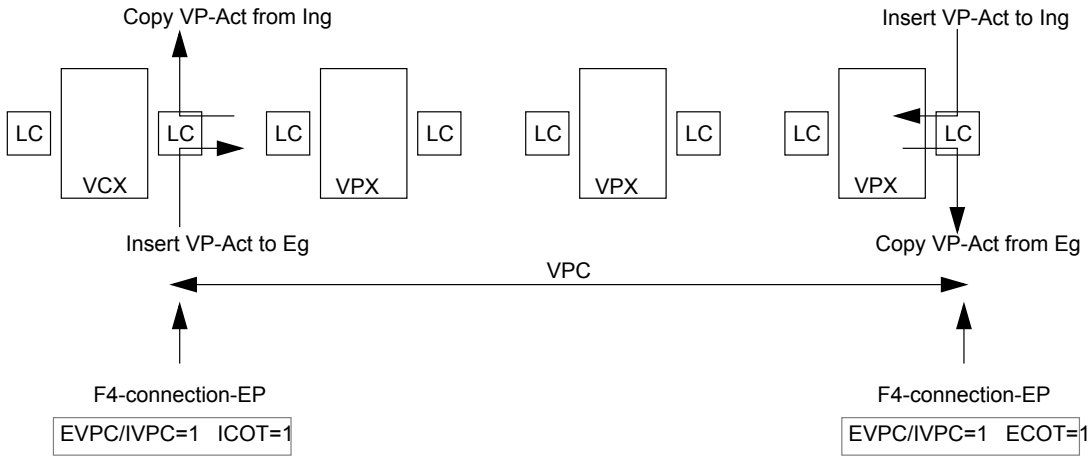


Figure 6-22. F4 OAM Flow for a VPC that Crosses the UNI

Figure 6-23 shows the F5 flows for a VCC that has one end point within the network and the other termination point outside the network. Therefore, the endpoint on the right is outside.

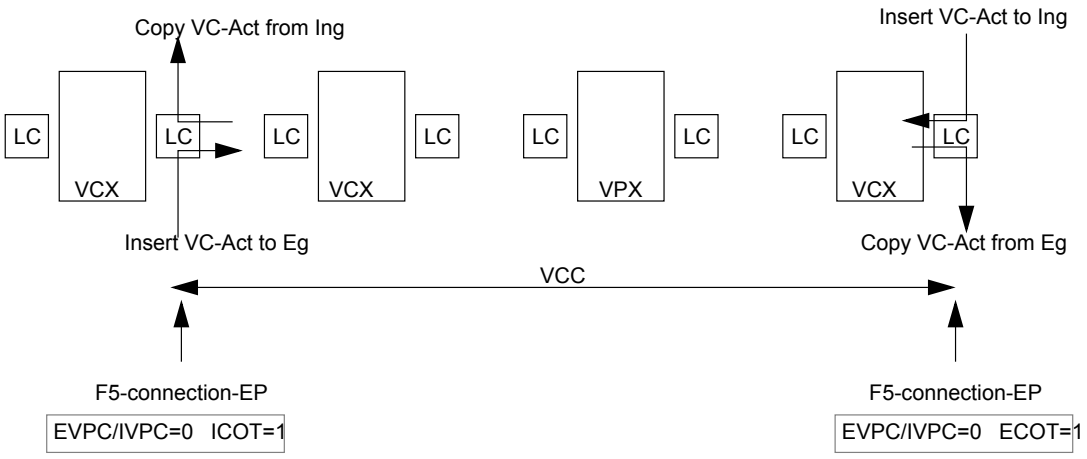


Figure 6-23. F5 OAM Flow

6.4 ABR Support

The MC92501 provides a full Available Bit Rate (ABR) solution for 'switch-behavior' relative-rate-marking and EFCI marking in accordance with TM-4.0. It also provides hooks for the switch-fabric in order to give higher priority for Resource Management (RM) cells. Following is a list of features:

- Performs RR marking on FRM and/or BRM cells, on selected connections. Enabled by either control registers or by fields that it gets from the overhead of cells which are received from the switch-fabric.
- Performs EFCI marking on non RM cells whose PTI[2]=0, on selected connections. Enabled by either control registers or by fields that it gets from the overhead of cells which are received from the switch-fabric.
- Resets EFCI on non RM cells whose PTI[2]=0, on selected connections.
- Checks CRC on received RM cells and generates CRC for transmitted RM cells in both the Ingress and the Egress flow.
- Provides different priority to RM cells.
- Can copy RM cells to the microprocessor or remove them from the flow.

6.4.1 RM Cell Definition

A cell is an RM cell if and only if at least one of the following conditions are met:

- The cell belongs to a VC connection and its PTI = 6.
- The cell belongs to a VP connection, its VCI = 6 and its PTI = 6.
- The cell belongs to a VP connection, its VCI = 6 and the *VP RM Cell PTI (VPRP)* bit is set.

6.4.2 RM Cell Fields

Figure 6-24 shows the RM Cell Fields. These fields include:

- PID = 1
- DIR = Direction (0 = Forward RM cell; 1 =Backward RM cell)
- BN = Backward Explicit Congestion Notification (0 = Generated by source; 1 = Not generated by the source)
- CI = Congestion Indication
- NI = No Increase Bit.
- ER = Explicit Rate
- CCR = Current Cell Rate
- MCR = Minimum Cell Rate
- CRC-10

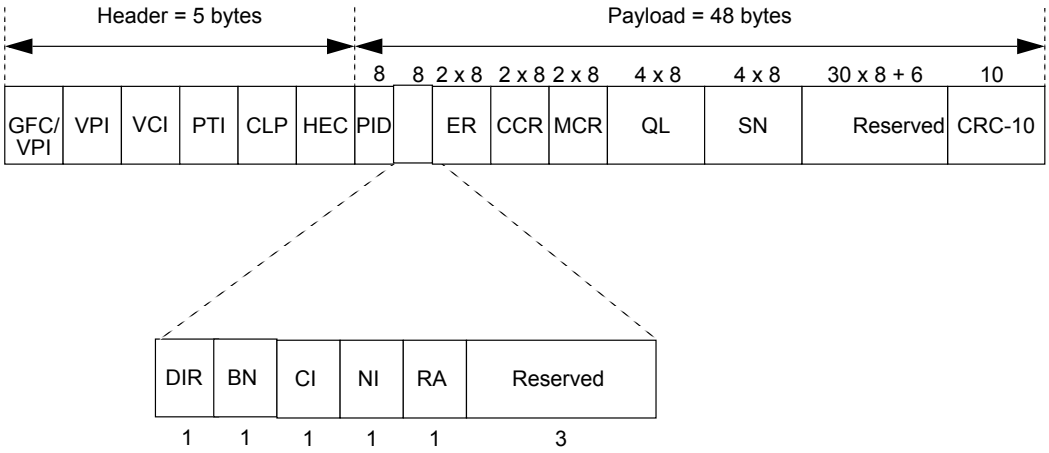


Figure 6-24. RM Cell Fields

6.4.3 Cell Marking (CI, NI, PTI)

Figure 6-25 illustrates two MC92501 devices connected to a switch fabric. For the sake of simplicity, consider an ABR flow that goes from left to right. This means that data cells are flowing from left to right, Forward Resource Management (FRM) cells are flowing from left to right, and Backward Resource Management (BRM) cells are flowing from right to left. The switch marks FRM and User cells flowing downstream and BRM cells flowing upstream. This switch function can be implemented in the Ingress of MC92501 #1 and in the Egress of MC92501 #2. MC92501 #1 marks cells because of the Ingress flow status (e.g., Ingress flow congestion) while MC92501 #2 marks cells because of the Egress flow status.

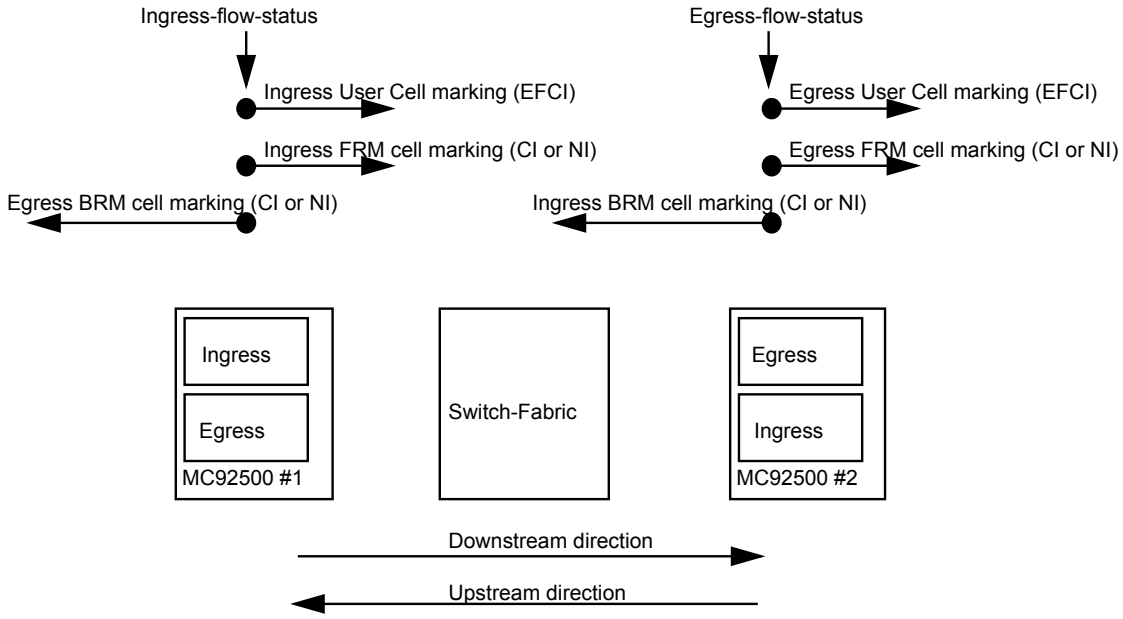


Figure 6-25. MC92501 to Switch Connections

The MC92501 can take the following actions in response to the Ingress flow status:

- Perform EFCI marking on Ingress cells (i.e., set PTI[1] bit in cells on which PTI[2] = 0).
- Set CI or NI in Ingress FRM cells.
- Set CI or NI in Egress BRM cells.

The MC92501 can take the following actions in response to the Egress flow status:

- Perform EFCI marking on Egress cells (i.e., set PTI[1] bit in cells on which PTI[2] = 0).
- Set CI or NI in Egress FRM cells.
- Set CI or NI in Ingress BRM cells.

Figure 6-26 presents an overview of the MC92501 marking scheme.

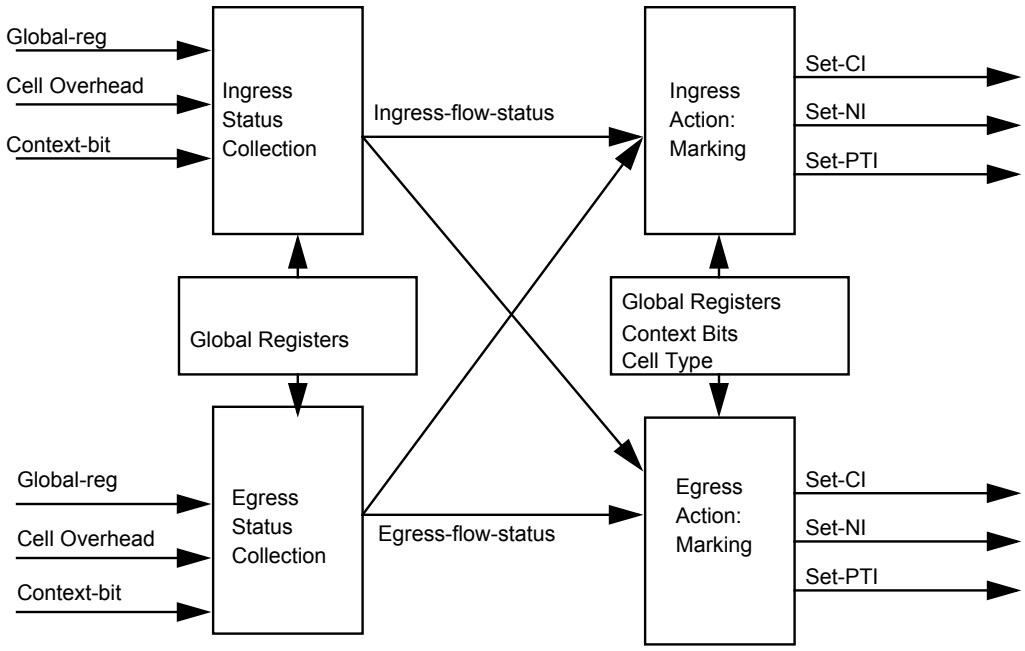


Figure 6-26. MC92501 Marking Scheme

This scheme shows that there are various ways to inform the MC92501 that it should mark a cell due to the Ingress flow status or the Egress flow status. It also shows that the status of the Ingress flow, the status of the Egress flow, global registers, a context bit and the cell type impact the decision of setting CI, NI and PTI.

6.4.3.1 Sources for Ingress Flow Status

The Ingress flow status is gathered from the following three sources:

- **Global Register**—The switch-fabric can notify the MC92501 to mark cells because of Ingress flow status by setting the *Global Ingress ABR Mark Enable (IAME)* bit in the Ingress Processing Control Register (IPLR)—see **Section 7.2.5.18**. In this case, the MC92501 can be programmed to mark cells in the Ingress and/or in the Egress.
- **Cell Overhead**—The switch-fabric can notify the MC92501 to mark cells because of the Ingress flow status of connection #n by setting the *Overhead Ingress Flow Status (IFS)* bit in the overhead of Egress cells belonging to that connection. The bit location in the overhead is programmed using the *IFS Byte Location (EIBY)* bit and the *IFS Bit Location (EIBI)* bit in the Egress Switch Overhead Information Register 1 (ESOIR1)—see **Section 7.2.6.8**. This bit is enabled by the *Global IFS Enable (EIAS)* bit in the Egress Switch Interface Configuration Register (ESWCR)—see **Section 7.2.6.6**. In this case, the MC92501 can be programmed to mark Egress BRM cells.
- **Context Bit**—The switch-fabric can notify the MC92501 to mark cells because of the Ingress flow status of connection #n by setting the *Overhead Ingress Flow Status (IFS)* bit in the overhead of Egress cells belonging to that connection. The bit

location in the overhead is programmed using the *IFS Byte Location (EIBY)* bit and the *IFS Bit Location (EIBI)* bit in the Egress Switch Overhead Information Register 1 (ESOIR1)—see **Section 7.2.6.8**. This bit is enabled by the *Global IFS Enable (EIAS)* bit in the Egress Switch Interface Configuration Register (ESWCR)—see **Section 7.2.6.6**. When the MC92501 receives the cell, it copies the bit into the *Connection Ingress Flow Status (CIFS)* bit in the Common Parameters Extension Word of connection #n. In this case, the MC92501 can be programmed to mark Ingress FRM cells or perform EFCI-marking.

Figure 6-27 shows the Ingress-flow-status logic.

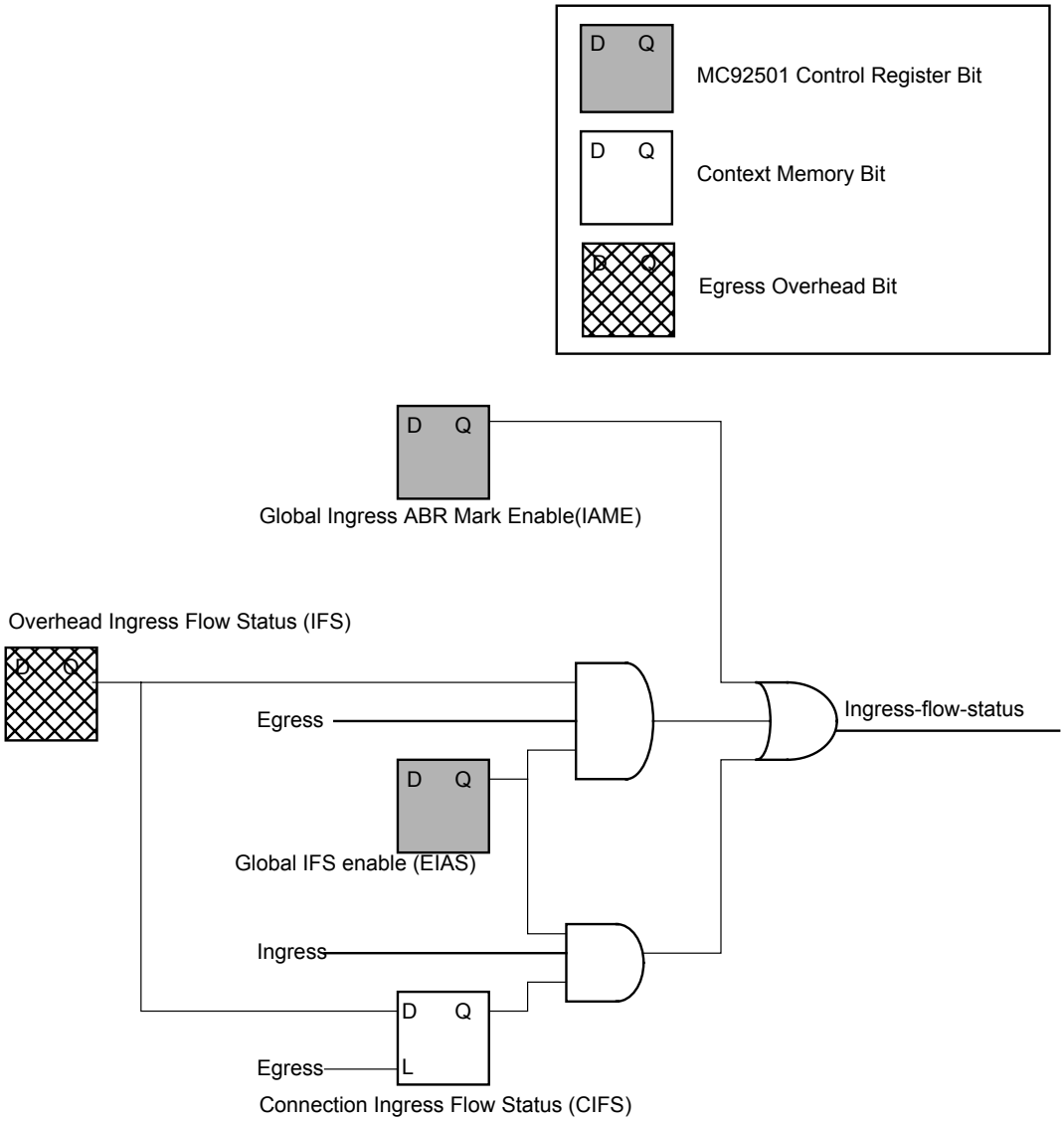


Figure 6-27. Ingress-Flow-Status Logic

6.4.3.2 Sources for Egress Flow Status

The Egress flow status is gathered from the following 3 sources:

- **Global Register**—The switch-fabric can notify the MC92501 to mark cells because of the Egress flow status by setting the *Global Egress ABR Mark Enable (EAME)* bit in the Egress Processing Control Register (EPLR). The MC92501 can be programmed to mark cells in the Ingress and/or in the Egress.
- **Cell Overhead**—The switch-fabric can notify the MC92501 to mark cells because of the Egress flow status of connection #n by setting the *Overhead Egress Flow Status (EFS)* bit in the overhead of Egress cells belonging to that connection. The location of this bit in the overhead is programmed using the *EFCI Byte Location (EFBY)* bit and the *EFCI Bit Location (EFBI)* bit in the Egress Switch Overhead Information Register 1 (ESOIR1)—see **Section 7.2.6.8**. This bit is enabled by the *Global EFS Enable (EEAS)* bit in the Egress Switch Interface Configuration Register (ESWCR)—see **Section 7.2.6.6**. In this case, the MC92501 can be programmed to mark Egress FRM cells or perform EFCI-marking.
- **Context Memory**—The switch-fabric can notify the MC92501 to mark cells because of the Egress flow status of connection #n by setting the *Overhead Egress Flow Status (EFS)* bit in the overhead of Egress cells belonging to that connection . When the MC92501 receives that cell, it copies the bit into the *Connection Egress Flow Status (CEFS)* bit in the Common Parameters Extension Word of connection #n. The MC92501 can be programmed to mark Ingress BRM cells.

Figure 6-28 shows the Egress-flow-status logic.

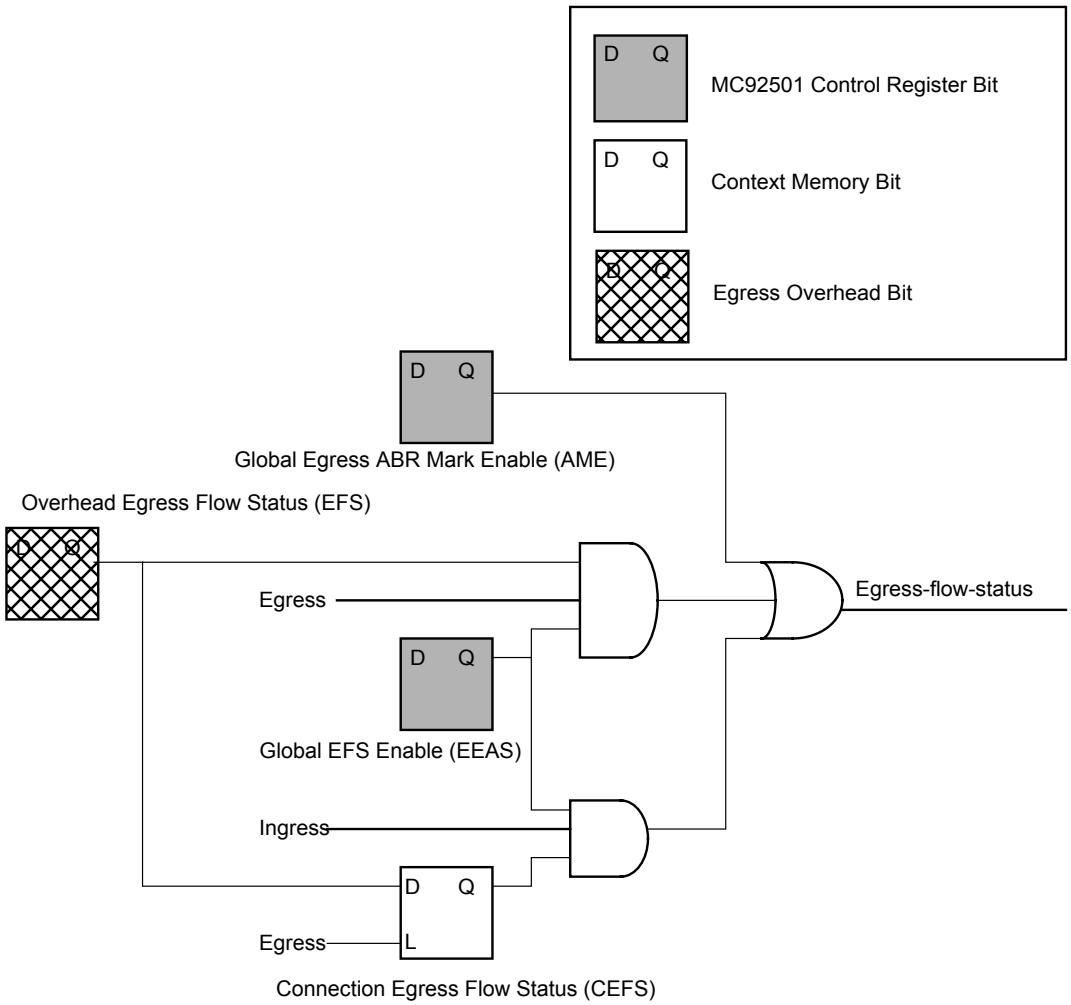


Figure 6-28. Egress-Flow-Status Logic

6.4.3.3 MC92501 Ingress Direction Actions

Figure 6-29 illustrates the actions taken by the MC92501 in the Ingress direction. The MC92501 can mark cells as a result of either Ingress-flow-status or Egress-flow-status.

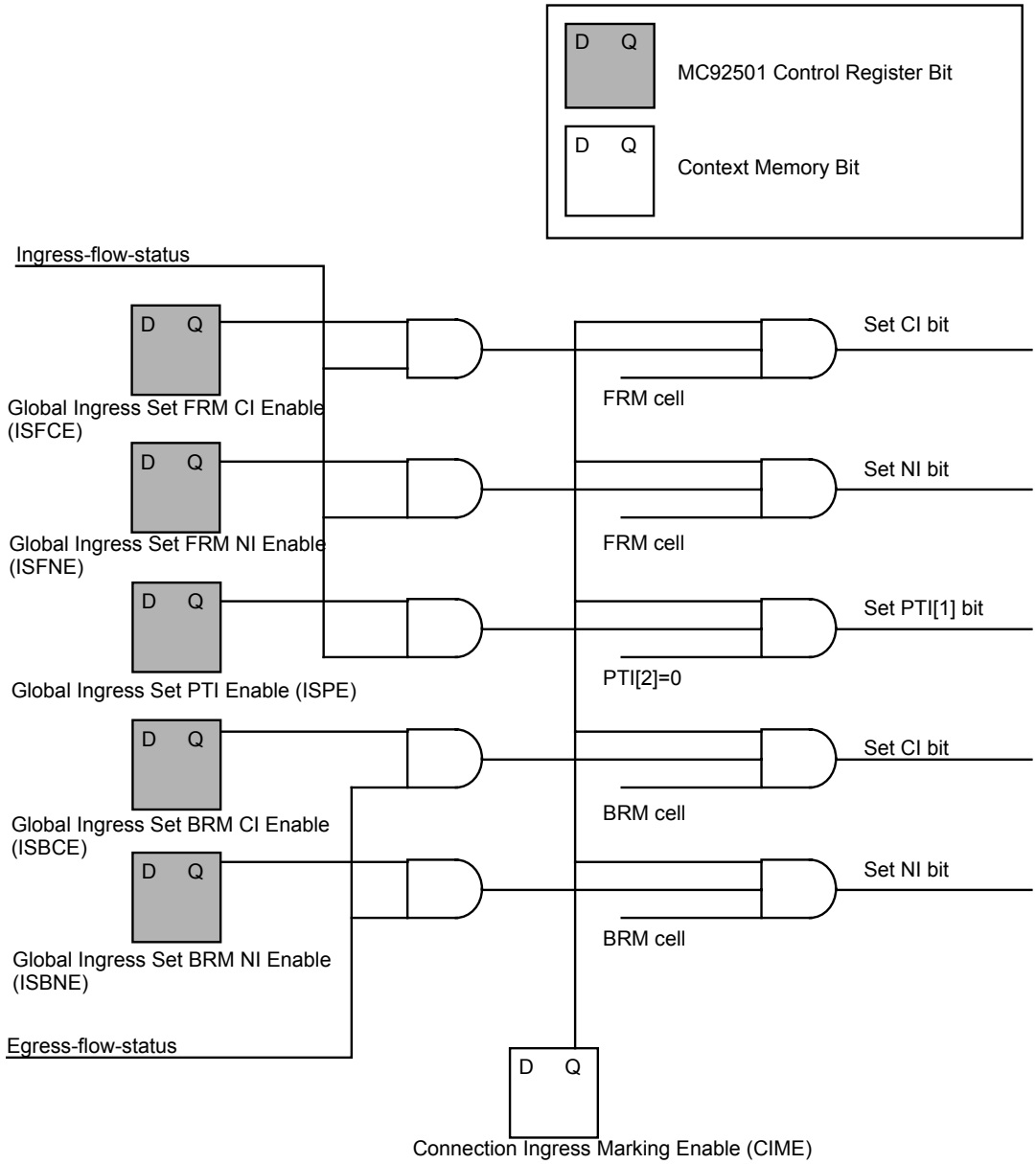


Figure 6-29. Ingress Direction Actions

In the case where the Ingress-flow-status is asserted, the MC92501 can perform one or more of the following:

- Set the CI bit in an Ingress FRM cell when the *Global Ingress Set FRM CI Enable (ISFCE)* bit in the Ingress Processing Configuration Register (IPCR) is set.
- Set the NI bit in an Ingress FRM cell when the *Global Ingress Set FRM NI Enable (ISFNE)* bit in the Ingress Processing Configuration Register (IPCR) is set.
- Set the PTI[1] bit in an Ingress cell whose PTI[2] = 0 when the *Global Ingress Set PTI Enable (ISPE)* bit in the Ingress Processing Configuration Register (IPCR) is set.

In the case where the Egress-flow-status is asserted, the MC92501 can perform one or more the following:

- Set the CI bit in an Ingress BRM cell when the *Global Ingress Set BRM CI Enable (ISBCE)* bit in the Ingress Processing Configuration Register (IPCR) is set.
- Set the NI bit in an Ingress BRM cell when the *Global Ingress Set BRM NI Enable (ISBNE)* bit in the Ingress Processing Configuration Register (IPCR) is set.

All cell marking on the Ingress is enabled on a per-connection basis by the *Connection Ingress Marking Enable (CIME)* bit in the Common Parameters Extension Word**.**

6.4.3.4 MC92501 Egress Direction Actions

Figure 6-30 illustrates the actions taken by the MC92501 in the Egress direction. The MC92501 can mark cells as a result of either Ingress-flow-status or Egress-flow-status. In the case where Egress-flow-status is asserted the MC92501 can perform one or more of the following:

- Set CI bit in an Egress FRM cell when the *Global Egress Set FRM CI Enable (ESFCE)* bit in the Egress Processing Configuration Register (EPCR) is set.
- Set NI bit in an Egress FRM cell when the *Global Egress Set FRM NI Enable (ESFNE)* bit in the Egress Processing Configuration Register (EPCR) is set.
- Set PTI[1] bit in an Egress cell whose PTI[2] = 0 when the *Global Egress Set PTI Enable (ESPE)* bit in the Egress Processing Configuration Register (EPCR) is set.

In the case where Ingress-flow-status is asserted the MC92501 can perform one or more the following:

- Set CI bit in an Egress BRM cell when the *Global Egress Set BRM CI Enable (ESBCE)* bit in the Egress Processing Configuration Register (EPCR) is set.
- Set NI bit in an Egress BRM cell—when the *Global Egress Set BRM NI Enable (ESBNE)* bit in the Egress Processing Configuration Register (EPCR) is set.

All cell marking on the Egress is enabled on a per-connection basis by the *Connection Egress Marking Enable (CEME)* bit in the Common Parameters Extension Word**.**

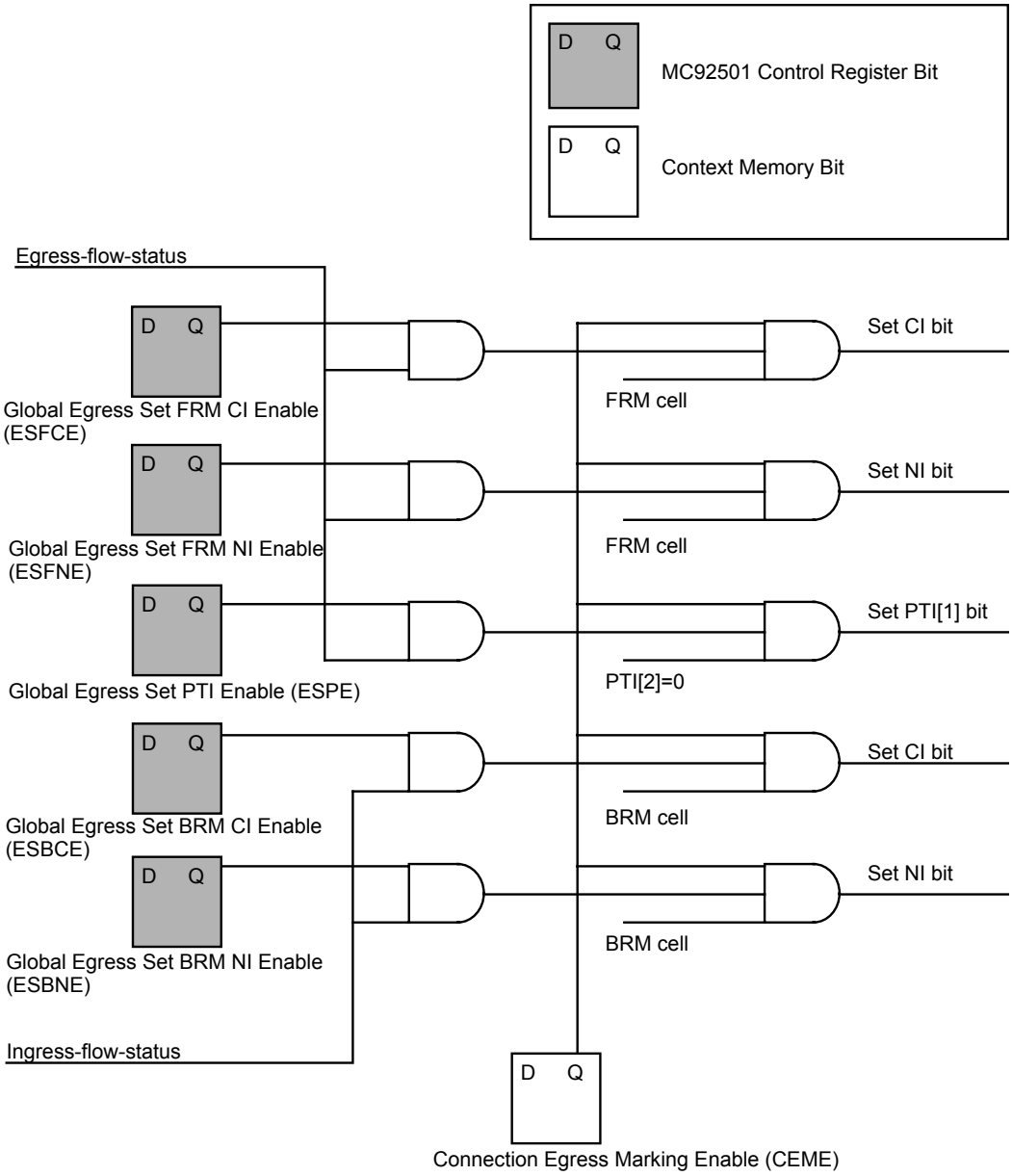
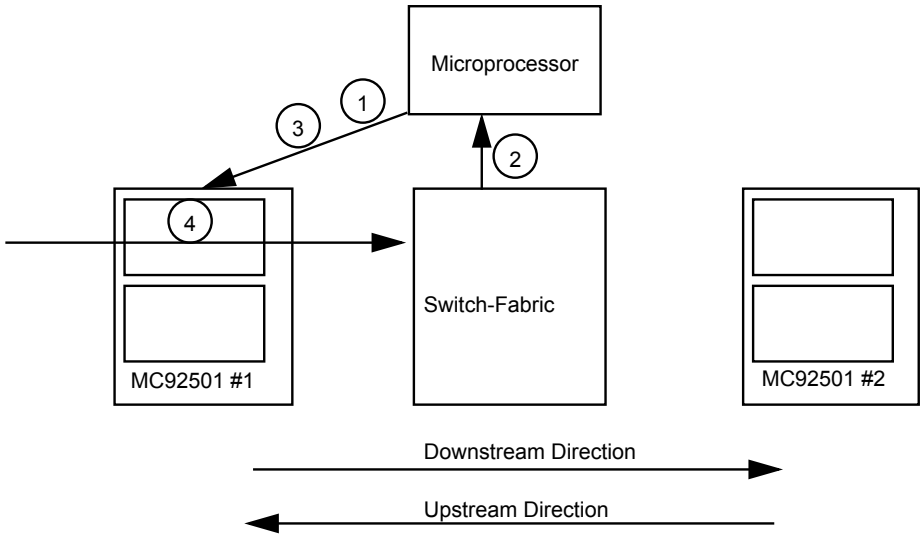


Figure 6-30. Egress Direction Actions

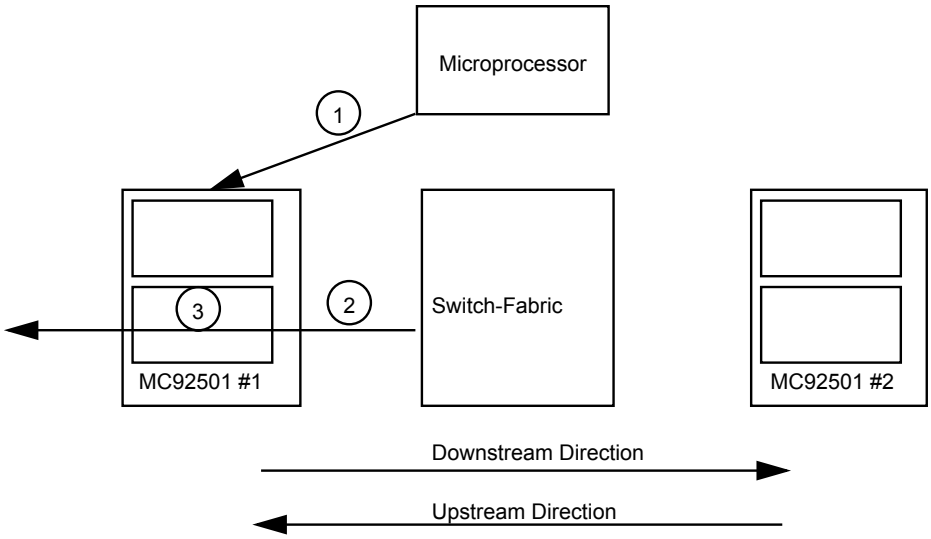
6.4.3.5 Cell Marking Examples

The following figures illustrate MC92501 cell marking examples.



1. Initially the microprocessor configures the MC92501 #1 as follows:
   - Sets the Global Ingress Set FRM CI Enable (ISFCE) bit.
   - Sets the Connection Ingress Marking Enable (CIME) bit for selected ABR connections.
2. The Switch Fabric informs the microprocessor that Ingress ABR queues have reached some limit.
3. The microprocessor sets the Global Ingress ABR Mark Enable (IAME) bit.
4. The MC92501 sets the CI bit for FRM cells belonging to the selected ABR connections.

Figure 6-31. Marking CI Bits of Ingress FRM Cells



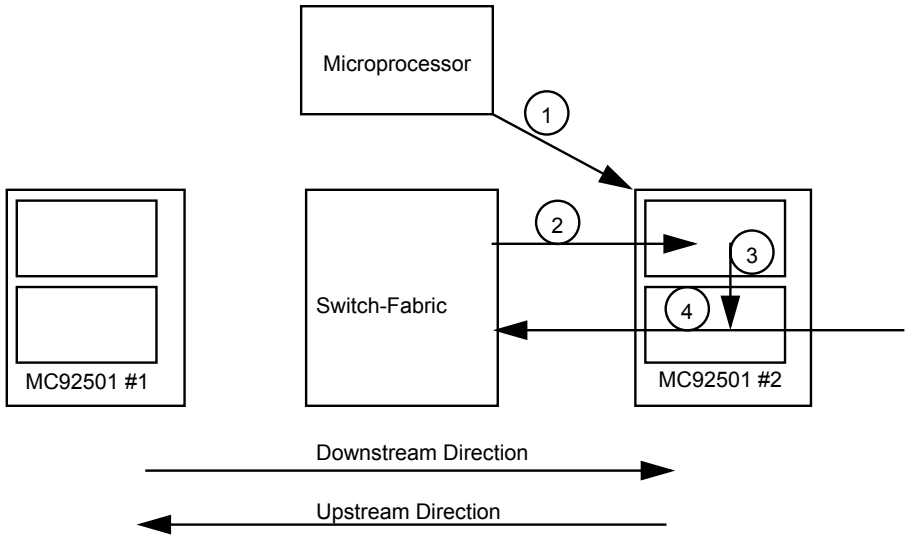
1 Initially the microprocessor configures the MC92501 #1 as follows:
- Sets the Global IFS Enable (EIAS) bit.
- Sets the Global Egress Set BRM NI Enable (ESBNE) bit.
- Programs the location of the Overhead Ingress Flow Status (IFS) bit by writing to the IFS Byte Location (EIBY) and the IFS Bit Location (EIBI) fields.

On connection setup the microprocessor configures the MC92501 #1 as follows:
- Sets the Connection Egress Marking Enable (CEME) bit for selected ABR connections.

2 The Switch detects that the Ingress queue of connection #n has reached a limit. It sets Ingress Flow status bit on the overhead of cells belonging to that connection.

3 The MC92501 sets the NI bit of BRM cell belonging to connection #n.

Figure 6-32. Marking a BRM Cell NI Field



1 Initially the microprocessor configures the MC92501 #2 as follows:
- Sets the Global EFS Enable (EEAS) bit.
- Sets the Global Ingress Set BRM CI Enable (ISBCE) bit.
- Programs the location of the Overhead Egress Flow Status (EFS) bit by writing to the EFCI Byte Location (EFBY) and the EFCI Bit Location (EFBI) fields.

On connection setup the microprocessor configures the MC92501 #2 as follows:
- Sets the Connection Egress Marking Enable (CEME) bit for selected ABR connections.

2 The Switch detects that the Egress queue of connection #n has reached a limit. It sets Egress Flow status bit on the overhead of cells belonging to that connection.

3 The MC92501 copies the overhead Egress flow status bit into the Connection Egress Flow Status (CEFS) bit and effectively sets it.

4 The MC92501 sets the CI bit of BRM cells belonging to connection #n.

Figure 6-33. Marking the CI Bit for All Ingress BRM Cells for a Connection

6.4.4 Ingress Switch Parameters Hooks

The MC92501 defines an 8-bit field which can be overlayed on bits of the Ingress switch parameters belonging to RM cells. In applications where the overlayed switch parameter field is a priority field used by the switch fabric, RM cells can gain higher priority in passing the switch. This enables shortening the feedback loop for ABR. The MC92501 performs this field overlay if one of the following occurs:

- An Ingress BRM cell is received and both the *Ingress BRM Overlay Enable (IBOE)* bit and the *Ingress RM Overlay Enable (IROE)* bit are set.
- An Ingress FRM cell is received and both the *Ingress FRM Overlay Enable (IFOE)* bit and the *Ingress RM Overlay Enable (IROE)* bit are set.

Once the MC92501 is enabled, it uses the *RM Overlay Location (ROL)* field to locate one byte out of 12 bytes in the Ingress switch parameters. This byte is overlaid by the *RM Overlay Field (ROF)* only on bits which are enabled by the *RM Overlay Mask (ROM)* field. See **Section 7.2.6.35** for a description of the RM Overlay Register fields.

For example, if the *ROF* = 11001101, the *ROM* = 01111000, and the *ROL* = 9, then the *IBOE* bit is set and the current cell is a backward RM cell, as shown in **Figure 6-34**.

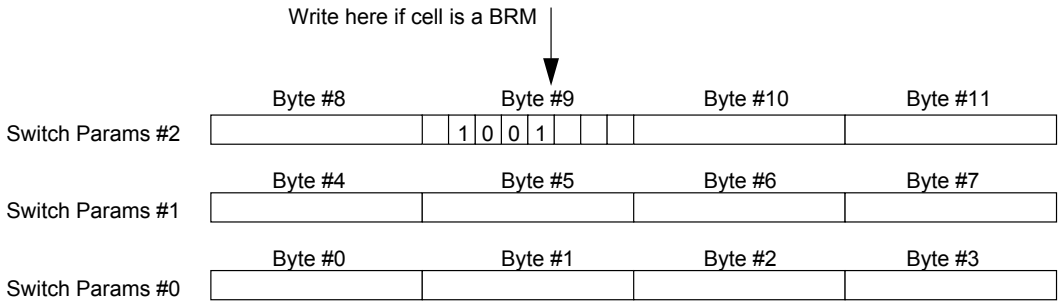


Figure 6-34. Ingress Switch Parameter Example

6.4.5 Egress Reset EFCI

The MC92501 resets PTI[1] on a cell that meets the following conditions:

- PTI[2] = 0.
- The cell is a non-RM cell.
- The *Egress Reset EFCI (EFREF)* bit in the Context Parameters Extension Table for the connection to which the cell belongs is set.

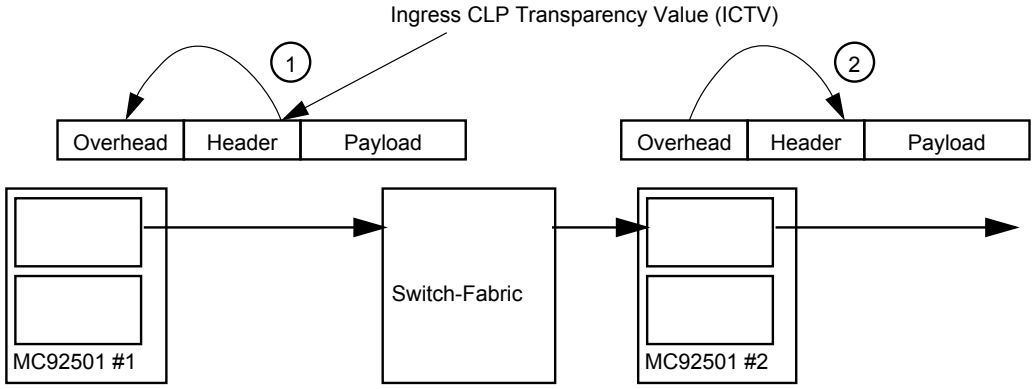
This feature can be used as part of ‘destination behavior’.

6.5 CLP Transparency Support

The Traffic Management specification defines two network operation models with relation to Cell Loss Priority (CLP) = 1 flow: CLP-Transparent and CLP-Significant. A connection which is CLP-Transparent doesn't have different Cell Loss Ratio (CLR) for CLP = 0 or CLP = 1 traffic, and therefore, doesn't prefer discarding CLP = 1 over CLP = 0 cells on congestion. Current switch fabrics do distinguish globally between CLP = 0 and CLP = 1 traffic. The latter is more susceptible to discarding in case of congestion.

The MC92501 solves the problem in the following manner:

- If a cell belongs to a connection that supports CLP transparency (i.e., the *Ingress CLP Transparency Enable (ICTE)* bit in the Common Parameters Extension Word is set), then the MC92501 moves the CLP from the cell's header to *Ingress Overhead CLP (IOCLP)* bit in the cell's overhead and assigns 0 to the header CLP. The cell is forwarded to the switch-fabric. The switch fabric deals with the cell as if CLP is 0.
- On the Egress side, the MC92501 reconstructs the header CLP from the *Egress Overhead CLP (EOCLP)* bit in the cell's overhead if enabled by the *Egress CLP Transparency Enable (ECTE)* bit.
- In order to support other applications the MC92501 function is extended. The *Ingress CLP Transparency Value (ICTV)* bit (defined in the Common Parameters Extension Word) is assigned to the cell header instead of 0.



1 If a cell belongs to a connection that supports CLP transparency (i.e., the Ingress CLP Transparency Enable (ICTE) bit in the Common Parameters Extension Word is set), then the MC92501 moves the CLP from the cell's header to Ingress Overhead CLP (IOCLP) bit in the cell's overhead and assigns 0 to the header CLP. The cell is forwarded to the switch fabric. The switch fabric deals with the cell as if CLP is 0. bit in the cell's overhead and assigns the Ingress CLP Transparency Value (ICTV) bit to the header CLP.

2 If the Connection Ingress Flow Status (CIFS) bit is set then the MC92501 reconstructs the header CLP from the On the Egress side, the MC92501 reconstructs the header CLP from the Egress Overhead CLP (EOCLP) bit in the cell's overhead if enabled by the Egress CLP Transparency Enable (ECTE) bit. bit in the cell's overhead.

Figure 6-35. CLP Transparency Example

6.6 Selective Discard Support

ATM Forum TM-4.0 defines procedures according to which cells can be discarded by network elements. A switching element may discard cells belonging to selected connections or cells whose CLP = 1 in case of congestion. This function is called selective-discard and it is implemented by the MC92501. Selective discard is enabled by the *Global Ingress Congestion Notification (ICNG)* bit in the Ingress Processing Control Register (IPLR). Selective discard is enabled on a per-connection basis by the *Connection Ingress Marking Enable (CIME)* field in the Common Parameters Extension Word. This field determines whether selective-discard is enabled and whether selective-discard is performed on CLP = 1 or on CLP = 0 + 1 traffic.

6.7 Multiple PHY Support

In order to reduce the system cost when using low-speed physical links, the MC92501 supports multiple PHY devices. Up to sixteen PHY devices can be supported by a single MC92501 as long as the total bandwidth does not exceed the MC92501's cell-processing rate (approximately 150 Mb/s at the ATM Layer). The MC92501 supports multiple PHY devices in the following ways:

- Both the Ingress and Egress PHY interfaces are defined to support multiple PHY devices. This involves additional pins to allow for transferring the number of the link to which the cell belongs.
- Per-link cell counters are provided for both the Ingress and Egress cell flows.
- The link number received from the Ingress PHY interface is used during the address compression stage. The address compression method is specified per link in the Ingress Link Registers (ILNK0 - ILNK15).
- The link number that identifies the destination link of an Egress cell is included in the connection's Connection Address word. This number is used by the Egress PHY interface to transfer the cell to the proper destination.
- The link number is included in the cell description information appended to cells that are copied to the Cell Extraction Queue.
- If Multicast Translation is performed on a link basis, the link number may be included in the calculation of the index to the Multicast Translation Table by providing it in the Multicast Translation Table Section field of the switch overhead information.

6.7.1 PHY Interfaces

When supporting multiple PHY devices, the PHY interfaces should be cell-based (i.e., the *Ingress PHY Operation Mode (IPOM)* bit of the Ingress PHY Configuration Register (IPHCR) and the *Egress PHY Operation Mode (EPOM)* bit of the Egress PHY Configuration Register (EPHCR) should be set. Both the Ingress and Egress PHY Interfaces add a bus to transfer the link number between the PHY devices and the MC92501.

The Ingress PHY Interface requires the PHY devices to arrange the priority among themselves. The chosen PHY device drives the RXPHYID bus with its link number while it transfers the cell.

The Egress PHY Interface drives the TxPHYID bus with the link number of the current or next cell, depending on the value of the *PHY ID Control (PHIDC)* bit defined in Section 7.2.6.14 General Configuration Register (GCR). Once the PHY device has acknowledged that it can receive the cell, the MC92501 transfers the cell. See Section 4.2.2 UTOPIA Level 1 Transmit PHY Interface (Egress) for details.

NOTE: It is critical that the switch provide the cells to the MC92501 in a sequence that the PHY devices can support. Otherwise, cells from an overloaded physical link may block cells from other links, and the multiple PHY devices are not properly supported. Most likely the switch needs to maintain a separate (virtual or physical) FIFO for each link as shown in **Figure 6-36**.

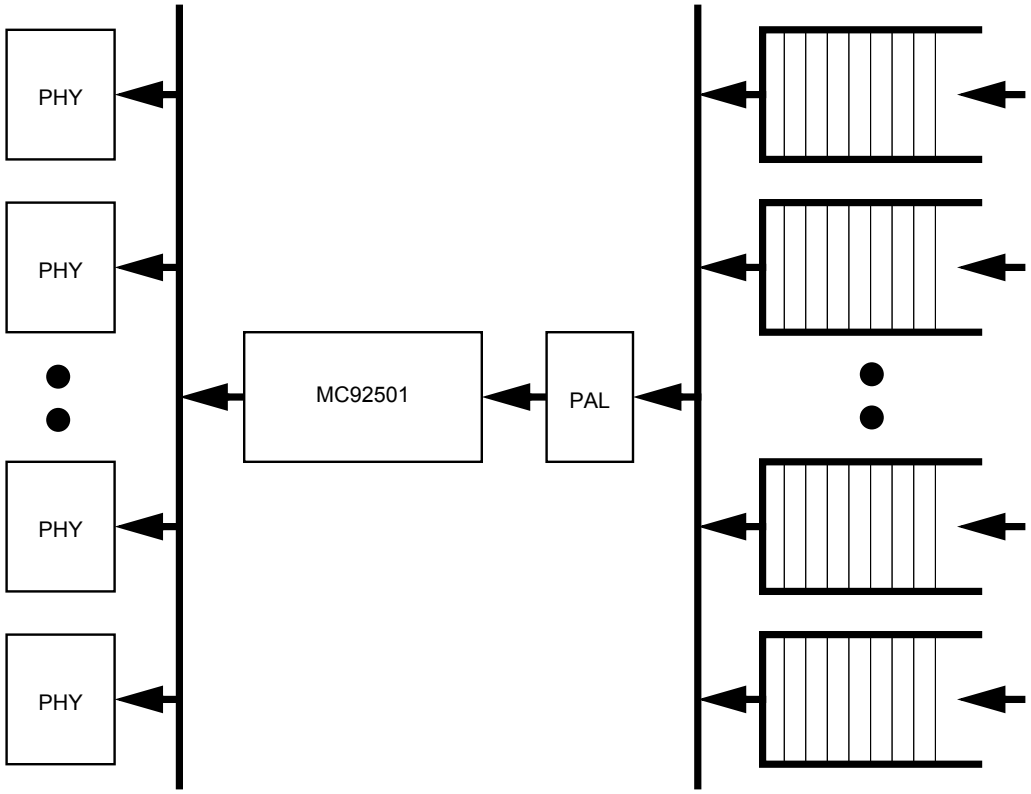


Figure 6-36. Multiple PHY Configuration on the Egress Side of the Switch

6.7.2 Cell Counters

In addition to the connection cell counters, link cell counters are provided for both the Ingress and Egress cell flows. Separate counts are maintained according to the cell classification as USER/OAM and the value of the CLP bit. Also, cells that arrive in the Ingress cell flow that do not belong to an active connection are counted separately in an Inactive Cell counter.

6.7.3 Address Compression

The MC92501 has a set of sixteen Link Registers so that the address compression parameters, including the address compression method, can be defined for each link independently. The Ingress Processing Unit uses the link number received from the PHY device as an index to the Link Registers.

6.7.4 Address Translation

Each connection supported by the MC92501 belongs to a specific link, so the link number can be considered a parameter of the connection. The link number is stored in the Connection Address word of the Context Parameters Table. The Egress Processing Unit reads the link number from the External Memory together with the new address for the cell. This number identifies the PHY device to which the cell must be transferred. It is broadcast on the TxPHYID bus when the cell is transferred as described above.

6.7.5 Cell Extraction Queue

The link number is provided together with each cell copied to the Cell Extraction Queue. (Although the link number can usually be determined since it is a parameter of the connection and the Connection ID is provided, this is not the case with inactive cells.) Cells copied from the Ingress cell flow use the link number provided by the PHY device, and cells copied from the Egress cell flow use the link number read from the Connection Address word.

6.7.6 Multicast Translation

When performing multicast in an environment in which a single MC92501 is supporting multiple PHY devices, a single cell arriving at the switch may be presented to the Egress MC92501 multiple times, once for each physical link. There are two possibilities for performing multicast translation in this environment. One is for the switch interface block to provide a unique multicast identifier for each link to which the cell is transmitted. In this case the MC92501 performs multicast translation in the same manner as for a single link.

Another possibility is for the switch to leave the same multicast identifier in all copies of the cell. In this case the Multicast Translation uses a unique region of the Multicast Translation Table for each link. In order to do so, the switch must provide the link number in the MTTS field of the cell overhead information. See **Section 5.3.2** Multicast Identifier Translation for details.

7

MC92501 Programming Model

7.1 Introduction

The MC92501 programming model includes registers, External Memory mapping, and a description of the data structures used in programming the chip. The following sections provide a detailed description of the programming model.

7.2 Registers Description

The MC92501 registers are divided into several groups. Some of the register groups may be accessed only when the MC92501 is in one of the two operating modes (Setup Mode and Operate Mode) defined in **Section 3.1**. The register groups are:

1. **Status Reporting Registers**—These registers report on the MC92501 status, and generally can be read and written by the processor in either of the MC92501 modes of operation (Setup Mode or Operate Mode).
2. **Control Registers**—These registers control the MC92501 operation, and may be read and written by the processor in either mode of operation.
3. **Configuration Registers**—These registers are used to define the MC92501 configuration and can be read by the processor in either mode of operation (Setup Mode or Operate Mode). These registers can be written by the processor only in the Setup Mode.
4. **Cell Insertion Registers**—These registers are used for cell insertion into the MC92501 cell flow, and are written by the processor when the MC92501 is in Operate Mode. In order to improve performance, the MC92501 Cell Insertion Registers receive special treatment and can be accessed without wait states. See **Section 4.5.1**.
5. **Cell Extraction Registers**—These registers are used for copying cells from the MC92501 cell flows, and can be read by the processor when the MC92501 is in Operate Mode. To improve performance, these registers receive special treatment and can be accessed without wait states. See **Section 4.5.1** for more information.
6. **Pseudo Registers**—The processor can write to these registers in either mode to perform certain operations on the MC92501.

7. **External Address Compression Device**—The processor uses this memory space to access the external address compression device, which can be accessed when the MC92501 is in Setup Mode or during maintenance slots. The MC92501 drives the External Memory Interface as described in **Section 7.2.8**.
8. **External Memory**—The processor uses this memory space to access External Memory. The MC92501 drives the External Memory Interface as described in **Section 4.4.3**. If the "Destructive" memory space is used, the MC92501 automatically provides a write-back of zeros to each External Memory location that is read. The External Memory can be accessed when the MC92501 is in Setup Mode or during maintenance slots.

The Status Reporting Registers, Control Registers, and Configuration Registers are all cleared to zero after Power Up Reset and Software Reset with the exception of the External Memory Timing Configuration Register (EMTCR), the ATMC CFB Revision Register (ARR), the MC92501 Revision Register (RR), and the *Cycle Mode (CM)* bit of the Interrupt Register (IR). **Figure 7-1** shows the MC92501 memory space addressable by the microprocessor using the MADD bus.

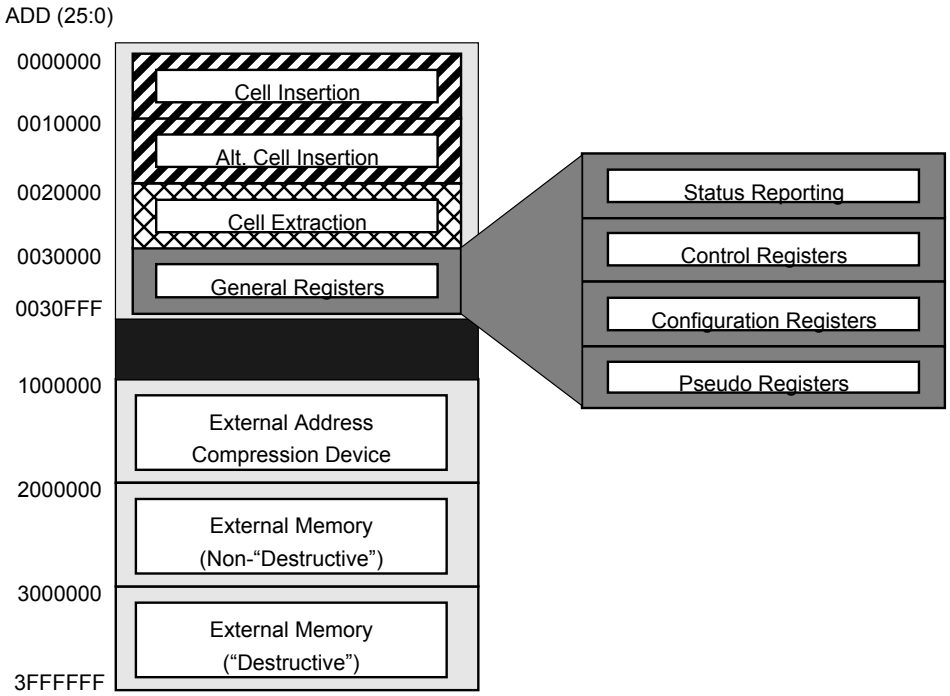


Figure 7-1. Memory Map

NOTE: All fields marked "0" or "reserved" in the register descriptions in this section must be written with zeros. The values read from these fields should be considered undefined and should be ignored.

7.2.1 Cell Insertion Registers (CIR0–CIR15)

These sixteen write-only 32-bit registers are used for Cell Insertion. For the Cell Insertion format, refer to **Section 7.4.1**. For the Cell Insertion modes of operation, refer to **Section 3.4.1**. The Cell Insertion Registers are mapped into two distinct address spaces, a full address space and an alternative address space.

7.2.1.1 Cell Insertion Address Space

In the Cell Insertion address space, all sixteen of the registers are addressable. The trigger register that instructs the MC92501 to transfer the Cell from the Cell Insertion Registers to the Ingress or Egress Insertion Queues is CIR15. The address space is intended for inserting full cells. The physical addresses of the Cell Insertion Registers are shown in **Figure 7-2**.

CIR0	00_0000_0000_xxxx_xxxx_xx00_0000
CIR1	00_0000_0000_xxxx_xxxx_xx00_0100
CIR2	00_0000_0000_xxxx_xxxx_xx00_1000
CIR3	00_0000_0000_xxxx_xxxx_xx00_1100
CIR4	00_0000_0000_xxxx_xxxx_xx01_0000
CIR5	00_0000_0000_xxxx_xxxx_xx01_0100
CIR6	00_0000_0000_xxxx_xxxx_xx01_1000
CIR7	00_0000_0000_xxxx_xxxx_xx01_1100
CIR8	00_0000_0000_xxxx_xxxx_xx10_0000
CIR9	00_0000_0000_xxxx_xxxx_xx10_0100
CIR10	00_0000_0000_xxxx_xxxx_xx10_1000
CIR11	00_0000_0000_xxxx_xxxx_xx10_1100
CIR12	00_0000_0000_xxxx_xxxx_xx11_0000
CIR13	00_0000_0000_xxxx_xxxx_xx11_0100
CIR14	00_0000_0000_xxxx_xxxx_xx11_1000
CIR15	00_0000_0000_xxxx_xxxx_xx11_1100

ACIR0	00_0000_0001_xxxx_xxxx_xxxx_x000
ACIR1	00_0000_0001_xxxx_xxxx_xxxx_x100

x = don't care

Figure 7-2. Cell Insertion Register Addresses

7.2.1.2 CIR Alternate Address Space

In the Cell Insertion Register alternate address space, only ACIR0 and ACIR1 are addressable. The trigger register that instructs the MC92501 to transfer the Cell from the Cell Insertion Registers to the Ingress or Egress Insertion Queues is ACIR1. This address space is used for inserting cells whose header and payload are generated by the MC92501. See **Figure 7-2** for the physical addresses of the CIR Alternate Address Space.

7.2.2 Cell Extraction Registers (CER0–CER15)

These sixteen read-only 32-bit registers are used for Cell Extraction. For the Cell Out format refer to **Section 7.4.2**, and for the Cell Extraction modes of operation refer to **Section 3.4.2**.

7.2.2.1 Cell Extraction Address Space

In the Cell Extraction address space, all sixteen registers are addressable. The trigger register, which informs the MC92501 that the cell in the Cell Extraction Registers was read and a new cell can be loaded from the Cell Extraction Queue into the Cell Extraction Registers, is CER15. This address space is intended for extracting full cells. The physical addresses of the Cell Insertion Registers are shown in **Figure 7-3**.

CER0	00_0000_0010_xxxx_xxxx_xx00_0000
CER1	00_0000_0010_xxxx_xxxx_xx00_0100
CER2	00_0000_0010_xxxx_xxxx_xx00_1000
CER3	00_0000_0010_xxxx_xxxx_xx00_1100
CER4	00_0000_0010_xxxx_xxxx_xx01_0000
CER5	00_0000_0010_xxxx_xxxx_xx01_0100
CER6	00_0000_0010_xxxx_xxxx_xx01_1000
CER7	00_0000_0010_xxxx_xxxx_xx01_1100
CER8	00_0000_0010_xxxx_xxxx_xx10_0000
CER9	00_0000_0010_xxxx_xxxx_xx10_0100
CER10	00_0000_0010_xxxx_xxxx_xx10_1000
CER11	00_0000_0010_xxxx_xxxx_xx10_1100
CER12	00_0000_0010_xxxx_xxxx_xx11_0000
CER13	00_0000_0010_xxxx_xxxx_xx11_0100
CER14	00_0000_0010_xxxx_xxxx_xx11_1000
CER15	00_0000_0010_xxxx_xxxx_xx11_1100
	x = don't care

Figure 7-3. Cell Extraction Register Addresses

7.2.3 General Register List

Table 7-1. General Register List

Register Group	Register Name	Mnemonic	ADD (25:0)	Ref. Page
Status Reporting	Interrupt Register	IR	00301E0	7-7
	Interrupt Mask Register	IMR	00301E4	7-10
	Last Cell Processing Time Register	LCPTR	0030EE0	7-11
	ATMC CFB Revision Register	ARR	0030FE0	7-11
	MC92501 Revision Register	RR	0030BFC	7-12

Table 7-1. General Register List (Continued)

Register Group	Register Name	Mnemonic	ADD (25:0)	Ref. Page
Control Registers	Microprocessor Control Register	MPCTLR	00301E8	7-12
	Maintenance Control Register	MACTLR	00301A0	7-13
	Cell Extraction Queue Filtering Register 0	CEQFR0	00301EC	7-14
	Cell Extraction Queue Filtering Register 1	CEQFR1	00301F0	7-14
	Cell Extraction Queue Priority Register 0	CEQPR0	00301F4	7-15
	Cell Extraction Queue Priority Register 1	CEQPR1	00301F8	7-15
	Ingress Insertion Leaky Bucket Register	IILB	0030224	7-16
	Ingress Insertion Bucket Fill Register	IIBF	0030220	7-16
	Egress Insertion Leaky Bucket Register	EILB	0030204	7-16
	Egress Insertion Bucket Fill Register	EIBF	0030200	7-17
	Internal Scan Control Register	ISCR	0030040	7-17
	Ingress Link Register 0–15	ILNK0–ILNK15	0030000–003003C	7-18
	Egress Link Enable Register	ELER	0030208	7-19
	Ingress Billing Counter Table Pointer	IBCTP	0030180	7-19
	Egress Billing Counter Table Pointer	EBCTP	0030184	7-20
	Policing Counter Table Pointer	PCTP	0030188	7-20
	Cell Time Register	CLTM	0030240	7-20
	Ingress Processing Control Register	IPLR	0030824	7-21
	Egress Processing Control Register	EPLR	0030828	7-21
	Indirect External Memory Access Address Register	IAAR	0030810	7-22
	Indirect External Memory Access Data Register	IADR	0030814	7-22

Table 7-1. General Register List (Continued)

Register Group	Register Name	Mnemonic	ADD (25:0)	Ref. Page
Configuration Registers	Microprocessor Configuration Register	MPCONR	0030E80	7-23
	Maintenance Configuration Register	MACONR	0030DA0	7-25
	Ingress PHY Configuration Register	IPHCR	0030CA0	7-26
	Egress PHY Configuration Register	EPHCR	0030C80	7-27
	Ingress Switch Interface Configuration Register	ISWCR	0030800	7-28
	Egress Switch Interface Configuration Register	ESWCR	0030804	7-30
	Egress Switch Overhead Information Register 0	ESOIR0	0030808	7-34
	Egress Switch Overhead Information Register 1	ESOIR1	0030818	7-35
	UNI Register	UNIR	0030EC0	7-36
	Ingress Processing Configuration Register	IPCR	0030E20	7-36
	Egress Processing Configuration Register	EPCR	0030E00	7-39
	Egress Multicast Configuration Register	EMCR	0030D00	7-41
	ATMC CFB Configuration Register	ACR	0030EA0	7-41
	MC92501 General Configuration Register	GCR	003080C	7-44
	Context Parameters Table Pointer	CPTP	0030D60	7-44
	OAM Table Pointer	OTP	0030D64	7-45
	Dump Vector Table Pointer	DVTP	0030D68	7-45
	VC Table Pointer	VCTP	0030D74	7-45
	Multicast Translation Table Pointer	MTTP	0030D6C	7-45
	Flags Table Pointer	FTP	0030D70	7-46
	Egress Link Counters Table Pointer	ELCTP	0030D80	7-46
	Ingress Link Counters Table Pointer	ILCTP	0030D84	7-46
	Context Parameters Extension Table Pointer	CPETP	0030D88	7-46
	Node ID Registers 0–3	ND0–ND3	0030E40–0030E4C	7-47–7-47
	Ingress VCI Copy Register	IVCR	0030E24	7-48

Table 7-1. General Register List (Continued)

Register Group	Register Name	Mnemonic	ADD (25:0)	Ref. Page
Configuration Registers (Cont'd)	Egress VCI Copy Register	EVCR	0030E04	7-48
	Ingress VCI Remove Register	IVRR	0030E28	7-49
	Egress VCI Remove Register	EVRR	0030E08	7-49
	Performance Monitoring Exclusion Register	PMER	0030E60	7-50
	External Memory Timing Configuration Register	EMTCR	0030CE0	7-50
	External Memory Interface Configuration Register	EMICR	0030CE4	7-52
	RM Overlay Register	RMOR	003081C	7-53
	CLP Transparency Overlay Register	CTOR	003082C	7-54
	Egress Overhead Manipulation Register	EGOMR	0030820	7-54
Pseudo Registers	Software Reset Register (Pseudo)	SRR	0030300	7-56
	Start Scan Register (Pseudo)	SSR	0030308	7-56
	Enter Operate Mode Register (Pseudo)	EOMR	0030304	7-56

7.2.4 Status Reporting Registers

The Status Reporting Registers may be read and written by the processor in either of the MC92501 modes of operation (Setup Mode or Operate Mode).

7.2.4.1 Interrupt Register (IR)

The MC92501 Interrupt Register (IR) includes all the MC92501 general status information that can cause an interrupt if the appropriate Interrupt Mask bit is set in the Interrupt Mask Register (IMR)—see **Section 7.2.4.2** for more information. The IR bits can be divided into three classes:

- **Real Status bit**—The MC92501 sets or clears a real status bit to indicate the current status. The user cannot change the value of this bit.
- **Threshold bit**—The MC92501 sets a threshold bit when a threshold is crossed. The user can reset the threshold bit by writing to the IR with the bit location set after the value drops below the threshold. While the value remains at or above the threshold, the threshold bit cannot be reset and any reset attempt is ignored.
- **Sticky bit**—The MC92501 sets a sticky bit when an event occurs. The user can reset the sticky bit by writing to the IR with the bit location set. However, if the event has occurred again since the IR was last read, the sticky bit is not reset. This prevents missed interrupts that could lead to deadlock situations. Unless it is defined as a specific class, IR bits are sticky bits by default.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OM	CM	MSE	0	0	0	0	SPD	0	0	FQF	CIQE	CEQR	CEQI	CEQL	CEQF
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	ESPE	ESHE	0	0	0	IPPE	IPHE	FQO	FQEO	MNAE

Figure 7-4. Interrupt Register (IR) Fields

Figure 7-4 shows the IR field locations within the register. The IR fields are:

- **Operate Mode (OM)**—The OM bit reports on the MC92501 mode of operation. It is a real status bit.
  - 0 = Setup Mode
  - 1 = Operate Mode
- **Cycle Mode (CM)**—The CM bit reports on the MC92501 Cycle Mode (Maintenance or Normal). It is a real status bit. The processor may use the *Maintenance Slot Enable (MSE)* bit, which issues an interrupt a programmable number of clock cycles before the Maintenance Cycle—to stop the normal flow of the program, and start a polling process on this bit until the MC92501 starts the Maintenance Cycle. Then the Processor may use this cycle to Read/Write the External Memory. The CM bit is set at reset since all cycles are Maintenance cycles after reset.
  - 0 = Normal Cycle. The MC92501 uses the External Memory.
  - 1 = Maintenance Cycle. The MC92501 does not use the External Memory, and the Processor (or the DMA device) may perform External Memory accesses.
- **Maintenance Slot Enable (MSE)**—The MC92501 sets this bit a programmable number of clock cycles before each Maintenance Cycle as specified in the *Maintenance Slot Interrupt Request (MSIR)* field defined in **Section 7.2.6.2**.
- **Scan Process Done (SPD)**—The SPD bit is set when the MC92501 has completed the Internal Scan Process operation.
- **FMC Queue Full (FQF)**—This bit reports that the internal FMC queue is full. The reason for this is that The FMC generation rate is higher then the allocated insertion bandwidth. Insertion rate is controlled by the insertion Leaky Bucket.

NOTE: The FQF bit is valid only if *PM on All Connections (PMAC)* in the ATMC CFB Configuration Register (ACR) is set.

- **Cell Insertion Queue Empty (CIQE)**—The Cell Insertion Queue is empty and a new cell may be inserted into the Ingress or Egress cell flow. The MCIREQ output signal is also asserted and may be used instead of the interrupt.
- **Cell Extraction Queue Ready (CEQE)**—This bit informs the Processor that a cell is ready to be read from the Cell Extraction Registers (CER0–CER15). The MCOREQ output signal is also asserted and may be used instead of the interrupt.
- **Cell Extraction Queue Interrupt Threshold (CEQI)**—This bit is set when the Cell Extraction Queue has reached the programmable Interrupt Threshold defined by the *Extraction Queue Interrupt Threshold (EQIT)* field of the Microprocessor Control Register (MPCTLR). Because this is a threshold bit, the MC92501 ignores any attempt to clear this bit while the number of cells in the Cell Extraction Queue equals or exceeds the Interrupt Threshold value.

Cell Extraction Queue Low Priority Threshold (CEQL)—This bit is set when the Cell Extraction Queue has reached the programmable Low Priority Threshold defined by the *Extraction Queue Low Priority Threshold (EQLPT)* field of the Microprocessor Control Register (MPCTLR). Low Priority Cells that are directed to this queue are dropped until the queue is read and contains fewer cells than the Low Priority Threshold value. This is a threshold bit, thus an attempt to reset this bit while the number of cells in the Cell Extraction Queue equals or exceeds the Low Priority Threshold value is ignored.

- **Cell Extraction Queue Full (CEQF)**—This bit is set when the Cell Extraction Queue is full. Additional cells that are directed to this queue are dropped until this queue is read. This is a threshold bit, thus an attempt to reset this bit while the Cell Extraction Queue is full is ignored.
- **Egress Switch Parity Error (ESPE)**—This bit is set when the Egress Switch Interface block detects a parity error.
- **Egress Switch Protocol Handshake Error (ESHE)**—This bit is set when the Egress Switch Interface block detects a Protocol Handshake Error. The MC92501 produces the ESHE interrupt in the following cases:
  - The STXSOC signal is asserted too early if the number of enabled bytes between one assertion of the STXSOC signal and the next is smaller than the data structure size as defined by the ESNB field.
  - The STXSOC signal is late if the number of enabled bytes between one assertion of the STXSOC signal and the next is larger than the data structure size as defined by the ESNB field.
  - The $\overline{\text{STXENB}}$ signal is asserted while STXCLAV is deasserted.
- **Ingress PHY Parity Error (IPPE)**—The IPPE bit is set when the Ingress PHY Interface detects a parity error. The cell that contains the Parity error may be discarded at the PHY interface (see **Section 5.2.1**), and the Ingress PHY Interface continues its operation without any effect on the next cell.
- **Ingress PHY Protocol Handshake Error (IPHE)**—The IPHE bit is set when the Ingress PHY Interface detects a Protocol Handshake Error. The cell during whose reception the Protocol Handshake Error occurred is discarded at the PHY interface (see **Section 5.2.1**), and the Ingress PHY Interface continues its operation without any effect on the next cell.
- **FMC Queue Overflow (FQO)**—The FQO bit is set when an attempt to insert a Forward Monitor Cell (FMC) into the cell flow within one block size fails. The new FMC is inserted into the FMC Queue. The previous FMC is not generated.
- **FMC Queue End-to-end Overflow (FQEO)**—The FQEO bit is set when an attempt to insert an end-to-end Forward Monitor Cell (FMC) into the cell flow within one half of the block size fails. The MC92501 continues its attempt to insert the FMC. If it is not inserted within one block size, the FQO bit is also set.
- **Maintenance Access Error (MNAE)**—The MNAE bit is set when the processor attempts to use the External Memory in a non-Maintenance Cycle, or the Processor is still using the External Memory when the MC92501 takes back control at the end of a Maintenance Cycle.

7.2.4.2 Interrupt Mask Register (IMR)

The IMR contains an Interrupt Enable bit for each bit in the IR. The MC92501 generates an interrupt when both the IR bit and its corresponding Enable bit are set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OME	CME	MSEE	0	0	0	0	SPDE	0	0	FQFE	CIQEE	CEQRE	CEQIE	CEQLE	CEQFE
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	ESPEE	ESHEE	0	0	0	IPPEE	IPHEE	FQOE	FQEOE	MNAEE

Figure 7-5. Interrupt Mask Register (IMR) Fields

- **Operate Mode Interrupt Enable (OME)**—When OME and OM are both set, an interrupt is generated.
- **Cycle Mode Interrupt Enable (CME)**—When CME and CM are both set, an interrupt is generated.
- **Maintenance Slot Enable Interrupt Enable (MSEE)**—When MSEE and MSE are both set, an interrupt is generated.
- **Scan Process Done Interrupt Enable (SPDE)**—When SPDE and SPD are both set, an interrupt is generated.
- **FMC Queue Full Interrupt Enable (FQFE)**—When FQF and FQFE are set, an interrupt is generated.
- **Cell Insertion Queue Empty Interrupt Enable (CIQEE)**—When CIQEE and CIQE are both set, an interrupt is generated.
- **Cell Extraction Queue Ready Interrupt Enable (CEQRE)**—When CEQRE and CEQR are both set, an interrupt is generated.
- **Cell Extraction Queue Interrupt Threshold Interrupt Enable (CEQIE)**—When CEQIE and CEQI are both set, an interrupt is generated.
- **Cell Extraction Queue Low Priority Threshold Interrupt Enable (CEQLE)**—When CEQLE and CEQL are both set, an interrupt is generated.
- **Cell Extraction Queue Full Interrupt Enable (CEQFE)**—When CEQFE and CEQF are both set, an interrupt is generated.
- **Egress Switch Parity Error Interrupt Enable (ESPEE)**—When ESPE and ESPEE are both set, an interrupt is generated.
- **Egress Switch Protocol Handshake Error Interrupt Enable (ESHEE)**—When ESHE and ESHEE are both set, an interrupt is generated.
- **Ingress PHY Parity Error Interrupt Enable (IPPEE)**—When IPPEE and IPPE are both set, an interrupt is generated.
- **Ingress PHY Protocol Handshake Error Interrupt Enable (IPHEE)**—When IPHEE and IPHE are both set, an interrupt is generated.
- **FMC Queue Overflow Interrupt Enable (FQOE)**—When FQOE and FQO are both set, an interrupt is generated.
- **FMC Queue End-to-end Overflow Interrupt Enable (FQEOE)**—When FQEOE and FQEO are both set, an interrupt is generated.
- **Maintenance Access Error Interrupt Enable (MNAEE)**—When MNAEE and MNAE are both set, an interrupt is generated.

7.2.4.3 Last Cell Processing Time Register (LCPTR)

This read-only register contains the cell time of the most recent non-maintenance cycle. This value may be used to find the most recent entries in the Dump Vector Table, which is updated in a cyclical fashion.

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

Figure 7-6. Last Cell Processing Time Register (LCPTR) Fields

7.2.4.4 ATMC CFB Revision Register (ARR)

This read-only register contains the ATMC CFB Revision Number.

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

Figure 7-7. ATMC CFB Revision Register (ARR) Fields

- **ATMC CFB Major Revision (AMRV)**—This field indicates the ATMC CFB Major Revision Number.
- **ATMC CFB Sub-Revision (ASRV)**—This field indicates the ATMC CFB Sub-Revision Number.

The following values of AMRV and ASRV are currently defined:

Table 7-2. Values of ATMC CFB Revision Fields

AMRV	ASRV	ATMC CFB Revision
000000	000000	Revision A
000001	000000	Revision B
000001	000001	Revision C

7.2.4.5 MC92501 Revision Register (RR)

This read-only register contains the MC92501 identification information.

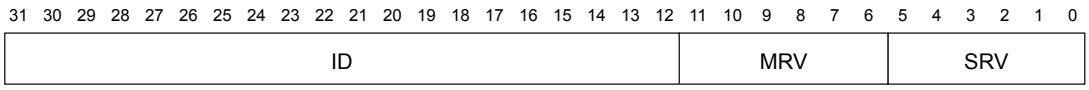


Figure 7-8. Revision Register (RR) Fields

- **CSP Identification Number (ID)**—The ID field value identifies the MC92501**.**
- MC92501 **Major Revision (MRV)**—This field indicates the MC92501 Major Revision Number.
- MC92501 **Sub-Revision (SRV)**—This field indicates the MC92501 Sub-Revision Number.

The following values of ID, MRV and SRV are currently defined:

Table 7-3. Values of MC92501 Revision Fields

ID	MRV	SRV	MC92501 Revision
10000000000000000001	000000	000000	Revision A

7.2.5 Control Registers

The Control Registers controls the MC92501 operation, and may be read and written by the processor in either of the MC92501 modes of operation (Setup Mode or Operate Mode).

7.2.5.1 Microprocessor Control Register (MPCTLR)

This register specifies microprocessor-related control parameters.

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

Table 7-4. Microprocessor Control Register (MPCTLR) Fields

- **Deassertion of Maintenance Request (DMR)**—The DMR field indicates the number of maintenance accesses to be performed in a single Maintenance Slot before the EMMREQ output signal is deasserted. Normally, DMR should be programmed to one less than the number of maintenance accesses intended to be performed so that EMMREQ is deasserted before the last access. If DMR is 0, EMMREQ remains asserted as long as maintenance accesses are allowed. This value might be used in Setup Mode to perform continuous maintenance accesses.

NOTE: Since DMR is 0 after reset, $\overline{\text{EMMREQ}}$ is asserted after reset until either a non-zero value is written to DMR or the MC92501 switches to Operate Mode.

- **Extraction Queue Interrupt Threshold (EQIT)—** This field defines the value of the Extraction Queue Interrupt Threshold. When the Extraction Queue reaches the Interrupt Threshold, the *Cell Extraction Queue Interrupt Threshold Interrupt Enable (CEQIE)* bit in the IR is set. If EQIT is 0, the Interrupt Threshold is undefined and CEQI is never set.
- **Extraction Queue Low Priority Threshold (EQLPT)—** This field defines the value of the Extraction Queue Low Priority Threshold. When the Extraction Queue reaches the Low Priority Threshold, the *Cell Extraction Queue Low Priority Threshold Interrupt Enable (CEQLE)* bit in the IR is set. Low Priority Cells that are directed to this queue are dropped until this queue is read and contains fewer cells than the Low Priority Threshold value. If EQLPT is 0, the Low Priority Threshold is undefined, CEQL is never set, and low-priority cells are not filtered out.

7.2.5.2 Maintenance Control Register (MACTLR)

This register defines parameters related to maintenance accesses.

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

Figure 7-9. Maintenance Control Register (MACTLR)

- **Maintenance Period Length (MPL)—** This field specifies the number of cell processing slots between maintenance slots. Note that higher values of MPL result in fewer maintenance slots. The fraction of slots that are maintenance slots in normal operation ranges from 1/2 (MPL = 1) to 1/64 (MPL = 63). If MPL is 0, all the slots are maintenance slots. This is the situation after reset, and it may be used for setup of the External Memory.

NOTE: The MC92501 does not process cells while MPL = 0. MPL should be programmed with a non-zero value before switching to Operate Mode. If MPL is programmed to zero in Operate Mode (for continuous maintenance accesses), cells may be lost.

7.2.5.3 Cell Extraction Queue Filtering Register 0 (CEQFR0)

The Cell Extraction Queue Filtering Register 0 controls the reason-based filtering of cells that are copied to the Cell Extraction Queue.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PR31	PR30	PR29	0	0	0	0	0	0	0	0	0	0	0	PR17	PR16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR15	PR14	PR13	PR12	PR11	PR10	PR9	0	PR7	PR6	PR5	PR4	0	0	PGFC	0

Figure 7-10. Cell Extraction Queue Filtering Register 0 (CEQFR0) Fields

- **Pass Reason *n* Cells (PR*n*)**—If the PRn bit is set, a cell that is copied to the Cell Extraction Queue with the *Extraction Reason (RSN)* field equal to n is actually placed in the Cell Extraction Queue, rather than being dropped.
- **Pass GFC Cells (PGFC)**—If the PGFC bit is set, a cell that is copied to the Cell Extraction Queue with the *GFC Reason (GFCR)* bit set is actually placed in the Cell Extraction Queue, rather than being dropped.

7.2.5.4 Cell Extraction Queue Filtering Register 1 (CEQFR1)

The Cell Extraction Queue Filtering Register 1 controls the cell-name-based filtering of cells that are copied to the Cell Extraction Queue.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	PN8	0	PN6	PN5	PN4	PN3	PN2	PN1	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	PC9	PC8	0	PC6	PC5	PC4	PC3	PC2	PC1	0

Figure 7-11. Cell Extraction Queue Filtering Register 1 (CEQFR1) Fields

- **Pass Name *n* Cells (PN*n*)**—If the PNn bit is set, a cell that is copied to the Cell Extraction Queue with the *Extraction Reason (RSN)* field indicating that it was copied because an OAM copy bit is set and with the *Indication Cell Name (ICN)* field equal to n is actually placed in the Cell Extraction Queue, rather than being dropped.
- **Pass Copy All Name *n* Cells (PC*n*)**—If the PCn bit is set, a cell that is copied to the Cell Extraction Queue with the *Extraction Reason (RSN)* field indicating that it was copied because the copy all bit is set and with the *Indication Cell Name (ICN)* field equal to n is actually placed in the Cell Extraction Queue, rather than being dropped.

7.2.5.5 Cell Extraction Queue Priority Register 0 (CEQPR0)

The Cell Extraction Queue Priority Register 0 controls the reason-based priority of cells that are copied to the Cell Extraction Queue.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HR31	HR30	HR29	0	0	0	0	0	0	0	0	0	0	0	HR17	HR16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HR15	HR14	HR13	HR12	HR11	HR10	HR9	0	HR7	HR6	HR5	HR4	0	0	HGFC	0

Figure 7-12. Cell Extraction Queue Priority Register 0 (CEQPR0) Fields

- **High-Priority Reason *n* Cells (HR*n*)**—If the HRn bit is set, a cell that is copied to the Cell Extraction Queue with the *Extraction Reason (RSN)* field equal to n is actually placed in the Cell Extraction Queue, rather than being dropped, when the Cell Extraction Queue is above the Low Priority Threshold.
- **High-Priority GFC Cells (HGFC)**—If the HGFC bit is set, a cell that is copied to the Cell Extraction Queue with the *GFC Reason (GFCR)* bit set is actually placed in the Cell Extraction Queue, rather than being dropped, when the Cell Extraction Queue is above the Low Priority Threshold.

7.2.5.6 Cell Extraction Queue Priority Register 1 (CEQPR1)

The Cell Extraction Queue Priority Register 1 controls the cell-name-based priority of cells that are copied to the Cell Extraction Queue.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	HN8	0	HN6	HN5	HN4	HN3	HN2	HN1	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	HC9	HC8	0	HC6	HC5	HC4	HC3	HC2	HC1	0

Figure 7-13. Cell Extraction Queue Priority Register 1 (CEQPR1) Fields

- **High-Priority Name *n* Cells (HN*n*)**—If the HNn bit is set, a cell that is copied to the Cell Extraction Queue with the *Extraction Reason (RSN)* field indicating that it was copied because an OAM copy bit is set and with the *Indication Cell Name (ICN)* field equal to n is actually placed in the Cell Extraction Queue, rather than being dropped, when the Cell Extraction Queue is above the Low Priority Threshold.
- **High-Priority Copy All Name *n* Cells (HC*n*)**—If the HCn bit is set, a cell that is copied to the Cell Extraction Queue with the *Extraction Reason (RSN)* field indicating that it was copied because the copy all bit is set and with the *Indication Cell Name (ICN)* field equal to n is actually placed in the Cell Extraction Queue, rather than being dropped, when the Cell Extraction Queue is above the Low Priority Threshold.

7.2.5.7 Ingress Insertion Leaky Bucket Register (IILB)

The Ingress Cell insertion rate is controlled by a Leaky Bucket algorithm whose parameters are defined in this register. For more information on the Ingress Insertion Leaky Bucket refer to **Section 5.2.6**.

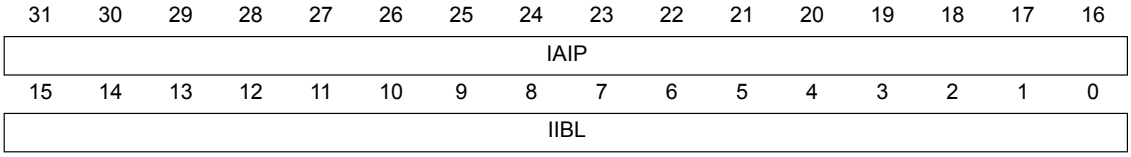


Figure 7-14. Ingress Insertion Leaky Bucket Register (IILB) Fields

- **Ingress Average Insertion Period (IAIP)**—This field determines how often, on average, cells may be inserted to the Ingress cell flow. It consists of a 12-bit integer part and a 4-bit fractional part.
- **Ingress Insertion Bucket Limit (IIBL)**—This field determines the burstiness of the cell insertions in the Ingress cell flow.

7.2.5.8 Ingress Insertion Bucket Fill Register (IIBF)

This register contains the current state of the Ingress Insertion Leaky Bucket.

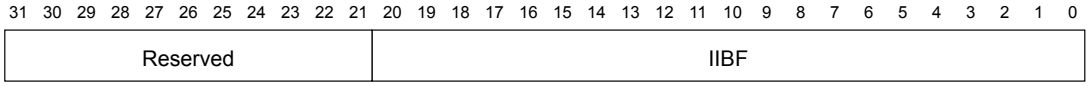


Figure 7-15. Ingress Insertion Bucket Fill Register (IIBF) Fields

- **Ingress Insertion Bucket Fill (IIBF)**—This field contains the current Ingress Insertion Leaky Bucket fill value. Like the IAIP, the IIBF is in units of 1/16 of a cell time, and the 4 Least Significant Bits represent the fractional part. Normally, the IIBF does not need to be user-defined, but it can be used to reset the Leaky Bucket.

7.2.5.9 Egress Insertion Leaky Bucket Register (EILB)

The Egress Cell insertion rate is controlled by a Leaky Bucket algorithm whose parameters are defined in this register. For more information on the Egress Insertion Leaky Bucket refer to **Section 5.3.3**.

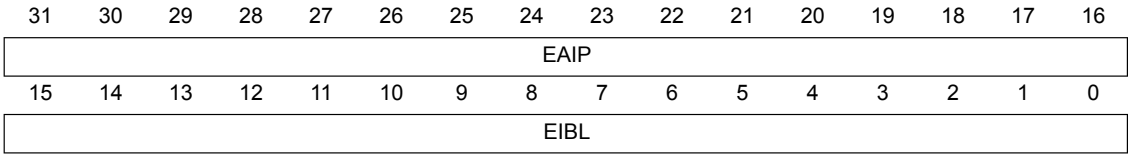


Figure 7-16. Egress Insertion Leaky Bucket Register (EILB) Fields

- **Egress Average Insertion Period (EAIP)**—This field determines how often, on average, cells may be inserted to the Egress cell flow. It consists of a 12-bit integer part and a 4-bit fractional part.
- **Egress Insertion Bucket Limit (EIBL)**—This field determines the burstiness of the cell insertions in the Egress cell flow.

7.2.5.10 Egress Insertion Bucket Fill Register (EIBF)

This register contains the current state of the Egress Insertion Leaky Bucket.

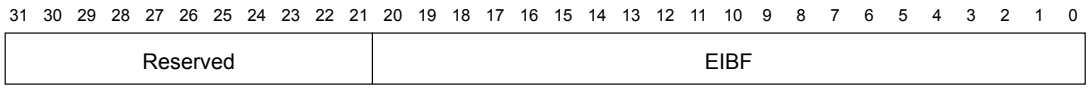


Figure 7-17. Egress Insertion Bucket Fill Register (EIBF) Fields

- **Egress Insertion Bucket Fill (EIBF)**—This field contains the current fill value of the Egress Insertion Leaky Bucket. Like the EAIP, the EIBF is in units of 1/16 of a cell time, and the 4 Least Significant Bits represent the fractional part. Normally, the EIBF does not need to be user-defined, but can be used to reset the Leaky Bucket.

7.2.5.11 Internal Scan Control Register (ISCR)

The Internal Scan Control Register contains the highest-valued Connection Identifier (CI) whose Context Parameters Table entry should be scanned. The Internal Scan process starts its scan from this value and scan downwards to CI = 0. The enable bits in the ISCR are used to define which cells should be inserted by the current internal scan. This register may be written at any time.

NOTE: If the ISCR is written while the Internal Scan is active, the updated values of the enable bits take effect immediately. Clearing the ISCR disables cell insertion from the Internal Scan process.

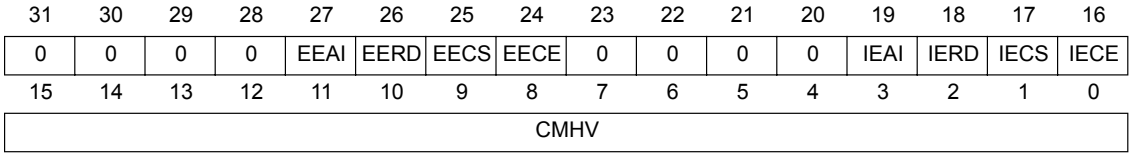


Figure 7-18. Internal Scan Register (ISCR) Fields

- **Egress Enable AIS (EEAI)**—This bit determines whether the Internal Scan inserts OAM AIS cells to the Egress cell flow.
  - 0 = AIS insertion to Egress disabled
  - 1 = AIS insertion to Egress enabled
- **Egress Enable RDI (EERD)**—This bit determines whether the Internal Scan inserts OAM RDI cells to the Egress cell flow.
  - 0 = RDI insertion to Egress disabled
  - 1 = RDI insertion to Egress enabled
- **Egress Enable Continuity Check Segment (EECS)**—This bit determines whether the Internal Scan inserts OAM segment CC cells to the Egress cell flow.
  - 0 = Segment CC insertion to Egress disabled
  - 1 = Segment CC insertion to Egress enabled

- **Egress Enable Continuity Check End-to-end (EECE)**—This bit determines whether the Internal Scan inserts OAM end-to-end CC cells to the Egress cell flow.
  - 0 = End-to-end CC insertion to Egress disabled
  - 1 = End-to-end CC insertion to Egress enabled
- **Ingress Enable AIS (IEAI)**—This bit determines whether the Internal Scan inserts OAM AIS cells to the Ingress cell flow.
  - 0 = AIS insertion to Ingress disabled
  - 1 = AIS insertion to Ingress enabled
- **Ingress Enable RDI (IERD)**—This bit determines whether the Internal Scan inserts OAM RDI cells to the Ingress cell flow.
  - 0 = RDI insertion to Ingress disabled
  - 1 = RDI insertion to Ingress enabled
- **Ingress Enable Continuity Check Segment (IECS)**—This bit determines whether the Internal Scan inserts OAM segment CC cells to the Ingress cell flow.
  - 0 = Segment CC insertion to Ingress disabled
  - 1 = Segment CC insertion to Ingress enabled
- **Ingress Enable Continuity Check End-to-end (IECE)**—This bit determines whether the Internal Scan inserts OAM end-to-end CC cells to the Ingress cell flow.
  - 0 = End-to-end CC insertion to Ingress disabled
  - 1 = End-to-end CC insertion to Ingress enabled
- **Context Memory Highest Value (CMHV)**—The highest-valued Connection Identifier whose Context Table entry is to be scanned.

7.2.5.12 Ingress Link Registers (ILNK0–ILNK15)

The Link Registers contain information (used by the Address Compression) about the treatment of the physical links that the MC92501 is supporting. If only one physical link is being supported (see **Section 7.2.6.3**), ILNK0 is used.

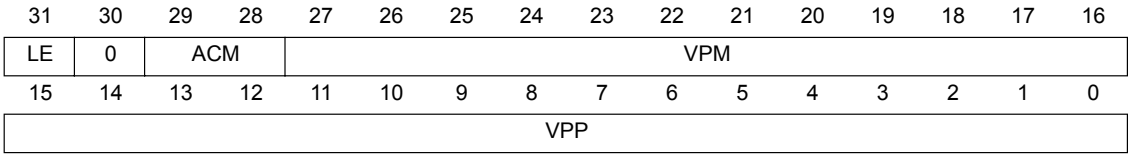


Figure 7-19. Ingress Link Register (ILNKn) Fields

- **Link Enable (LE)**—This bit, when set, indicates that the physical link is enabled. This bit should only be set for physical links that are supported.
  - 0 = Reception from this link is disabled
  - 1 = Reception from this link is enabled
- **Address Compression Method (ACM)**—This field determines the method used for address compression on the cells arriving from the PHY on this link. See **Section 5.2.2** for more information.
  - 00 = VP and VC Table Lookup
  - 01 = VP Table Lookup only

— 10 = External Address Compression without VP Table Lookup
— 11 = External Address Compression with VP Table Lookup

- **VPI Mask (VPM)** — This field indicates which bits of the VPI are allocated on this link. Each bit that is set in this field indicates that the corresponding bit of the VPI should be used in the VP Table Lookup. See **Section 5.2.2** for more information.

NOTE: At a UNI the four most significant bits of the VPM field should be zero since the VPI contains only eight bits.

- **VP Pointer (VPP)** — This field contains the 16 MSBs of the External Memory address of the VP Table belonging to the link. Eight 0s are concatenated to the right of this field to construct the actual External Memory address.

7.2.5.13 Egress Link Enable Register (ELER)

The Egress Link Enable Register is used to enable/disable transmission to each physical link individually. Each bit that is set enables transmission to the corresponding link.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LE15	LE14	LE13	LE12	LE11	LE10	LE9	LE8	LE7	LE6	LE5	LE4	LE3	LE2	LE1	LE0

Figure 7-20. Egress Link Enable Register (ELER) Fields

- **Link Enable (LE0–LE15)** — Each LE bit refers to one of the physical links supported by the MC92501. The LE bit, when set, enables transmission to the link.

7.2.5.14 Ingress Billing Counters Table Pointer Register (IBCTP)

This register contains the pointer to the first word of the Ingress Billing Counters Table. The pointer is in units of 256 bytes. This pointer register may be written in Operate Mode to allow the processor to relocate the Ingress Billing Counters Table. In this way the counters of all the connections can be frozen simultaneously (e.g., at 15 minute intervals) and read at leisure.

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

Figure 7-21. Ingress Billing Counters Table Pointer (IBCTP) Fields

7.2.5.15 Egress Billing Counters Table Pointer Register (EBCTP)

This register contains the pointer to the first word of the Egress Billing Counters Table. The pointer is in units of 256 bytes. This pointer register may be written in Operate Mode to allow the processor to relocate the Egress Billing Counters Table. In this way the counters of all the connections can be frozen simultaneously (e.g., at 15 minute intervals) and read at leisure.

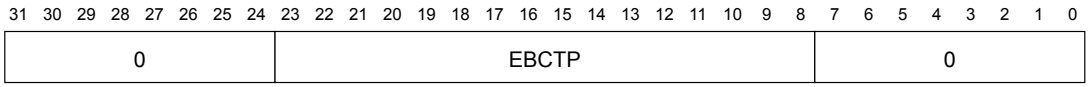


Figure 7-22. Egress Billing Counters Table Pointer (EBCTP) Fields

7.2.5.16 Policing Counters Table Pointer Register (PCTP)

This register contains the pointer to the first word of the Policing Counters Table. The pointer is in units of 256 bytes. This pointer register may be written in Operate Mode to allow the processor to relocate the Policing Counters Table. In this way the counters of all the connections can be frozen simultaneously (for example, at 15 minute intervals) and read at leisure.

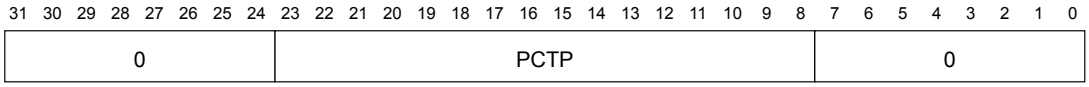


Figure 7-23. Policing Counters Table Pointer (PCTP) Fields

7.2.5.17 Cell Time Register (CLTM)

This register contains a count of the cell processing times. It is a 32-bit counter that is initialized to zero on reset and counts cell processing times when the MC92501 is in Operate Mode.

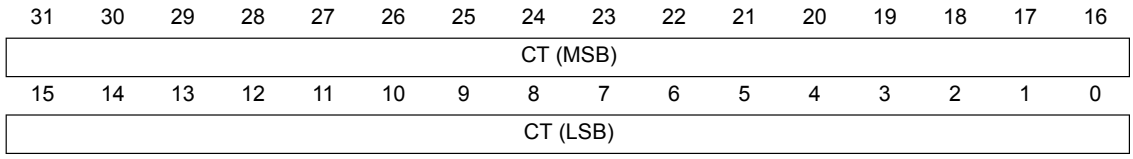


Figure 7-24. Cell Time Register (CLTM) Fields

7.2.5.18 Ingress Processing Control Register (IPLR)

This register defines the MC92501 Ingress processing parameters that can be changed in Operate Mode.

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

Figure 7-25. Ingress Processing Control Register (IPLR) Fields

- **Global Ingress Congestion Notification (ICNG)**—This bit notifies the MC92501 whether there is congestion in the Ingress flow.
  - 0 = No Ingress Congestion.
  - 1 = Ingress Congestion. The MC92501 performs Selective Discard according to per-connection *Connection Ingress Marking Enable (CIME)* bit.
- **Global Ingress ABR Mark Enable (IAME)**—This bit, when set, indicates that currentIngressngress flow status implies that the MC92501 should perform RR-marking and/or EFCI marking if enabled. See **Section 6.4.3.1** for more information.

7.2.5.19 Egress Processing Control Register (EPLR)

This register defines the MC92501 Ingress processing parameters that can be changed in Operate Mode.

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

Figure 7-26. Egress Processing Control Register (EPLR) Fields

- **Global Egress ABR Mark Enable (EAME)**—This bit, when set, indicates that current Egress flow status implies that the MC92501 should perform RR-marking and/or EFCI marking if enabled. See **Section 6.4.3.2** for more information.

7.2.5.20 Indirect External Memory Access Address Register (IAAR)

This register contains the address, width and busy bit for accessing the MC92501 External Memory or the External Memory Device. Refer to **Section 3.3.4** for details.

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

Figure 7-27. Indirect External Memory Access Address Register (IAAR) Fields

- **Indirect External Memory Access Busy (IAB)** — This bit indicates that indirect External Memory access mechanism is busy.
  - 0 = Indirect access mechanism is free and therefore indirect External Memory access data register can be accessed
  - 1 = Indirect access mechanism is busy and therefore indirect access data register should not be accessed
- **Indirect External Memory Access Direction (IAD)** — This bit indicates indirect access direction.
  - 0 = Indirect write access
  - 1 = Indirect read access
- **Indirect External Memory Access Size (IAW)** — This bit indicates the size of the access.
  - 0 = 32 bits
  - 1 = 16 bits
- **Indirect External Memory Access Address Space (IAAS)** — This field indicates the accessed address space.
  - 00 = Reserved
  - 01 = External Address Compression Device
  - 10 = Non Destructive External Memory
  - 11 = Reserved
- **Indirect External Memory Access Address (IAA)** — This field indicates bits 23:1 of the address within the address space specified in the *Indirect External Memory Access Address Space (IAAS)* field.

7.2.5.21 Indirect External Memory Access Data Register (IADR)

This register contains the data which should be written to the External Memory in case of an indirect write access or the data that was last read from External Memory in case of an indirect read access. Refer to **Section 3.3.4** for details.

7.2.6 Configuration Registers

The Configuration Registers are used to define the MC92501 configuration, and may be read by the processor in either of the MC92501 modes of operation (Setup Mode or Operate Mode). These registers may be written by the processor only in Setup Mode of operation.

7.2.6.1 Microprocessor Configuration Register (MPCONR)

This register defines the Processor interface configuration parameters.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DO	0	0	0	0	WSSM	0	0	RQ0		0	RQ1		0	RQ2	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	MDC	0	DDDS	0	DDGR		0	DDEM		0	DDCI	0	DDCE	

Figure 7-28. Microprocessor Configuration Register (MPCONR) Fields

- **Data Order (DO)**—This bit defines the order of the bytes on the data bus inside the 32-bit data structures of the Cell Insertion Payload Registers (CIR4–CIR15) and the Cell Extraction Payload Registers (CER4–CER15). It also defines the order of the records in the External Memory tables that contain 16-bit records.
  - 0 = Most significant byte first (Big Endian—Motorola or IBM style)
  - 1 = Least significant byte first (Little Endian—Intel or Digital style)
- **Word Select Signals Mode (WSSM)**—This bit defines the functionality of the *MP Word Write Enable High / Address 1 (MWSH/A1)* and the *MP Word Write Enable Low / SIZE (MWSL/SIZE)* signals. See **Section 4.5.1.2.3** and **Section 4.5** for details.
  - 0 = MWSH/A1 functions as MWSH-word write enable high and MWSL/SIZE functions as MWSL-word write enable low.
  - 1 = MWSH/A1 functions as A1 and MWSL/SIZE functions as SIZE.
- **MREQ0 Signal Functionality (RQ0)**—This field defines the functionality of the *MP Request 0 (MREQ0)* signal. See **Section 4.5.2** for details.
  - 00 = Cell in Request
  - 01 = Cell In Request
  - 10 = Cell Out Request
  - 11 = External Memory Request
- **MREQ1 Signal Functionality (RQ1)**—This field defines the functionality of the *MP Request 1 (MREQ1)* signal. See **Section 4.5.2** for details.
  - 00 = Cell Out Request
  - 01 = Cell In Request
  - 10 = Cell Out Request
  - 11 = External Memory Request
- **MREQ2 Signal Functionality (RQ2)**—This field defines the functionality of the *MP Request 2 (MREQ2)* signal. See **Section 4.5.2** for details.
  - 00 = External Memory Request
  - 01 = Cell In Request
  - 10 = Cell Out Request
  - 11 = External Memory Request

- **MDTACK*n* Drive Control (MDC)**—This bit determines which MDTACK signals are driven.
  - 0 = MDTACK0 is driven and MDTACK1 is not driven.
  - 1 = Both MDTACK0 and MDTACK1 are driven.
- **Drive DTACK at Data Strobe (DDDS)**—This bit determines the timing of the assertion of the MDTACK output signal during Cell Insertion Register write accesses and Maintenance write accesses. See **Section 4.5.1.2.2** and **Section 4.5.1.2.3** for details.
  - 0 = MDTACK is asserted with the standard timing.
  - 1 = MDTACK is not asserted until the MCLK rising edge after MDS has been detected asserted at a falling edge of MCLK.
- **DTACK Drive During General Register Accesses (DDGR)**—This field determines the timing of the assertion of the MDTACK output signal during General Register read accesses. See **Section 4.5.1.1.1** for details.
  - 00 = The MC92501 tri-states MDTACK during General Register read accesses.
  - 01 = During General Register read accesses, MDTACK is asserted as soon as MDATA contains valid data.
  - 10 = During General Register read accesses, the assertion of MDTACK is delayed by one ACLK cycle.
  - 11 = During General Register read accesses, the assertion of MDTACK is delayed by two ACLK cycles.

NOTE: If DDGR ≠ 00, MDTACK is driven during Maintenance write accesses.

- **DTACK Drive During External Memory Accesses (DDEM)**—This field determines the timing of the assertion of the MDTACK output signal during Maintenance read accesses. See **Section 4.5.1.1.3** for details.
  - 00 = The MC92501 tri-states MDTACK during Maintenance read and write accesses.
  - 01 = During Maintenance read accesses, MDTACK is asserted at the MCLK rising edge following the falling edge at which MSEL is detected asserted.
  - 10 = During Maintenance read accesses, the assertion of MDTACK is delayed by one MCLK cycle.
  - 11 = During Maintenance read accesses, the assertion of MDTACK is delayed by two MCLK cycles.

NOTE: If DDEM ≠ 00, MDTACK is driven during Maintenance write accesses. The timing is determined by the *Drive DTACK at Data Strobe (DDDS)* bit.

- **Drive DTACK During Cell Insertion Accesses (DDCI)**—This bit determines whether the MC92501 drives the MDTACK output signal during accesses to the Cell Insertion Registers. See **Section 4.5.1.2.2** for details.
  - 0 = The MC92501 tri-states MDTACK during Cell Insertion Register accesses.
  - 1 = The MC92501 drives MDTACK during Cell Insertion Register accesses. The timing is determined by the *Drive DTACK at Data Strobe (DDDS)* bit.

- **DTACK Drive during Cell Extraction Register Accesses (DDCE)**—This field determines the timing of the assertion of the MDTACK output signal during Cell Extraction Register accesses. See **Section 4.5.1.1.2** for details.
  - 00 = The MC92501 tri-states MDTACK during Cell Extraction Register accesses.
  - 01 = During Cell Extraction Register accesses, $\overline{\text{MDTACK}}$ is asserted in response to the assertion of MSEL.
  - 10 = During Cell Extraction Register accesses, the assertion of $\overline{\text{MDTACK}}$ is delayed by one MCLK cycle.
  - 11 = During Cell Extraction Register accesses, the assertion of $\overline{\text{MDTACK}}$ is delayed by two MCLK cycles.

7.2.6.2 Maintenance Configuration Register (MACONR)

This register defines the Processor interface configuration parameters.

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

Figure 7-29. Maintenance Configuration Register (MACONR) Fields

- **Maintenance Slot DMA Request (MSDR)**—The MSDR field determines the number of ACLK cycles before a Maintenance Slot that the internal maintenance request signal is asserted. This signal is synchronized to MCLK and then the EMMREQ output signal is asserted. The synchronization delay of 1–2 MCLK cycles should be taken into account when programming the MSDR field. Note that the MSDR ranges was increased with relation to revision A from 0–63 to 0–511.
  - 000000000 = The maintenance request signal is asserted together with the *Cycle Mode (CM)* bit of the Interrupt Register at the beginning of the Maintenance Slot. EMMREQ is asserted 1.5–2.5 MCLK cycles later.
  - 000000001 = The maintenance request signal is asserted one ACLK cycle before the CM bit. EMMREQ is asserted 1.5–2.5 MCLK cycles later.
  - 000000010 = The maintenance request signal is asserted two ACLK cycles before the CM bit. EMMREQ is asserted 1.5–2.5 MCLK cycles later.
  - ***
  - 111111111 = The maintenance request signal is asserted 511 ACLK cycles before the CM bit. EMMREQ is asserted 1.5–2.5 MCLK cycles later.

- **Maintenance Slot Interrupt Request (MSIR)**—This field defines the number of ACLK cycles before a Maintenance Slot that the *Maintenance Slot Enable (MSE)* bit in the IR is set. This interrupt notifies the processor that the next cell slot is a Maintenance Slot. The processor should monitor the *Cycle Mode (CM)* bit in the Interrupt Register before starting its Maintenance operation.
  - 000000 = The MSE bit is asserted together with the *Cycle Mode (CM)* bit of the Interrupt Register at the beginning of the Maintenance Slot.
  - 000001 = The MSE bit is asserted one clock cycle before the CM bit.
  - 000010 = The MSE bit is asserted two clock cycles before the CM bit.
  - ***
  - 111111 = The MSE bit is asserted 63 clock cycles before the CM bit.

7.2.6.3 Ingress PHY Configuration Register (IPHCR)

This register controls the operation of the Ingress PHY interface block. See **Section 4.2.1** for more details.

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

Figure 7-30. Ingress PHY Configuration Register (IPHCR) Fields

- **Ingress UTOPIA Mode (IUM)**—This bit defines the UTOPIA level mode of the Ingress PHY. See **Section 4.2** for more information.
  - 0 = UTOPIA level 1
  - 1 = UTOPIA level 2
- **Invalid Pattern Discard (INVPD)**—The INVPD bit determines the treatment of cells received with the "invalid" pattern (VPI/VCI/; CLP = 1) in the header.
  - 0 = Cells with the "invalid" pattern are removed from the cell flow and copied to the Cell Extraction Queue.
  - 1 = Cells with the "invalid" pattern are discarded at the PHY interface in the same manner as unassigned cells and do not occupy a cell processing slot.
- **Ingress PHY Operation Mode (IPOM)**—The IPOM bit determines whether the Ingress PHY interface operates in an octet-based mode or a cell-based mode.
  - 0 = The PHY interface is octet-based.
  - 1 = The PHY interface is cell-based.
- **Ingress PHY Parity Enable (IPPR)**—This bit defines whether there is parity checking on the Ingress PHY Interface. If a parity error is detected on the cell header by the Ingress PHY Interface, the cell that contains the parity error is discarded, and the Ingress PHY continues its operation without any effect on the next cell.
  - 0 = Parity checking is disabled.
  - 1 = Parity checking is enabled.

- **Ingress Payload Parity Enable (IPLP)**—This bit determines whether a parity error that is detected on the payload of a cell arriving from the PHY should cause the cell to be removed from the cell flow. Parity checking is enabled by *Ingress PHY Parity Enable (IPPR)*.
  - 0 = A parity error detected on the payload of a cell does not cause the cell to be removed.
  - 1 = A parity error detected on the payload of a cell causes the cell to be removed from the cell flow and copied to the Cell Extraction Queue.

Table 7-5. Parity Checking at Ingress PHY Interface

IPPR	IPLP	Action when Parity Error is Detected on a Header Byte	Action when Parity Error is Detected on a Payload Byte
0	0	Ignore	Ignore
0	1	Ignore	Ignore
1	0	Discard cell at the PHY interface	Ignore
1	1	Discard cell at the PHY interface	Remove cell and copy to the Cell Extraction Queue

7.2.6.4 Egress PHY Configuration Register (EPHCR)

This register controls the operation of the Egress PHY interface block.

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

Figure 7-31. Egress PHY Configuration Register (EPHCR) Fields

- **Egress UTOPIA Mode (EUM)**—This bit defines the UTOPIA level mode of the Egress PHY. See **Section 4.2** for more information.
  - 0 = UTOPIA level 1
  - 1 = UTOPIA level 2
- **Egress PHY Interface FIFO Control (EPFC)**—This bit determines the size of the Egress PHY Interface FIFO.
  - 0 = The Egress PHY Interface has a 4-cell FIFO.
  - 1 = The Egress PHY Interface has a 2-cell FIFO.

NOTE: If the PHY device has at least a 4-cell FIFO, EPFC may be 1 to reduce the delay of the Egress cell flow. If the PHY device has only a small FIFO, EPFC should be 0.

- **Egress PHY Operation Mode (EPOM)**—The EPOM bit determines whether the Egress PHY interface operates in an octet-based mode or a cell-based mode.
  - 0 = The PHY interface is octet-based.
  - 1 = The PHY interface is cell-based.
- **Egress Cell Generation Enable (ECGE)**—This bit determines whether the Egress PHY Interface block generates Unassigned/Idle cells when there are no cells for transmission. The cell type generated by the Egress PHY is defined by *Egress Generate Idle Cells (EGIC)*.
  - 0 = Do not generate Unassigned/Idle cells
  - 1 = Generate Unassigned/Idle cells

NOTE: If multiple PHY devices are used or if UL2 is used, ECGE should not be set.

- **Egress Generate Idle Cells (EGIC)**—This bit determines the type of cells generated by the Egress PHY Interface. The cell generation is enabled by *Egress Cell Generation Enable (ECGE)*.
  - 0 = "Unassigned" (ATM layer) cells are generated
  - 1 = "Idle" (PHY layer) cells are generated

Table 7-6. Cell Generation at Egress PHY Interface

ECGE	EGIC	Cell Generation
0	0	None
0	1	None
1	0	"Unassigned" cells
1	1	"Idle" cells

7.2.6.5 Ingress Switch Interface Configuration Register (ISWCR)

This register controls the operation of the Ingress Switch Interface block.

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

Figure 7-32. Ingress Switch Interface Configuration Register (ISWCR)

- **Ingress Switch SRXDATA Driver Control (ISSDC)**—This bit determines whether the outputs of the Ingress Switch Interface block are tri-stated when SRXENB is deasserted.
  - 0 = SRXDATA, SRXPRTY, and SRXSOC are driven only if $\overline{\text{SRXENB}}$ is asserted on the previous clock.
  - 1 = SRXDATA, SRXPRTY, and SRXSOC are always driven.

- **Ingress Switch Parity Mode (ISPM)**—This bit selects the ISWI parity mode—odd or even.
  - 0 = Odd parity is generated over the data transferred to the switch
  - 1 = Even parity is generated over the data transferred to the switch
- **Ingress Switch HEC Field (ISHF)**—The ISHF field determines if the HEC octet is inserted before the payload of cells transferred to the switch and what data it should contain.
  - 00 = No HEC octet is inserted
  - 01 = Reserved
  - 10 = The HEC octet is inserted and contains 00000000
  - 11 = The HEC octet is inserted and contains the MSB of the most-significant long word of the switch parameters.
- **Ingress Switch Number Of Bytes (ISNB)**—The ISNB field determines the size of the data structure transferred to the switch.
  - 0000 = 64 bytes per cell are transferred to the switch
  - 0001 = Reserved
  - 0010 = Reserved
  - 0011 = Reserved
  - 0100 = 52 bytes per cell are transferred to the switch
  - 0101 = 53 bytes per cell are transferred to the switch
  - 0110 = 54 bytes per cell are transferred to the switch
  - 0111 = 55 bytes per cell are transferred to the switch
  - 1000 = 56 bytes per cell are transferred to the switch
  - 1001 = 57 bytes per cell are transferred to the switch
  - 1010 = 58 bytes per cell are transferred to the switch
  - 1011 = 59 bytes per cell are transferred to the switch
  - 1100 = 60 bytes per cell are transferred to the switch
  - 1101 = 61 bytes per cell are transferred to the switch
  - 1110 = 62 bytes per cell are transferred to the switch
  - 1111 = 63 bytes per cell are transferred to the switch

7.2.6.6 Egress Switch Interface Configuration Register (ESWCR)

This register determines the operation of the Egress Switch Interface block.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EIAS	EEAS	VPS	0	0	IHAF	0	ESFC	EFE	MTSE	EATD	ELNS	ESPC	ESPR	EPLP	ESHF
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ESNB					0	IMSB					ILSB				

Figure 7-33. Egress Switch Interface Configuration Register (ESWCR) Fields

- **Global IFS Enable (EIAS)**—This bit enables the MC92501 to use *Overhead Ingress Flow Status (IFS)* bit in the Egress switch overhead. See **Section 6.4.3.1** for details.
  - 0 = The *Overhead Ingress Flow Status (IFS)* bit is not defined in the Egress overhead fields so it cannot trigger ABR cell marking.
  - 1 = The *Overhead Ingress Flow Status (IFS)* bit is defined in the Egress overhead fields and is used by the MC92501 for marking cells as part of Relative Rate ABR.
- **Global EFS Enable (EEAS)**—This bit enables the MC92501 to use *Overhead Egress Flow Status (EFS)* bit in the Egress switch overhead. See **Section 6.4.3.2** for details.
  - 0 = The *Overhead Egress Flow Status (EFS)* bit is not defined in the Egress overhead fields so it cannot trigger ABR cell marking.
  - 1 = The *Overhead Egress Flow Status (EFS)* bit is defined in the Egress overhead fields and is used by the MC92501 for marking cells as part of Relative Rate ABR.
- **VPI Size in ECI on Header mode (VPS)**—This bit determines the size of the VPI field for 'ECI on Header mode' (IHAF=1). See **Section 5.3.1** for details.
  - 0 = VPI size is 12 bits
  - 1 = VPI size is 8 bits
- **Identifier in Header Address Fields (IHAF)**—The IHAF bit, when set, indicates that the ECI is located in the header address fields of the cell structure as follows: If the VPI field of the header is non-zero, it is used as the ECI. If the VPI field of the header is zero, the VCI field is used as the ECI. When IHAF is set, the *Identifier Most-Significant Byte (IMSB)* and *Identifier Least-Significant Byte (ISLB)* fields are not used.
- **Egress Switch FIFO Control (ESFC)**—The ESFC bit determines the size of the FIFO in the Egress Switch Interface block.
  - 0 = 6-cell FIFO
  - 1 = 4-cell FIFO

NOTE: Normally, use a 6-cell FIFO. For applications in which it is necessary to reduce maximum cell delay through the MC92501, a 4-cell FIFO might be appropriate with certain switch interface architectures.

- **EFCI Enable (EFE)**—This bit enables the MC92501 to set the middle bit of the PTI in the header of an incoming cell if the EFCI bit received from the switch is set.
  - 0 = The EFCI bit is not defined in the overhead fields so EFCI = 0 is provided to

the Cell Processing block.
  - 1 = The EFCI bit from the overhead fields causes the PTI1 bit of the cell header to be set.
- **Multicast Translation Table Section Enable (MTSE)**—This bit determines whether the MTTS field is defined in the overhead fields received from the switch.
  - 0 = The MTTS field is not defined in the overhead fields so MTTS = 0000 is provided to the Cell Processing block
  - 1 = The MTTS field provided to the Cell Processing block is taken from the location in the overhead fields defined by the *MTTS Byte Location (MTBY)* and *MTTS Bit (MTBI)* location fields
- **Egress Address Translation Disable (EATD)**—This bit determines whether the MC92501 performs Address Translation on the cells arriving from the switch.
  - 0 = The MC92501 performs Address Translation in the Egress direction
  - 1 = The MC92501 does not perform Address Translation in the Egress direction, and the cell header is transferred transparently to the PHY.
- **Egress Link Number Selection (ELNS)**—This bit selects the source for the link number when *Egress Address Translation Disable (EATD)* is set.
  - 0 = The link number is zero
  - 1 = The link number is taken from the MTTS field whose location is determined by the *MTTS Byte (MTBY)* and *MTTS Bit (MTBI)* location fields

Table 7-7. Destination Link Number

EATD	ELNS	Destination Link Number
0	0	Taken from the Egress Translation Address word in the Context Parameters Table
0	1	Taken from the Egress Translation Address word in the Context Parameters Table
1	0	0
1	1	Taken from MTTS field

- **Egress Switch Parity Control (ESPC)**—This bit selects the Egress Switch parity mode: odd or even.
  - 0 = Odd parity is expected
  - 1 = Even parity is expected
- **Egress Switch Parity Enable (ESPR)**—This bit defines whether there is parity checking on the Egress Switch Interface.
  - 0 = Parity checking is disabled
  - 1 = Parity checking is enabled

- **Egress Payload Parity Enable (EPLP)**—This bit determines whether a parity error that is detected on the payload of a cell arriving from the switch should cause the cell to be removed from the cell flow. Parity checking is enabled by the *Egress Switch Parity Enable (ESPR)* bit.
  - 0 = A parity error detected on the payload of a cell does not cause the cell to be dropped
  - 1 = A parity error detected on the payload of a cell causes the cell to be dropped

Table 7-8. Parity Checking at Egress Switch Interface

EPPR	EPLP	Action when Parity Error is Detected on a Header/ Overhead/HEC Byte	Action when Parity Error is Detected on a Payload Byte
0	0	Ignore	Ignore
0	1	Ignore	Ignore
1	0	Discard cell at the switch interface	Ignore
1	1	Discard cell at the switch interface	Remove cell and copy to the Cell Extraction Queue

- **Egress Switch HEC Field (ESHF)**—The ESHF bit determines if the cells that are transferred from the switch contain the HEC octet.
  - 0 = The Switch does not transfer the HEC octet.
  - 1 = The Switch transfers the HEC octet. This octet is discarded by the MC92500.
- **Egress Switch Number Of Bytes (ESNB)**—The ESNB field determines the number of bytes in the data structure received from the switch.
  - 00000 = 65 bytes per cell are received from the switch
  - 00001 = 66 bytes per cell are received from the switch
  - 00010 = 67 bytes per cell are received from the switch
  - 00011 = 68 bytes per cell are received from the switch
  - 00100 = 69 bytes per cell are received from the switch
  - 00101 = 70 bytes per cell are received from the switch
  - 00110 = 71 bytes per cell are received from the switch
  - 00111 = 72 bytes per cell are received from the switch
  - 01000 = 73 bytes per cell are received from the switch
  - 01001 = 74 bytes per cell are received from the switch
  - 01010 = 75 bytes per cell are received from the switch
  - 01011 = 76 bytes per cell are received from the switch
  - 01100 = 77 bytes per cell are received from the switch
  - 01101 = 78 bytes per cell are received from the switch
  - 01110 = 79 bytes per cell are received from the switch

- 01111 = 80 bytes per cell are received from the switch
- 10000 = Reserved
- 10001 = Reserved
- 10010 = Reserved
- 10011 = 52 bytes per cell are received from the switch
- 10100 = 53 bytes per cell are received from the switch
- 10101 = 54 bytes per cell are received from the switch
- 10110 = 55 bytes per cell are received from the switch
- 10111 = 56 bytes per cell are received from the switch
- 11000 = 57 bytes per cell are received from the switch
- 11001 = 58 bytes per cell are received from the switch
- 11010 = 59 bytes per cell are received from the switch
- 11011 = 60 bytes per cell are received from the switch
- 11100 = 61 bytes per cell are received from the switch
- 11101 = 62 bytes per cell are received from the switch
- 11110 = 63 bytes per cell are received from the switch
- 11111 = 64 bytes per cell are received from the switch

- **Identifier Most-Significant Byte (IMSB)**—The IMSB field contains the byte number of the most-significant byte of the MI/ECI field within the switch data structure. The byte on which STXSOC is asserted is byte number 0.
- **Identifier Least-Significant Byte (ILSB)**—The ILSB field contains the byte number of the least-significant byte of the MI/ECI field within the switch data structure. The byte on which STXSOC is asserted is byte number 0.

7.2.6.7 Egress Switch Overhead Information Register (ESOIR0)

This register determines the location of the overhead information in the data structure received from the switch.

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

Figure 7-34. Egress Switch Overhead Information Register (ESOIR0) Fields

- **EFCI Byte (EFBY)** — The EFBY field contains the byte number of the switch data structure in which the EFCI can be found (overhead, header, and HEC bytes). The byte on which STXSOC is asserted is byte number 0.
- **EFCI Bit (EFBI)** — The EFBI field contains the number of the EFCI bit within the byte specified by the *EFCI Byte (EFBY)* location field. The most-significant bit is number 7, and the least-significant bit is number 0.
- **M Byte (MBY)** — The MBY field contains the byte number of the switch data structure in which the M bit can be found. The byte on which STXSOC is asserted is byte number 0.
- **M Bit (MBI)** — The MBI field contains the number of the M bit within the byte specified by the *M Byte (MBY)* location field. The most-significant bit is number 7 and the least-significant bit is number 0.
- **MTTS Byte (MTBY)** — The MTBY field contains the byte number of the switch data structure in which the MTTS field can be found. The byte on which STXSOC is asserted is byte number 0.
- **MTTS Bit (MTBI)** — This field indicates the location of the MTTS field within the byte specified by *MTTS Byte (MTBY)* Location field.
  - 0 = MTTS equals the value that reside in bits 7:5 of the byte pointed to by *MTTS Byte (MTBY)* Location field.
  - 1 = MTTS equals the value that reside in bits 7:6 of the byte pointed to by *MTTS Byte (MTBY)* Location field.
  - 2 = MTTS equals the value that reside in bit 7 of the byte pointed to by *MTTS Byte (MTBY)* Location field.
  - 3 = MTTS equals the value that reside in bits 3:0 of the byte pointed to by *MTTS Byte (MTBY)* Location field.
  - 4 = MTTS equals the value that reside in bits 4:1 of the byte pointed to by *MTTS Byte (MTBY)* Location field.
  - 5 = MTTS equals the value that reside in bits 5:2 of the byte pointed to by *MTTS Byte (MTBY)* Location field.
  - 6 = MTTS equals the value that reside in bits 6:3 of the byte pointed to by *MTTS Byte (MTBY)* Location field.
  - 7 = MTTS equals the value that reside in bits 7:4 of the byte pointed to by *MTTS Byte (MTBY)* Location field.

7.2.6.8 Egress Switch Overhead Information Register 1 (ESOIR1)

This register determines the location of the overhead information in the data structure received from the switch. The register has the following structure:

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

Figure 7-35. Egress Switch Overhead Information Register 1 (ESOIR1) Fields

- **EOCLP Byte (EOBY)**—This field contains the byte number of the switch data structure in which the *Egress Overhead CLP (EOCLP)* bit can be found (overhead, header and HEC bytes). The byte on which STXSOC is asserted is byte number 0. See **Section 6.5** for details.
- **EOCLP Bit Location (EOBI)**—This field contains the number of the *Egress Overhead CLP (EOCLP)* bit within the byte specified by the *EOCLP Byte (EOBY)* location field. The most-significant bit is 7, and the least-significant bit is 0. See **Section 6.5** for details.
- **IFS Byte Location (EIBY)**—This field contains the byte number of the switch data structure in which the *Overhead Ingress Flow Status (IFS)* bit can be found (overhead, header and HEC bytes). The byte on which STXSOC is asserted is byte number 0. See **Section 6.4.3.1** for details.
- **IFS Bit Location (EIBI)**—This field contains the number of the *Overhead Ingress Flow Status (IFS)* bit within the byte specified by the *EFS Byte (EEBY)* location field. The most-significant bit is 7, and the least-significant bit is 0. See **Section 6.4.3.1** for details.
- **EFS Byte Location (EEBY)**—This field contains the byte number of the switch data structure in which the *Overhead Egress Flow Status (EFS)* bit can be found (overhead, header and HEC bytes). The byte on which STXSOC is asserted is byte number 0. See **Section 6.4.3.2** for details.
- EFS Bit Location (EEBI)—This field contains the number of the *Overhead Egress Flow Status (EFS)* bit within the byte specified by the *EFCI Byte (EFBY)* location field. The most-significant bit is 7, and the least-significant bit is 0. See **Section 6.4.3.2** for details.

7.2.6.9 UNI Register (UNIR)

The UNI Register determines whether each of the links is treated as a UNI or as an NNI.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNI15	UNI14	UNI13	UNI12	UNI11	UNI10	UNI9	UNI8	UNI7	UNI6	UNI5	UNI4	UNI3	UNI2	UNI1	UNI0

Figure 7-36. UNI Register (UNIR) Fields

- **User-Network Interface (UNI15–UNI0)**—The UNI bits define the type of interface that the physical link crosses. It affects the interpretation of the four most significant bits of the ATM cell header.
  - 0 = The physical link does not cross a User-Network Interface (UNI). It may cross a Network-Network Interface (NNI), or it may be an intranetwork link.
  - 1 = The physical link crosses a UNI.

7.2.6.10 Ingress Processing Configuration Register (IPCR)

This register defines the MC92501 Ingress processing parameters.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	IGCTE	ICCR	IRCR	ISFCE	ISFNE	ISPE	ISBCE	ISBNE
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPCC			0	IGZC	IUHC	IIP	IROE	0	IBCC			0	IAPE		IACE

Figure 7-37. Ingress Processing Configuration Register (IPCR)

- **Global Ingress CLP Transparency Enable (IGCTE)**—This bit enables CLP transparency function on the Ingress. See **Section 6.5** for details.
- **Ingress Check CRC on RM cells (ICCR)**—This bit determines whether the CRC of RM cells that are received in the Ingress is checked.
  - 0 = The CRC of RM cells that are received in the Ingress is not checked.
  - 1 = The CRC of RM cells that are received in the Ingress is checked and if it is not o.k. then the cell is removed and can be copied to the microprocessor.
- **Ingress Recalculate CRC on RM cells (IRCR)**—This bit determines whether the CRC of Ingress RM cells is recalculated.
  - 0 = The CRC of Ingress RM cells is not recalculated.
  - 1 = The CRC of Ingress RM cells is recalculated.
- **Global Ingress Set FRM CI Enable (ISFCE)**—This bit enables setting CI bit in forward RM cells received in Ingress. See **Section 6.4.3.3** for details.
  - 0 = Setting CI bit in forward RM cells received in Ingress is disabled.
  - 1 = Setting CI bit in forward RM cells received in Ingress is enabled.

- **Global Ingress Set FRM NI Enable (ISFNE)**—This bit enables setting NI bit in forward RM cells received in Ingress. See **Section 6.4.3.3** for details.
  - 0 = Setting NI bit in forward RM cells received in Ingress is disabled.
  - 1 = Setting NI bit in forward RM cells received in Ingress is enabled.
- **Global Ingress Set PTI Enable (ISPE)**—This bit enables setting PTI[1] bit in cells with PTI[2]=0 which are received in Ingress. See **Section 6.4.3.3** for details.
  - 0 = Setting PTI[1] bit in cells with PTI[2] = 0 which are received in Ingress is disabled.
  - 1 = Setting PTI[1] bit in cells with PTI[2] = 0 which are received in Ingress is enabled.
- **Global Ingress Set BRM CI Enable (ISBCE)**—This bit enables setting CI bit in Backward RM cells received in Ingress. See **Section 6.4.3.3** for details.
  - 0 = Setting CI bit in backward RM cells received in Ingress is disabled.
  - 1 = Setting CI bit in backward RM cells received in Ingress is enabled.
- **Global Ingress Set BRM NI Enable (ISBNE)**—This bit enables setting NI bit in backward RM cells received in Ingress. See **Section 6.4.3.3** for details.
  - 0 = Setting NI bit in backward RM cells received in Ingress is disabled.
  - 1 = Setting NI bit in backward RM cells received in Ingress is enabled.
- **Ingress Policing Counters Control (IPCC)**—This field determines which counters appear in the Policing Counters Table if Ingress UPC is enabled (The *UPC Flow (UPCF)* bit in the ATMC CFB Configuration Register (ACR) is reset). It also determines the size of each record in the table. See **Section 7.3.6** for more details.
  - 000 = The Policing Table does not exist.
  - 001 = The Policing Table contains three counters and one reserved long word: DSCD0, DSCD1, TAG, Reserved.
  - 010 = The Policing Table contains three counters: DSCD0, DSCD1, TAG.
  - 011 = The Policing Table contains two counters: DSCD, TAG.
  - 100 = The Policing Table contains one counter: TAG.
  - 101 = The Policing Table contains one counter: DSCD.
  - 110 = Reserved
  - 111 = Reserved
- **Ingress GFC Zero Check (IGZC)**—This bit determines whether the MC92501 checks that the GFC header field is zero on cells arriving over a UNI.
  - 0 = The GFC field is not checked.
  - 1 = The GFC field is checked. If it is non-zero, the cell is copied to the Cell Extraction Queue.
- **Ingress Unallocated Header Bits Check (IUHC)**—This bit determines whether the MC92501 checks that unallocated bits of the cell header (as defined by the VPI Mask and VCI Mask—see **Section 5.2.2**) are zero.
  - 0 = The unallocated bits of the header are not checked
  - 1 = The unallocated bits of the header are checked. If one or more bits are set, the cell is copied to the Cell Extraction Queue and removed from the cell flow.

- **Ingress Insertion Priority (IIP)**—This bit determines the priority between inserted/generated cells and Ingress received cells. Note that Insertion is always limited by the Leaky Bucket mechanism.
  - 0 = Ingress received cells priority is higher than Inserted/generated cells.
  - 1 = Inserted/generated cells priority is higher than Ingress received cells.
- **Ingress RM Overlay Enable (IROE)**—This bit enables updating switch parameters words in the case of RM cell. See **Section 6.4.4** for details.
- **Ingress Billing Counters Control (IBCC)**—The IBCC field determines which counters appear in the Ingress Billing Counters Table. It also determines the size of each record in the table. See **Section 7.3.4** for more details.
  - 000 = The Ingress Billing Table does not exist.
  - 001 = The Ingress Billing Table contains four counters: IUCLP0, IUCLP1, IOCLP0, IOCLP1.
  - 010 = The Ingress Billing Table contains three counters: IUCLP0, IUCLP1, IOAM.
  - 011 = The Ingress Billing Table contains three counters and one reserved long word: IUCLP0, IUCLP1, IOAM, reserved.
  - 100 = The Ingress Billing Table contains two counters: IU, IOAM.
  - 101 = The Ingress Billing Table contains two counters: ICLP0, ICLP1.
  - 110 = The Ingress Billing Table contains one counter: ICNTR.
  - 111 = Reserved
- **Ingress Address Translation VPI Enable (IAPE)**—The IAPE field determines whether and how address translation is performed on the VPI field in the Ingress cell flow.
  - 00 = No translation is performed on the VPI field.
  - 01 = The entire VPI field is replaced.
  - 10 = The VPI field is replaced based on the UNI bit of the link (see **Section 7.2.6.9**). If the link is a UNI, only bits 7:0 of the VPI are replaced. Otherwise, the entire VPI field is replaced.
  - 11 = Reserved
- **Ingress Address Translation VCI Enable (IACE)**—The IACE bit determines whether address translation is performed on the VCI field in the Ingress cell flow.
  - 0 = No translation is performed on the VCI field.
  - 1 = The entire VCI field is replaced if the *Ingress Virtual Path Connection (IVPC)* bit of the connection's Ingress Parameters word is reset, indicating VC switching.

7.2.6.11 Egress Processing Configuration Register (EPCR)

This register defines the MC92501 Egress processing parameters.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	EGCTE	ECCR	ERCR	ESFCE	ESFNE	ESPE	ESBCE	ESBNE
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPCC			0	0	0	EIP	0	0	EBCC			0	RGFC	0	0

Figure 7-38. Egress Processing Configuration Register (EPCR) Fields

- **Global Egress CLP Transparency Enable (EGCTE)**—This bit enables CLP transparency function on the Egress. See **Section 6.5** for details.
- **Egress Check CRC on RM cells (ECCR)**—This bit determines whether the CRC of RM cells that are received in the Egress is checked.
  - 0 = The CRC of RM cells that are received in the Egress is not checked.
  - 1 = The CRC of RM cells that are received in the Egress is checked and if it is not o.k. then the cell is removed and can be copied to the microprocessor.
- **Egress Recalculate CRC on RM cells (ERCR)**—This bit determines whether the CRC of Egress RM cells is recalculated.
  - 0 = The CRC of Egress RM cells is not recalculated.
  - 1 = The CRC of Egress RM cells is recalculated.
- **Global Egress Set FRM CI Enable (ESFCE)**—This bit enables setting CI bit in forward RM cells received in Egress. See **Section 6.4.3.4** for details.
  - 0 = Setting CI bit in forward RM cells received in Egress is disabled.
  - 1 = Setting CI bit in forward RM cells received in Egress is enabled.
- **Global Egress Set FRM NI Enable (ESFNE)**—This bit enables setting NI bit in forward RM cells received in Egress. See **Section 6.4.3.4** for details.
  - 0 = Setting NI bit in forward RM cells received in Egress is disabled.
  - 1 = Setting NI bit in forward RM cells received in Egress is enabled.
- **Global Egress Set PTI Enable (ESPE)**—This bit enables setting PTI[1] bit in cells with PTI[2]=0 which are received in Egress. See **Section 6.4.3.4** for details.
  - 0 = Setting PTI[1] bit in cells with PTI[2]=0 which are received in Egress is disabled.
  - 1 = Setting PTI[1] bit in cells with PTI[2]=0 which are received in Egress is enabled.
- **Global Egress Set BRM CI Enable (ESBCE)**—This bit enables setting CI bit in backward RM cells received in Egress. See **Section 6.4.3.4** for details.
  - 0 = Setting CI bit in backward RM cells received in Egress is disabled.
  - 1 = Setting CI bit in backward RM cells received in Egress is enabled.
- **Global Egress Set BRM NI Enable (ESBNE)**—This bit enables setting NI bit in backward RM cells received in Egress. See **Section 6.4.3.4** for details.
  - 0 = Setting NI bit in backward RM cells received in Egress is disabled.
  - 1 = Setting NI bit in backward RM cells received in Egress is enabled.

- **Egress Policing Counters Control (EPCC)—** This field determines which counters appear in the Policing Counters Table if Egress UPC is enabled (The *UPC Flow (UPCF)* bit in the ATMC CFB Configuration Register (ACR) is set). It also determines the size of each record in the table. See **Section 7.3.6** for more details.
  - 000 = The Policing Table does not exist.
  - 001 = The Policing Table contains three counters and one reserved long word: DSCD0, DSCD1, TAG, Reserved.
  - 010 = The Policing Table contains three counters: DSCD0, DSCD1, TAG.
  - 011 = The Policing Table contains two counters: DSCD, TAG.
  - 100 = The Policing Table contains one counter: TAG.
  - 101 = The Policing Table contains one counter: DSCD.
  - 110 = Reserved
  - 111 = Reserved
- **Egress Insertion Priority (EIP)—** This bit determines the priority between inserted/generated cells and Egress received cells.

NOTE: Insertion is always limited by the Leaky Bucket mechanism.

  - 0 = Inserted/generated cells priority is higher than Egress received cells.
  - 1 = Egress received cells priority is higher than Inserted/generated cells.
- **Egress Billing Counters Control (EBCC)—** The EBCC field determines which counters appear in the Egress Billing Counters Table. It also determines the size of each record in the table. See **Section 7.3.5** for more details.
  - 000 = The Egress Billing Table does not exist.
  - 001 = The Egress Billing Table contains four counters: EUCLP0, EUCLP1, EOCLP0, EOCLP1.
  - 010 = The Egress Billing Table contains three counters: EUCLP0, EUCLP1, EOAM.
  - 011 = The Egress Billing Table contains three counters and one reserved long word: EUCLP0, EUCLP1, EOAM, reserved.
  - 100 = The Egress Billing Table contains two counters: EU, EOAM.
  - 101 = The Egress Billing Table contains two counters: ECLP0, ECLP1.
  - 110 = The Egress Billing Table contains one counter: ECNTR.
  - 111 = Reserved
- **Replace GFC Field (RGFC)—** The RGFC bit determines whether, during address translation, the GFC field of the ATM header is replaced by the value provided in the Egress Translation Address word in the External Memory. This bit is only relevant for connections belonging to a UNI.
  - 0 = The GFC field is not replaced.
  - 1 = The GFC field is replaced.

7.2.6.12 Egress Multicast Configuration Register (EMCR)

This register contains the MC92501 Egress multicast translation parameters.

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

Figure 7-39. Egress Multicast Configuration Register (EMCR) Fields

- **Multicast Translation Table Control (MLTC)**—The MLTC bit determines the existence of the Multicast Translation Table in External Memory.
  - 0 = The Multicast Translation Table does not exist.
  - 1 = The Multicast Translation Table exists.
- **Egress Multicast Identifier Control (EMIC)**—This field determines the number of bits of the Multicast Identifier that are allocated for Multicast Translation (see **Section 5.3.2**).
  - 000 = 9 bits of the Multicast Identifier are allocated
  - 001 = 10 bits of the Multicast Identifier are allocated
  - 010 = 11 bits of the Multicast Identifier are allocated
  - 011 = 12 bits of the Multicast Identifier are allocated
  - 100 = 13 bits of the Multicast Identifier are allocated
  - 101 = 14 bits of the Multicast Identifier are allocated
  - 110 = 15 bits of the Multicast Identifier are allocated
  - 111 = 16 bits of the Multicast Identifier are allocated

7.2.6.13 ATMC CFB Configuration Register (ACR)

This register defines ATMC CFB general processing parameters.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ATC		SPC		COMC	INPC	EGPC	DVTC			FLGC	OAMC	VPRP	FTM	CRRP	PMAC
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UPCF	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-40. ATMC CFB Configuration Register (ACR) Fields

- **Address Translation Control (ATC)**—The ATC bit determines whether the Address Translation long words exist in the Context Parameters Table record.
  - 00 = No Address Translation words exist.
  - 01 = One common Address Translation word exists.
  - 10 = Both Address Translation words exist.
  - 11 = Reserved

- **Ingress Switch Parameter Control (SPC)** — This field determines the number of long words of Switch Parameters per record in the Context Parameters Table in External Memory.
  - 00 = No Switch Parameters
  - 01 = One long word of Switch Parameters per record in the Context Parameters Table
  - 10 = Two long words of Switch Parameters per record in the Context Parameters Table
  - 11 = Three long words of Switch Parameters per record in the Context Parameters Table
- **Common Parameters Control (COMC)** — The COMC bit determines whether the Common Parameters long word exists in the Context Parameters Table record.
  - 0 = The Common Parameters long word does not exist.
  - 1 = The Common Parameters long word exists.
- **Ingress Parameters Control (INPC)** — The INPC bit determines whether the Ingress Parameters long word exists in the Context Parameters Table record.
  - 0 = The Ingress Parameters long word does not exist.
  - 1 = The Ingress Parameters long word exists.
- **Egress Parameters Control (EGPC)** — The EGPC bit determines whether the Egress Parameters long word exists in the Context Parameters Table record.
  - 0 = The Egress Parameters long word does not exist.
  - 1 = The Egress Parameters long word exists.
- **Dump Vector Table Control (DVTC)** — The DVTC field determines the size of the Dump Vector Table in External Memory. Each record consists of 2 long words. See **Section 7.3.13** for details.
  - 000 = The Dump Vector Table does not exist.
  - 001 = The Dump Vector Table contains 128 records.
  - 010 = The Dump Vector Table contains 512 records.
  - 011 = The Dump Vector Table contains 2K records.
  - 100 = The Dump Vector Table contains 8K records.
  - 101 = The Dump Vector Table contains 32K records.
  - 110 = The Dump Vector Table contains 128K records.
  - 111 = The Dump Vector Table contains 512K records.
- **Flags Table Control (FLGC)** — The FLGC bit determines the existence of the Flags Table in External Memory.
  - 0 = The Flags Table does not exist.
  - 1 = The Flags Table exists.
- **OAM Table Control (OAMC)** — The OAMC bit determines the existence of the OAM Table in External Memory.
  - 0 = The OAM Table does not exist.
  - 1 = The OAM Table exists.

- **VP RM Cell PTI (VPRP)**—This bit determines whether a cell is a VP RM cell only if its PTI = 6.
  - 0 = A cell is a VP RM cell if and only if it belongs to a VP connection, its VCI = 6 and its PTI = 6.
  - 1 = A cell is a VP RM cell if and only if it belongs to a VP connection and its VCI = 6.
- **FMC Time-Stamp Enable (FTM)**—The FTM bit determines the encoding of the Time-Stamp field of OAM Forward Monitoring Cells generated by the MC92501.
  - 0 = The Time-Stamp field is encoded with the default value of all 1s.
  - 1 = The Time-Stamp field is encoded with the current value of the Cell Time Register (CLTM).
- **VC RM cell Removal Point (CRRP)**—This bit determines whether a VC cell whose PTI = 6 or 7 is removed at the OAM termination point or whether its removal is subjected to the per-connection enable bits for PTI = 6 or PTI = 7.
  - 0 = A VC cell whose PTI = 6 or 7 is removed at the OAM termination point as defined by the *Egress End-to-end OAM Termination (EEOT)* bit in the Egress and by the *Ingress End-to-end OAM Termination (IEOT)* bit in the Ingress.
  - 1 = A VC cell is removed at the Egress if the *Egress PTI 6 Remove (EP6R)* bit is set and its PTI = 6 or if the *Egress PTI 7 Remove (EP7R)* bit is set and its PTI = 7. A VC cell is removed at the Ingress if the *Ingress PTI 6 Remove (EP6R)* bit is set and its PTI = 6 or if the *Ingress PTI 7 Remove (IP7R)* bit is set and its PTI = 7.
- **PM on All Connections (PMAC)**—This bit determines whether OAM Performance Monitoring test can be done on all connections or on 64 connections.
  - 0 = Performance Monitoring can be done only on 64 selected connections.
  - 1 = Performance Monitoring can be done on all connections.
- **UPC Flow (UPCF)**—This bit determines whether the UPC is active in the Ingress flow or in the Egress flow.
  - 0 = The UPC is active in the Ingress flow.
  - 1 = The UPC is active in the Egress flow.

7.2.6.14 General Configuration Register (GCR)

This register determines the configuration of those sections of the MC92501 not contained in the ATMC CFB.

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

Figure 7-41. General Configuration Register (GCR) Fields

- **Context Parameters Extension Table Control (CMPC)**—This bit determines the existence of the Context Parameters Extension Table in External memory. See **Section 7.3.16** for details.
  - 0 = The Context Parameters Extension Table does not exist.
  - 1 = The Context Parameters Extension Table exists.
- **PHY ID Control (PHIDC)**—The PHIDC bit determines the functionality of the TXPHYID pins. See **Section 4.2.2** for details.
  - 0 = The TXPHYID pins contain the ID of the currently transmitted cell.
  - 1 = The TXPHYID pins contain the ID of the next cell to be transmitted.
- **Ingress Link Counters Table Control (ILCC)**—The ILCC bit determines the existence of the Ingress Link Counters Table in External Memory.
  - 0 = The Ingress Link Counters Table does not exist.
  - 1 = The Ingress Link Counters Table exists.
- **Egress Link Counters Table Control (ELCC)**—The ILCC bit determines the existence of the Egress Link Counters Table in External Memory.
  - 0 = The Egress Link Counters Table does not exist.
  - 1 = The Egress Link Counters Table exists.

7.2.6.15 Context Parameters Table Pointer Register (CPTP)

This register contains the pointer to the first word of the Context Parameters Table. The pointer is in units of 256 bytes.

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

Figure 7-42. Context Parameters Table Pointer (CPTP) Fields

7.2.6.16 OAM Table Pointer Register (OTP)

This register contains the pointer to the first word of the OAM Table. The pointer is in units of 256 bytes.

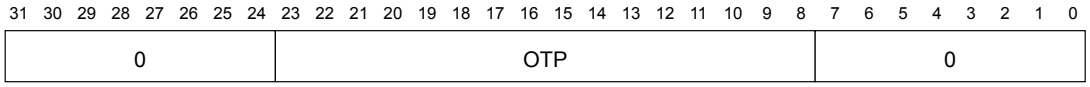


Figure 7-43. OAM Table Pointer (OTP) Register Fields

7.2.6.17 Dump Vector Table Pointer Register (DVTP)

This register contains the pointer to the first word of the Dump Vector Table. The pointer is in units of 256 bytes.

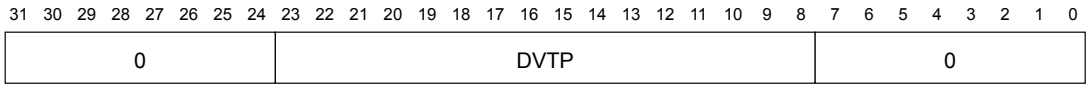


Figure 7-44. Dump Vector Table Pointer (DVTP) Register Fields

7.2.6.18 VC Table Pointer Register (VCTP)

This register contains the pointer to the first word of the VC Table. The pointer is in units of 256 bytes.

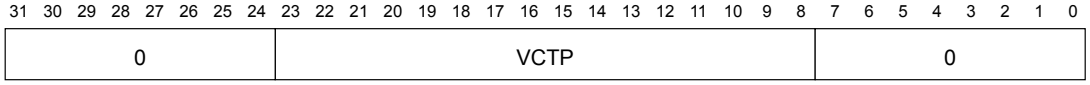


Figure 7-45. VC Table Pointer (VCTP) Register Fields

7.2.6.19 Multicast Translation Table Pointer Register (MTTP)

This register contains the pointer to the first word of the Multicast Translation Table. The pointer is in units of 256 bytes.

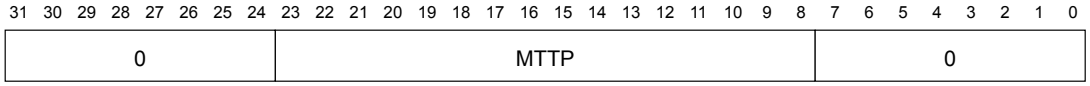


Figure 7-46. Multicast Translation Table Pointer (MTTP) Register Fields

7.2.6.20 Flags Table Pointer Register (FTP)

This register contains the pointer to the first word of the Flags Table. The pointer is in units of 256 bytes.

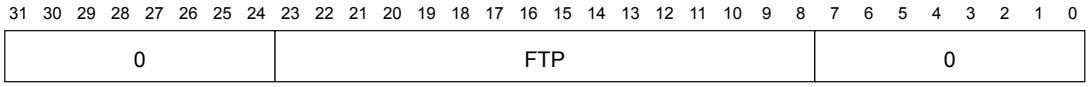


Figure 7-47. Flags Table Pointer (FTP) Register Fields

7.2.6.21 Egress Link Counters Table Pointer Register (ELCTP)

This register contains the pointer to the first word of the Egress Link Counters Table. Each pointer is in units of 256 bytes.

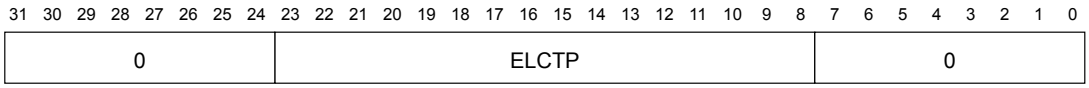


Figure 7-48. Egress Link Counters Table Pointer (ELCTP) Register Fields

7.2.6.22 Ingress Link Counters Table Pointer Register (ILCTP)

This register contains the pointer to the first word of the Ingress Link Counters Table. Each pointer is in units of 256 bytes.

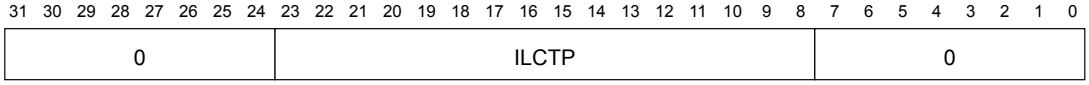


Figure 7-49. Ingress Link Counters Table Pointer (ILCTP) Register Fields

7.2.6.23 Context Parameters Extension Table Pointer Register (CPETP)

This register contains the pointer to the first word of the Context Parameters Extension Table. The pointer is in units of 256 bytes.

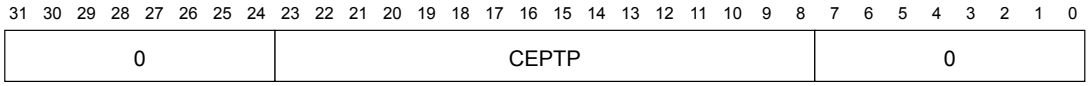


Figure 7-50. Context Parameters Extension Table Pointer (CPETP) Register Fields

7.2.6.24 Node ID Register 0 (ND0)

This register contains the most significant bits of the Node ID.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Node_ID (127:112)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Node_ID (111:96)															

Figure 7-51. Node ID Register 0 (ND0) Fields

7.2.6.25 Node ID Register 1 (ND1)

This register contains the upper-middle portion of the Node ID.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Node_ID (95:80)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Node_ID (79:64)															

Figure 7-52. Node ID Register 1 (ND1) Fields

7.2.6.26 Node ID Register 2 (ND2)

This register contains the lower-middle portion of the Node ID.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Node_ID (63:48)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Node_ID (47:32)															

Figure 7-53. Node ID Register 2 (ND2) Fields

7.2.6.27 Node ID Register 3 (ND3)

This register contains the least significant bits of the Node ID.

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

Figure 7-54. Node ID Register 3 (ND3) Fields

7.2.6.28 Ingress VCI Copy Register (IVCR)

The Ingress VCI Copy Register is used to select specific reserved values of VCI for copying to the processor from the Ingress cell flow. The values contained in this register are used by all VPCs which have the *Ingress VCR/VRR Registers Enable (IVRE)* bit of the Ingress Parameters word in External Memory set.

NOTE: The definition of this register may change in future revisions of the MC92501 to reflect the evolving ATM standards.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IVC31	IVC30	IVC29	IVC28	IVC27	IVC26	IVC25	IVC24	IVC23	IVC22	IVC21	IVC20	IVC19	IVC18	IVC17	IVC16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IVC15	IVC14	IVC13	IVC12	IVC11	IVC10	IVC9	IVC8	IVC7	IVC6	IVC5	0	0	IVC2	IVC1	IVC0

Figure 7-55. Ingress VCI Copy Register (IVCR) Fields

- **Ingress VCI Copy (IVC31–IVC5, IVC2–IVC0)**—Each IVC bit refers to a VCI value. Each IVC bit, when set, indicates that cells received in the Ingress cell flow with that VCI value should be copied to the Cell Extraction Queue.

7.2.6.29 Egress VCI Copy Register (EVCR)

The Egress VCI Copy Register is used to select specific reserved values of VCI for copying to the processor from the Egress cell flow. The values contained in this register are used by all VPCs which have the *Egress VCR/VRR Registers Enable (EVRE)* bit of the Egress Parameters word in External Memory set.

NOTE: The definition of this register may change in future revisions of the MC92501 to reflect the evolving ATM standards.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EVC31	EVC30	EVC29	EVC28	EVC27	EVC26	EVC25	EVC24	EVC23	EVC22	EVC21	EVC20	EVC19	EVC18	EVC17	EVC16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVC15	EVC14	EVC13	EVC12	EVC11	EVC10	EVC9	EVC8	EVC7	EVC6	EVC5	0	0	EVC2	EVC1	EVC0

Figure 7-56. Egress VCI Copy Register (EVCR) Fields

- **Egress VCI Copy (EVC31–EVC5, EVC2–EVC0)**—Each EVC bit refers to a VCI value. Each EVC bit, when set, indicates that cells received in the Egress cell flow with that VCI value should be copied to the Cell Extraction Queue.

7.2.6.30 Ingress VCI Remove Register (IVRR)

The Ingress VCI Remove Register is used to select specific reserved values of VCI for removal from the cell flow. The values contained in this register are used by all VPCs which have the *Ingress VCR/VRR Registers Enable (IVRE)* bit of the Ingress Parameters word in External Memory set.

NOTE: The definition of this register may change in future revisions of the MC92501 to reflect the evolving ATM standards.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IVR31	IVR30	IVR29	IVR28	IVR27	IVR26	IVR25	IVR24	IVR23	IVR22	IVR21	IVR20	IVR19	IVR18	IVR17	IVR16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IVR15	IVR14	IVR13	IVR12	IVR11	IVR10	IVR9	IVR8	IVR7	IVR6	IVR5	0	0	IVR2	IVR1	IVR0

Figure 7-57. Ingress VCI Remove Register (IVRR) Fields

- **Ingress VCI Remove (IVR31–IVR5, IVR2–IVR0)**—Each IVR bit refers to a VCI value. The IVR bit, when set, indicates that cells received in the Ingress cell flow with that VCI value should be removed from the cell flow.

7.2.6.31 Egress VCI Remove Register (EVRR)

The Egress VCI Remove Register is used to select specific reserved values of VCI for removal from the cell flow. The values contained in this register are used by all VPCs which have the *Egress VCR/VRR Registers Enable (EVRE)* bit of the Egress Parameters word in External Memory set.

NOTE: The definition of this register may change in future revisions of the MC92501 to reflect the evolving ATM standards.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EVR31	EVR30	EVR29	EVR28	EVR27	EVR26	EVR25	EVR24	EVR23	EVR22	EVR21	EVR20	EVR19	EVR18	EVR17	EVR16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVR15	EVR14	EVR13	EVR12	EVR11	EVR10	EVR9	EVR8	EVR7	EVR6	EVR5	0	0	EVR2	EVR1	EVR0

Figure 7-58. Egress VCI Remove Register (EVRR) Fields

- **Egress VCI Remove (EVR31–EVR5, EVR2–EVR0)**—Each EVR bit refers to a VCI value. The EVR bit, when set, indicates that cells received in the Egress cell flow with that VCI value should be removed from the cell flow.

7.2.6.32 Performance Monitoring Exclusion Register (PMER)

The Performance Monitoring Exclusion Register determines which of the reserved values of VCI and PTI should be excluded from OAM Performance Monitoring blocks. This register is provided because this issue has not yet been standardized, and any aberration from the standard may destroy the block test results.

NOTE: The definition of this register may change in future revisions of the MC92501 to reflect the evolving ATM standards.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PVE31	PVE30	PVE29	PVE28	PVE27	PVE26	PVE25	PVE24	PVE23	PVE22	PVE21	PVE20	PVE19	PVE18	PVE17	PVE16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PVE15	PVE14	PVE13	PVE12	PVE11	PVE10	PVE9	PVE8	PVE7	PVE6	PVE5	PTE7	PTE6	PVE2	PVE1	PVE0

Figure 7-59. Performance Monitoring Exclusive Register (PMER) Fields

- **Performance Monitoring VCI Exclude (PVE31–PVE5, PVE2–PVI0)**—Each PVE bit refers to a VCI value. The PVE bit, when set, indicates that cells belonging to a Virtual Path Connection and having that VCI value should be excluded from any Performance Monitoring block test at the F4 level.
- **Performance Monitoring PTI Exclude (PTE7–PTE6)**—Each PTE bit refers to a PTI value. The PTE bit, when set, indicates that cells belonging to a Virtual Channel Connection and having that PTI value should be excluded from any Performance Monitoring block test at the F5 level.

7.2.6.33 External Memory Timing Configuration Register (EMTCR)

The External Memory Timing Configuration Register determines the timing of the External Memory control and data signals with respect to ACLK. See **Section 4.4.2** for details.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EMLBSW			0	EMTBSW			0	EMLBSR			0	0	0	0	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EMLWR			0	EMTWR			0	EMDDSP			0	EMDDEP			0

Figure 7-60. External Memory Timing Configuration Register (EMTCR) Fields

All of the fields are defined as follows:

- 000 = The event occurs at the ACLK rising edge.
- 001 = The event occurs 1/8 of an ACLK period following the ACLK rising edge.
- 010 = The event occurs 2/8 of an ACLK period following the ACLK rising edge.
- 011 = The event occurs 3/8 of an ACLK period following the ACLK rising edge.
- 100 = The event occurs 4/8 of an ACLK period following the ACLK rising edge.

- 101 = The event occurs 5/8 of an ACLK period following the ACLK rising edge.
- 110 = The event occurs 6/8 of an ACLK period following the ACLK rising edge.
- 111 = The event occurs 7/8 of an ACLK period following the ACLK rising edge.

- **External Memory Leading Edge of Bank Select for Write Cycle (EMLBSW)** — The EMLBSW field determines the timing of the assertion of the External Memory bank select signals with respect to ACLK during a write cycle.
- **External Memory Trailing Edge of Bank Select for Write Cycle (EMTBSW)** — The EMTBSW field determines the timing of the deassertion of the External Memory bank select signals with respect to ACLK during a write cycle.
- **External Memory Leading Edge of Bank Select for Read Cycle (EMLBSR)** — The EMLBSR field determines the timing of the assertion of the External Memory bank select signals with respect to ACLK during a read cycle.
- **External Memory Leading Edge of Write (EMLWR)** — The EMLWR field determines the timing of the assertion of the EMWR signal with respect to ACLK during a write cycle.
- **External Memory Trailing Edge of Write (EMTWR)** — The EMTWR field determines the timing of the deassertion of the EMWR signal with respect to ACLK during a write cycle.
- **External Memory Data Drive Start Point (EMDDSP)** — The EMDDSP field determines when the MC92501 begins to drive the External Memory Data bus during a write cycle.
- **External Memory Data Drive End Point (EMDDEP)** — The EMDDEP field determines when the MC92501 stops driving the External Memory Data bus during a write cycle.

Table 7-9 shows the legal values for each field and the value after reset (shaded).

Table 7-9. External Memory Timing Parameters

EMLBSW	EMTBSW	EMLBSR	EMLWR	EMTWR	EMDDSP	EMDDEP
			001		001	
010		010	010		010	
011		011	011	011	011	
100	100	100	100	100	100	100
101	101	101	101	101	101	101
	110			110		110
	111			111		111
						000

7.2.6.34 External Memory Interface Configuration Register (EMICR)

The External Memory Interface Configuration Register defines the configuration of the External Memory and affects the use of the External Memory Bank Enable signals. See **Section 4.4.1** for details.

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

Figure 7-61. External Memory Interface Configuration Register (EMICR) Fields

- **Number of External Memory Banks (NEMB)**—The NEMB field determines the number of banks of External Memory being used.
  - 00 = All of the External Memory is one bank. All of the External Memory Bank Select signals are asserted for every access.
  - 01 = The External Memory is divided into two banks. The $\overline{\text{EMBSH/L0}}$ and $\overline{\text{EMBSH/L2}}$ signals are asserted together when Bank 0 is addressed, and the EMBSH/L1 and EMBSH/L3 signals are asserted together when Bank 1 is addressed. The bank is determined by the most significant active bit (as defined by the *External Memory Address Space (EMAS)* field) of the External Memory Address.
  - 10 = The External Memory is divided into four banks. Each pair of External Memory Bank Select signals (0–3) is asserted when the corresponding bank is addressed. The bank is determined by the two most significant active bits (as defined by the *External Memory Address Space (EMAS)* field) of the External Memory Address.
  - 11 = Reserved
- **External Memory Address Space (EMAS)**—The EMAS field determines the number of active bits of the External Memory Address. This value affects the determination of which External Memory Bank Select signals to assert.
  If NEMB is 0, EMAS is not used.
  - 000 = EMADD(23) is the most significant active bit. (16 MB of EM address space)
  - 001 = EMADD(22) is the most significant active bit. (8 MB of EM address space)
  - 010 = EMADD(21) is the most significant active bit. (4 MB of EM address space)
  - 011 = EMADD(20) is the most significant active bit. (2 MB of EM address space)
  - 100 = EMADD(19) is the most significant active bit. (1 MB of EM address space)
  - 101 = EMADD(18) is the most significant active bit. (512 KB of EM address space)
  - 110 = Reserved
  - 111 = Reserved

7.2.6.35 RM Overlay Register (RMOR)

This register contains all the parameters which are related to RM cell overlay. Refer to **Section 6.4.4** for details. The register has the following structure:

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

Figure 7-62. RM Overlay Register (RMOR) Fields

- **Ingress BRM Overlay Enable (IBOE)** — This bit determines whether the MC92501 overlays the *RM Overlay Field (ROF)* on the switch parameters for Ingress Backward RM cells.
  - 0 = Switch parameters are not overlayed when a Backward RM cell is received in the Ingress.
  - 1 = Switch parameters are overlayed when a Backward RM cell is received in the Ingress
- **Ingress FRM Overlay Enable (IFOE)** — This bit determines whether the MC92501 overlays the *RM Overlay Field (ROF)* on the switch parameters for Ingress Forward RM cells.
  - 0 = Switch parameters are not overlayed when a Forward RM cell is received in the Ingress.
  - 1 = Switch parameters are overlayed when a Forward RM cell is received in the Ingress
- **RM Overlay Location (ROL)** — This field contains the number of the switch parameters byte which should be overlayed.
- **RM Overlay Mask (ROM)** — This field contains the byte mask which serves for overlaying the *RM Overlay Field (ROF)* over Ingress switch parameters byte.
- **RM Overlay Field (ROF)** — This field contains the byte which is overlayed on Ingress switch parameters byte. Each bit in this field is overlayed on the corresponding bit in the Ingress switch parameters only if it is enabled by the corresponding bit in the *RM Overlay Mask (ROM)* field.

7.2.6.36 CLP Transparency Overlay Register (CTOR)

This register contains the location of the *Ingress Overhead CLP (IOCLP)* bit in the Ingress switch parameters. See **Section 6.5** for details. The register has the following structure:

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

Figure 7-63. CLP Transparency Overlay Register (CTOR) Fields

- **IOCLP Byte location (OCBL)**—This field contains the byte number within the switch parameter word on which the *Ingress Overhead CLP (IOCLP)* bit is located. The most-significant byte is byte number 0 and the least-significant byte is byte number 3.
- **IOCLP Bit location (OCBI)**—This field contains the number of the *Ingress Overhead CLP (IOCLP)* bit within the byte specified by the *IOCLP Byte Location (OCBL)* field. The most-significant bit is number 7, and the least-significant bit is number 0.

7.2.6.37 Egress Overhead Manipulation Register (EGOMR)

This register contains fields for manipulating Egress overhead fields. See **Section 5.3.1** for details.

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

Figure 7-64. Egress Overhead Manipulation Register (EGOMR) Fields

- **Egress Cell Processing Block M bit Source (ECMS)**— This field contains the source for the M bit which is used by the Egress Cell Processing Block
  - 00 = The M bit used by the Egress Cell Processing block is taken from the M bit which is extracted from the switch cell data structure.
  - 01 = The M bit used by the Egress Cell Processing block is taken from the logical NOT of the M bit which is extracted from the switch cell data structure.
  - 10 = The M bit used by the Egress Cell Processing block is 0.
  - 11 = The M bit used by the Egress Cell Processing block is 1.
- **Egress Cell Processing Block MTTS Size (ECTS)**—This field contains the size of the MTTS field which is used by the Egress Cell Processing block.
  - 00 = The MTTS field which is used by the Egress Cell processing block is the MTTS field which is extracted from the switch cell data structure.
  - 01 = The MTTS field which is used by the Egress Cell processing block is the least significant bit of the MTTS field which is extracted from the switch cell data structure.

- 10 = The MTTS field which is used by the Egress Cell processing block is the 2 least significant bits of the MTTS field which is extracted from the switch cell data structure.
- 11 = The MTTS field which is used by the Egress Cell processing block is the 3 least significant bits of the MTTS field which is extracted from the switch cell data structure.

- **Egress Cell Processing Block ECI size (ECES)** — This field contains the size of the ECI field which is used by the Egress Cell Processing block.
  - 0000 = The ECI field which is used by the Egress Cell processing block is the ECI field which is extracted from the switch cell data structure.
  - 0001 = Reserved.
  - 0010 = Reserved.
  - 0011 = Reserved.
  - 0100 = Reserved.
  - 0101 = Reserved.
  - 0110 = The ECI field which is used by the Egress Cell processing block is the 6 least significant bits of the ECI field which is extracted from the switch cell data structure.
  - 0111 = The ECI field which is used by the Egress Cell processing block is the 7 least significant bits of the ECI field which is extracted from the switch cell data structure.
  - 1000 = The ECI field which is used by the Egress Cell processing block is the 8 least significant bits of the ECI field which is extracted from the switch cell data structure.
  - 1001 = The ECI field which is used by the Egress Cell processing block is the 9 least significant bits of the ECI field which is extracted from the switch cell data structure.
  - 1010 = The ECI field which is used by the Egress Cell processing block is the 10 least significant bits of the ECI field which is extracted from the switch cell data structure.
  - 1011 = The ECI field which is used by the Egress Cell processing block is the 11 least significant bits of the ECI field which is extracted from the switch cell data structure.
  - 1100 = The ECI field which is used by the Egress Cell processing block is the 12 least significant bits of the ECI field which is extracted from the switch cell data structure.
  - 1101 = The ECI field which is used by the Egress Cell processing block is the 13 least significant bits of the ECI field which is extracted from the switch cell data structure.
  - 1110 = The ECI field which is used by the Egress Cell processing block is the 14 least significant bits of the ECI field which is extracted from the switch cell data structure.
  - 1111 = The ECI field which is used by the Egress Cell processing block is the 15 least significant bits of the ECI field which is extracted from the switch cell data structure.

7.2.7 Pseudo-Registers

These registers are use to perform certain operations on the MC92501, and may be written by the processor in either of the MC92501 modes of operation (Setup Mode or Operate Mode).

7.2.7.1 Software Reset Register (SRR)

A write access to this pseudo-register location resets the MC92501. All the internal registers and FIFOs are loaded with their default values, and the MC92501 is transferred to (remains in) Setup Mode. See **Section 3.1.3** for more information.

7.2.7.2 Start SCAN Register (SSR)

A write access to this pseudo-register location starts the Internal Scan operation. If the Internal Scan process is still operating (didn't finish scanning all the connections), this access is ignored.

7.2.7.3 Enter Operate Mode Register (EOMR)

A write access to this pseudo-register location transfers the MC92501 to Operate Mode within 1–2 cell times. The processor may use the *Operate Mode (OM)* bit in the Interrupt Register (IR) to find out when the MC92501 is in Operate Mode. (See **Section 3.1.2** for more information.)

7.2.8 External Address Compression Device Access

This memory space (ADD(25:24) = 01) provides the processor with access to the external address compression device. The value of bits 23:2 of the MADD input pins are copied to the EMADD output pins, and the EACEN pin is asserted. The data is transferred between MDATA and EMDATA. See **Section 7.2.8** for details.

7.2.9 Maintenance Access

This memory space (ADD(25) = 1) provides the processor with access to the MC92501 External Memory for maintenance purposes. The value of bits 23:2 of the MADD input pins are copied to the EMADD output pins, and the External Memory Interface control pins are driven as necessary. The data is transferred between MDATA and EMDATA. If bit 24 of MADD is set, the MC92501 automatically writes back zero to each word of External Memory that is read. See **Section 4.4.3** for details.

Table 7-10. Address Space for Accesses that use the External Memory Interface

ADD(25:0)	Access Type
01_AAAA_AAAA_AAAA_AAAA_AAAA_AAA0	External Address Compression Device
10_AAAA_AAAA_AAAA_AAAA_AAAA_AAA0	External Memory
11_AAAA_AAAA_AAAA_AAAA_AAAA_AAA0	External Memory with write back of 0

7.3 External Memory Description

The MC92501 uses External Memory to store the database of information relating to the processing of cells on a per-connection basis. The MC92501 accesses the External Memory using 16- or 32-bit accesses.

7.3.1 Memory Partitioning

The External Memory is partitioned into several tables:

- **Context Parameters Table**—This table contains a record for each active connection that contains connection-specific information for processing and routing the cells belonging to the connection. See **Section 7.3.3** for details.
- **Context Parameters Extension Table**—This table includes a record for each active connection that contains connection additional connection-specific information for processing and routing the cells belonging to the connection. See **Section 7.3.16** for details.
- **Ingress Billing Counters Table**—This table includes a record for each active connection that contains the cell counters used by the connection during the normal Ingress cell flow. The table is dynamic and updated by the MC92501. The microprocessor is responsible for collecting the contents of the counters on a regular basis. See **Section 7.3.4** for details.
- **Egress Billing Counters Table**—This table includes a record for each active connection that contains the counters used by the connection during the normal Egress cell flow. The table is dynamic and updated by the MC92501. The microprocessor is responsible for collecting the contents of the counters on a regular basis. See **Section 7.3.5** for details.
- **Policing Counters Table**—This table includes a record for each active connection that contains the counters used to record the results of the UPC/NPC policing. This table is dynamic and updated by the MC92501. The microprocessor is responsible for collecting the contents of the counters on a regular basis. See **Section 7.3.6** for details.
- **Flags Table**—This table includes a record for each active connection that contains OAM flags used by all the connections during the normal cell flow. This table is dynamic and updated by the MC92501. The microprocessor is responsible for checking the flags on a regular basis. See **Section 7.3.7** for details.
- **VP Table(s)**—The VP Table Register(s) contain either an Ingress Connection Identifier (ICI) defined by the microprocessor as an active connection or a reference to the VC Table (see **Section 5.2.2**). The size and location of the VP Table are determined by the Link Register. If multiple links are supported, each Link Register defines a separate VP Table. Multiple VP Tables are not required to be contiguous. See **Section 7.3.8** for details.
- **VC Table**—This table contains a list of all the Ingress Connection Identifiers (ICIs) defined by the microprocessor as active Virtual Channel Connections. This table exists only if the Table Lookup method of Address Compression is used with VC Table Lookup enabled. See **Section 7.3.9** for details.

- **Multicast Translation Table**—This table contains the Egress Connection Identifiers (ECIs) associated with multicast identifiers. See **Section 7.3.10** for details.
- **Virtual Bucket Table**—Each record in this table contains the information for the UPC/NPC enforcement. This is not a physical table, but a virtual one. Since the Parameters Table contains a full address for the location of the Bucket record of each connection, there is no need to put all the Bucket records in consecutive physical locations. Although the user can distribute the records in any manner, this document uses the term "Bucket Table" in this document to refer to the set of all the Bucket records. See **Section 7.3.11** for details.
- **OAM Table**—This table contains the additional information required to run OAM Performance Monitoring See **Section 7.3.12** for details.
- **Dump Vector Table**—This table contains the dump vectors describing the recent history of the cell processing. This table is generally used for debugging purposes only. See **Section 7.3.13** for details.
- **Ingress Link Counters Table**—This table includes a record for each link that contains the cell counters used by the link during the normal Ingress cell flow. This table is dynamic and updated by the MC92501. The microprocessor is responsible for collecting the contents of the counters on a regular basis. See **Section 7.3.14** for details.
- **Egress Link Counters Table**—This table includes a record for each link that contains the cell counters used by the link during the normal Egress cell flow. This table is dynamic and updated by the MC92501. The microprocessor is responsible for collecting the contents of the counters on a regular basis. See **Section 7.3.15**.

Each of the per-connection tables is defined by a pointer only. The sizes of the tables are determined by the number of connections being used. The pointers are programmable with a granularity of 64 long words. These tables include the following:

- Context Parameters Table Pointer Register (CPTP)
- Ingress Billing Counters Table Pointer Register (IBCTP)
- Egress Billing Counters Table Pointer Register (EBCTP)
- Policing Counters Table Pointer Register (PCTP)
- Flags Table Pointer Register (FTP)

The other tables are also defined by pointers that are programmable with a granularity of 64 long words.These tables are:

- VC Table Pointer Register (VCTP)
- Multicast Translation Table Pointer Register (MTTP)
- OAM Table Pointer Register (OTP)
- Dump Vector Table Pointer Register (DVTP)
- Egress Link Counters Table Pointer Register (ELCTP)
- Ingress Link Counters Table Pointer Register (ILCTP)

NOTE: All fields marked "0" or "reserved" in the descriptions in this section must be written with zeros. The values read from

these fields should be considered undefined and should be ignored.

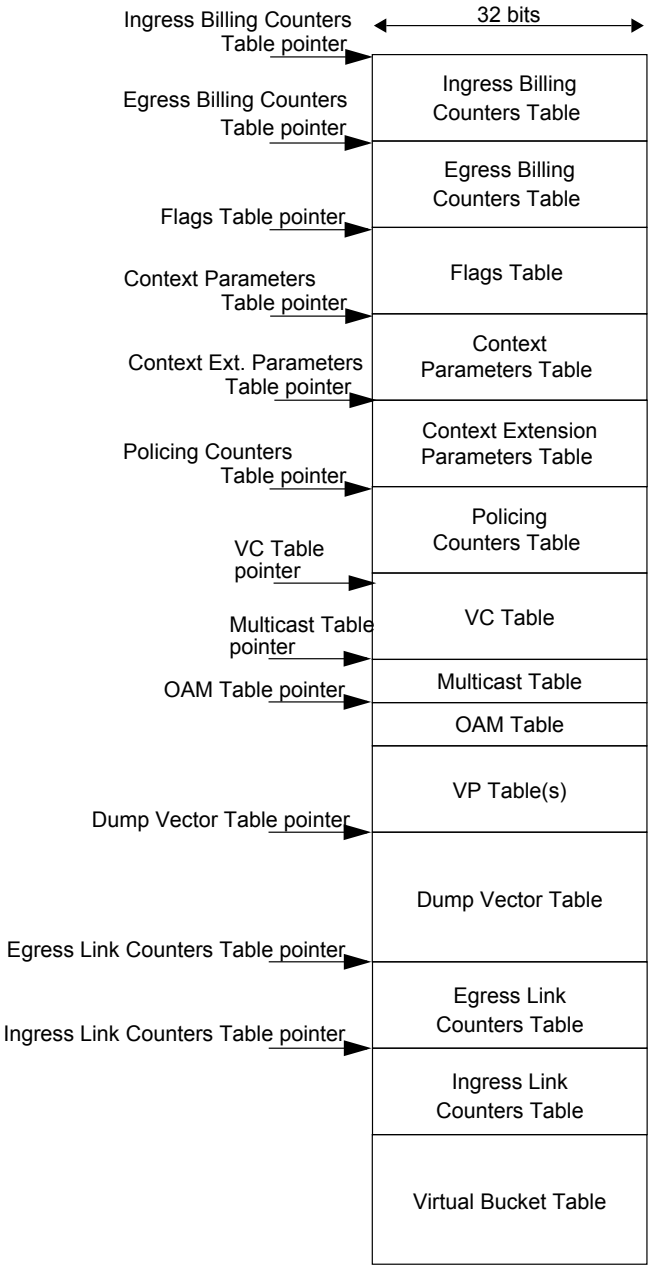


Figure 7-65. External Memory Partitioning

7.3.2 Memory Allocation

The amount of External Memory required and its allocation among the various tables can be derived from the following tables. **Table 7-11** deals with the tables whose size depends on the number of active connections, while **Table 7-12**, **Table 7-13**, and **Table 7-14** deal with the remaining tables. More details on these calculations, as well as example configurations, can be found in **Appendix C**.

Table 7-11. Number of Long Words per Connection in Each Table

Table name	Option	Long Words
Ingress Billing Counters		1–4
Policing Counters		1–4
Egress Billing Counters		1–4
Flags		1
Context Parameters		1–8
Context Extension Parameters		1
VP Table		see **Table 7-12**
VC Table		see **Table 7-13**
Multicast Translation Table		see **Table 7-14**
Buckets Table	One Bucket	3
	Two Buckets	5
	Three Buckets	7
	Four Buckets	9
OAM Table		8

Note: At most, sixty-four connections have associated records in the OAM Table.

Table 7-12. VP Table Size (per link)

Allocated VPI Bits	VP Table Records	VP Table Long Words (without VC Table Lookup)	VP Table Long Words (with VC Table Lookup)
1	2	1	2
2	4	2	4
3	8	4	8
4	16	8	16

Table 7-12. VP Table Size (per link) (Continued)

Allocated VPI Bits	VP Table Records	VP Table Long Words (without VC Table Lookup)	VP Table Long Words (with VC Table Lookup)
5	32	16	32
6	64	32	64
7	128	64	128
8	256	128	256
9	512	256	512
10	1024	512	1024
11	2048	1024	2048
12	4096	2048	4096

Note: See **Section 5.2.2.2** for more information.

Table 7-13. VC Sub-Table Size (per VPI)

Allocated VCI Bits	VC Table Records	VC Table Long Words
1 (see note)	2	1
2 (see note)	4	2
3 (see note)	8	4
4 (see note)	16	8
5 (see note)	32	16
6	64	32
7	128	64
8	256	128
9	512	256
10	1024	512
11	2048	1024
12	4096	2048
13	8192	4096
14	16384	8192
15	32768	16384
16	65536	32768

Note: ATM standards indirectly require at least 5–6 allocated VCI bits since the smallest VCI values are reserved.

Table 7-14. Multicast Translation Table Size (per link)

Allocated Multicast ID Bits	Multicast Table Records	Multicast Table Long Words
9	512	256
10	1K	512
11	2K	1K
12	4K	2K
13	8K	4K
14	16K	8K
15	32K	16K
16	64K	32K

Table 7-15. Addressing External Memory Records

Table	Width	Address Calculation	Parameters
Ingress Billing Counters	Long Word	{IBCTP[23:8], 0000_0000} + {ICI[15:0] * N, 00}	IBCTP: Ingress Billing Counters Table Pointer from Register ICI: Ingress Connection Identifier from Address Compression N: number of long words per record (1–4)
Policing Counters	Long Word	{PCTP[23:8], 0000_0000} + {ICI[15:0] * N, 00}	PCTP: Policing Counters Table Pointer from Register ICI: Ingress Connection Identifier from Address Compression N: number of long words per record (1–4)
Egress Billing Counters	Long Word	{EBCTP[23:8], 0000_0000} + {ECI[15:0] * N, 00}	ECTP: Egress Counters Table Pointer from Register ECI: Egress Connection Identifier from cell overhead bytes or Multicast Translation N: number of long words per record (1–4)
Flags	Word	{FTP[23:8], 0000_0000} + {ICI[15:0], 00}	FTP: Flags Table Pointer from Register ICI: Ingress Connection Identifier from Address Compression

Table 7-15. Addressing External Memory Records (Continued)

Table	Width	Address Calculation	Parameters
Context Parameters	Long Word	{CPTP[23:8], 0000_0000} + {CI[15:0] * N, 00}	CPTP: Context Parameters Table Pointer from Register CI: ECI or ICI as above N: number of long words per record (1–8)
Context Parameters Extension	Long Word	{CEPTP[23:8], 0000_0000} + {CI[15:0] * 1, 00}	CEPTP: Context Extension Parameters Table Pointer from Register CI: ECI or ICI as above
VP Table (Full Lookup)	Long Word	{VPP[23:8], 0000_0000} + {Index[11:0], 00}	VPP: VP Pointer from ILNKn register Index: VP Index = VPI from cell masked by VP_MASK from ILNKn
VP Table (VC Lookup Disable)	Word	{VPP[15:0], 0000_0000} + {Index[11:0], 0}	VPP: VP Pointer from ILNKn register Index: VP Index = VPI from cell masked by VP_MASK from ILNKn
VC Table	Word	{VCTP[23:8], 0000_0000} + {Offset[15:0], 00} + {Index[15:0], 0}	VCTP: VC Table Pointer from Register Offset: VC Offset from VP Table Index: VC Index = VCI from cell masked by VC_MASK from VP Table
Multicast Translation	Word	{MTTP[23:8], 0000_0000} + {MTTS[3:0], MI[(EMIC+8):0], 0}	MTTP: Multicast Translation Table Pointer from Register MTTS: Multicast Translation Table Section from cell overhead bytes MI: Multicast Identifier from cell overhead bytes EMIC: Egress Multicast Identifier Control from Egress Multicast Configuration Register (EMCR)
Buckets	Long Word	{BKT_Ptr[21:0], 00}	BKT_Ptr: Bucket Pointer from Context Parameters Table

Table 7-15. Addressing External Memory Records (Continued)

Table	Width	Address Calculation	Parameters
OAM Table	Long Word	{OTP[23:8], 0000_0000} + {OAM_Ptr, 0_0000}	OTP: OAM Table Pointer from Register OAM_Ptr: OAM Pointer from Context Parameters Table
Dump Vector	Long Word	{DVTP[23:8], 0000_0000} + {CT[(2*(DVTC+2)):0], 000}	DVTP: Dump Vector Table Pointer from Register CT: continuous cell time counter DVTC: Dump Vector Table Control from ATMC CFB Configuration Register (ACR)
Ingress Link Counters	Long Word	{ILCTP[23:8], 0000_0000} + {LINK * 5, 00}	ILCTP: Ingress Link Counters Table Pointer from Register LINK: the link from which the cell is received
Egress Link Counters	Long Word	{ELCTP[23:8], 0000_0000} + {LINK * 4, 00}	ELCTP: Egress Link Counters Table Pointer from Register LINK: the link to which the cell is transmitted

7.3.3 Context Parameters Table

The Context Parameters Table consists of specific records for each connection. There are various options regarding the format (size and contents) of each record. These options are programmable on a global basis using the ATMC CFB Configuration Register (ACR), such that all of the records in this table have the same format. The long words that may be included in a record of the Context Parameter Table are shown in **Figure 7-66** in the order in which they appear.

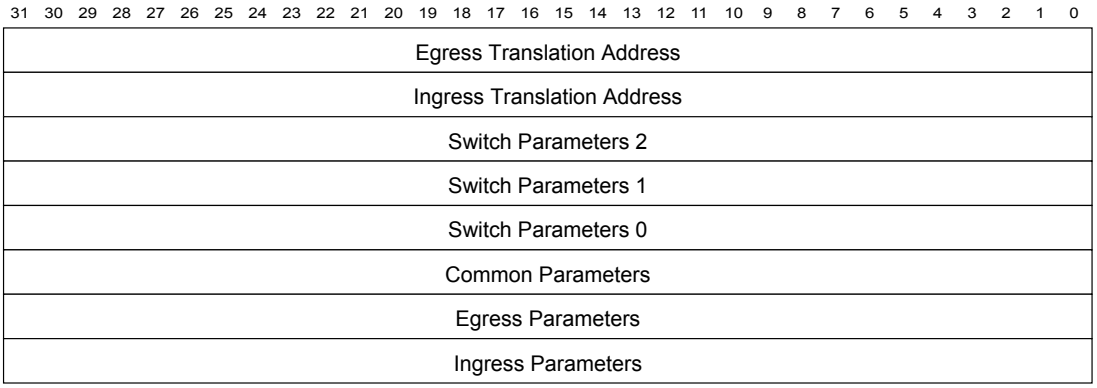


Figure 7-66. Context Parameters Table Record (Full Configuration)

If some of the long words do not exist, the record is shortened, but the order of the long words that remain does not change. An example is shown in **Figure 7-67**. This chapter describes in detail the long words that make up each record.

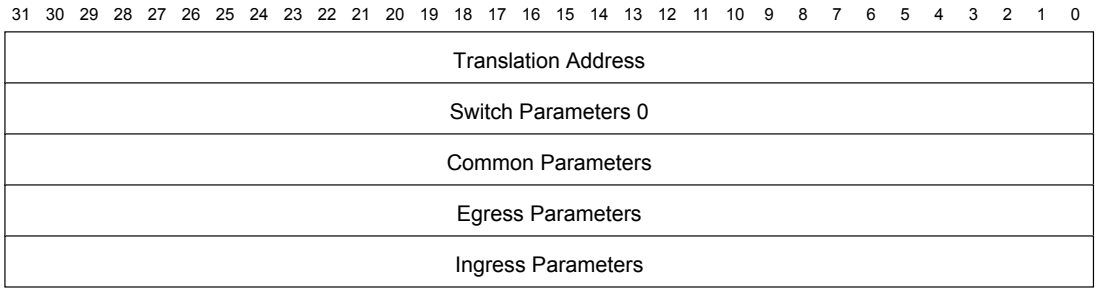


Figure 7-67. Context Parameters Table Record (ATC=01, SPC=01)

7.3.3.1 Egress Translation Address

This long word contains the connection address of the connection. This long word should be initialized during the connection setup process. It is used for address translation in the Egress cell flow. The long word structure is shown in **Figure 7-68**.

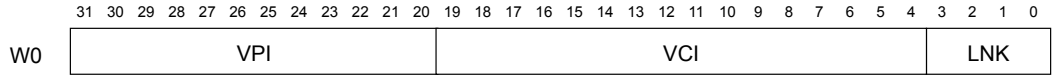


Figure 7-68. Egress Translation Address Long Word

- **Virtual Path Identifier (VPI)**—This field is part of the new address used by the Egress address translation mechanism. At a UNI the four most significant bits of this field are the GFC and they are user-defined.
- **Virtual Channel Identifier (VCI)**—This field is part of the new address used by the Egress address translation mechanism.
- **Physical Link Number (LNK)**—This field is valid only when multiple PHY devices are supported (see **Section 7.2.6.3** for details). It determines the physical link to which this connection belongs. If a single PHY device is used, the LNK field should be programmed as 0.

7.3.3.2 Ingress Translation Address

This long word is generally used if Ingress Address Translation is enabled. It contains the new address of the connection and is used for address translation in the Ingress cell flow. This long word should be initialized during the connection setup process. The long word structure is shown in **Figure 7-69**.

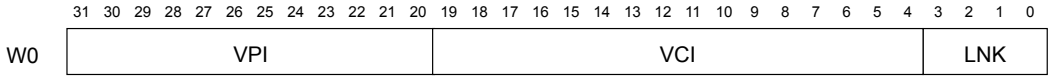


Figure 7-69. Ingress Address Translation Long Word

- **Virtual Path Identifier (VPI)**—This field is part of the new address used by the Ingress address translation mechanism.
- **Virtual Channel Identifier (VCI)**—This field is part of the new address used by the Ingress address translation mechanism.
- **Physical Link Number (LNK)**—This field is valid only when multiple PHY devices are supported (see **Section 7.2.6.3** for more information). It determines the physical link to which this connection belongs. If a single PHY device is used, the LNK field should be programmed as 0.

7.3.3.3 Switch Parameters

These optional long words contain the switch parameter records for the connection. Any of the three long words that exist are transferred to the Switch Interface along with each processed cell. For more details see **Section 5.2.8**. These long words should be initialized during the connection setup process.

7.3.3.4 Common Parameters

This long word is shared by the Ingress and Egress processing. It contains static parameters for the connection. This long word should be initialized during the connection setup process. The size and the location of some of the fields is changed according to the *PM on All Connections (PMAC)* bit in the ATMC CFB Configuration Register (ACR). See **Section 7.2.4.4** for details. **Figure 7-70** shows the long word fields when the PMAC bit is cleared.

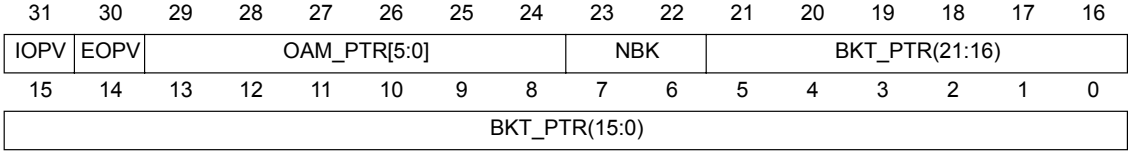


Figure 7-70. Common Parameters Long Word Fields (PMAC = 0)

Figure 7-71 shows the long word field definition when the *PMAC* bit is set.

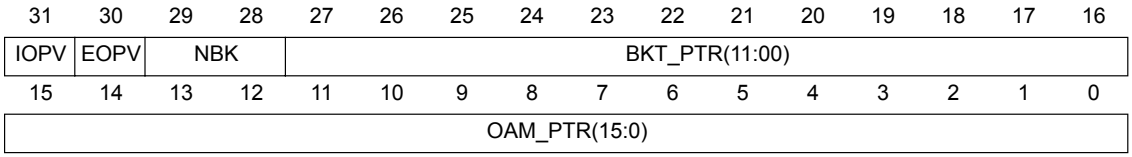


Figure 7-71. Common Parameters Long Word Fields (PMAC = 1)

- **Ingress OAM Pointer is Valid (IOPV)**—When IOPV is set, the OAM Pointer is valid. Note that even if IOPV is reset, the OAM Pointer is still valid if EOPV is set.
- **Egress OAM Pointer is Valid (EOPV)**—When EOPV is set, the OAM Pointer is valid. Note that even if EOPV is reset, the OAM Pointer is still valid if IOPV is set.
- **OAM Pointer (OAM_PTR)**—This field is used as a pointer to the OAM Table in case a Block Test is activated and performed on the connection. This pointer is valid only if *Ingress OAM Pointer is Valid (IOPV)* or *Egress OAM Pointer is Valid (EOPV)* is set (see **Section 6.3.7** for details).
- **Number of Buckets (NBK)**—This field defines the number of UPC/NPC Leaky Buckets that are active on this connection.
  - 00 = Four Leaky Buckets
  - 01 = One Leaky Bucket
  - 10 = Two Leaky Buckets
  - 11 = Three Leaky Buckets

NOTE: Zero Leaky Buckets (no UPC/NPC) is indicated by all 1s in the BKT_PTR field, in which case the value in the NBK field is not used.

- **Bucket Pointer (BKT_PTR)**—This pointer points to the Buckets record in the External Memory that contains the relevant UPC information used by Ingress for enforcement. The reserved value of all 1s is used to indicate that there is no Buckets record for this connection. (See **Appendix A** for more details). When the *PM on All Connections (PMAC)* bit is set, then the Bucket Pointer [21:12] is located in the Common Parameters Extension Word. See **Section 7.3.16** for details.

7.3.3.5 Egress Parameters

This long word is used by the Egress cell processing. It contains static parameters for the connection. This long word should be initialized during the connection setup process. The long word structure is shown in **Figure 7-72**.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECIV	EVPC	EEOT	ESOT	ESOO	Rsvd	ECAS	ECRD	ECOT	ECAO	ECSF	ECEF	ECSB	ECEB	Rsvd	Rsvd
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ESAI	ESRD	ESCS	ESCE	ECA	ERA	EP6C	EP7C	EVRE	EP6R	EP7R	Reserved				

Figure 7-72. Egress Parameters Long Word Fields

- **Egress Connection Identifier Valid (ECIV)**—This bit defines if the CI record is active or should be ignored.
  - 0 = Not valid
  - 1 = Valid
- **Egress Virtual Path Connection (EVPC)**—This bit defines whether the connection is a virtual channel or virtual path.
  - 0 = The connection is a Virtual Channel Connection.
  - 1 = The connection is a Virtual Path Connection.
- **Egress End-to-end OAM Termination (EEOT)**—When this bit is set, the Egress flow is treated as the terminating point of the OAM end-to-end cell flow for the connection. Additionally if the *VC RM cell Removal Point (CRRP)* bit is reset then cells with PTI = 6 or 7 are removed at this point.
- **Egress Segment OAM Termination (ESOT)**—When ESOT is set, the Egress flow is treated as the terminating point of the OAM segment cell flow for the connection.
- **Egress Segment OAM Origin (ESOO)**—When ESOO is set, the Egress flow is treated as the originating point of the OAM segment cell flow for the connection. Incoming segment OAM cells are removed at this point to prevent contamination of the OAM flows within the segment.
- **Egress Copy Received AIS Cells (ECAS)**—When ECAS is set, OAM end-to-end AIS cells belonging to this connection that are received in the Egress cell flow are copied to the Cell Extraction Queue. See **Section 6.3.5.1** for more details.
- **Egress Copy Received RDI Cells (ECRD)**—When ECRD is set, OAM end-to-end RDI cells belonging to this connection that are received in the Egress cell flow are copied to the Cell Extraction Queue. See **Section 6.3.5.1** for more details.
- **Egress Copy Received Other OAM Cells (ECOT)**—When ECOT is set, miscellaneous or unidentified OAM cells belonging to this connection that are received in the Egress cell flow are copied to the Cell Extraction Queue. See **Section 6.3.4.2** for more details.
- **Egress Copy All Received OAM Cells (ECAO)**—When ECAO is set, all OAM cells belonging to this connection that are received in the Egress cell flow are copied to the Cell Extraction Queue.

- **Egress Copy Segment FMCs (ECSF)**—When ECSF is set, OAM segment Forward Monitoring Cells belonging to this connection that are received in the Egress cell flow are copied to the Cell Extraction Queue. See **Section 6.3.7** for more details.
- **Egress Copy End-to-end FMCs (ECEF)**—When ECEF is set, OAM end-to-end Forward Monitoring Cells received in the Egress cell flow belonging to the connection are copied to the Cell Extraction Queue. See **Section 6.3.7** for details.
- **Egress Copy Segment BRCs (ECSB)**—When ECSB is set, OAM segment Backward Reporting Cells belonging to this connection that are received in the Egress cell flow are copied to the Cell Extraction Queue. See **Section 6.3.7** for details.
- **Egress Copy End-to-end BRCs (ECEB)**—When ECEB is set, OAM end-to-end Backward Reporting Cells belonging to this connection that are received in the Egress cell flow are copied to the Cell Extraction Queue. See **Section 6.3.7** for details.
- **Egress Send AIS cell (ESAI)**—When ESAI is set, an AIS cell is inserted in the Egress cell flow during the Internal Scan.
- **Egress Send RDI cell (ESRD)**—When ESRD is set, an RDI cell is inserted in the Egress cell flow during the Internal Scan.
- **Egress Send Continuity Check Segment OAM cell (ESCS)**—When ESCS is set, a segment Continuity Check cell is inserted in the Egress cell flow during the Internal Scan.
- **Egress Send Continuity Check End-to-end OAM cell (ESCE)**—When ESCS is set, an end-to-end Continuity Check cell is inserted in the Egress cell flow during the Internal Scan.
- **Egress Copy All cells (ECA)**—When ECA is set, ALL cells belonging to this connection that are received in the Egress cell flow are copied to the Cell Extraction Queue.
- **Egress Remove All cells (ERA)**—When ERA is set, ALL cells belonging to this connection that are received from the Switch Interface are removed from the Egress cell flow.
- **Egress PTI 6 Copy (EP6C)**—When EP6C is set, all cells belonging to this connection with PTI = 6 that are received in the Egress cell flow are copied to the Cell Extraction Queue.
- **Egress PTI 7 Copy (EP7C)**—When EP7C is set, all cells belonging to this connection with PTI = 7 that are received in the Egress cell flow are copied to the Cell Extraction Queue.
- **Egress VCR/VRR Registers Enable (EVRE)**—The EVRE bit, when set, enables the copying and/or removing of cells with reserved VCI values according to the programming of the Egress VCI Copy Register (EVCR) and the Egress VCI Remove Register (EVRR).
- **Egress PTI 6 Remove (EP6R)**—When this bit is set and the *VC RM cell Removal Point (CRRP)* bit is set , and then an Egress cell whose PTI = 6 is removed provided that the connection is a VC connection.
- **Egress PTI 7 Remove (EP7R)**—When this bit is set and the *VC RM cell Removal Point (CRRP)* bit is set, and then an Egress cell whose PTI = 7 is removed provided that the connection is a VC connection.

7.3.3.6 Ingress Parameters

This long word is used by the Ingress cell processing. It contains static parameters for the connection. This long word should be initialized during the connection setup process. The long word structure is shown in **Figure 7-73**.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ICIV	IVPC	IEOT	ISOT	ISOO	Rsvd	ICAS	ICRD	ICOT	ICAO	ICSF	ICEF	ICSB	ICEB	Rsvd	Rsvd
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ISAI	ISRD	ISCS	ISCE	ICA	IRA	IP6C	IP7C	IVRE	IP6R	IP7R	Reserved				UDT

Figure 7-73. Ingress Parameters Long Word Fields

- **Ingress Connection Identifier Valid (ICIV)**—This bit defines if the CI record is active or should be ignored.
  - 0 = Not valid
  - 1 = Valid
- **Ingress Virtual Path Connection (IVPC)**—This bit defines whether the connection is a virtual channel or virtual path.
  - 0 = The connection is a Virtual Channel Connection.
  - 1 = The connection is a Virtual Path Connection.
- **Ingress End-to-end OAM Termination (IEOT)**—When IEOT is set, the Ingress flow is treated as the terminating point of the OAM end-to-end cell flow for the connection. Also, if the *VC RM cell Removal Point (CRRP)* bit is cleared, then cells that belong to a VC connection and whose PTI = 6 or 7 are removed at this point.
- **Ingress Segment OAM Termination (ISOT)**—When ISOT is set, the Ingress flow is treated as the terminating point of the connection OAM segment cell flow.
- **Ingress Segment OAM Origin (ISOO)**—When ISOO is set, the Ingress flow is treated as the originating point of the OAM segment cell flow for the connection. Incoming segment OAM cells are removed at this point to prevent contamination of the OAM flows within the segment.
- **Ingress Copy Received AIS Cells (ICAS)**—When ICAS is set, OAM end-to-end AIS cells belonging to this connection that are received in the Ingress cell flow are copied to the Cell Extraction Queue. See **Section 6.3.5.1** for more details.
- **Ingress Copy Received RDI Cells (ICRD)**—When ICRD is set, OAM end-to-end RDI cells belonging to this connection that are received in the Ingress cell flow are copied to the Cell Extraction Queue. See **Section 6.3.5.1** for more details.
- **Ingress Copy Received Other OAM Cells (ICOT)**—When ICOT is set, miscellaneous or unidentified OAM cells belonging to this connection that are received in the Ingress cell flow are copied to the Cell Extraction Queue. See **Section 6.3.4.2** for more details.
- **Ingress Copy All Received OAM Cells (ICAO)**—When ICAO is set, all OAM cells belonging to this connection that are received in the Ingress cell flow are copied to the Cell Extraction Queue.
- **Ingress Copy Segment FMCs (ICSF)**—When ICSF is set, OAM segment Forward Monitoring Cells belonging to this connection that are received in the Ingress cell

flow are copied to the Cell Extraction Queue. See **Section 6.3.7** for more details.

- **Ingress Copy End-to-end FMCs (ICEF)**—When ICEF is set, OAM end-to-end Forward Monitoring Cells received in the Ingress cell flow that belong to this connection are copied to the Cell Extraction Queue. See **Section 6.3.7** for details.
- **Ingress Copy Segment BRCs (ICSB)**—When ICSB is set, OAM segment Backward Reporting Cells belonging to this connection that are received in the Ingress cell flow are copied to the Cell Extraction Queue. See **Section 6.3.7** for details.
- **Ingress Copy End-to-end BRCs (ICEB)**—When ICEB is set, OAM end-to-end Backward Reporting Cells received in the Ingress cell flow that belong to this connection are copied to the Cell Extraction Queue. See **Section 6.3.7** for details.
- **Ingress Send AIS cell (ISAI)**—When ISAI is set, an AIS cell is inserted in the Ingress cell flow during the Internal Scan.
- **Ingress Send RDI cell (ISRD)**—When ISRD is set, an RDI cell is inserted in the Ingress cell flow during the Internal Scan.
- **Ingress Send Continuity Check Segment OAM cell (ISCS)**—When ISCS is set, a segment Continuity Check cell is inserted in the Ingress cell flow during the Internal Scan.
- **Ingress Send Continuity Check End-to-end OAM cell (ISCE)**—When ISCS is set, an end-to-end Continuity Check cell is inserted in the Ingress cell flow during the Internal Scan.
- **Ingress Copy All cells (ICA)**—When ICA is set, ALL cells received in the Ingress cell flow that belong to this connection are copied to the Cell Extraction Queue.
- **Ingress Remove All cells (IRA)**—When IRA is set, ALL cells received from the PHY that belong to this connection are removed from the Ingress cell flow.
- **Ingress PTI 6 Copy (IP6C)**—When IP6C is set, all cells received in the Ingress cell flow with PTI = 6 that belong to the connection are copied to the Cell Extraction Queue.
- **Ingress PTI 7 Copy (IP7C)**—When IP7C is set, all cells received in the Ingress cell flow with PTI = 7 belonging to this connection are copied to the Cell Extraction Queue.
- **Ingress VCR/VRR Registers Enable (IVRE)**—The IVRE bit, when set, enables the copying and/or removing of cells with reserved VCI values according to the programming of the Ingress VCI Copy Register (IVCR) and the Ingress VCI Remove Register (IVRR).
- **Ingress PTI 6 Remove (IP6R)**—When this bit is set and the RRP bit is set, then an Ingress cell with PTI = 6 is removed if the connection is a VC connection.
- **Ingress PTI 7 Remove (IP7R)**—When this bit is set and the RRP bit is set, then an Ingress cell with PTI = 7 is removed if the connection is a VC connection.
- **UPC/NPC Don't Touch (UDT)**—The UDT bit indicates that the UPC/NPC block should perform the normal operation, but the cell shouldn't be touched (i.e., tagged or discarded) regardless of the UPC results. When UDT is set, the UPC/NPC counters are incremented and they indicate the number of cells that would have been tagged/discarded if the cells were processed normally.
  - 0 = Process the incoming cell normally.
  - 1 = Don't touch the incoming cell.

7.3.4 Ingress Billing Counters Table

This table contains the Ingress Billing Counter records for all the active connections. Details such as when the counters are updated and how and when the microprocessor aggregates them can be found in Section 9.2.3 Ingress Switch Interface Signals and **Section 3.3** External Memory Maintenance. Each counter wraps to zero after reaching its maximum value. A number of options exist regarding the number of counters and their definition. Note that in each case one and only one counter is incremented for each received cell belonging to this connection. There is also an option to eliminate the Ingress Billing Counters Table entirely. The *Ingress Billing Counters Control (IBCC)* field of the Ingress Processing Configuration Register (IPCR) is used for programming the various options. If all four counters exist, each record has structure shown in **Figure 7-74**.

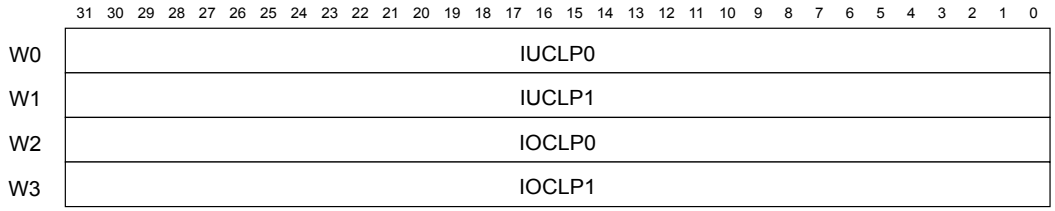


Figure 7-74. Ingress Counters (Full Configuration—IBCC = 001)

- **Ingress User CLP0 counter (IUCLP0)**—Ingress uses the 32-bit counter to count the incoming user (non-OAM) cells from the PHY with Cell Loss Priority (CLP) = 0.
- **Ingress User CLP1 counter (IUCLP1)**—Ingress uses this 32-bit counter to count the incoming user (non-OAM) cells from the PHY with CLP = 1.
- **Ingress OAM CLP0 counter (IOCLP0)**—Ingress uses this 32-bit counter to count the incoming OAM cells from the PHY with CLP = 0.
- **Ingress OAM CLP1 counter (IOCLP1)**—Ingress uses this 32-bit counter to count the incoming OAM cells from the PHY whose CLP = 1.

If the two OAM counters are combined, each record has one of the following structures, shown in **Figure 7-75** or **Figure 7-76** to keep a round number of long words per record.

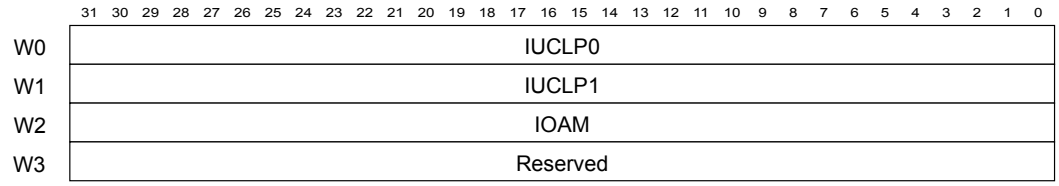


Figure 7-75. Ingress Counters (Single OAM Configuration—IBCC = 011)

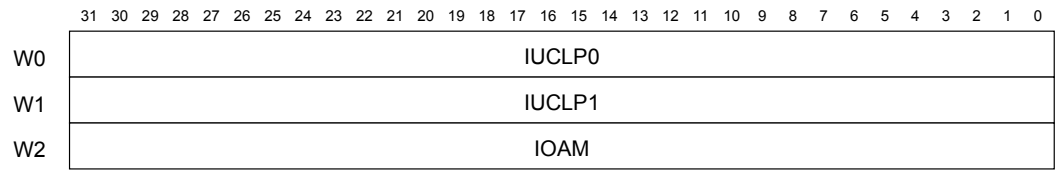


Figure 7-76. Ingress Counters (Single OAM Configuration—IBCC = 010)

- **Ingress User CLP0 counter (IUCLP0)**—This 32-bit counter is used by the Ingress to count the incoming user (non-OAM) cells from the PHY with CLP = 0.
- **Ingress User CLP1 counter (IUCLP1)**—This 32-bit counter is used by the Ingress to count the incoming user (non-OAM) cells from the PHY with CLP = 1.
- **Ingress OAM counter (IOAM)**—This 32-bit counter is used by the Ingress to count the incoming OAM cells from the PHY.

If the two user counters are also combined, each record has the structure shown in **Figure 7-77**.

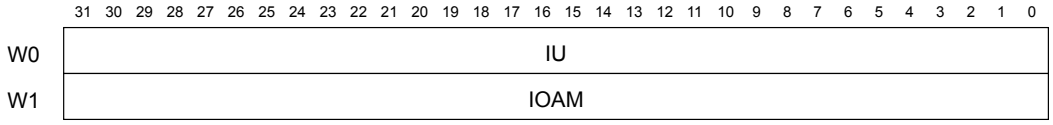


Figure 7-77. Ingress Counters (No CLP Distinction Configuration—IBCC = 100)

- **Ingress User counter (IU)**—This 32-bit counter is used by the Ingress to count the incoming user (non-OAM) cells from the PHY.
- **Ingress OAM counter (IOAM)**—This 32-bit counter is used by the Ingress to count the incoming OAM cells from the PHY.

If the OAM counter is eliminated, each record has the structure shown in **Figure 7-78**.

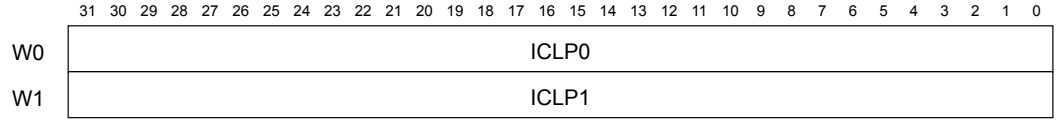


Figure 7-78. Ingress Counters (No OAM Distinction Configuration—IBCC = 101)

- **Ingress CLP0 counter (ICLP0)**—This 32-bit counter is used by the Ingress to count the incoming cells from the PHY whose Cell Loss Priority (CLP) is zero.
- **Ingress CLP1 counter (ICLP1)**—This 32-bit counter is used by the Ingress to count the incoming cells from the PHY whose Cell Loss Priority (CLP) is one.

Finally, all of the counters can be combined into one, in which case each record has the following structure:

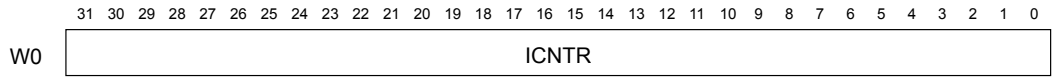


Figure 7-79. Ingress Counters (Single Counter Configuration—IBCC = 110)

- **Ingress Counter (ICNTR)**—This 32-bit counter is used by the Ingress to count the incoming cells from the PHY.

7.3.5 Egress Billing Counters Table

This table contains the Egress billing counter records for all the active connections. Details such as when the counters are updated and how and when the microprocessor aggregates them can be found in **Section 5.3.4** Egress Context Table Lookup and **Section 3.3** External Memory Maintenance. Each counter wraps to zero after reaching its maximum value. A number of options exist regarding the number of counters and their definition. In each case one and only one counter is incremented for each received cell belonging to this connection. There is also an option to eliminate the Egress Billing Counters Table entirely. The *Egress Billing Counters Control (EBCC)* field of the Egress Processing Configuration Register (EPCR) is used for programming the various options. If all four counters exist, each record has the structure shown in **Figure 7-80**.

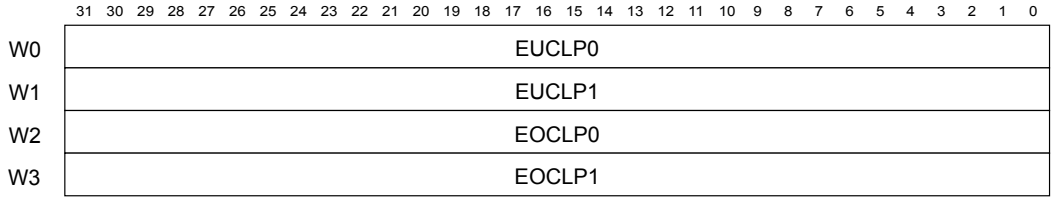


Figure 7-80. Egress Counters (Full Configuration—EBCC = 001)

- **Egress User CLP0 counter (EUCLP0)**—This 32-bit counter is used by the Egress to count the outgoing user (non-OAM) cells whose Cell Loss Priority (CLP) is zero.
- **Egress User CLP1 counter (EUCLP1)**—This 32-bit counter is used by the Egress to count the outgoing user (non-OAM) cells whose Cell Loss Priority (CLP) is one.
- **Egress OAM CLP0 counter (EOCLP0)**—This 32-bit counter is used by the Egress to count the outgoing OAM cells whose Cell Loss Priority (CLP) is zero.
- **Egress OAM CLP1 counter (EOCLP1)**—This 32-bit counter is used by the Egress to count the outgoing OAM cells whose Cell Loss Priority (CLP) is one.

If the two OAM counters are combined, each record has one of the following structures, shown in **Figure 7-81** or **Figure 7-82** to keep a round number of long words per record:

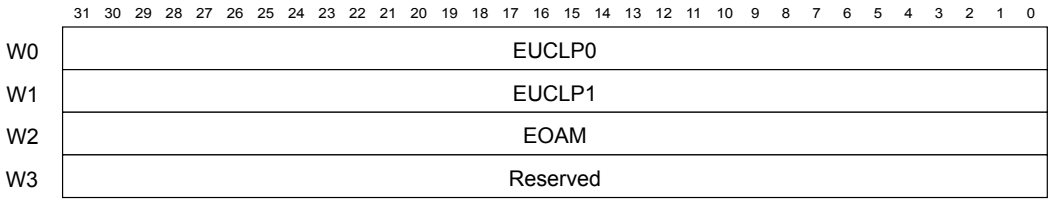


Figure 7-81. Egress Counters (Single OAM Configuration—EBCC = 011)

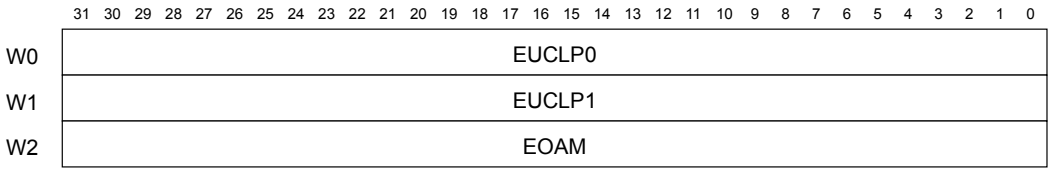


Figure 7-82. Egress Counters (Single OAM Configuration—EBCC = 010)

- **Egress User CLP0 counter (EUCLP0)**—This 32-bit counter is used by the Egress to count the outgoing user (non-OAM) cells whose Cell Loss Priority (CLP) is zero.
- **Egress User CLP1 counter (EUCLP1)**—This 32-bit counter is used by the Egress to count the outgoing user (non-OAM) cells whose Cell Loss Priority (CLP) is one.
- **Egress OAM counter (EOAM)**—This 32-bit counter is used by the Egress to count the outgoing OAM cells.

For two combined user counters , **Figure 7-83** shows the structure for each record.

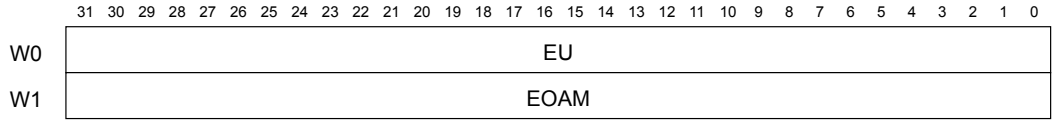


Figure 7-83. Egress Counters (No CLP Distinction Configuration—EBCC = 100)

- **Egress User counter (EU)**—This 32-bit counter is used by the Egress to count the outgoing user (non-OAM) cells.
- **Egress OAM counter (EOAM)**—This 32-bit counter is used by the Egress to count the outgoing OAM cells.

If the OAM counter is eliminated, each record has the structure shown in **Figure 7-84**.

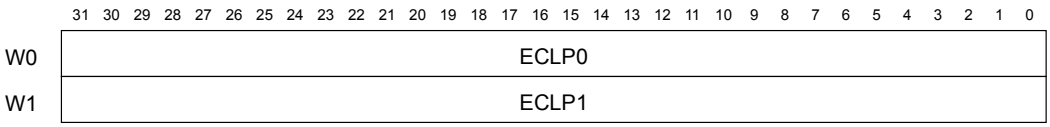


Figure 7-84. Egress Counters (No OAM Distinction Configuration—EBCC = 101)

- **Egress CLP0 counter (ECLP0)**—This 32-bit counter is used by the Egress to count the outgoing cells whose Cell Loss Priority (CLP) is zero.
- **Egress CLP1 counter (ECLP1)**—This 32-bit counter is used by the Egress to count the outgoing cells whose Cell Loss Priority (CLP) is one.

Finally, all of the counters can be combined into one, in which case each record has the structure shown in **Figure 7-85**.

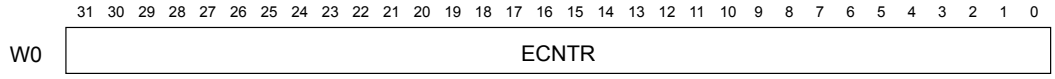


Figure 7-85. Egress Counters (Single Counter Configuration—EBCC = 110)

- **Egress Counter (ECNTR)**—This 32-bit counter is used by the Egress to count the outgoing cells.

7.3.6 Policing Counters Table

This table contains the policing counter records for all the active connections. Details such as when the counters are updated and how and when the microprocessor aggregates them can be found in **Section 5.2.5** UPC/NPC and **Section 3.3** External Memory Maintenance. Each counter wraps to zero after reaching its maximum value. A number of options exist regarding the number of counters and their definition. Note that in each case at most one policing counter is incremented for each received cell belonging to this connection. There is also an option to eliminate the policing Counters Table entirely. The *Ingress Policing Counters Control (IPCC)* field of the Ingress Processing Configuration Register (IPCR) is used in the case of Ingress UPC for programming the various options. The *Egress Policing Counters Control (EPCC)* field of the Egress Processing Configuration Register (EPCR) is used in the case of Egress UPC for programming the various options.

If all three counters exist, each record has one of the following structures, shown in **Figure 7-86** and **Figure 7-87**, to keep a round number of long words in each record.

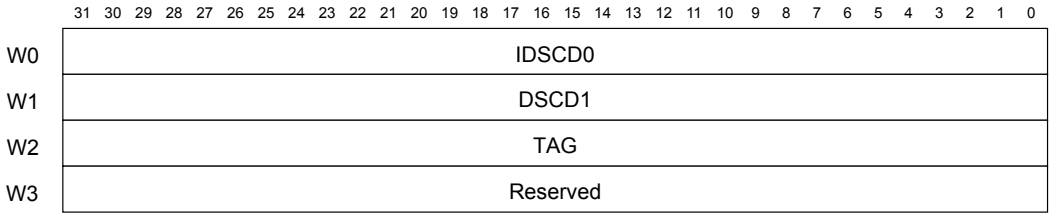


Figure 7-86. Policing Counters (Full Configuration—PCC = 001)

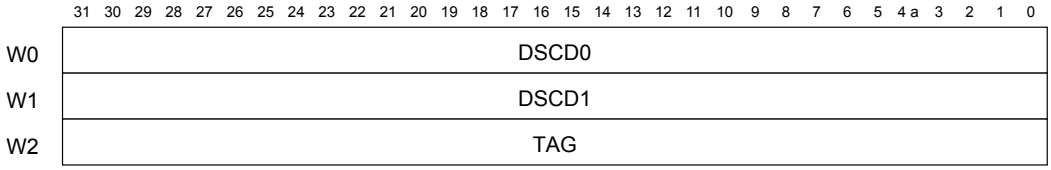


Figure 7-87. Policing Counters (Full Configuration—PCC = 010)

- **Ingress Discard CLP0 counter (DSCD0)**—This 32-bit counter is used by the UPC/NPC to count cells that were discarded (removed from the cell flow) by the enforcer whose Cell Loss Priority (CLP) was zero.
- **Ingress Discard CLP1 counter (DSCD1)**—This 32-bit counter is used by the UPC/NPC to count cells that were discarded (removed from the cell flow) by the enforcer whose Cell Loss Priority (CLP) was one.
- **Ingress TAG counter (TAG)**—This 32-bit counter is used by the UPC/NPC to count cells that were tagged (CLP was changed from zero to one) by the enforcer.

Figure 7-88 shows the individual record structure if two discard counters are combined.

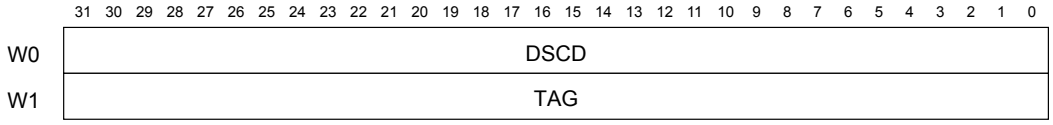


Figure 7-88. Policing Counters (No CLP Distinction Configuration—PCC = 011)

- **Ingress Discard counter (DSCD)**—This 32-bit counter is used by the UPC/NPC to count cells that were discarded (removed from the cell flow) by the enforcer.
- **Ingress TAG counter (TAG)**—This 32-bit counter is used by the UPC/NPC to count cells that were tagged (CLP was changed from zero to one) by the enforcer.

Options using only the TAG or DSCD counters (as defined above) use individual records with one long word as shown in **Figure 7-89** and **Figure 7-90**.

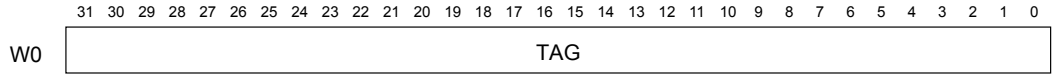


Figure 7-89. Policing Counters (Only Tag Counter Configuration—PCC = 100)

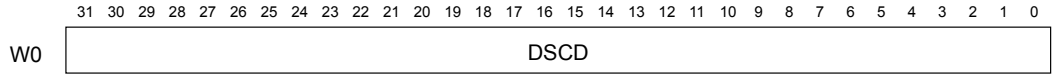


Figure 7-90. Policing Counters (Only Discard Counter Configuration—PCC=101)

7.3.7 Flags Table

This table contains the dynamic flags records for all the active connections. The ATMC is responsible for updating them during the cell flow, and the microprocessor is responsible for reading them on a regular basis. These flags are used mostly for Operation And Maintenance tasks (Fault Management tests such as Alarm Surveillance and Continuity Check). (See **Section 6.3.5.1** for more details.)

The records of the Flags Table consist of one long word as follows:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERAS	ERRD	ERST	ERET	Reserved											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRAS	IRRD	IRST	IRET	Reserved											

Figure 7-91. Flags Table Fields

- **Egress Received an AIS cell (ERAS)—**This bit is set when an OAM end-to-end AIS cell is received in the Egress. This bit remains set until the microprocessor reads the entry and writes back zeros.
  - 0 = No AIS cell has been received in the Egress cell flow since this bit was cleared by the processor.
  - 1 = One or more AIS cells have been received in the Egress cell flow since this bit was cleared by the processor.
- **Egress Received an RDI cell (ERRD)—**This bit is set when an OAM end-to-end RDI cell is received in the Egress. This bit remains set until the microprocessor reads the entry and writes back zeros.
  - 0 = No RDI cell has been received in the Egress cell flow since this bit was cleared by the processor.
  - 1 = One or more RDI cells have been received in the Egress cell flow since this bit was cleared by the processor.
- **Egress Received a Segment Traffic cell (ERST)—**This bit is set when a user cell or any type of Continuity Check cell is received in the Egress. This bit remains set until the microprocessor reads the entry and writes back zeros.
  - 0 = No segment traffic cell has been received in the Egress cell flow since this bit was cleared by the processor.
  - 1 = One or more segment traffic cells have been received in the Egress cell flow since this bit was cleared by the processor.
- **Egress Received an End to end Traffic cell (ERET)—**This bit is set when a user cell or End to End Continuity Check cell is received in the Egress. This bit remains set until the microprocessor reads the entry and writes back zeros.
  - 0 = No end-to-end traffic cell has been received in the Egress cell flow since this bit was cleared by the processor.
  - 1 = One or more end-to-end traffic cells have been received in the Egress cell flow since this bit was cleared by the processor.

- **Ingress Received an AIS cell (IRAS)**—This bit is set when an OAM end-to-end AIS cell is received in the Ingress. This bit remains set until the microprocessor reads the entry and writes back zeros.
  - 0 = No AIS cell has been received in the Ingress cell flow since this bit was cleared by the processor.
  - 1 = One or more AIS cells have been received in the Ingress cell flow since this bit was cleared by the processor.
- **Ingress Received an RDI cell (IRRD)**—This bit is set when an OAM end-to-end RDI cell is received in the Ingress. This bit remains set until the microprocessor reads the entry and writes back zeros.
  - 0 = No RDI cell has been received in the Ingress cell flow since this bit was cleared by the processor.
  - 1 = One or more RDI cells have been received in the Ingress cell flow since this bit was cleared by the processor.
- **Ingress Received a Segment Traffic cell (IRST)**—This bit is set when a user cell or any type of Continuity Check cell is received in the Ingress. This bit remains set until the microprocessor reads the entry and writes back zeros.
  - 0 = No segment traffic cell has been received in the Ingress cell flow since this bit was cleared by the processor.
  - 1 = One or more segment traffic cells have been received in the Ingress cell flow since this bit was cleared by the processor.
- **Ingress Received an End to end Traffic cell (IRET)**—This bit is set when a user cell or End to End Continuity Check cell is received in the Ingress. This bit remains set until the microprocessor reads the entry and writes back zeros.
  - 0 = No end-to-end traffic cell has been received in the Ingress cell flow since this bit was cleared by the processor.
  - 1 = One or more end-to-end traffic cells have been received in the Ingress cell flow since this bit was cleared by the processor.

7.3.8 VP Table

As described earlier, the VP Table is used to find the Ingress Connection Identifier (ICI) for VPCs or a pointer to the VC Table for VCCs during the address compression process. The table contains up to 4K records for the maximum of 4K VPCs that can be defined on a single link. There are several formats for the VP Table record. If VC Table lookup is disabled, each record is 16 bits wide and two records are stored in each long word. If VC Table lookup is enabled, each record occupies one long word. See **Section 5.2.2.2** for more details.

7.3.8.1 VP Table Record without VC Table Lookup

When VC Table lookup is not performed (ACM = 01 or 11), each long word entry of the VP Table contains the records of two connections in order to save space. Since each record is 16 bits wide, the order of the records within a long word of External Memory is reversed if low-endian ordering is being used on the system bus (see **Figure 7-92**). The MC92501 interprets the long words according to the *Data Order (DO)* bit in the Microprocessor Configuration Register (MPCONR).

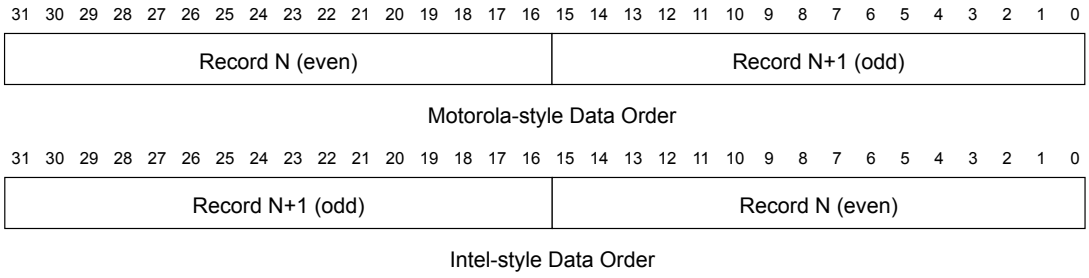


Figure 7-92. Arrangement of 16-bit Records in External Memory

Each 16-bit record has the structure shown in **Figure 7-93**.

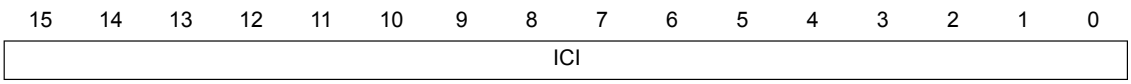


Figure 7-93. VP Table Structure

- **Ingress Connection Identifier (ICI)**—This pointer is the index to the connection records in the Context Parameter Table and the Counter Tables. A value of all 1's indicates that the ICI is not valid.

7.3.8.2 VP Table Record with VC Table Lookup

When VC Table lookup is enabled, each long word entry contains the record of one connection. Each record has one of the formats shown in the following figures.

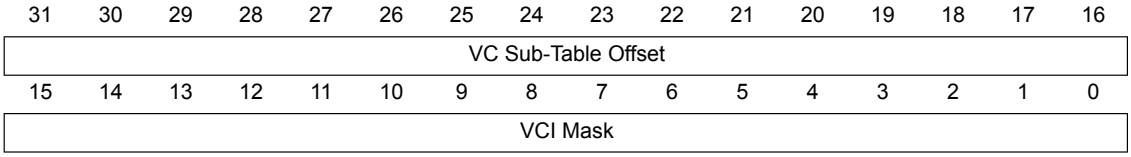


Figure 7-94. VP Table Record Fields with VC Switching

This format shown in **Figure 7-94** is used when VC switching is being performed on the connection. It is identified by the VC Sub-Table Offset field containing any value other than all 1s.

- **VC Sub-Table Offset**—This field points to the beginning of this VP's VC Sub-Table within the VC Table. It is in units of long words and is added to the VC Table Pointer (see **Section 7.2.6.18**) to obtain the address of the VC Sub-Table.

- **VCI Mask**—This field indicates which bits of the cell's VCI are used to index the VC connection in the VC Sub-Table. Each bit of the VCI Mask that is set indicates that the corresponding bit in the VCI should be included in the index. The bits that are included are shifted to the right, such that the number of bits in the index is equal to the number of 1's in the VCI Mask.

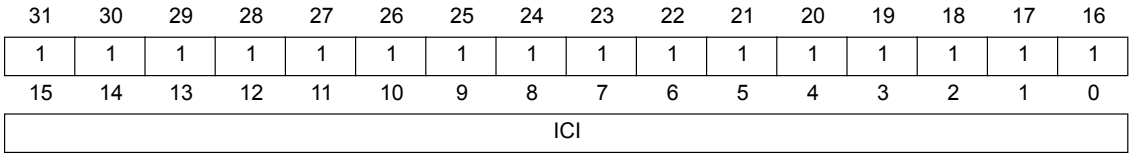


Figure 7-95. VP Table Record Fields with VP Switching

The format shown in **Figure 7-95** is used when VP switching is being performed on the connection. It is identified by the 16 most significant bits being set.

- **Ingress Connection Identifier (ICI)**—This pointer is the index to the connection records in the Context Parameter Table and the Counter Tables. A value of all 1s indicates that the ICI is not valid, and no entry exists in the Context Parameters Table for this VPI value.

7.3.9 VC Table

As described earlier, the VC Table is used to find the Ingress Connection Identifier (ICI) of VCCs during the address compression process (see **Section 5.2.2**). Each long word entry includes the records of two connections in order to save space. Since each record is 16 bits wide, the order of the records within a long word of External Memory is reversed if low-endian ordering is being used on the system bus (see **Figure 7-92**). The ATMC interprets the long words according to the *Data Order (DO)* bit in the Microprocessor Configuration Register (MPCONR). The microprocessor is responsible for initializing the records during the connection set-up process. Each record uses the structure shown in **Figure 7-96**.

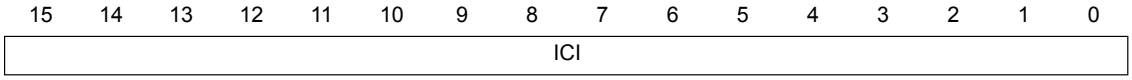


Figure 7-96. VC Record Structure

- **Ingress Connection Identifier (ICI)**—This pointer is the index to the connection records in the Context Parameter Table and the Counter Tables. A value of all 1's indicates that the ICI is not valid, and no entry exists in the Context Parameters Table for this VPI/VCI value.

7.3.10 Multicast Translation Table

The Multicast Translation Table is used to look up the Egress Connection Identifier (ECI) of cells that have been multicast in the switch. The Multicast Identifier (MI), described in **Section 5.3.2,** is used as the index to this table.Each long word contains two connection records to save space. Since each record is 16 bits wide, the order of the records within a long word of External Memory is reversed if low-endian ordering is being used on the system bus (see **Figure 7-92**). The ATMC interprets the long words according to the *Data Order (DO)* bit in the Microprocessor Configuration Register (MPCONR). Each record has the following structure:

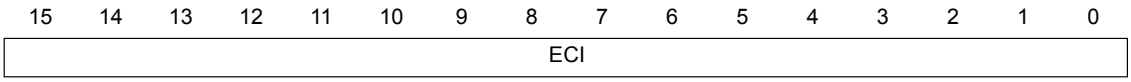


Figure 7-97. Multicast Translation Table Record Structure

- **Egress Connection Identifier (ECI)**—This pointer is the index to the connection records in the Context Parameter Table and the Counter Tables. A value of all 1s indicates that the ECI is not valid, and no entry exists in the Context Parameters Table for this MI value.

7.3.11 Buckets Record

As described earlier, the Buckets information is organized in records. Each record contains the buckets information of one connection. Since the Bucket Pointer (BKT_PTR) is a full address, the Buckets records may be distributed anywhere in External Memory.

7.3.11.1 Bucket Entries for One Connection

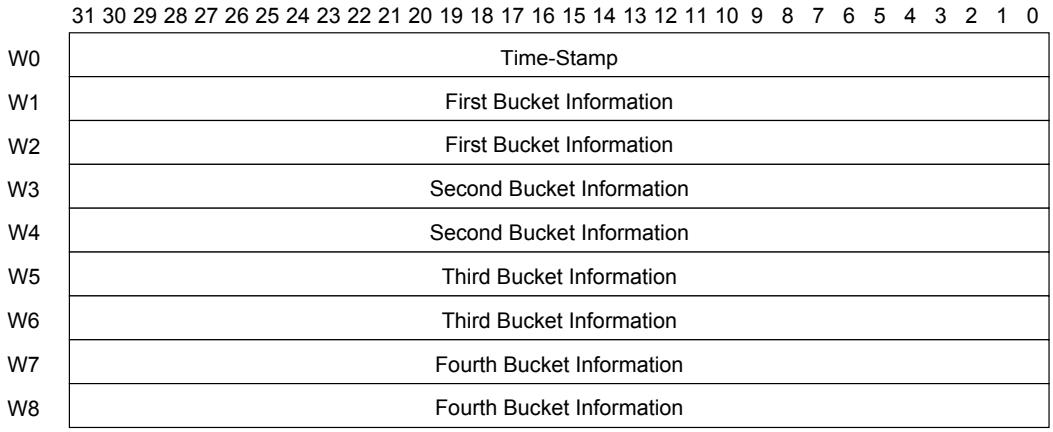


Figure 7-98. Bucket Entries

- **Time-Stamp**—This field is used to store the last time at which a cell was admitted in order to be able to calculate the leak from the bucket. This field is mandatory.
- **First Bucket Information**—These fields contain the dynamic and static information of a bucket. These fields are mandatory.
- **Second, Third and Fourth Bucket Information**— The NBK field in the Context Parameters Table determines the number of buckets (and fields) being used.

7.3.11.2 Bucket Information

The structure of each of the four Bucket information entries is identical (W1=W3=W5=W7 contain TSC and BKC as shown below, W2=W4=W6=W8 contain LMS, SCP, BKL, TAG, and CAP as shown below). See **Section A.3** for more details.

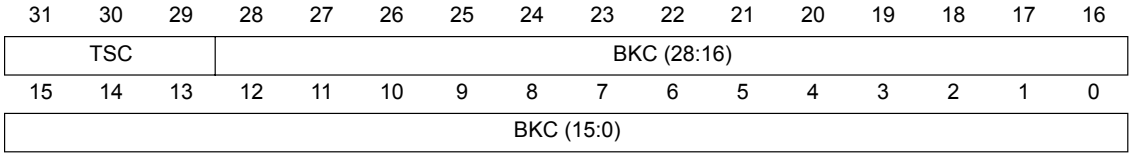


Figure 7-99. Bucket Information Word 1, 3, 5, and 7 Fields

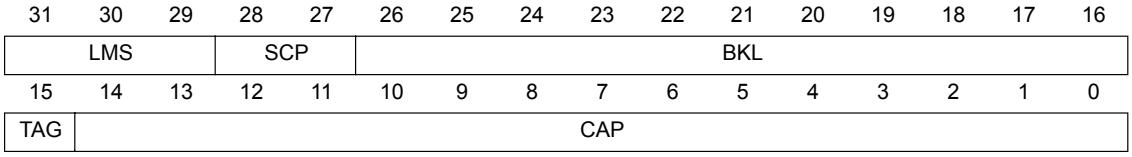


Figure 7-100. Bucket Information Word 2, 4, 6, and 8 Fields

- **Timescale (TSC)**—The eight timescale values are used for encoding the BKC and CAP fields by moving the binary point in a way that keeps the UPC error to a minimum. See **Section A.6** for details.
- **Bucket Contents (BKC)**—This field stores the contents of a bucket in units of cell slots and is updated dynamically whenever a cell is admitted.
- **Limit Shift (LMS)**—The limit shift is used for encoding the BKL by moving the binary point. See **Section A.6** for details.
- **Scope (SCP)**—This field defines the scope on which the enforcer should work (on CLP0, CLP1, or both).
  - 00 = No enforcement by this bucket.
  - 01 = Enforcement only on cells with CLP = 0.
  - 10 = Enforcement only on cells with CLP = 1.
  - 11 = Enforcement on both CLPs.
- **Bucket Limit (BKL)**—The bucket limit is used by the enforcer to limit the burstiness of the channel. It is defined as one Cell Arrival Period less than the bucket size. See **Section A.6** for details.
- **TAG**—This bit defines if a violating cell is tagged (set CLP to one) or discarded (removed from the cell flow).
  - 0 = Violating cell is discarded.
  - 1 = Violating cell is tagged.
- **Cell Arrival Period (CAP)**—This field represents the bandwidth of the connection by defining the average time (in cell slots) between cells.

7.3.12 OAM Table

Each OAM Table record consists of eight long words and contains the fields necessary for running a bidirectional Performance Monitoring test on a connection. The OAM Pointer in a connection Context Parameters Table record is used as an index to the OAM Table record relating to the connection. Each record is logically divided into two parts: an Egress area (consisting of the first four long words) and an Ingress area (consisting of the last four long words), as illustrated in **Figure 7-101**. At most, one of the two areas may be used as an originating point (where Forward Monitoring cells are generated).

	31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
W0 = E0	Control Bits	MCSN	TUC
W1 = E1	BEDC		TUC0
W2 = E2	Reserved	BMCSN	ECI
W3 = E3	Reserved	LMCSN	TUCD
W4 = I0	Control Bits	MCSN	TUC
W5 = I1	BEDC		TUC0
W6 = I2	Reserved	BMCSN	ICI
W7 = I3	Reserved	LMCSN	TUCD

Figure 7-101. OAM Table Record

- **MCSN**—The MC92501 uses this field for storing the Monitoring Cell Sequence Number of the next Forward Monitoring cell.
- **TUC**—The MC92501 uses this field for storing the running Total User Cell count of the current block.
- **BEDC**—The MC92501 uses this field for storing the running Block Error Detection Code (BIP-16) of the current block of user cells.
- **TUC0**—The MC92501 uses this field for storing the running Total CLP=0 User Cell count of the current block.
- **BMCSN**—The MC92501 uses this field for storing the Monitoring Cell Sequence Number of the next Backward Reporting cell.
- **ECI/ICI**—This field contains the connection identifier of the connection on which the Performance Monitoring test is being performed. In the Egress area this is the Egress Connection Identifier, and in the Ingress area it is the Ingress Connection Identifier. This field is initialized by the user when the test is set up.
- **LMCSN**—The MC92501 uses this field for storing the Monitoring Cell Sequence Number of the previous FMC. This field does not require initialization by the user.
- **TUCD**—The MC92501 uses this field for storing the difference between the Total User Cell counts when the previous FMC arrived. This field does not require initialization by the user

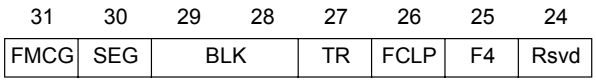


Figure 7-102. Control Bit Fields

- **Forward Monitoring Cell Generation (FMCG)** — The FMCG bit indicates that this is the originating point of the Performance Monitoring test, and FMCs should be generated here.
  - 0 = FMCs should not be generated.
  - 1 = FMCs should be generated.
- **Segment (SEG)** — The SEG bit determines whether Performance Monitoring is performed on the entire connection or only on a connection segment.
  - 0 = The Performance Monitoring test is being performed on an entire connection. Generated monitoring cells should be End-to-end OAM cells.
  - 1 = The Performance Monitoring test is being performed on a connection segment. Generated monitoring cells should be Segment OAM cells.
- **Block Size (BLK)** — This field defines the block size for the Performance Monitoring test. It should be initialized by the user and is not changed by the MC92501.
  - 00 = 128 cells.
  - 01 = 256 cells.
  - 10 = 512 cells.
  - 11 = 1024 cells.
- **Test Running (TR)** — This bit indicates that the test is currently running. It should be reset when the record is initialized. It is set by the MC92501 when an FMC is processed. It may be reset by the processor in case of a failure. See **Section 6.3.7** for details.
- **FMC CLP bit (FCLP)** — This bit is used as the CLP bit in the header of generated FMCs. It should be initialized by the user and is not changed by the MC92501.
- **F4** — This bit determines whether Performance Monitoring is performed at the F4 or F5 level.
  - 0 = The Performance Monitoring test is being performed at the F5 level.
  - 1 = The Performance Monitoring test is being performed at the F4 level.

7.3.13 Dump Vector Table

The Dump Vector Table contains a trace of the activities of the MC92501. The MC92501 writes one record per cell time to the Dump Vector Table. Each record consists of two long words, one for the Egress processing (shown in **Figure 7-103**) and one for the Ingress processing (shown in **Figure 7-104**). The size of the Dump Vector Table is determined by the *Dump Vector Table Control (DVTC)* field of the ATMC CFB Configuration Register (ACR). When the end of the table is reached, the write pointer wraps around to the beginning of the table and the records are overwritten.

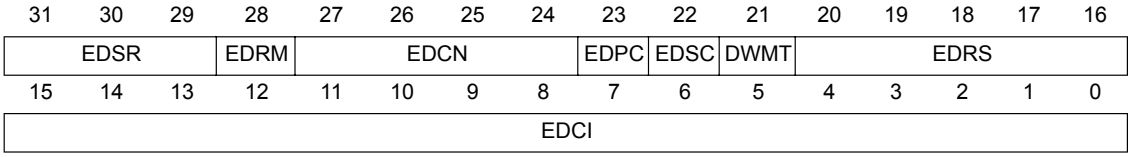


Figure 7-103. Dump Vector Table Egress Long Word Fields

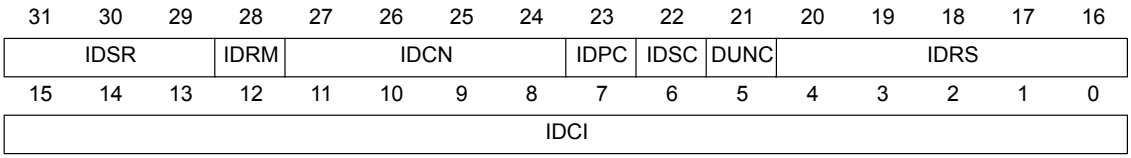


Figure 7-104. Dump Vector Table Ingress Long Word Fields

- **Egress Dump Cell Source (EDSR)**—The EDSR field specifies the source of the Egress-processed dumped cell. See **Section 7.4.2.1** for the field definition.
- **Egress Dump Removed (EDRM)**—The EDRM bit, when set, indicates that the cell was removed from the Egress cell flow.
- **Egress Dump Cell Name (EDCN)**—The EDCN field specifies the type of cell that was processed. See **Section 7.4.4.1** for the field definition.
- **Egress Dump Primary Copy (EDPC)**—When set, this bit indicates that the primary cell copied to the Cell Extraction Queue from the Egress during the PREVIOUS cell time was copied to the MPI and read by the Microprocessor.
- **Egress Dump Secondary Copy (EDSC)**—When set, this bit indicates that the MPI did not filter out the secondary cell copied to the Cell Extraction Queue from the Egress during the PREVIOUS cell time and was read by the Microprocessor.
- **Dump Was Maintenance (DWMT)**—The DWMT bit indicates that the previous cell time was a Maintenance Slot. Therefore, the previous record is invalid.
- **Egress Dump Action Reason (EDRS)**—The EDRS field specifies the conclusion of the cell processing. If the cell is copied to the microprocessor, this value is the reason that the cell was copied. See **Section 7.4.4.2** for the field definition.
- **Egress Dump Connection Identifier (EDCI)**—The EDCI field contains the ECI of the processed cell.
- **Ingress Dump Cell Source (IDSR)**—The IDSR field specifies the source of the cell processed in the Ingress cell flow. See **Section 7.4.2.1** for the field definition.
- **Ingress Dump Removed (IDRM)**—The IDRM bit, when set, indicates that the cell was removed from the Ingress cell flow.

- **Ingress Dump Cell Name (IDCN)**—The IDCN field specifies the type of cell that was processed. See **Section 7.4.4.1** for the field definition.
- **Ingress Dump Primary Copy (IDPC)**—The IDPC bit, when set, indicates that the primary cell copied to the Cell Extraction Queue from the Ingress cell flow during the PREVIOUS cell time was copied to the MPI and is actually read by the Microprocessor.
- **Ingress Dump Secondary Copy (IDSC)**—The IDSC bit, when set, indicates that the secondary cell copied to the Cell Extraction Queue from the Ingress cell flow during the PREVIOUS cell time was not filtered out by the MPI and is actually read by the Microprocessor.
- **Dump UPC Non-Conforming (DUNC)**—The DUNC bit indicates that the UPC/NPC mechanism found the cell to be non-conforming.
- **Ingress Dump Action Reason (IDRS)**—The IDRS field specifies the conclusion of the cell processing. If the cell is copied to the microprocessor, this value is the reason that the cell was copied. See **Section 7.4.4.2** for the field definition.
- **Ingress Dump Connection Identifier (IDCI)**—The IDCI field contains the ICI of the processed cell.

7.3.14 Ingress Link Counters Table

This table contains the Ingress Link Counter records. Each counter wraps to zero after reaching its maximum value. **Figure 7-105** shows the record structure.

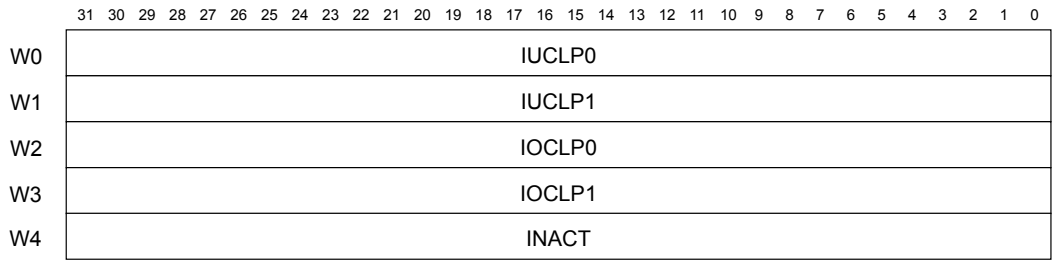


Figure 7-105. Ingress Link Counters

- **Ingress User CLP0 counter (IUCLP0)**—This 32-bit counter is used to count the incoming user (non-OAM) cells from the PHY whose Cell Loss Priority (CLP) is zero.
- **Ingress User CLP1 counter (IUCLP1)**—This 32-bit counter is used to count the incoming user (non-OAM) cells from the PHY whose Cell Loss Priority (CLP) is one.
- **Ingress OAM CLP0 counter (IOCLP0)**—This 32-bit counter is used to count the incoming OAM cells from the PHY whose Cell Loss Priority (CLP) is zero.
- **Ingress OAM CLP1 counter (IOCLP1)**—This 32-bit counter is used to count the incoming OAM cells from the PHY whose Cell Loss Priority (CLP) is one.
- **Inactive Cell Counter (INACT)**—This 32-bit counter is used to count the incoming cells from the PHY on which the address compression algorithm failed to produce a valid ICI value.

7.3.15 Egress Link Counters Table

This table contains the Egress Link Counter records. Each counter wraps to zero after reaching its maximum value. **Figure 7-106** shows the record structure.

	31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
W0	EUCLP0
W1	EUCLP1
W2	EOCLP0
W3	EOCLP1

Figure 7-106. Ingress Link Counters

- **Egress User CLP0 counter (EUCLP0)**—This 32-bit counter is used to count the outgoing user (non-OAM) cells to the PHY whose Cell Loss Priority (CLP) is zero.
- **Egress User CLP1 counter (EUCLP1)**—This 32-bit counter is used to count the outgoing user (non-OAM) cells to the PHY whose Cell Loss Priority (CLP) is one.
- **Egress OAM CLP0 counter (EOCLP0)**—This 32-bit counter is used to count the outgoing OAM cells to the PHY whose Cell Loss Priority (CLP) is zero.
- **Egress OAM CLP1 counter (EOCLP1)**—This 32-bit counter is used to count the outgoing OAM cells to the PHY whose Cell Loss Priority (CLP) is one.

7.3.16 Context Parameters Extension Table

Each Context Parameters Extension Table record contain one Common Parameters Extension Word. There are 4 modes for this word (shown in **Figure 7-107** through **Figure 7-110**) depending on the values of the *PM on All Connections (PMAC)* bit and the *UPC Flow (UPCF)* bit in the *ATMC CFB Configuration Register (ACR)*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BKPT[21:12]										0	0	CIFS	CEFS	ECTE	CEME
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PDV			0	0	EREF	0	ISDM		UOM		0	ICTV	ICTE	CIME

Figure 7-107. Parameter Extension Word Fields (PMAC = 1 and UPCF = 0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	PDV			0	0	EREF	0	ISDM		0	0	CIFS	CEFS	ECTE	CEME
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BKPT[21:12]										UOM		0	ICTV	ICTE	CIME

Figure 7-108. Parameter Extension Word Fields (PMAC = 1 and UPCF = 1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	CIFS	CEFS	ECTE	CEME
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PDV			0	0	EREF	0	ISDM		UOM		0	ICTV	ICTE	CIME

Figure 7-109. Parameter Extension Word Fields (PMAC = 0 and UPCF = 0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	PDV			0	0	EREF	0	ISDM		0	0	CIFS	CEFS	ECTE	CEME
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	UOM		0	ICTV	ICTE	CIME

Figure 7-110. Parameter Extension Word Fields (PMAC = 0 and UPCF = 1)

- **Bucket Pointer[21:12] (BKPT[21:12])**—When the *PM on All Connections (PMAC)* bit is set, this field contains bits 21–12 of the Bucket Pointer. When the *PMAC* is cleared, this field is reserved and should equal 0.
- **Connection Ingress Flow Status (CIFS)**—The MC92501 copies the *Overhead Ingress Flow Status (IFS)* bit to this bit. This bit is used by the Ingress processing block for ABR cell marking and is, therefore, intended for MC92501 internal use. See **Section 6.4.3.1** for details.
- **Connection Egress Flow Status (CEFS)**—The MC92501 copies the *Overhead Egress Flow Status (EFS)* bit to this bit. This bit is used by the Ingress processing block for ABR cell marking and is, therefore, intended for MC92501 internal use. See **Section 6.4.3.2** for details.
- **Egress CLP Transparency Enable (ECTE)**—This bit determines whether CLP should be copied from the *Egress Overhead CLP (EOCLP)* bit to the cell header. See **Section 6.5** for details.
  - 0 = CLP should not be copied from the switch overhead to the cell header.
  - 1 = CLP should be copied from the switch overhead to the cell header.
- **Connection Egress Marking Enable (CEME)**—This bit enables marking of cells which are received in the Egress. See **Section 6.4.3.4** for details.
  - 0 = Marking of cells which are received in the Egress is disabled.
  - 1 = Marking of cells which are received in the Egress is enabled.
- **Packet Discard Variables (PDV)**—This field is accessed only by the MC92501.
- **Egress Reset EFCI (EREF)**—This bit determines if PTI[1] of an Egress cell is reset.
  - 0 = PTI[1] of an Egress cell is not reset.
  - 1 = PTI[1] of an Egress cell is to be reset.
- **Ingress Selective Discard Operation Mode (ISDM)**—This field determines the Selective Discard operation mode. See **Section 6.6** for details.
  - 00 = No Selective Discard.
  - 01 = Reserved.
  - 10 = Selective Discard on CLP = 1 flow.
  - 11 = Selective discard on CLP = 0+1 flow.
- **UPC Operation Mode (UOM)**—This field determines the UPC operation mode.
  - 00 = cell-based UPC.
  - 01 = Partial Packet Discard (PPD). See **Section 6.2.4** for details
  - 10 = Early Packet Discard (EPD). See **Section 6.2.5** for details.
  - 11 = Limited EPD. See **Section 6.2.6** for details.

- **Ingress CLP Transparency Value (ICTV)**—This bit determines the value that should be written to a cell's header if the *Ingress CLP Transparency Enable (ICTE)* bit is set. See **Section 6.5** for details.
- **Ingress CLP Transparency Enable (ICTE)**—This bit determines whether CLP should be copied to the *Ingress Overhead CLP (IOCLP)* bit and whether the *Ingress CLP Transparency Value (ICTV)* bit should be written to the cell header CLP. See **Section 6.5** for details.
  - 0 = The Ingress header CLP bit is not touched.
  - 1 = CLP should be copied from the cell header to the Ingress switch parameters. The ICTV bit should be written to the cell header CLP.
- **Connection Ingress Marking Enable (CIME)**—This bit enables marking of cells which are received in the Ingress. See **Section 6.4.3.3** for details.
  - 0 = Marking of cells which are received in the Ingress is disabled.
  - 1 = Marking of cells which are received in the Ingress is enabled.

7.4 Data Structures

NOTE: All fields marked "0" or "reserved" in the descriptions in this section must be written with zeros. The values read from these fields should be considered undefined and should be ignored.

7.4.1 Inserted Cell

The Inserted Cell structure contains 16 long words as shown in **Figure 7-111**. The contents of the first and second long words are described below. The third long word is unused. The fourth long word contains the ATM cell header (not including the HEC

octet), and the last twelve long words contain the cell payload.

CIR0	Cell Descriptor	ACIR0
CIR1	Connection Descriptor	ACIR1
CIR2	Unused	
CIR3	ATM cell header (VPI,VCI,PTI,CLP)	
CIR4	Payload 1–4	
CIR5	Payload 5–8	
CIR6	Payload 9–12	
CIR7	Payload 13–16	
CIR8	Payload 17–20	
CIR9	Payload 21–24	
CIR10	Payload 25–28	
CIR11	Payload 29–32	
CIR12	Payload 33–36	
CIR13	Payload 37–40	
CIR14	Payload 41–44	
CIR15	Payload 45–48	

Figure 7-111. Inserted Cell Structure

7.4.1.1 Cell Descriptor

The cell descriptor informs the MC92501 what should be done with the Inserted Cell. Five formats are used for the cell descriptor, depending on how the cell header is provided. If the user provides the ATM header and payload, Format I is used, shown in **Figure 7-112**.

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

Figure 7-112. Cell Descriptor Format I

When the user provides the non-address[1] part of the ATM header and payload, and the MC92501 performs address translation, Format II is used, as shown in **Figure 7-113**. This

1.The non-address part of the ATM header consists of the fields that are not modified during address translation. Generally, this includes the PTI and CLP fields. If VP switching is performed, it generally includes the VCI field. See **Section 7.2.6.10** Ingress Processing Configuration Register (IPCR) and **Section 7.2.6.11** Egress Processing Configuration Register (EPCR) for more details.

format is also used for inserting holes in the cell flow.

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

Figure 7-113. Cell Descriptor Format II

When the user provides the ATM header and an OAM fields template, Format III is used (shown in **Figure 7-114**).

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

Figure 7-114. Cell Descriptor Format III

When the user provides the non-address part of the ATM header and an OAM fields template, and the MC92501 performs address translation, Format IV is used (see **Figure 7-115**).

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

Figure 7-115. Cell Descriptor Format IV

When the MC92501 generates the OAM cell header and payload, Format V is used (see **Figure 7-116**).

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

Figure 7-116. Cell Descriptor Format V

- **Descriptor Code (DCODE)**—The DCODE field indicates the type of processing the MC92501 should perform on the inserted cell. It also determines the format of the remainder of the Cell Descriptor word (listed in parentheses).
  - 00000 = The user provides the header and payload. The MC92501 generates the CRC-10 field. (Format I)
  - 00001 = The user provides the header and payload. The MC92501 does not modify the cell. (Format I)
  - 00010 = The user provides the non-address part of the header and payload. The ATMC CFB performs address translation and generates the CRC-10 field.

(Format II)

- 00011 = The user provides the non-address part of the header and payload. The MC92501 performs address translation. (Format II)
- 00100 = The MC92501 generates the header and the payload. This value is only used for OAM cells. (Format V)
- 00110 = The user provides the non-address part of the header and an OAM fields template (see **Section 7.4.1.4**. The MC92501 performs address translation and generates the payload from the OAM fields template. (Format IV)
- 00111 = The user provides the header and an OAM fields template as defined in **Section 7.4.1.4**. The MC92501 generates the payload from the OAM fields template. (Format III)
- 01000 = The user provides the header and the payload, but does not provide a connection identifier (see **Section 7.4.1.2**). The MC92501 generates the CRC-10 field and inserts the cell, but does not perform any connection-specific processing. (Format I)
- 01001 = The user provides the header and the payload, but does not provide a connection identifier (see **Section 7.4.1.2**). The MC92501 inserts the cell, but does not perform any connection-specific processing. (Format I)
- 01100 = The MC92501 inserts a hole (i.e., a cell processing slot is used, but no cell is inserted). (Format II)
- 01101 = The user provides the header, the payload, and a connection identifier (see **Section 7.4.1.2**). The MC92501 generates the CRC-10 field and inserts the cell, but does not perform any connection-specific processing except for appending the switch parameters. (Format I)
- 01110 = The user provides the header, the payload, and a connection identifier (see **Section 7.4.1.2**). The MC92501 inserts the cell, but does not perform any connection-specific processing except for appending the switch parameters. (Format I)

NOTE: If Ingress Address Translation is enabled, it is performed for all cells that undergo connection-specific processing, even if the user provides the header (DCODE = 00000 or 00001).

When a cell is inserted into the Ingress cell flow without a connection identifier (DCODE = 01000 or 01001), switch parameters from the Context Parameters Table are not provided and address translation is not performed. To insert a cell that is not processed, but does have switch parameters appended, use DCODE = 01101 or 01110.

- **Descriptor Cell Flow (DCF)**—This bit determines into which cell flow the cell is to be inserted (Ingress or Egress).
  - 0 = The cell is to be inserted into the Egress cell flow.
  - 1 = The cell is to be inserted into the Ingress cell flow.
- **Descriptor Segment (DSEG)**—The DSEG bit determines whether the inserted OAM cell is end-to-end or segment, and the header is generated accordingly.
  - 0 = The inserted cell is an end-to-end OAM cell.

— 1 = The inserted cell is a segment OAM cell.

- **Descriptor CLP (DCLP)** — The DCLP bit is used as the CLP bit in the ATM header of the generated cell.
- **Descriptor Cell Name (DCN)** — The DCN field identifies the type of OAM cell to be generated. The coding of this field can be found in Table 7-16. Currently, the only values valid in the Cell Descriptor are BRC for Formats III and IV and AIS, RDI, CC, and FMC for Format V.
- **Descriptor Link (DLINK)** — For a cell inserted into the Ingress cell flow, the DLINK field indicates to which physical link the cell belongs. The MC92501 ignores this field, but the Customer-Specific Logic or switch interface blocks may use this information. For a cell inserted into the Egress cell flow, the DLINK field indicates to which physical link the cell should be transferred.

7.4.1.2 Connection Descriptor

The second long word of the inserted cell contains the connection descriptor. The fields contained in this word determine the connection into which the cell is inserted.

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

Figure 7-117. Connection Descriptor Structure

- **Egress Connection Identifier / Ingress Connection Identifier (ECI/ICI)** — This field identifies the connection to which the inserted cell belongs. Depending on the Cell Flow bit described in **Section 7.4.1.1**, this field is interpreted as either the Egress Connection Identifier or the Ingress Connection Identifier. In either case, it is used as the index to the Context Parameters Table during the cell processing.

7.4.1.3 ATM Cell Header

The fourth long word of the inserted cell contains the ATM cell header. The fields are defined by ATM standards.

NOTE: Depending on the DCODE of the inserted cell, portions of the header may be altered during the cell processing.

At a UNI, the header fields are structured as shown in **Figure 7-118**.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GFC				VPI								VCI(15:12)			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VCI(11:0)												PTI			CLP

Figure 7-118. Inserted Cell ATM Cell Header Fields (UNI)

At an NNI, the header fields are structured as shown in **Figure 7-119**.

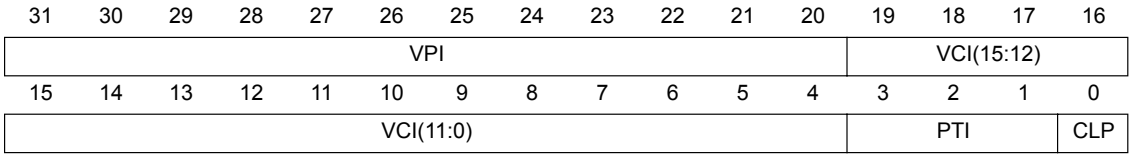


Figure 7-119. Inserted Cell ATM Cell Header Fields (NNI)

- Generic Flow Control (GFC)
- Virtual Path Identifier (VPI)
- Virtual Channel Identifier (VCI)
- Payload Type Identifier (PTI)
- Cell Loss Priority (CLP)

7.4.1.4 Inserted OAM Fields Template

When the MC92501 is requested to generate the payload of an OAM cell from a template, several of the long words of the payload portion of the inserted cell structure contains the OAM fields template. The remaining payload long words are undefined. The only type of OAM fields template currently defined is a BRC fields template. The BRC fields template is normally copied directly from the template provided by the MC92501 (see **Section 7.4.2.5** Extracted OAM Fields Template).

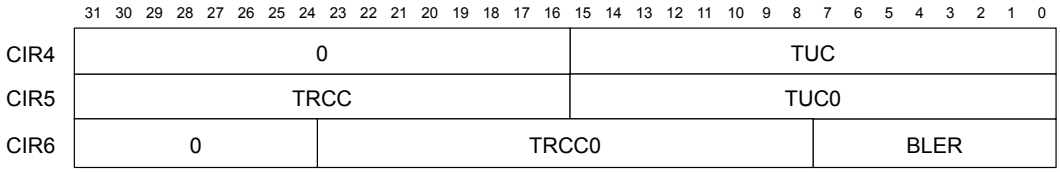


Figure 7-120. Inserted BRC Fields Template

- **TUC**—This field is used for the third and fourth octets of the generated cell payload. See **Section 6.3.6** Performance Management.
- **TRCC**—This field is used for the forty-fifth and forty-sixth octets of the generated cell payload. See **Section 6.3.6** Performance Management.
- **TUC0**—This field is used for the seventh and eighth octets of the generated cell payload. See **Section 6.3.6** Performance Management.
- **TRCC0**—This field is used for the forty-second and forty-third octets of the generated cell payload. See **Section 6.3.6** Performance Management.
- **Block Error Result (BLER)**—This field is used for the forty-fourth octet of the generated cell payload. See **Section 6.3.6** Performance Management.

7.4.2 Extracted Cell

The extracted cell structure contains 16 long words as shown in **Figure 7-121**. The contents of the first through third long words are described below. The fourth long word contains the ATM cell header (not including the HEC octet), and the last twelve long words contain the cell payload.

CER0	Cell Indication
CER1	Connection Indication
CER2	Time-Stamp
CER3	ATM cell header (VPI,VCI,PTI,CLP)
CER4	Payload 1–4
CER5	Payload 5–8
CER6	Payload 9–12
CER7	Payload 13–16
CER8	Payload 17–20
CER9	Payload 21–24
CER10	Payload 25–28
CER11	Payload 29–32
CER12	Payload 33–36
CER13	Payload 37–40
CER14	Payload 41–44
CER15	Payload 45–48

Figure 7-121. Extracted Cell Structure

7.4.2.1 Cell Indication

The cell indication informs the processor about where the cell came from and why it was extracted. There are five formats for the cell indication. The formats can be distinguished by the values of the most-significant bits.

7.4.2.1.1 User/OAM Cell Indication

This format is used for a user or OAM cell. When this format is used, the entire cell structure contains valid data.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	USR	ICF	Reserved			ISRC			ICN			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						GFCR	RSN					ILINK			

Figure 7-122. User/OAM Cell Indication Fields

- **User Cell (USR)**—This bit distinguishes between OAM cells and user cells.
  - 0 = OAM cell
  - 1 = User cell

- **Indication Cell Flow (ICF)** — This bit indicates from which cell flow the cell has been copied (Ingress or Egress).
  - 0 = The cell was in the Egress cell flow.
  - 1 = The cell was in the Ingress cell flow.
- **Indication Source (ISRC)** — The ISRC field indicates the source of the cell.
  - 001 = The cell arrived from the PHY/Switch.
  - 010 = The cell was inserted through the processor interface.
  - 011 = The FMC was internally generated.
  - 100 = The cell was generated by the Internal Scan process.
  - 101 = The cell was inserted by the Customer-Specific Logic.
  - All unused values are reserved.
- **Indication Cell Name (ICN)** — This field identifies the type of cell being extracted. The coding of this field can be found in Table 7-16.
- **GFC Reason (GFCR)** — The GFCR bit, when set, indicates that the GFC field in the cell's header is non-zero.
- **Extraction Reason (RSN)** — This field contains the reason that the MC92501 copied this cell to the Cell Extraction Queue. The coding of this field can be found in **Table 7-17**. Note that there may be instances in which more than one reason applies, but only one reason is provided.
- **Indication Link (ILINK)** — This field identifies the physical link on which the cell arrived (Ingress) or would be transmitted (Egress).

7.4.2.1.2 Egress Multicast Translation Failure Cell Indication

This format is used for a cell from the Egress flow if the Multicast Translation failed. When this format is used, the entire cell structure contains valid data.

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

Figure 7-123. Egress Multicast Translation Failure Cell Indication Fields

- **Indication Cell Flow (ICF)** — This bit is reset to indicate that the cell was copied from the Egress cell flow.
- **Indication Source (ISRC)** — The ISRC field is 001 which indicates that the cell arrived from the switch.
- **Extraction Reason (RSN)** — This field contains the reason that the MC92501 copied this cell to the Cell Extraction Queue. The coding of this field can be found in Table 7-17. The only defined value is Address Compression failure.

When this Cell Indication is used, the Connection Indication word appears as described in **Section 7.4.2.2**.

7.4.2.1.3 Error Cell Indication

This format is used if an error was found in the first stages of processing. When this format is used, the entire cell structure contains valid data.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	1	1	ICF	Reserved			ISRC			DCN			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDCODE					Reserved	RSN						DLINK			

Figure 7-124. Erro Cell Indication Fields

- **Indication Cell Flow (ICF)**—This bit indicates from which cell flow the cell has been copied (Ingress or Egress).
  - 0 = The cell was in the Egress cell flow.
  - 1 = The cell was in the Ingress cell flow.
- **Indication Source (ISRC)**—The ISRC field indicates the source of the cell.
  - 001 = The cell arrived from the PHY/Switch.
  - 010 = The cell was inserted through the processor interface.
  - 011 = The FMC was internally generated.
  - 100 = The cell was generated by the Internal Scan process.
  - 101 = The cell was inserted by the Customer-Specific Logic.
  - All unused values are reserved.
- **Descriptor Cell Name (DCN)**—This field contains the Cell Name field from the Cell Descriptor. Depending on the source of the cell, this field might not be meaningful. The coding of the Cell Name can be found in Table 7-16.
- **Cell Descriptor Code (CDCODE)**—This field contains the DCODE field from the Cell Descriptor. Depending on the source of the cell, the value of this field might not be identifiable.
- **Extraction Reason (RSN)**—This field contains the reason that the MC92501 copied this cell to the Cell Extraction Queue. The coding of this field can be found in **Table 7-17**. Note that there may be instances in which more than one reason applies, but only one reason is provided.
- **Descriptor Link (DLINK)**—This field contains the DLINK field from the Cell Descriptor. Depending on the source of the cell, this field might not be meaningful.

7.4.2.1.4 Short Report Indication

This format is used to report various events that occurred in the course of the cell processing. When this format is used only CER0–CER2 contain valid data.

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

Figure 7-125. Short Report Indication Fields

- **Indication Cell Flow (ICF)**—This bit indicates from which cell flow the report originated (Ingress or Egress).
  - 0 = The report is from the Egress cell flow.
  - 1 = The report is from the Ingress cell flow.
- **Indication Source (ISRC)**—The ISRC field indicates the source of the cell that triggered the report generation.
  - 001 = The cell arrived from the PHY/Switch.
  - 010 = The cell was inserted through the processor interface.
  - 101 = The cell was inserted by the Customer-Specific Logic.
  - All unused values are reserved.
- **Extraction Reason (RSN)**—This field contains the reason that the MC92501 generated this report. The coding of this field can be found in **Table 7-17**. Currently, the only defined reasons for the Short Report indication are the two values regarding FMCs not being inserted.

When using the Short Report indication, the connection identifier is found in the Connection Indication word described in **Section 7.4.2.2** Connection Indication.

7.4.2.1.5 OAM Fields Template Indication

This format is used for an OAM fields template.

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

Figure 7-126. OAM Fields Template Fields

- **Indication Cell Flow (ICF)**—This bit indicates from cell flow in which the template was generated (Ingress or Egress).
  - 0 = The template is from the Egress cell flow.
  - 1 = The template is from the Ingress cell flow.
- **Indication Source (ISRC)**—The ISRC field indicates the source of the cell that triggered the generation of the template.
  - 001 = The cell arrived from the PHY/Switch.
- **Indication Cell Name (ICN)**—This field identifies the cell type to generate from the template. Field coding is in Table 7-16. Currently, only BRC is defined.
- **Extraction Reason (RSN)**—This field defines the template type. The field coding is in **Table 7-17**. Currently, only BRC is defined.
- **Indication Link (ILINK)**—This field identifies the physical link on which the cell that triggered the generation of the template arrived (Ingress) or would be transmitted (Egress).

When the OAM Fields Template indication is used, the payload portion of the extracted cell structure contains the OAM fields template as described in **Section 7.4.2.5** Extracted OAM Fields Template.

7.4.2.2 Connection Indication

The second long word of the extracted cell contains the connection indication. The fields contained in this word indicate the connection to which the cell/report/template belongs.

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

Figure 7-127. Connection Indication Fields

- **Egress Connection Identifier / Ingress Connection Identifier (ECI/ICI)** — This field identifies the connection to which the extracted cell/report/template belongs. Depending on the Cell Flow bit of the Cell Indication, this field is interpreted as either the Egress Connection Identifier or the Ingress Connection Identifier.

When the Egress Multicast Translation Failure Indication described in **Section 7.4.2.1.2** Egress Multicast Translation Failure Cell Indication is used, the Connection Indication word contains overhead information obtained from the switch which may be helpful in determining the cause of the failure.

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

Figure 7-128. Egress Multicast Translation Failure Connection Overhead Information

- **Address Translation Disable (ATD)** — The ATD bit, when set, indicates that Address Translation is disabled. This bit is taken from the *Egress Address Translation Disable (EATD)* bit of the Egress Switch Interface Configuration Register (ESWCR).
- **Multicast Translation Table Section (MTTS)** — The MTTS field is used in Multicast Identifier Translation. See **Section 5.3.2** Multicast Identifier Translation.
- **LINK** — The LINK field contains the link number if ATD is set. Otherwise, this field contains 0000.
- **Multicast Identifier (MI)** — This field contains the Multicast Identifier used in Multicast Identifier Translation. See **Section 5.3.2** Multicast Identifier Translation.

7.4.2.3 Time-Stamp

The third long word of the extracted cell contains a time-stamp indicating when the cell was processed. The Time-Stamp field is in units of MC92501 cell processing times and represents the number of cell processing times since the MC92501 was reset (modulo 2^{32}).

7.4.2.4 ATM Cell Header

The fourth long word of the extracted cell contains the incoming ATM cell header. The fields are defined by ATM standards. At a UNI, the header fields are:

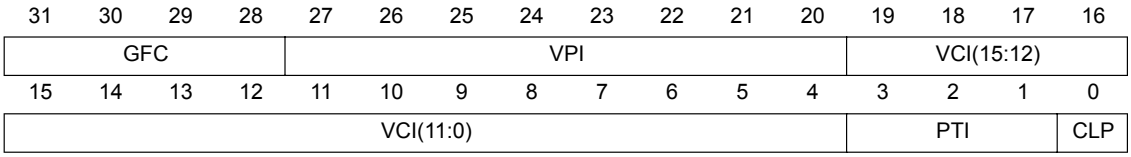


Figure 7-129. Extracted Cell ATM Cell Header (UNI)

At an NNI, the header fields are:

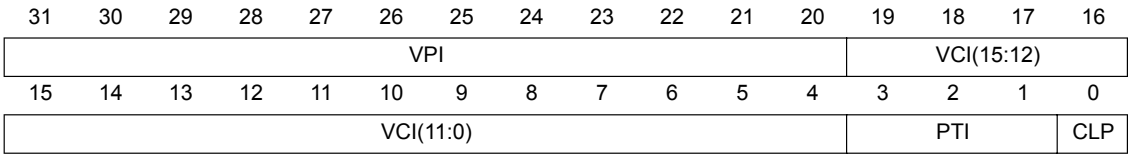


Figure 7-130. Extracted Cell ATM Cell Header (NNI)

- Generic Flow Control (GFC)
- Virtual Path Identifier (VPI)
- Virtual Channel Identifier (VCI)
- Payload Type Identifier (PTI)
- Cell Loss Priority (CLP)

7.4.2.5 Extracted OAM Fields Template

When the OAM Fields Template indication is used, several of the long words of the payload portion of the extracted cell structure contain the OAM fields template. The remaining payload long words are undefined. The only type of OAM fields template currently defined is a BRC fields template. The BRC fields template is normally copied directly to the template inserted to the MC92501 (see **Section 7.4.1.4** Inserted OAM Fields Template).

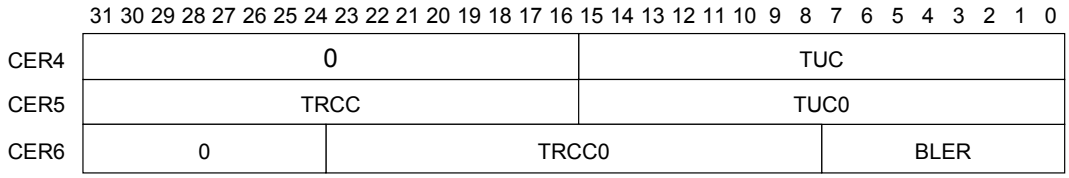


Figure 7-131. Extracted BRC Fields Template

- **TUC**—This field is copied from the TUC of the received FMC. See **Section 6.3.6** Performance Management.
- **TRCC**—This field is copied from the TUC of the OAM Table entry. See **Section 6.3.6** Performance Management.
- **TUC0**—This field is copied from the TUC0 of the received FMC. See **Section 6.3.6** Performance Management.
- **TRCC0**—This field is copied from the TUC0 of the OAM Table entry. See **Section 6.3.6** Performance Management.
- **Block Error Result (BLER)**—This field is computed from the BEDC of the received FMC and the OAM Table entry. See **Section 6.3.6** Performance Management.

7.4.3 External Address Compression

The address of a cell received from the PHY layer is presented on the EMDATA bus using the following structure:

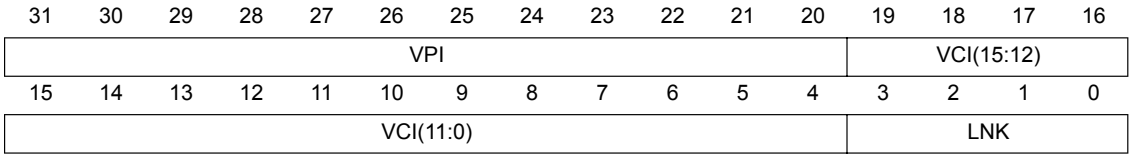


Figure 7-132. Received Cell Address Structure

- **Virtual Path Identifier (VPI)**—This field is taken from the ATM cell header.
- **Virtual Channel Identifier (VCI)**—This field is taken from the ATM cell header.
- **Physical Link Number (LNK)**—This field is valid only when the MC92501 is supporting multiple PHY devices. It contains the number of the physical link from which the cell was received. When only one PHY device is supported, this field is set to 0000.

The user's response to the External Address Compression uses the structure shown in **Figure 7-133**.

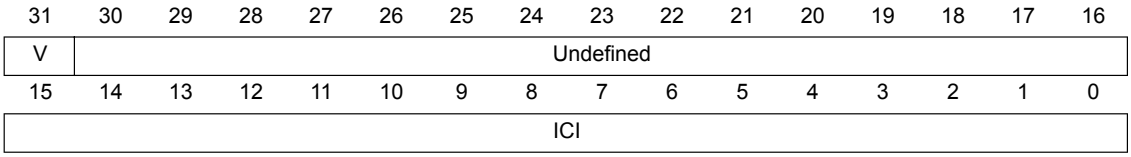


Figure 7-133. User Response to EAC Structure

- **Valid (V)**—If this bit is set, the ICI is valid. Otherwise, the compression has failed, and the cell is treated as inactive.
- **Ingress Connection Identifier (ICI)**—This field contains an index to the Context Parameter Table records of the connection to which the cell belongs. The reserved value of all 1s indicates that the address compression has failed even if the Valid bit is set. This value should not be used as the connection identifier of any connection.

7.4.4 General Fields

This section contains the coding for fields that are shared by multiple data structures. Not all of the values are valid for all of the structures.

7.4.4.1 Cell Name

Table 7-16 describes the coding of the Cell Name field.

Table 7-16. Cell Name Coding

CN	Description
0000	Undefined
0001	OAM AIS Cell
0010	OAM RDI Cell
0011	OAM Loopback Cell
0100	OAM Continuity Check Cell
0101	OAM Forward Monitoring Cell
0110	OAM Backward Reporting Cell
1000	Other OAM Cell
1001	User Cell

7.4.4.2 Reason

Table 7-17 describes the coding of the Reason field. The reasons are listed in order of increasing priority, rather than in strict numerical order. If more than one reason applies, the last reason listed is the one that is provided with the cell.

Table 7-17. Reason Coding

RSN	Description
00000	Undefined
00001	The cell was copied because the Egress Copy All (ECA) cells or Ingress Copy All (ICA) cells bit was set. See **Section 7.3.3** Context Parameters Table.
00010	The cell is an OAM cell that was copied because one of the OAM copy bits in the Context Parameters Table was set. See **Section 7.3.3** Context Parameters Table.
00100	The cell is a loopback cell that returned to its source.
00101	The cell is a loopback cell to be looped back at this point.
00110	The cell was copied due to a reserved VCI value as indicated by the VCR register. See **Section 7.2.6.28** Ingress VCI Copy Register (IVCR) and **Section 7.2.6.29** Egress VCI Copy Register (EVCR).
00111	The cell was copied due to a reserved value of PTI. See **Section 7.3.3.5** Egress Parameters and **Section 7.3.3.6** Ingress Parameters.
01001	A CRC error was detected.
01010	Illegal OAM cell. See **Section 6.3.4.1** Illegal OAM Cells.
01100	The PHY/Switch interface detected a parity error.
01101	The cell has an invalid link number. (Egress only)
01110	The connection ECIV/ICIV bit is reset.
01111	The Address Compression failed.
10000	One or more of the unmasked header bits of the cell is set.
01011	Illegal cell
10001	The cell is an invalid (PHY-layer) cell.
10010	The PHY/Switch interface detected a protocol error.
10011	No cell was processed because the output queue is full.
10100	No cell was processed because there was no cell to process.
10101	No cell was processed because it was a Maintenance Slot.
11101	This is a BRC fields template.
11110	This connection is running a PM block test, and an end-to-end FMC was not inserted on time.
11111	This connection is running a PM block test, and a potential FMC was discarded because it could not be inserted.

Note: Some of the reasons listed here never appear in the Cell Extraction Queue.

8

Test Operation

The IEEE JTAG Boundary Scan Architecture and Test Access Port (TAP) Controller and the RAM Built-In Self-Test (BIST) operations are discussed in the following subsections.

8.1 JTAG Overview

The MC92501 provides a test access port (TAP) that is compatible with the IEEE 1149.1 *Standard Test Access Port and Boundary Scan Architecture* [6]. This implementation of IEEE 1149.1 allows boundary scan compatibility with other JTAG components in a circuit-board. A boundary scan description language (BSDL) file for this device is provided in **Appendix F**.

The TAP includes five dedicated signal pins, a 16-state TAP controller, a bypass register, an instruction register, and a device Identification (ID) Register. The ID register contains the specific JTAG ID code. There is a boundary scan register that links all device I/O signal pins into a shift-register chain around the periphery of the device. This path is provided with serial input and output signal pins and is controlled by appropriate clock and control signals.

The JTAG test logic is independent of the device system logic, except for the requirement that the device be in System Mode (TESTSCAN = 0). The JTAG implementation on the MC92501 provides for the following capabilities:

- Boundary scan operations
- BYPASS mode, which bypasses the MC92501 boundary scan chain
- Sampling system data from I/O signals (for inputs) and core system data (for outputs) and shifting out the result through the Boundary Scan Register
- Test of component interconnect with EXTEST mode. This instruction updates I/O and system logic with data that is shifted in serially. Output pins are driven with the updated values, and input pins have the updated values driven into the core logic.
- CLAMP mode, which drives as in EXTEST mode, but puts boundary scan chain in BYPASS mode
- IDCODE mode allows the shifting out of the ID code from the ID register.
- HIGHZ mode tristates all system outputs and bi-directionals for component isolation.

8.1.1 Functional Blocks

The MC92501 implementation of the IEEE 1149.1 standard includes a TAP controller, an instruction register, a bypass register, an idcode register, and the I/O boundary scan register. The associated signal pins for the JTAG logic are:

- **TCK**—Test Clock input for the JTAG test logic.
- **TMS**—Test Mode Select input used to sequence the TAP controller's state machine, sampled on the rising edge of TCK.
- **TDI**—Test Data Input is sampled on the rising edge of TCK.
- **TDO**—Test Data Output is active during the Shift-DR and Shift-IR controller states.
- **TRST**—Asynchronous Test Reset signal that initializes the TAP controller and other JTAG logic.
- **TESTSCAN**—Mode pin for the MC92501. Needs to be in a low (0) state for system JTAG operation.

A block diagram overview is shown in **Figure 8-1**.

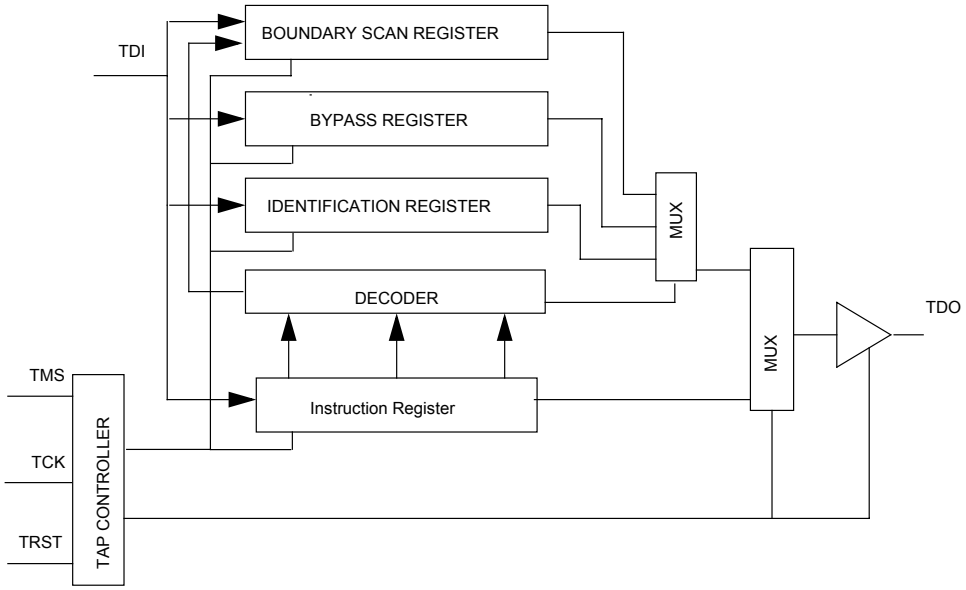


Figure 8-1. JTAG Logic Block Diagram

8.1.1.1 TAP Controller

The TAP Controller is a synchronous, 16-state machine which controls and manages the mode of operation for the test circuitry. The controller interprets the sequence of logical values on the TMS signal. The TAP controller states are explained in the IEEE 1149.1 document. The state diagram for the controller is shown in **Figure 8-2**.

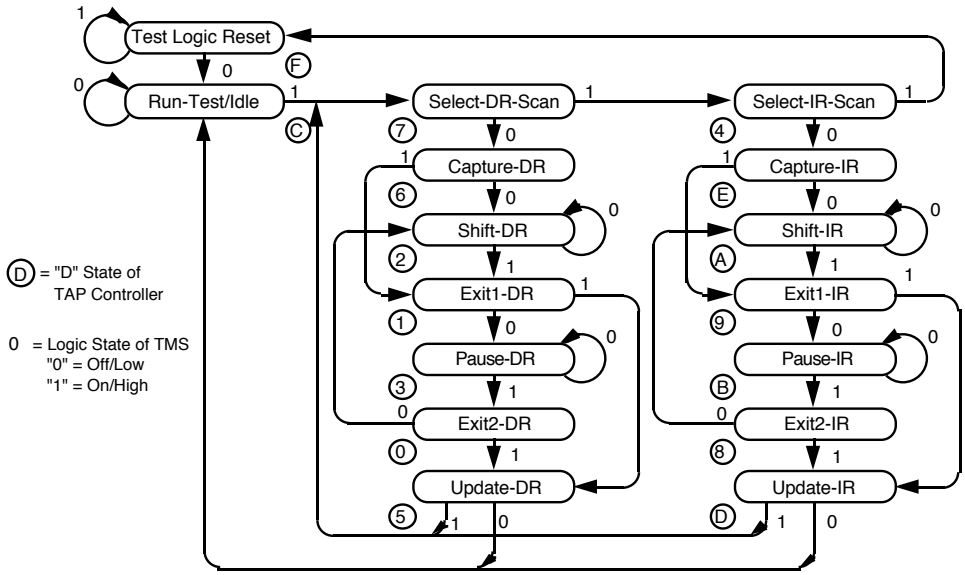


Figure 8-2. TAP Controller State Diagram

8.1.1.2 Instruction Register

The 4-bit instruction register holds various instructions that define which test registers are to be used and the serial test data register path between TDI and TDO signals. The instructions are described in **Section 8.1.2** JTAG Instruction Support.

8.1.1.3 Device Identification Register

The device identification register is a 32-bit register that holds the manufacturer's identity code, part number, and version code. The bit assignment for the ID code is shown in **Table 8-1**. The idcode data is shifted out with the least significant bit (0) shifted out first.

Table 8-1. Device Identification Register ID Codes

Bit #	Code Use	MC92501 Value
31-28	Version Number	(Revision A) 0000
27-22	Motorola ASIC Identification	01 0100
21-12	Sequence Number	00_0011_1011
11-0	Motorola Identification	0000_0001_1101

8.1.1.4 Bypass Register

The 1-bit bypass register is connected between TDI and TDO when either the BYPASS instruction or the CLAMP instruction is active. This allows for bypassing the boundary scan register while other components on a board are being tested. When the bypass register is selected by the current instruction, the data that is shifted out on the first clock edge is the previously latched data.

8.1.1.5 Boundary Scan Register

The boundary scan register allows testing of circuitry external to the MC92501. It also allows the signals flowing through the system pins to sampled and examined without interfering with the operation of the MC92501. The boundary scan register includes all the functional pins of the MC92501 and additional stages that control the direction and driving state of some of the pins.

8.1.2 JTAG Instruction Support

The following instructions are supported by the MC92501:

- **SAMPLE**—The SAMPLE instruction captures the input or output data from the I/O pads. The data is captured on the rising edge of TCK when the TAP controller is in the Capture-DR state, and can be observed when shifted out through the boundary scan register. This instruction does not update the core logic or the I/O pads. This makes it useful for preloading the boundary scan register with known data when entering the EXTEST instruction.
- **CORE SAMPLE**—The CORE SAMPLE private instruction is almost identical to the standard SAMPLE instruction. However, when the CORE SAMPLE instruction is active, the data that is captured during the Capture-DR state originates from the chip core, rather than the I/O pad. This difference is only detectable on bi-directional pins that are currently in the input state.
- **BYPASS**—The BYPASS instruction selects the bypass register for serial access between TDI and TDO. This instruction does not update the core logic or the I/O pads.
- **EXTEST**—The EXTEST instruction selects the boundary scan register. When the EXTEST instruction is active, the output pins and the internal core input signals are updated with test data that has been previously shifted into the boundary scan register. Also, the state of all input ports of the chip core are defined by the data shifted into the boundary scan register, and are updated on the falling edge of TCK in the Update-DR controller state. Refer to the IEEE 1149.1 document for more details on the function and use of EXTEST.
- **IDCODE**—The MC92501 includes a device identification register. The IDCODE instruction selects only the device identification register to be connected for serial access between TDI and TDO in the Shift-DR controller state. When the IDCODE instruction is selected, the vendor identification code is loaded into the ID register in the Capture-DR state.
- **CLAMP**—The CLAMP instruction selects the bypass register for serial access between TDI and TDO. The CLAMP instruction drives the core data and the chip I/O signals in the same manner as EXTEST, while the bypass register is selected as the serial TDI/TDO path.

- **HIGHZ**—The HIGHZ instruction selects the bypass register for serial access between TDI and TDO. This instruction also puts all system logic output pins (including bidirectional pins) in their inactive drive state. This instruction is provided to help isolate the component during circuit board testing.

Table 8-2 shows the coding of the instructions supported by the MC92501. Bit 0 is the first bit shifted in from TDI and the first bit shifted out through TDO. The shaded rows of the table indicate the code point that is defined in the BSDL file for each instruction. The other values are provided for completeness, but are not expected to be used.

Table 8-2. Instruction Decoding

Code				Instruction
Bit3	Bit2	Bit1	Bit0	
0	0	0	0	EXTEST
0	0	0	1	IDCODE
0	0	1	0	SAMPLE
0	0	1	1	HIGHZ
0	1	0	0	CLAMP
0	1	0	1	BYPASS
0	1	1	0	BYPASS
0	1	1	1	BYPASS
1	0	0	0	EXTEST
1	0	0	1	IDCODE
1	0	1	0	CORE SAMPLE
1	0	1	1	HIGHZ
1	1	0	0	CLAMP
1	1	0	1	BYPASS
1	1	1	0	BYPASS
1	1	1	1	BYPASS

8.1.3 Boundary Scan Register Path

The MC92501 has a 104-bit Boundary Scan Register. This register contains the bits for all the signal I/O pins and the JTAG logic ENSCAN control pins. There are 2 bits associated with each bidirectional pin and a single bit for each dedicated input and output pin. This is due to the nature of the design of the I/O macros used. **Table 8-3** defines the signal name and the scan bit(s) associated with that signal. The first column of the table lists the functional signal pin name. The second column lists the chip I/O cell type. The third and fourth columns define the scan bit number(s) in the boundary scan register. The first bit to be shifted out is bit 0. The last column lists the ENSCAN macro that controls the output enable for that particular signal pin. Pins that are Input only or Output only do not require an output enable, and the table reflects this with a blank entry in the Output Enable column.

There are two flip-flops in the scan chain for each bidirectional pin. The leftmost of the two bits in the "Scan Bit" column of the table is associated with the Input portion of the bidirectional. This bit, representing a flip-flop, is used to capture I/O data. It also is the storage register for data used to update (drive) input core data. The second bit in the "Scan Bit" column represents the flip-flop associated with the output portion of a bidirectional. This flip-flop does not capture any meaningful data during a Capture state, but does store shifted in data. The data it stores is used to update (drive) the output buffer of the bidirectional pin.

There are three ENSCAN boundary scan bits that are associated with the bidirectional and three-state pins. They control the drive state of these pins when either the EXTEST instruction or the CLAMP instruction is active. If the ENSCAN boundary scan bit is set, the controlled pins are driven (i.e., they are in an active or output state). If the ENSCAN boundary scan bit is reset, the controlled pins are not driven (i.e., they are in an inactive or input state).

Table 8-3. Boundary Scan Bit Definition

Signal Name	I/O Cell Type	System Mode	Scan Bit #		Output Enable
STXCLK	In	In	360		
STXCLAV	Bidirectional	Out	359	358	
STXSOC	Bidirectional	In	357	356	
STXPRTY	Bidirectional	In	355	354	
STXDATA7	Bidirectional	In	353	352	
STXDATA6	Bidirectional	In	351	350	
STXDATA5	Bidirectional	In	349	348	
STXDATA4	Bidirectional	In	347	346	
STXDATA3	Bidirectional	In	345	344	

Table 8-3. Boundary Scan Bit Definition (Continued)

Signal Name	I/O Cell Type	System Mode	Scan Bit #		Output Enable
STXDATA2	Bidirectional	In	343	342	
STXDATA1	Bidirectional	In	341	340	
STXDATA0	Bidirectional	In	339	338	
STXENB	Bidirectional	In	337	336	
TXENB	Bidirectional	Out	335	334	
TXFULL	In	In	333		
TXCCLR	Bidirectional	In	332	331	
TXPHYIDV	Three-state	Out	330		
TXPRTY	Bidirectional	Out	329	328	
TXSOC	Bidirectional	Out	327	326	
TXDATA7	Bidirectional	Out	325	324	
TXDATA6	Bidirectional	Out	323	322	
TXDATA5	Bidirectional	Out	321	320	
TXDATA4	Bidirectional	Out	319	318	
TXDATA3	Bidirectional	Out	317	316	
TXDATA2	Bidirectional	Out	315	314	
TXDATA1	Bidirectional	Out	313	312	
TXDATA0	Bidirectional	Out	311	310	
TXPHYID3	Three-state	Out	309		
TXPHYID2	Three-state	Out	308		
TXPHYID1	Three-state	Out	307		
TXPHYID0	Three-state	Out	306		
MDATA31	Bidirectional	Bidirectional	305	304	ENSCAN1
MDATA30	Bidirectional	Bidirectional	303	302	ENSCAN1
MDATA29	Bidirectional	Bidirectional	301	300	ENSCAN1
MDATA28	Bidirectional	Bidirectional	299	298	ENSCAN1
MDATA27	Bidirectional	Bidirectional	297	296	ENSCAN1

Table 8-3. Boundary Scan Bit Definition (Continued)

Signal Name	I/O Cell Type	System Mode	Scan Bit #		Output Enable
MDATA26	Bidirectional	Bidirectional	295	294	ENSCAN1
MDATA25	Bidirectional	Bidirectional	293	292	ENSCAN1
MDATA24	Bidirectional	Bidirectional	291	290	ENSCAN1
MDATA23	Bidirectional	Bidirectional	289	288	ENSCAN1
MDATA22	Bidirectional	Bidirectional	287	286	ENSCAN1
MDATA21	Bidirectional	Bidirectional	285	284	ENSCAN1
MDATA20	Bidirectional	Bidirectional	283	282	ENSCAN1
MDATA19	Bidirectional	Bidirectional	281	280	ENSCAN1
MDATA18	Bidirectional	Bidirectional	279	278	ENSCAN1
MDATA17	Bidirectional	Bidirectional	277	276	ENSCAN1
MDATA16	Bidirectional	Bidirectional	275	274	ENSCAN1
MDATA15	Bidirectional	Bidirectional	273	272	ENSCAN1
MDATA14	Bidirectional	Bidirectional	271	270	ENSCAN1
MDATA13	Bidirectional	Bidirectional	269	268	ENSCAN1
MDATA12	Bidirectional	Bidirectional	267	266	ENSCAN1
MDATA11	Bidirectional	Bidirectional	265	264	ENSCAN1
MDATA10	Bidirectional	Bidirectional	263	262	ENSCAN1
MDATA9	Bidirectional	Bidirectional	261	260	ENSCAN1
MDATA8	Bidirectional	Bidirectional	259	258	ENSCAN1
MDATA7	Bidirectional	Bidirectional	257	256	ENSCAN1
MDATA6	Bidirectional	Bidirectional	255	254	ENSCAN1
MDATA5	Bidirectional	Bidirectional	253	252	ENSCAN1
MDATA4	Bidirectional	Bidirectional	251	250	ENSCAN1
MDATA3	Bidirectional	Bidirectional	249	248	ENSCAN1
MDATA2	Bidirectional	Bidirectional	247	246	ENSCAN1
MDATA1	Bidirectional	Bidirectional	245	244	ENSCAN1
MDATA0	Bidirectional	Bidirectional	243	242	ENSCAN1

Table 8-3. Boundary Scan Bit Definition (Continued)

Signal Name	I/O Cell Type	System Mode	Scan Bit #		Output Enable
MADD25	Bidirectional	In	241	240	
MADD24	Bidirectional	In	239	238	
MADD23	Bidirectional	In	237	236	
MADD22	Bidirectional	In	235	234	
MADD21	Bidirectional	In	233	232	
MADD20	Bidirectional	In	231	230	
MADD19	Bidirectional	In	229	228	
MADD18	Bidirectional	In	227	226	
MADD17	Bidirectional	In	225	224	
MADD16	Bidirectional	In	223	222	
MADD15	Bidirectional	In	221	220	
MADD14	Bidirectional	In	219	218	
MADD13	Bidirectional	In	217	216	
MADD12	Bidirectional	In	215	214	
MADD11	Bidirectional	In	213	212	
MADD10	Bidirectional	In	211	210	
MADD9	Bidirectional	In	209	208	
MADD8	Bidirectional	In	207	206	
MADD7	Bidirectional	In	205	204	
MADD6	Bidirectional	In	203	202	
MADD5	Bidirectional	In	201	200	
MADD4	Bidirectional	In	199	198	
MADD3	Bidirectional	In	197	196	
MADD2	Bidirectional	In	195	194	
MSEL	In	In	193		
MREQ0	Three-state	Out	192		
MREQ1	Three-state	Out	191		

Table 8-3. Boundary Scan Bit Definition (Continued)

Signal Name	I/O Cell Type	System Mode	Scan Bit #		Output Enable
MDTACK0	Three-state	Three-state	190		ENSCAN2
MINT	Three-state	Out	189		
MREQ2	Three-state	Out	188		
MCLK	In	In	187		
MWR	In	In	186		
MWSH	In	In	185		
MWSL	In	In	184		
MDS	In	In	183		
SRXENB	In	In	182		
SRXDATA7	Bidirectional	Three-state	181	180	ENSCAN4
SRXDATA6	Bidirectional	Three-state	179	178	ENSCAN4
SRXDATA5	Bidirectional	Three-state	177	176	ENSCAN4
SRXDATA4	Bidirectional	Three-state	175	174	ENSCAN4
SRXDATA3	Bidirectional	Three-state	173	172	ENSCAN4
SRXDATA2	Bidirectional	Three-state	171	170	ENSCAN4
SRXDATA1	Bidirectional	Three-state	169	168	ENSCAN4
SRXDATA0	Bidirectional	Three-state	167	166	ENSCAN4
SRXCLK	In	In	165		ENSCAN4
SRXCLAV	Bidirectional	Out	164	163	
SRXSOC	Bidirectional	Three-state	162	161	ENSCAN4
SRXPRTY	Bidirectional	Three-state	160	159	ENSCAN4
MDTACK1	Three-state	Three-state	158		ENSCAN6
RXPHYID4	Three-state	Three-state	157		ENSCAN3
RXSOC	Bidirectional	In	156	155	
RXENB	Bidirectional	Out	154	153	
RXEMPTY	Bidirectional	In	152	151	
RXPHYID3	Bidirectional	Bidirectional	150	149	ENSCAN3

Table 8-3. Boundary Scan Bit Definition (Continued)

Signal Name	I/O Cell Type	System Mode	Scan Bit #		Output Enable
RXPHYID2	Bidirectional	Bidirectional	148	147	ENSCAN3
RXPHYID1	Bidirectional	Bidirectional	146	145	ENSCAN3
RXPHYID0	Bidirectional	Bidirectional	144	143	ENSCAN3
RXPRTY	Bidirectional	In	142	141	
RXDATA7	Bidirectional	In	140	139	
RXDATA6	Bidirectional	In	138	137	
RXDATA5	Bidirectional	In	136	135	
RXDATA4	Bidirectional	In	134	133	
RXDATA3	Bidirectional	In	132	131	
RXDATA2	Bidirectional	In	130	129	
RXDATA1	Bidirectional	In	128	127	
RXDATA0	Bidirectional	In	126	125	
EMDATA31	Bidirectional	Bidirectional	124	123	ENSCAN5
EMDATA30	Bidirectional	Bidirectional	122	121	ENSCAN5
EMDATA29	Bidirectional	Bidirectional	120	119	ENSCAN5
EMDATA28	Bidirectional	Bidirectional	118	117	ENSCAN5
EMDATA27	Bidirectional	Bidirectional	116	115	ENSCAN5
EMDATA26	Bidirectional	Bidirectional	114	113	ENSCAN5
EMDATA25	Bidirectional	Bidirectional	112	111	ENSCAN5
EMDATA24	Bidirectional	Bidirectional	110	109	ENSCAN5
EMDATA23	Bidirectional	Bidirectional	108	107	ENSCAN5
EMDATA22	Bidirectional	Bidirectional	106	105	ENSCAN5
EMDATA21	Bidirectional	Bidirectional	104	103	ENSCAN5
EMDATA20	Bidirectional	Bidirectional	102	101	ENSCAN5
EMDATA19	Bidirectional	Bidirectional	100	99	ENSCAN5
EACEN	Three-state	Out	98		
EMWR	Three-state	Out	97		

Table 8-3. Boundary Scan Bit Definition (Continued)

Signal Name	I/O Cell Type	System Mode	Scan Bit #		Output Enable
EMDATA18	Bidirectional	Bidirectional	96	95	ENSCAN5
EMDATA17	Bidirectional	Bidirectional	94	93	ENSCAN5
EMDATA16	Bidirectional	Bidirectional	92	91	ENSCAN5
EMDATA15	Bidirectional	Bidirectional	90	89	ENSCAN5
EMDATA14	Bidirectional	Bidirectional	88	87	ENSCAN5
EMDATA13	Bidirectional	Bidirectional	86	85	ENSCAN5
EMDATA12	Bidirectional	Bidirectional	84	83	ENSCAN5
EMDATA11	Bidirectional	Bidirectional	82	81	ENSCAN5
EMDATA10	Bidirectional	Bidirectional	80	79	ENSCAN5
EMDATA9	Bidirectional	Bidirectional	78	77	ENSCAN5
EMDATA8	Bidirectional	Bidirectional	76	75	ENSCAN5
EMDATA7	Bidirectional	Bidirectional	74	73	ENSCAN5
EMDATA6	Bidirectional	Bidirectional	72	71	ENSCAN5
EMDATA5	Bidirectional	Bidirectional	70	69	ENSCAN5
EMDATA4	Bidirectional	Bidirectional	68	67	ENSCAN5
EMDATA3	Bidirectional	Bidirectional	66	65	ENSCAN5
EMDATA2	Bidirectional	Bidirectional	64	63	ENSCAN5
EMDATA1	Bidirectional	Bidirectional	62	61	ENSCAN5
EMDATA0	Bidirectional	Bidirectional	60	59	ENSCAN5
EMADD23	Bidirectional	Out	58	57	
EMADD22	Bidirectional	Out	56	55	
EMADD21	Bidirectional	Out	54	53	
EMADD20	Bidirectional	Out	52	51	
EMADD19	Bidirectional	Out	50	49	
EMADD18	Bidirectional	Out	48	47	
EMADD17	Bidirectional	Out	46	45	
EMADD16	Bidirectional	Out	44	43	

Table 8-3. Boundary Scan Bit Definition (Continued)

Signal Name	I/O Cell Type	System Mode	Scan Bit #		Output Enable
EMADD15	Bidirectional	Out	42	41	
EMADD14	Bidirectional	Out	40	39	
EMADD13	Bidirectional	Out	38	37	
EMADD12	Bidirectional	Out	36	35	
EMADD11	Bidirectional	Out	34	33	
EMADD10	Bidirectional	Out	32	31	
EMADD9	Bidirectional	Out	30	29	
EMADD8	Bidirectional	Out	28	27	
EMADD7	Bidirectional	Out	26	25	
EMADD6	Bidirectional	Out	24	23	
EMADD5	Bidirectional	Out	22	21	
EMADD4	Bidirectional	Out	20	19	
EMADD3	Bidirectional	Out	18	17	
EMADD2	Bidirectional	Out	16	15	
EMBSH0	Three-state	Out	14		
EMBSH1	Three-state	Out	13		
EMBSH2	Three-state	Out	12		
EMBSH3	Three-state	Out	11		
EMBSL0	Three-state	Out	10		
EMBSL1	Three-state	Out	9		
EMBSL2	Three-state	Out	8		
EMBSL3	Three-state	Out	7		
ARST	In	In	6		
ENSCAN1	(Core Macro)		5		
ENSCAN2	(Core Macro)		4		
ENSCAN3	(Core Macro)		3		
ENSCAN4	(Core Macro)		2		

Table 8-3. Boundary Scan Bit Definition (Continued)

Signal Name	I/O Cell Type	System Mode	Scan Bit #		Output Enable
ENSCAN5	(Core Macro)		1		
ENSCAN6	(Core Macro)		0		

9

MC92501 Product Specifications

9.1 Introduction

This section includes information about the MC92501 product specifications, including:

- Signal Names
- Physical and Electrical Characteristics
- Ordering information
- Packaging specifications

The following sections provide detailed descriptions of the product specifications.

9.2 Signal Description

This section contains brief descriptions of the input and output signals in their functional groups, as shown in **Figure 9-1**. Each signal is explained briefly in a paragraph with references to other sections that contain more details about the signal and the related operations.

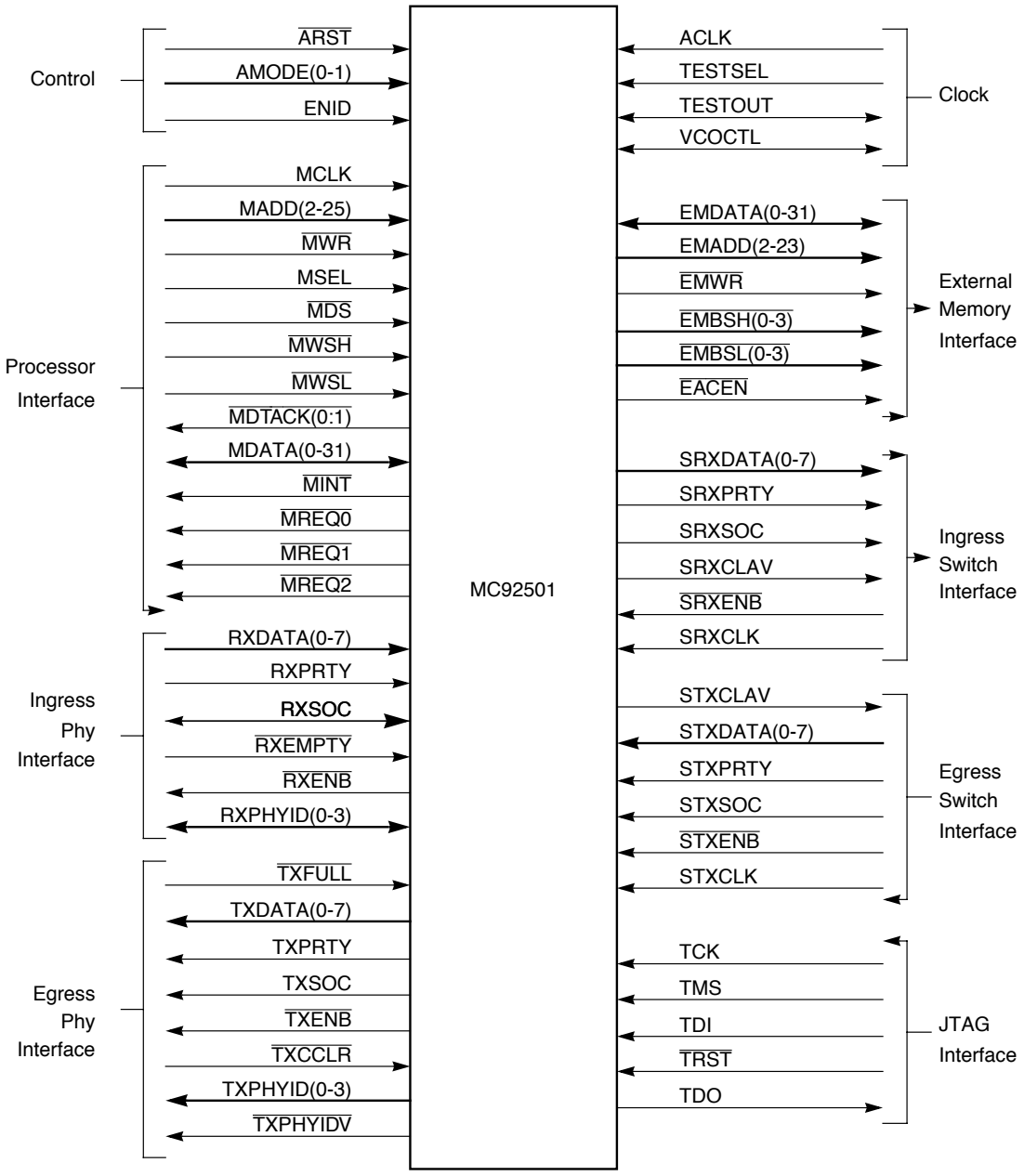


Figure 9-1. Functional Signal Groups

9.2.1 Ingress PHY Signals

The following signals relate to the PHY interface that is connected to the PHY chip(s) using the UTOPIA standard. Refer to **Section 4.2.1** UTOPIA Level 1 Receive PHY Interface (Ingress) for more information. All of the input signals are sampled at the rising edge of ACLK, and all of the output signals are updated at the rising edge of ACLK.

- **Receive Data Bus (RXDATA0–RXDATA7)**—This input data bus receives octets from the PHY chip. When RXENB is active, RXDATA is sampled into the MC92501.
- **Receive Data Bus Parity (RXPRTY)**—This input is the odd parity over RXDATA. This input is ignored if RXENB was not active or the parity check is disabled (see **Section 7.2.6.3** Ingress PHY Configuration Register (IPHCR)).
- **Receive Start Of Cell (RXSOC)**—This input, when high, indicates that the current RXDATA is the first byte of a cell. This input is sampled when RXENB is active.
- **Receive PHY Empty (RXEMPTY)**—This input, when low, indicates that currently the PHY chip has no available data.
- **Receive Enable (RXENB)**—This output, when low, indicates that the MC92501 is ready to receive data.
- **Receive PHY ID 0–3/ Receive Address 0–3 (RXPHYID0–RXPHYID3/RXADD0–RXADDR3)**—This bus has 2 modes depending on the *Ingress UTOPIA Mode (IUM)* bit of the Ingress PHY Configuration Register (IPHCR):
  - In UTOPIA level 1, it is **RXPHYID0 - RXPHYID3**—an input bus that indicates the ID number of the PHY device currently transferring data to the MC92501. If only a single PHY device is supported, this bus should be tied low. This bus is sampled along with the first octet of each cell.
  - In UTOPIA level 2, it is **RXADD0-RXADDR3**—an output bus that indicates the 4 least significant bits of the ID number of the PHY device which is being polled or selected by the MC92501. See **Section 4.2.3** UTOPIA Level 2 Receive PHY Interface (Ingress) for details.
- **Receive Address 4 (RXADD4)**—This signal is an output signal that indicates the most significant bit of the ID number of the PHY device which is being polled or selected by the MC92501. See **Section 4.2.3** UTOPIA Level 2 Receive PHY Interface (Ingress) for details.

9.2.2 Egress PHY Signals

The following signals relate to the PHY interface that is connected to the PHY chip(s) using the UTOPIA standard. Refer to **Section 4.2.4** UTOPIA Level 2 Transmit PHY Interface (Egress) for more information. All of the input signals are sampled at the rising edge of ACLK, and all of the output signals are updated at the rising edge of ACLK.

- **Transmit Data Bus (TXDATA0–TXDATA7)**—This output data bus transmits octets to the PHY chip. When TXENB is active, TXDATA contains a valid octet for the PHY.
- **Transmit Data Bus Parity (TXPRTY)**—This output signal is the odd parity over TXDATA. When TXENB is active, TXPRTY is a valid parity bit for the PHY.
- **Transmit Enable (TXENB)**—This output signal, when low, indicates that TXDATA, TXPRTY, and TXSOC are valid data for the PHY.
- **Transmit Start Of Cell (TXSOC)**—This output signal indicates, when high, that the current data on TXDATA is the first byte of a cell. TXSOC is valid when TXENB is asserted.
- **Transmit PHY Full (TXFULL)**—This input signal indicates, when low, that the PHY device is full.
- **Transmit Cell Clear (TXCCLR)**—This input signal indicates, when low, that the current cell should be cleared from the Egress PHY interface.
- **Transmit PHY ID 0–3 / Transmit Address 0–3 (TXPHYID0–TXPHYID3/ TXADDR0–TXADDR3)**—This bus has two modes depending on the *Egress UTOPIA Mode (EUM)* bit:
  - In UTOPIA level 1, it is **TXPHYID0–TXPHYID3**—an output bus that indicates the ID number of the PHY device to which either the current cell or the next cell is directed. The functionality is controlled by the MC92501 General Configuration Register (GCR).
  - In UTOPIA level 2, it is **TXADDR0-TXADDR3**—an output bus that indicates the 4 less significant bits of the ID number of the PHY device which is being polled or selected by the MC92501. See **Section 4.2.4** UTOPIA Level 2 Transmit PHY Interface (Egress) for details.
- **Transmit Next PHY ID Valid/Transmit Address 4 (TXPHYIDV/TXADDR4)**—This bit has 2 modes depending on the *Egress UTOPIA Mode (EUM)* bit:
  - In UTOPIA level 1, it is **TXPHYIDV**—an output signal, when low, indicates that TXPHYID (when configured as the next cell's ID) is valid. If TXPHYID is configured to refer to the current cell, TXPHYIDV is not used.
  - In UTOPIA level 2, it is **TXADDR4**—an output signal that indicates the most significant bit of the ID number of the PHY device which is being polled or selected by the MC92501. See **Section 4.2.4** UTOPIA Level 2 Transmit PHY Interface (Egress) section for details.

9.2.3 Ingress Switch Interface Signals

The following signals relate to the Ingress switch interface. Refer to **Section 4.3.1** Receive Interface (Ingress) for more information. All of the input signals are sampled at the rising edge of SRXCLK, and all of the output signals are updated at the rising edge of SRXCLK.

- **Receive Clock (SRXCLK)**—This input is used to clock the Ingress switch interface signals.
- **Receive DATA BUS (SRXDATA0-SRXDATA7)**—This three-state output data bus transmits bytes to the switch. When SRXENB is active, SRXDATA contains valid data for the switch. This bus is updated on the rising edge of SRXCLK.
- **Receive Data Bus Parity (SRXPRTY)**—This three-state output is the parity protection of SRXDATA transmitted to the switch. The type of parity (even/odd) is defined in **Section 7.2.6.5** Ingress Switch Interface Configuration Register (ISWCR).
- **Receive Start Of Cell (SRXSOC)**—This three-state output, when high, indicates that the current data on SRXDATA is the first byte of a cell structure (including the overhead bytes).
- **Receive Switch Cell Available (SRXCLAV)**—This output, when asserted, indicates that the MC92501 has a cell ready to transfer to the switch. When deasserted, it indicates that currently there is no data available for the switch.
- **Receive Enable (SRXENB)**—This input, when low, enables new values on SRXDATA, SRXPRTY and SRXSOC.

9.2.4 Egress Switch Interface Signals

The following signals relate to the Egress switch interface. Refer to **Section 4.3.2** Transmit Interface (Egress) for more information. All of the input signals are sampled at the rising edge of STXCLK, and all of the output signals are updated at the rising edge of STXCLK.

- **Transmit Clock (STXCLK)**—This input signal is used to clock the Egress switch interface signals.
- **Transmit Data Bus (STXDATA0–STXDATA7)**—This input data bus receives bytes from the switch. When STXENB is asserted, STXDATA is sampled into the MC92501 on the rising edge of STXCLK.
- **Transmit Data Bus Parity(STXPRTY)**—This input is the parity over STXDATA. The type of parity (even/odd) and the parity check control are defined in **Section 7.2.6.6** Egress Switch Interface Configuration Register (ESWCR). This input is ignored if STXENB is deasserted or the parity check is disabled. It is sampled on the rising edge of STXCLK.
- **Transmit Start Of Cell (STXSOC)**—This input indicates, when high, that the current data is the first byte of a cell structure (including the overhead bytes). This input is sampled on the rising edge of STXCLK when STXENB is asserted.
- **Transmit Enable (STXENB)**—This input, when low, enables STXDATA, STXPRTY, and STXSOC.
- **Transmit Cell Available (STXCLAV)**—This output, when asserted, indicates that the MC92501 is prepared to receive a complete cell.

9.2.5 External Memory Signals

The following signals relate to thE External Memory Interface. Refer to **Section 4.4** External Memory Interface for more information.

- **External Memory Data Bus (EMDATA0–EMDATA31)** — This tri-statable bidirectional bus is the data path between the MC92501 and External Memory.
- **External Memory Address Bus (EMADD2–EMADD23)** — This output bus is the general address bus used by the MC92501 to access the External Memory.
- **External Memory Write (EMWR)** — When asserted (low), this output signal indicates that the current cycle to the External Memory is a write cycle. This signal is active low and is asserted within the cycle.
- **External Memory Bank Select High (EMBSH0–EMBSH3)** — These output signals are used to select the high word of the appropriate memory bank. One or more of these signals is asserted for each External Memory access according to the value of EMADD. See **Section 4.4.1** EM Bank Select Signals for more information. During a maintenance write access from the microprocessor, the value detected on MWSH is driven on the appropriate EMBSH signal. These signals are active low.
- **External Memory Bank Select Low (EMBSL0–EMBSL3)** — These output signals are used to select the low word of the appropriate memory bank. One or more of these signals is asserted for each External Memory access according to the value of EMADD. See **Section 4.4.1** EM Bank Select Signals for more information. During a maintenance write access from the microprocessor, the value detected on MWSL is driven on the appropriate EMBSL signal. These signals are active low.
- **External Address Compression Enable (EACEN)** — This output signal is asserted when data is being written to or read from an external address compression device using the External Memory Data Bus. This signal is active low.

9.2.6 Control Signals

These signals are used to control the MC92501.

- **ATMC Power-Up Reset (ARST)** — This input signal is used for power-up reset of the entire chip. It must be asserted for at least the time required by the PLL [15].
- **ATMC Mode (AMODE0–AMODE1)** — These input signals determine the operating mode of the chip's test features. In normal usage these pins should be grounded.
- **Enable IDD (ENID)** — This input pin is used for test purposes. In normal usage the ENID pin must be grounded.

9.2.7 Microprocessor Signals (MP)

The following signals relate to the microprocessor interface. Refer to **Section 4.5** Microprocessor Interface for more information.

- **MP Clock (MCLK)** — This input signal is used as the Microprocessor clock inside the MC92501. This signal drives the microprocessor logic in the MC92501. The duty cycle should be in the range of 40–60%.
- **MP Data Bus (MDATA0–MDATA31)** — This three-state bidirectional bus provides the general data path between the MC92501 and the microprocessor.
- **MP Address Bus (MADD2–MADD25)** — This input bus contains the address which is used by the microprocessor to define the register being accessed. This bus is used by the MC92501 at the assertion of MSEL and sampled on the falling edge of MCLK.
- **MP Select ($\overline{\text{MSEL}}$)** — This input signal is used to determine that the current access to the MC92501 is valid. This signal is active low and sampled by the MC92501 on the falling edge of MCLK.
- **MP Data Select ($\overline{\text{MDS}}$)** — This input signal is used to indicate when the data on MDATA is valid during a write access to the MC92501. This signal is active low and sampled by the MC92501 on the falling edge of MCLK.
- **MP Write ($\overline{\text{MWR}}$)** — This input signal is used to determine whether the MP is reading from the MC92501 or writing to it. This signal is active low and sampled by the MC92501 on the falling edge of MCLK. The MC92501 drives MDATA when MSEL = 0 and MWR = 1.
- **MP Word Write Enable High / Address 1 ($\overline{\text{MWSH}}$/A1)** — This input signal can be programmed by the *Word Select Signals Mode (WSSM)* bit as follows:
  - *Write-enable Mode*: This signal indicate that the high word is being written. During a maintenance write access, the value detected on $\overline{\text{MWSH}}$/A1 is driven on the appropriate EMBSH signal. During read access EMBSH signal is always asserted. This signal is active low.
  - *Add1-Size Mode*: This signal serves as address 1 during a maintenance write access. During a read access, this signal is ignored.

NOTE: This signal is sampled by the MC92501 on the falling edge of MCLK. See **Section 7.2.6.1** Microprocessor Configuration Register (MPCONR) for details.

- **MP Word Write Enable Low / SIZE ($\overline{\text{MWSL}}$/SIZE)** — This signal is sampled by the MC92501 on the falling edge of MCLK and can be programmed by the *Word Select Signals Mode (WSSM)* bit as follows:
  - *Write-enable Mode*: This signal indicates that the low word is being written. During a maintenance write access, the value detected on $\overline{\text{MWSL}}$/SIZE is driven on the appropriate EMBSL signal. (During read access EMBSL signal is always asserted). This signal is active low.
  - *Add1-Size Mode*: this signal indicates the size of the maintenance write access: either 32 bits or 16 bits access. During a read access, this signal is ignored and the access width is 32 bits.

.

NOTE: **Table 9-1** describes the combined $\overline{\text{MWSH}}$/A1 and $\overline{\text{MWSL}}$/Size functionality:

Table 9-1. Host Interface Fields

WSSM = 0		WSSM = 1 and DO-Data Order = 0		WSSM = 1 and DO-Data Order=1		Function
$\overline{\text{MWSH}}$	$\overline{\text{MWSL}}$	A1	Size	A1	Size	
0	0	x	0	x	0	Write D(31:0)
0	1	0	1	1	1	Write D(31:16)
1	0	1	1	0	1	Write D(15:00)

NOTE: All Cell Extraction Register, Cell Insertion Register, and General Register accesses are long-word (32-bit) accesses, so both MWSH/A1 and MWSL/SIZE should be asserted low for these write accesses when write-enable mode is selected.

- **MP Data Acknowledge0 ($\overline{\text{MDTACK0}}$)** — This tri-statable output signal is used to indicate the end of an access from the MC92501. At the end of each access, this signal is actively pulled up and then released. The user may program the MC92501 not to drive this signal during certain types of accesses. See **Section 7.2.6.1** Microprocessor Configuration Register (MPCONR) for details. This signal is active low and is output asynchronously to MCLK.
- **MP Data Acknowledge1 ($\overline{\text{MDTACK1}}$)** — This tri-statable output signal is used to indicate the end of an access from the MC92501. At the end of each access, this signal is actively pulled up and then released. The user may program the MC92501 not to drive this signal during certain types of accesses. See **Section 7.2.6.1** Microprocessor Configuration Register (MPCONR) for details. This signal is active low and is output asynchronously to MCLK.
- **MP Interrupt ($\overline{\text{MINT}}$)** — This output signal is used to notify the microprocessor of the occurrence of interrupting events. This signal is asserted on the rising edge of ACLK (asynchronous with respect to MCLK).
- **MP Request 0 (MREQ0)** — This output signal can be programmed to one of three options (described below in the note). Its default value is option #1: MP Cell In Request (MCIREQ). See **Section 7.2.6.1** Microprocessor Configuration Register (MPCONR) for details.
- **MP Request 1 (MREQ1)** — This output signal can be programmed to one of three options (described below in the note). Its default value is option #2: MP Cell Out Request (MCOREQ).See **Section 7.2.6.1** Microprocessor Configuration Register (MPCONR) for details.
- **MP Request 2 (MREQ2)** — This output signal can be programmed to one of of three options (described below in the note). Its default value is option #3: External Memory Maintenance Request (EMMREQ). See **Section 7.2.6.1** Microprocessor Configuration Register (MPCONR) for details.

NOTE: $\overline{\text{MREQ0}}$, $\overline{\text{MREQ1}}$ and $\overline{\text{MREQ2}}$ signals are fully backward compatible to the MC92501 Revision A $\overline{\text{MCIREQ}}$, $\overline{\text{MCOREQ}}$ and $\overline{\text{EMMREQ}}$ signals, respectively. The MREQ[n] signals are used by DMA devices and can be programmed to support DMA requests as follows:

1. *MP Cell In Request*: MREQ[n] is an output signal that can be used by an external DMA device as a control line indicating when to start a new cell insertion cycle into the MC92501. It is asserted whenever the Cell Insertion Register array is available to be written. Refer to **Section 3.4.1** Cell Insertion for more information. This signal is active low and is output on the falling edge of MCLK.
2. *MP Cell Out Request*: MREQ[n] is an output signal may be used by an external DMA device as a control line indicating when to start a new cell extraction cycle from the MC92501. It is asserted whenever the Cell Extraction Register array is available to be read. Refer to **Section 3.4.2** Cell Extraction for more information. It is active low and is output on the falling edge of MCLK.
3. *External Memory Maintenance Request*: MREQ[n] is an output signal is asserted a programmable number (see **Section 7.2.6.2** Maintenance Configuration Register (MACONR)) of clock cycles before the start of an External Memory maintenance cycle. It is deasserted after a programmable number of maintenance accesses have been performed. See **Section 7.2.6.2** Maintenance Configuration Register (MACONR) for details. It is active low and is output on the falling edge of MCLK.

9.2.8 PLL Signals

The following signals are connected to the analog PLL macro which must be used in the MC92501. An application notedescribing the use of the PLL is available [see Reference 15, **Appendix G**] .

- **ATMC Master Clock (ACLK)**—This input signal is used by the PLL to generate the internal master clock of MC92501. The duty cycle should be in the range of 40–60%.
- **TESTSEL**—This is a dedicated test signal that must be grounded during normal system operation.
- **TESTOUT**—This is a dedicated test signal that must be connected to the analog ground (AVSS) during normal system operation.
- **VCOCTL**—This is a dedicated test signal that must be connected to the analog ground (AVSS) during normal system operation.

9.2.9 Test Signals

The following test signals are provided by the MC92501:

- **Test Clock (TCK)** — This input pin is the JTAG clock. The TDO, TDI, and TMS pins are synchronized by this pin.
- **Test Mode Select (TMS)** — This input pin is sampled on the rising edge of TCK. TMS is responsible for the state change in the test access port state machine.
- **Test Data Input (TDI)** — This input pin is sampled on the rising edge of TCK. TDI is the data to be shifted toward the TDO output.
- **Test Data Output (TDO)** — This three-state output pin changes its logical value on the falling edge of TCK.
- **Test Reset (TRST)** — This input pin is the JTAG asynchronous reset. When asserted low, the test access port is forced to the Test_Logic_Reset state. When JTAG is not being used, this signal should be tied to ARST or hard-wired to GND.

9.3 Electrical and Physical Characteristics

This section provides the following sets of physical and electrical specifications for the MC92501:

- Absolute Maximum Ratings
- Recommended Operating Conditions
- DC Electrical Characteristics
- Clocks
- Microprocessor Interface Timing
- PHY Interface Timing
- Switch Interface Timing
- External Memory Interface Timing

9.3.1 Absolute Maximum Ratings

Table 9-2. Absolute Maximum Ratings

Symbol	Parameter	Value/Value Range[1]	Unit
V_{DD}	DC Supply Voltage	-0.5 to 3.8	V
V_{IN}[2]	DC Input Voltage (5 V Tolerant)	-0.5 to 5.8	V
V_{OUT}[2,3]	DC Output Voltage	-0.5 to V_{DD} + 0.5	V
I	DC Current Drain per Pin, Any Single Input or Output	±50	mA
I	DC Current Drain VDD and VSS Pins	±100	mA
T_{STG}	Storage Temperature	-65 to +150	°C
T_L	Lead Temperature (10 s soldering)	300	°C

Notes:
1. Maximum ratings are those values beyond which damage to the device may occur.
2. All input, bidirectional, and MDTACK are 5 V Tolerant. For proper operation it is recommended that Vin and Vout be constrained to $0 \le (V_{IN}, V_{OUT}) \le 5.5$ V.
3. SRXDATAx, SRXSOC, SRXPRTY, TDO tri-state outputs must be constrained to $0 \le V_{OUT} \le V_{DD}$ in the high impedance state.
4. This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit.

9.3.2 Recommended Operating Conditions

Table 9-3. Recommended Operating Conditions to Guarantee Functionality

Symbol	Parameter	Min	Max	Unit
V_{DD}	DC Supply Voltage, V_{DD} = 3.3V (Nominal)	3.0	3.6	V
V_{in}	Input Voltage (5V Tolerant)	0	5.5	V
T_A	Industrial Operating Temperature	–40	85	°C

Notes:
1. All parameters are characterized for DC conditions after thermal equilibrium has been established.
2. Unused inputs must always be tied to an appropriate logic voltage level (e.g., either V_{ss} or V_{DD}).
3. This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit.

9.3.3 DC Electrical Characteristics

Table 9-4. DC Electrical Characteristics

Symbol	Parameter	Condition	Min.	Max.	Unit
V_{IH}	TTL Inputs (5V Tolerant)	—	2.2	5.5	V
V_{IL}	TTL Inputs (5V Tolerant)	—	–0.3	0.8	V
I_{IN}	Input Leakage Current, No Pull Resistor	$V_{IN} = V_{DD}$ or V_{SS}	–5	5	µA
	With Pullup Resistor *		–720	–5	
	With Pulldown Resistor *		5	720	
I_{OH}	Output High Current, LVTTL Output Type Outputs: EACEN, EMWR, EMADDx, EMBSHx, EMBSLx	V_{DD} = Min, V_{OH} Min = 0.8 V_{DD}	–20	—	mA
	Output High Current, LVTTL Output Type Outputs: All other outputs		–4	—	
I_{OL}	Output Low Current, LVTTL Output Type Outputs: EACEN, EMWR, EMADDx, EMBSHx, EMBSLx	V_{DD} = Min, V_{OL} Max = 0.4 V	20	—	mA
	Output Low Current, LVTTL Output Type Outputs: All other outputs		4	—	
I_{OZ}	Output Leakage Current, Tri-State Output	Output = High Impedance $V_{OUT} = V_{DD}$ or V_{SS}	–10	10	µA
I_{DDQ}	Max Quiescent Supply Current	I_{OUT} = 0 mA $V_{IN} = V_{DD}$ or V_{SS}	10		µA
I_{DD}	Max Dynamic Supply Current	Nominal Load Capacitance, ACLK = 25.6 Mhz, MCLK = 33 Mhz	300[1]		mA
C_I	Input Capacitance (TTL)	—	—	10	pF

Notes:
1. Under Typical Loca, 25 Mhz ACLK/MCLK
2. T_A = –40˚C to 85˚C, V_{DD}=3.3 V ±0.3 V Guaranteed
3. Inputs may be modified to include pullup resistors at any time.
4. See **Section 9.2** Signal Description for pin input/output type.

9.3.4 Clocks

Table 9-5. Clock Timing

Num	Characteristics	Min	Max	Unit
C1	ACLK Cycle Time	39	80	ns
C2	ACLK Pulse Width Low	15	—	ns
C3	ACLK Pulse Width High	15	—	ns
C4	ACLK Rise/Fall Time	—	5	ns
C5	MCLK Cycle Time	30	—	ns
C6	MCLK Pulse Width Low	12	—	ns
C7	MCLK Pulse Width High	12	—	ns
C8	MCLK Rise/Fall Time	—	5	ns
C9	SRXCLK/STXCLK Cycle Time	30	—	ns
C10	SRXCLK/STXCLK Pulse Width Low	12	—	ns
C11	SRXCLK/STXCLK Pulse Width High	12	—	ns
C12	SRXCLK/STXCLK Rise/Fall Time	—	5	ns

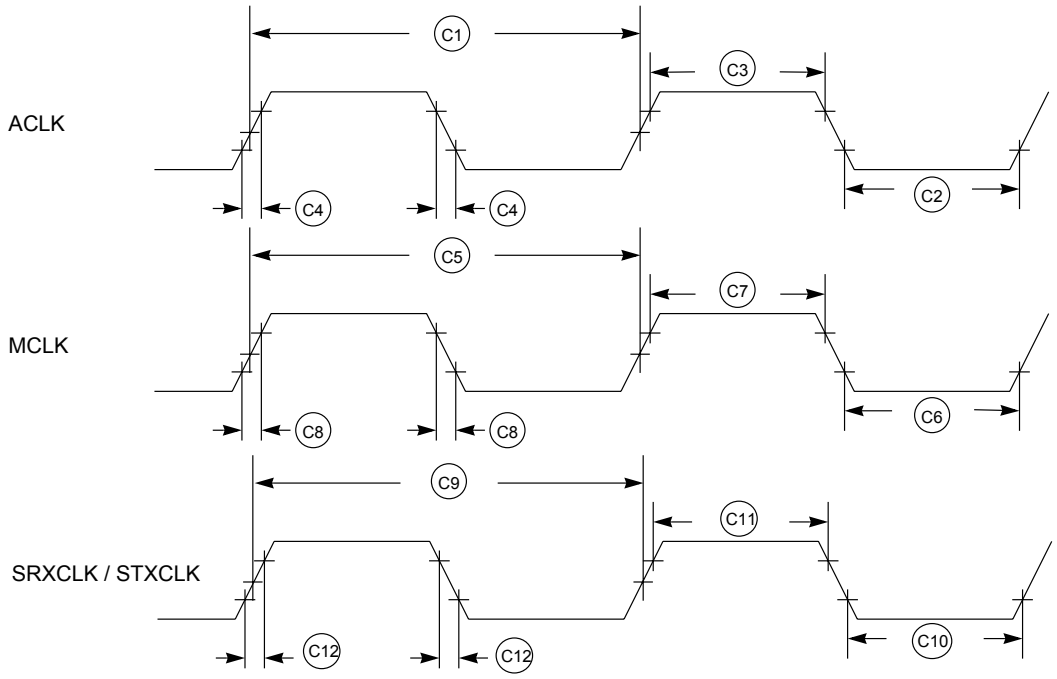


Figure 9-2. Clock Timing Diagrams

9.3.5 Microprocessor Interface Timing

The timing diagrams in this section are intended to convey setup and hold values for input signals and propagation delay values for output signals. For functional timing diagrams, see **Section 4.5** Microprocessor Interface.

Table 9-6. Microprocessor Interface Timings

Num	Characteristics	Min	Max	Unit
1	MSEL setup time before MCLK falling edge	5		ns
2	MSEL hold time after MCLK falling edge	1		ns
3	MADD/MWR setup time before MSEL assertion	5		ns
4	MADD/MWR hold time after MCLK falling edge[1]	3		ns
5	MDS setup time before MCLK falling edge	5		ns
6	MDS hold time after MCLK falling edge	1		ns
7	MDATA setup time before MCLK falling edge	4		ns
8	MDATA hold time after MCLK falling edge	1		ns
9	MSEL assertion to MDATA active	0		ns
11	MCLK falling edge to MDATA valid for CER Accesses[2]		26	ns
12	MSEL deassertion to MDATA invalid	1		ns
13	MSEL deassertion to MDATA inactive		11	ns
14	MWR assertion to MDATA invalid	1		ns
15	MWR assertion to MDATA inactive		11	ns
16	MCLK rising edge to MDATA valid for Maintenance Accesses[2,3]		T_D	ns
17	MCLK falling edge to MDATA valid for General Register Accesses[2,4]		T_R	ns
19	MSEL assertion to MDTACK active	0		ns
20	MCLK falling edge to MDTACK inactive		12	ns
21	MSEL assertion to MDTACK0 asserted[5]		9	ns
22	MSEL deassertion to MDTACK deasserted[5]		13	ns
23	MCLK rising edge to MDTACK asserted[5]		13	ns
24	MWSH, MWSL setup time before MCLK falling edge[1]	2		ns

Table 9-6. Microprocessor Interface Timings (Continued)

Num	Characteristics	Min	Max	Unit
25	$\overline{MWSH}$, $\overline{MWSL}$ hold time after MCLK falling edge[1]	3		ns
26	MCLK falling edge to $\overline{REQ}$ valid	0	14	ns
27	MCLK falling edge to $\overline{MDTACK}$ asserted for General Register Read Accesses[5,6]		T_{RD}	ns
28	MCLK falling edge to $\overline{MDTACK}$ asserted for General Register Write Accesses[5,7]		T_{WD}	ns
29	Access width (MCLK falling edge to $\overline{MSEL}$ deassertion) for General Register Write Accesses[8]	T_W		
30	$\overline{MSEL}$ assertion to $\overline{MDTACK1}$ asserted[9]		12	ns
31	$\overline{MDTACK0}$ assertion to $\overline{MDTACK1}$ assertion	0	5	ns

Notes:
1. This refers only to the first falling edge of MCLK in each access at which $\overline{MSEL}$ is asserted.
2. This is for a 150 pF load. Add 0.9 ns for each additional 10 pF. For a 100 pF load, subtract 4 ns.
3. T_D = External Memory access time + 18 ns
4. T_R = 4 * ACLK period + 20 ns
5. This is for a 50 pF load.
6. T_{RD} = 4 * ACLK period + 11 ns
7. T_{WD} is measured from the MCLK falling edge at which $\overline{MDS}$ is sampled as asserted. T_{WD} = 4 * ACLK period + 11 ns
8. T_W is measured from the MCLK falling edge at which MDS is sampled as asserted. T_W = 4 * ACLK period. Note that the setup and hold times with respect to MCLK (timing values 1 and 2) still apply.
9. This is for a 50 pF load.

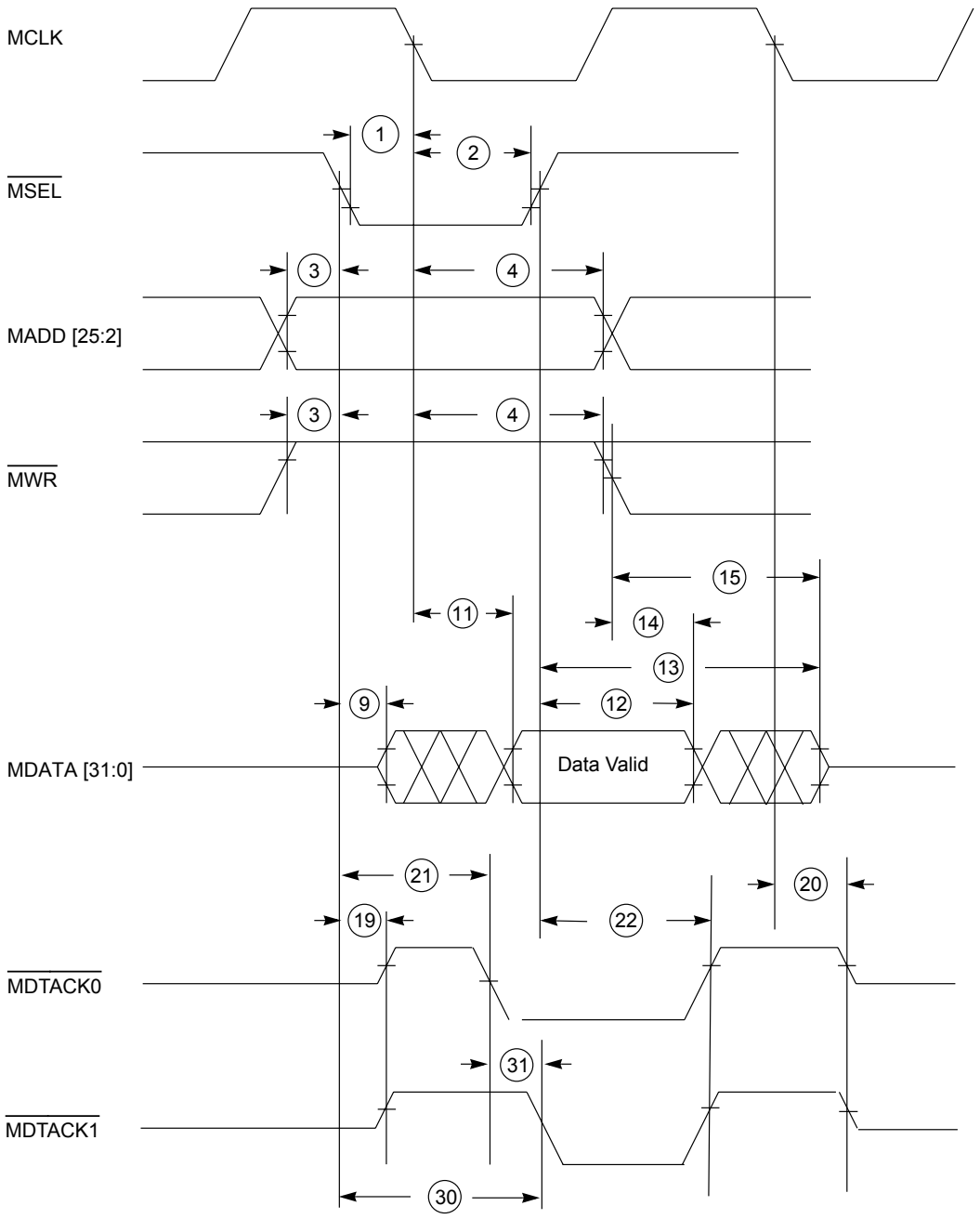


Figure 9-3. Cell Extraction Register Read Access Timing

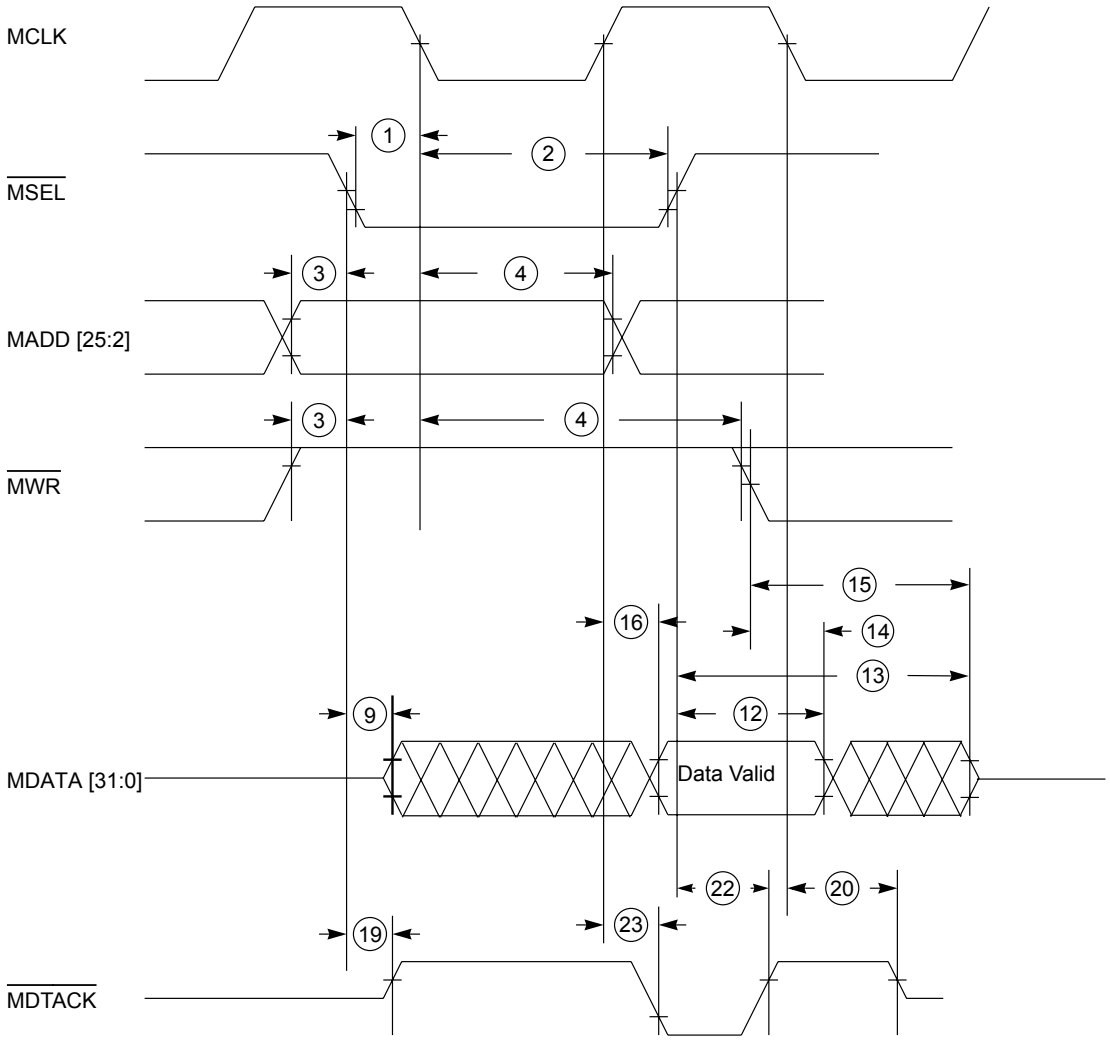


Figure 9-4. Maintenance Read Access Timing

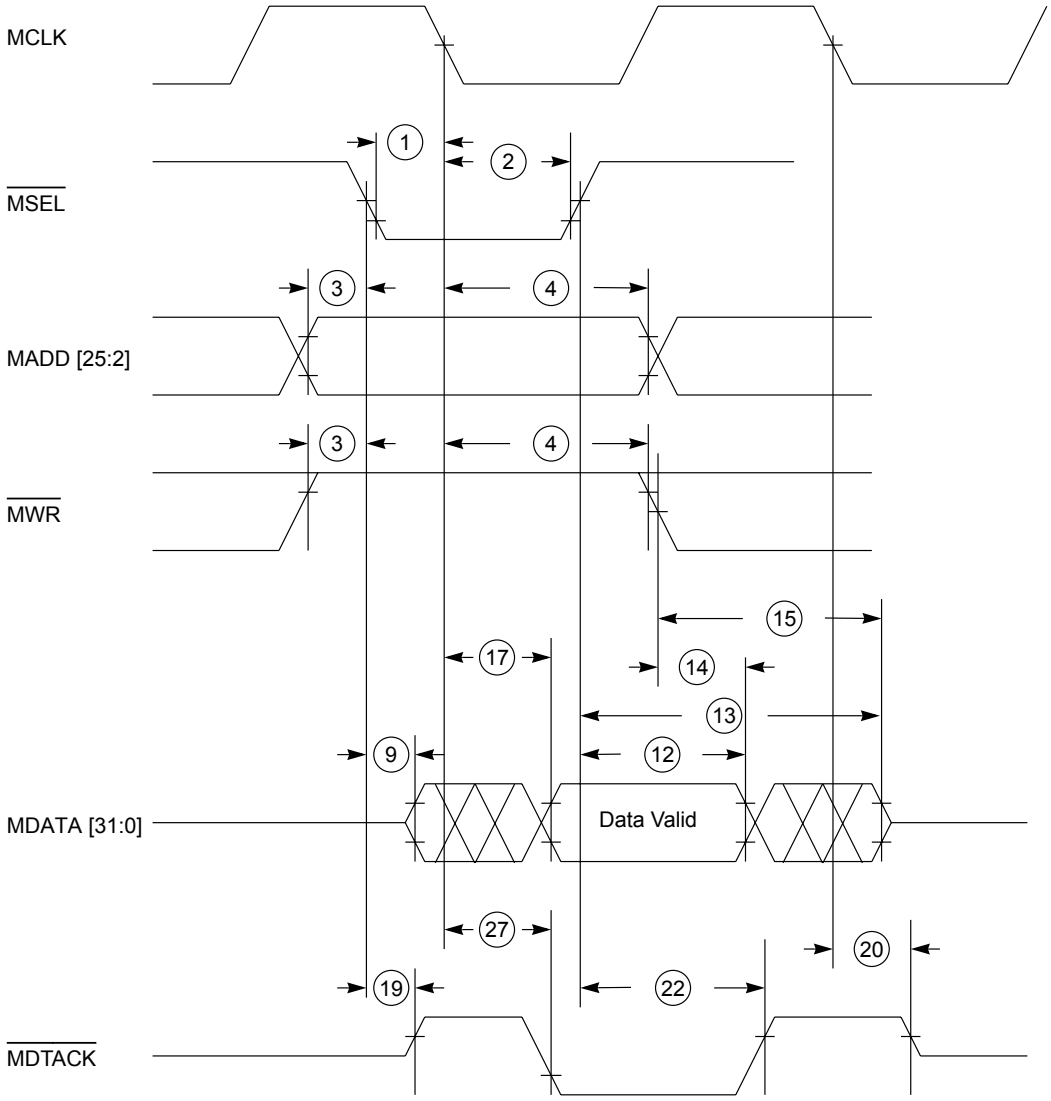


Figure 9-5. General Register Read Access Timing

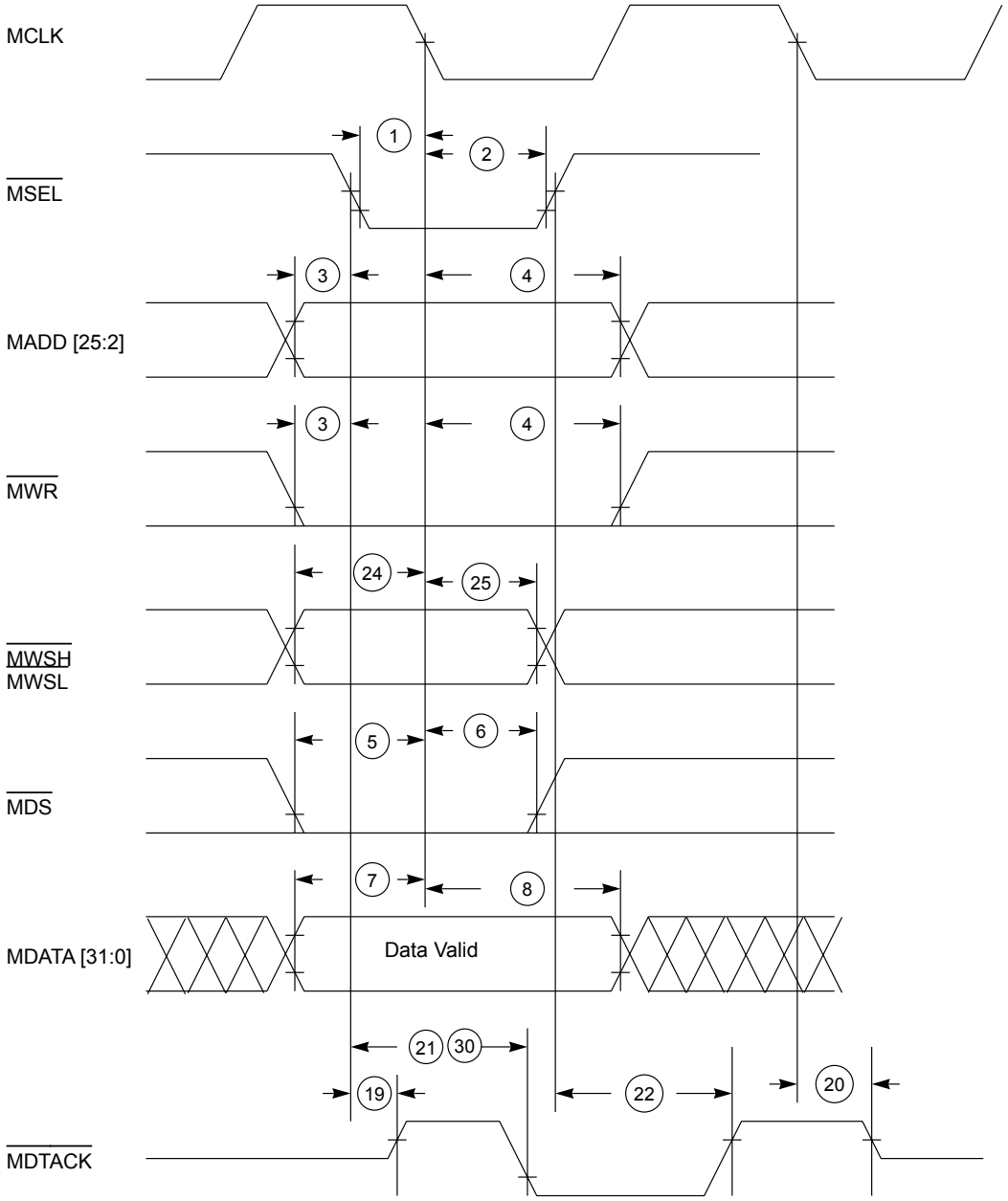


Figure 9-6. Cell Insertion Register Write Access/Maintenance Write Access Timing

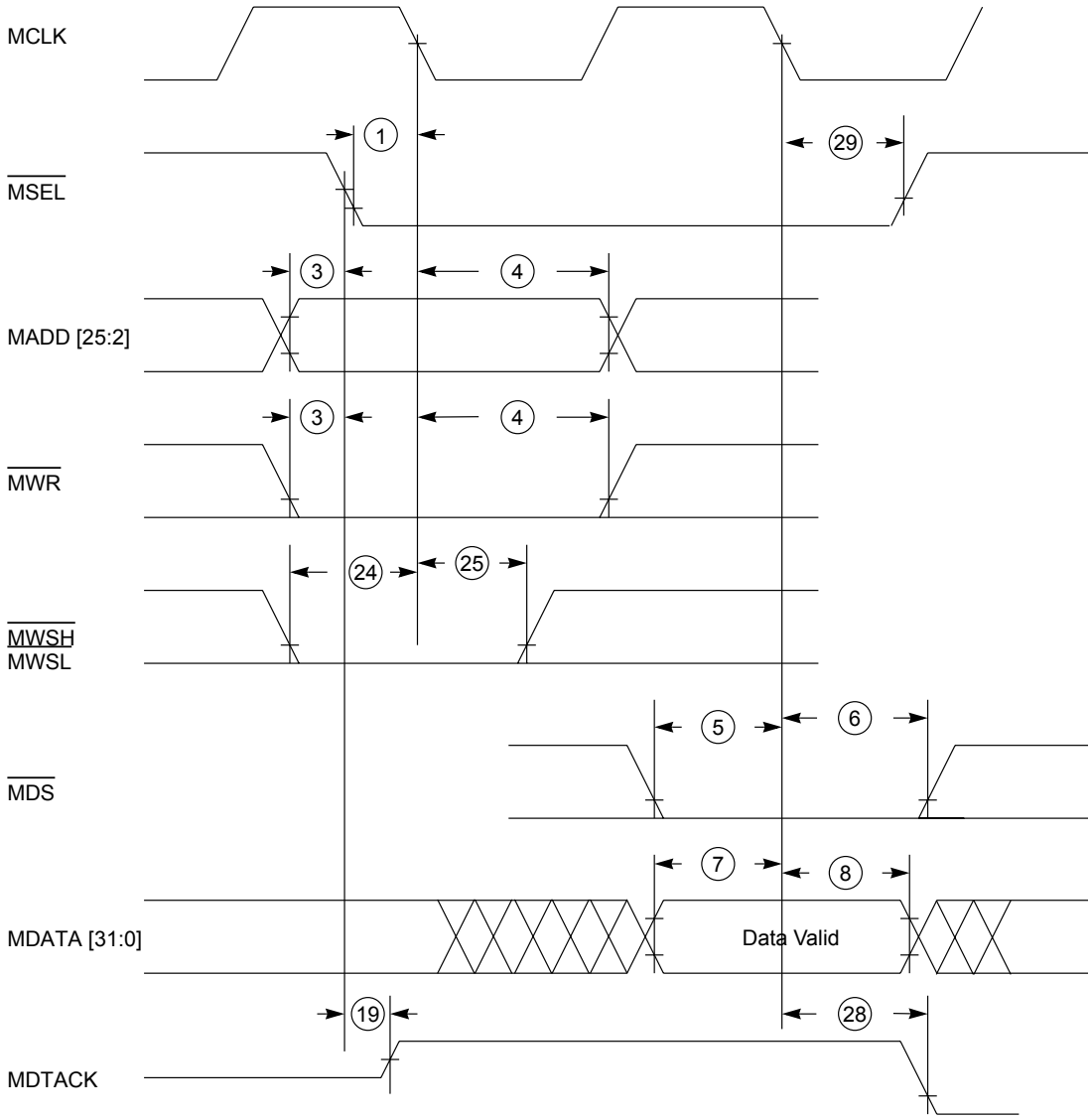


Figure 9-7. General Register Write Access Timing

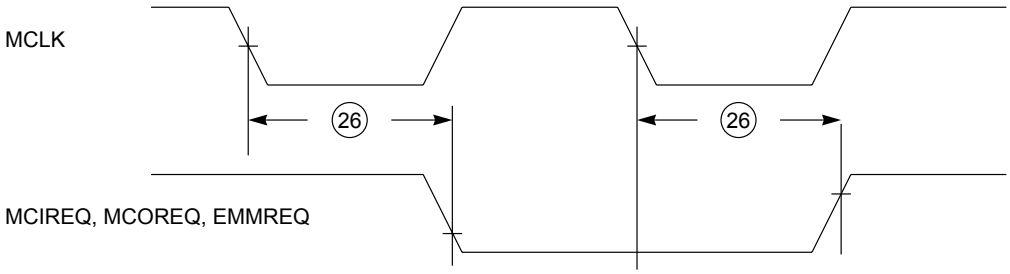


Figure 9-8. DMA Request Signals Timing

9.3.6 PHY Interface Timing

Table 9-7. PHY Interface Timings

Num	Characteristics	Min	Max	Unit
51	Setup time before ACLK rising edge	10	—	ns
52	Hold time after ACLK rising edge	1	—	ns
53	Propagation delay from rising edge of ACLK[1]	1	16[2]	ns

Notes: 1. For a 200 pF load. Add 0.25 ns for each additional 10 pF. For 100 pF subtruct 2.5 ns.
2. 16 ns for 70°C, 17 ns for 85°C

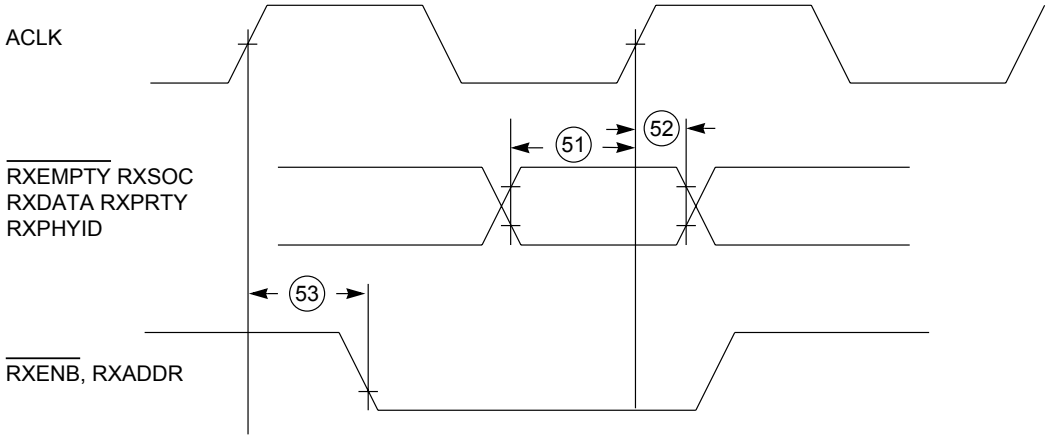


Figure 9-9. Receive PHY Interface Timing

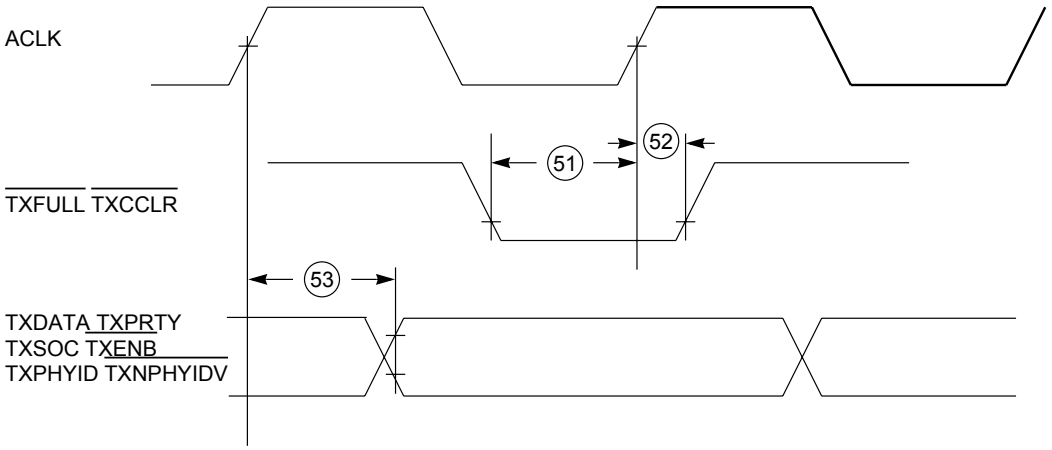


Figure 9-10. Transmit PHY Interface Timing

9.3.7 Switch Interface Timing

Table 9-8. Switch Interface Timing

Num	Characteristics	Min	Max	Unit
61	Setup time before SRXCLK/STXCLK rising edge	4		ns
62	Hold time after SRXCLK/STXCLK rising edge	1		ns
63	Propagation delay from rising edge of SRXCLK/STXCLK	1	18	ns
64	SRXCLK rising edge to outputs active	1		
65	SRXCLK rising edge to outputs inactive	1	16	

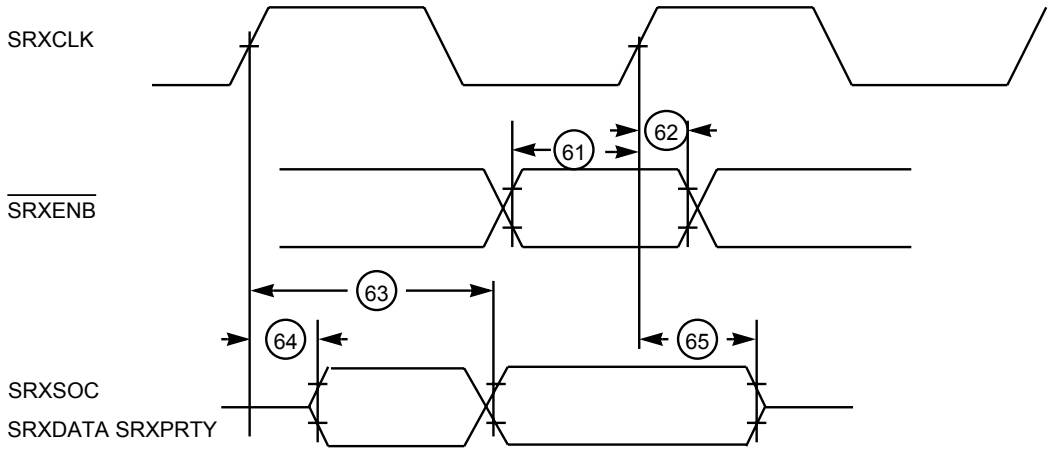


Figure 9-11. Ingress Switch Interface Timing

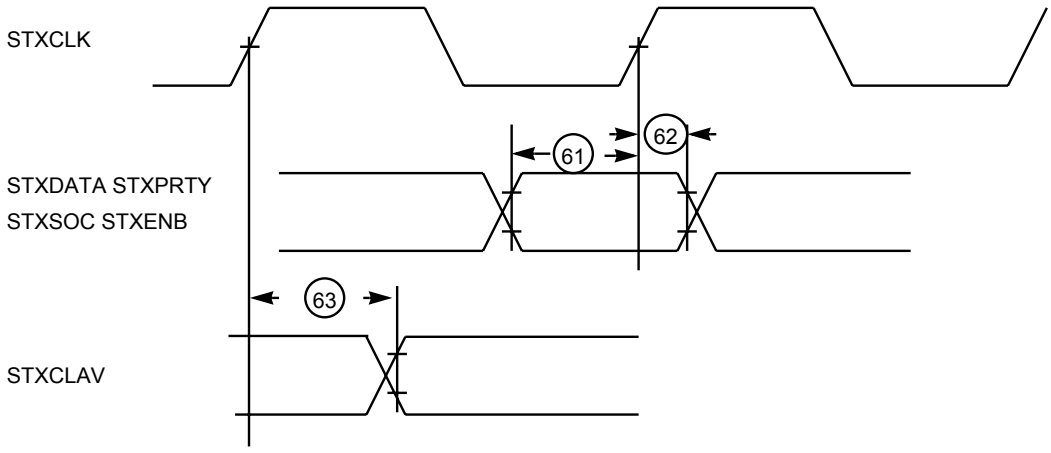


Figure 9-12. Egress Switch Interface Timing

9.3.8 External Memory Interface Timing

This section represents External Memory timing parameters for the default definition of the External Memory Timing Configuration Register (EMTCR). These values are for a load of up to 50 pF, which is the rated maximum load for the External Memory interface pins.

9.3.8.1 Write Cycle Timing

Table 9-9. Write Cycle Timing

Num	Characteristics	Min	Max	Unit
81	Write Pulse Width	16	—	ns
82	EMWR assertion time. EMWR low to end of Write.	22	—	ns
83	Address Setup Time. EMADD Valid to Beginning of Write.	6	—	ns
84	Address Valid Time. During this Time EMADD is Valid.	32	—	ns
85	Address Hold Time. End of Write to EMADD Invalid.	6	—	ns
87	Data Setup Time. EMDATA Valid to End of Write.	13	—	ns
88	Data Hold Time. End of Write to EMDATA Invalid.	6	—	ns
Note: A write occurs during the overlap of EMBSH0-3, EMBSL0-3, EACEN low and EMWR low.				

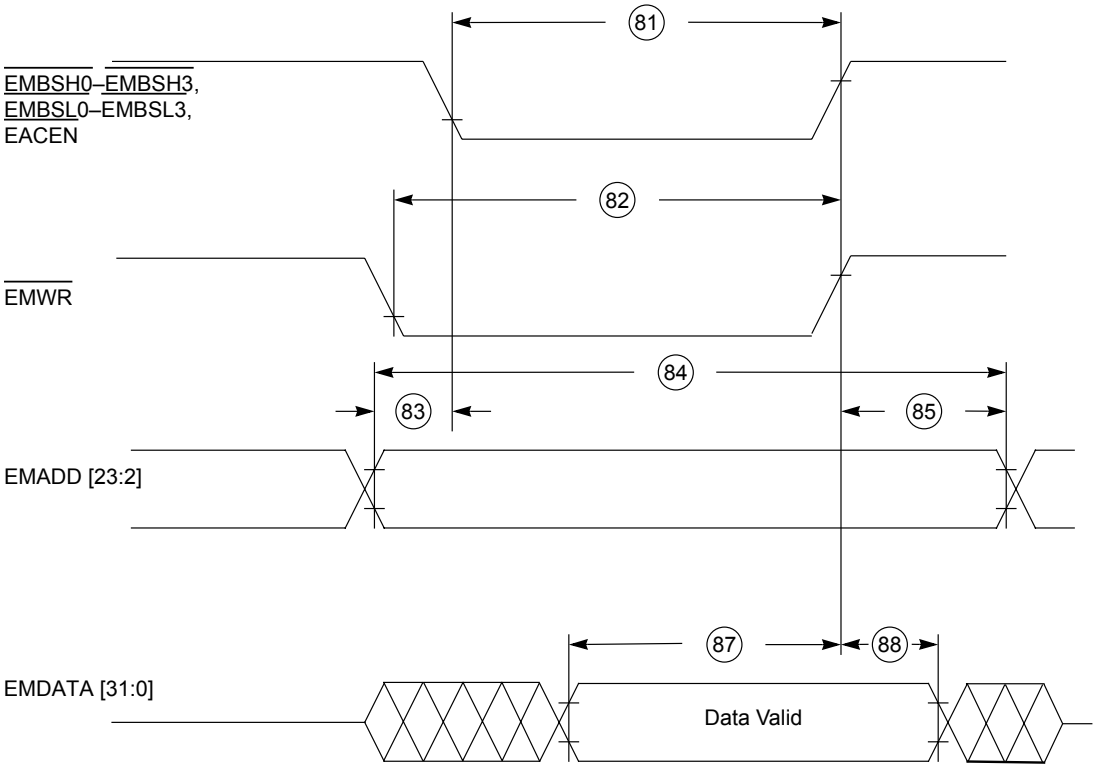


Figure 9-13. External Memory Write Access Timing

9.3.8.2 Read Cycle Timing

Table 9-10. Read Cycle Timing

Num	Characteristics	Min	Max	Unit
90	Enable Pulse Width. EMBSH0–EMBSH3, EMBSL0–EMBSL3, EACEN Pulse Width.	28	—	ns
92	Address Setup Time. EMBSH0–EMBSH3, EMBSL0–EMBSL3, EACEN High.	33	—	ns
93	Address Hold Time. EMADD Invalid to EMBSH0–EMBSH3, EMBSL0–EMBSL3, EACEN High	—	1[1]	ns
94	Data Driving Start Point. EMBSH0–EMBSH3, EMBSL0–EMBSL3, EACEN Low to EMDATA Active.	0	—	ns
95	Data Setup Time. EMDATA Valid to EMBSH0–EMBSH3, EMBSL0–EMBSL3, EACEN High.	5	—	ns
96	Data Hold Time. EMBSH0–EMBSH3, EMBSL0–EMBSL3, EACEN High to EMDATA Invalid.	0	—	ns
97	Data Driving End Point. EMBSH0–EMBSH3, EMBSL0–EMBSL3, EACEN High to EMDATA Inactive[2]	—	9	ns

Notes: 1. A RAM with hold time from address change to data change is required.
2. Failure to meet this value may result in contention on EMDATA if a write access follows.

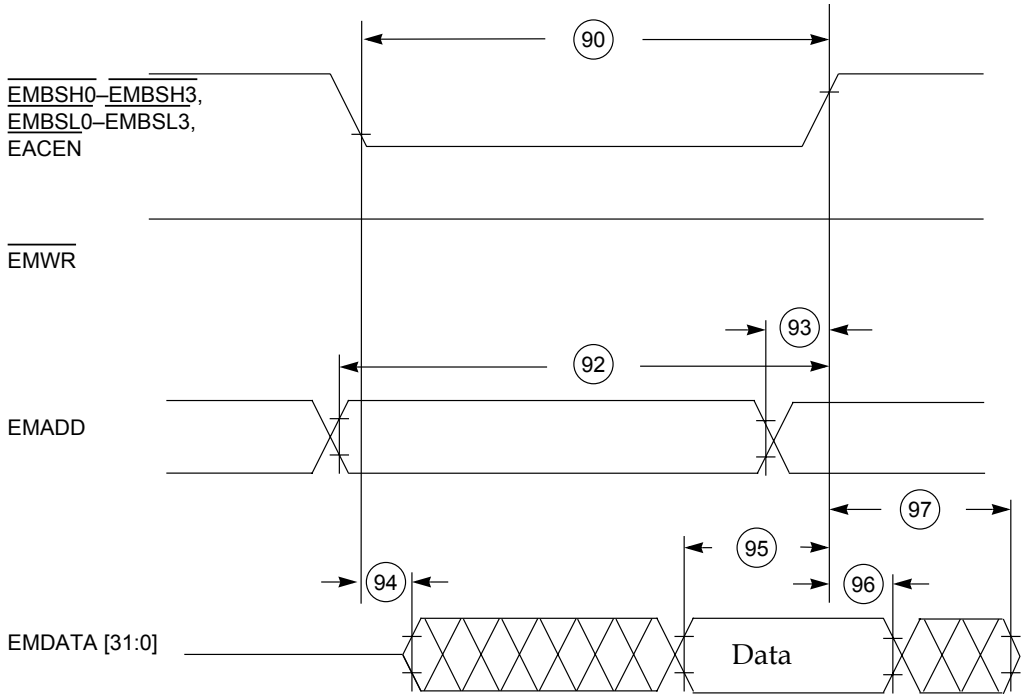


Figure 9-14. External Memory Read Access Timing

9.4 Ordering Information

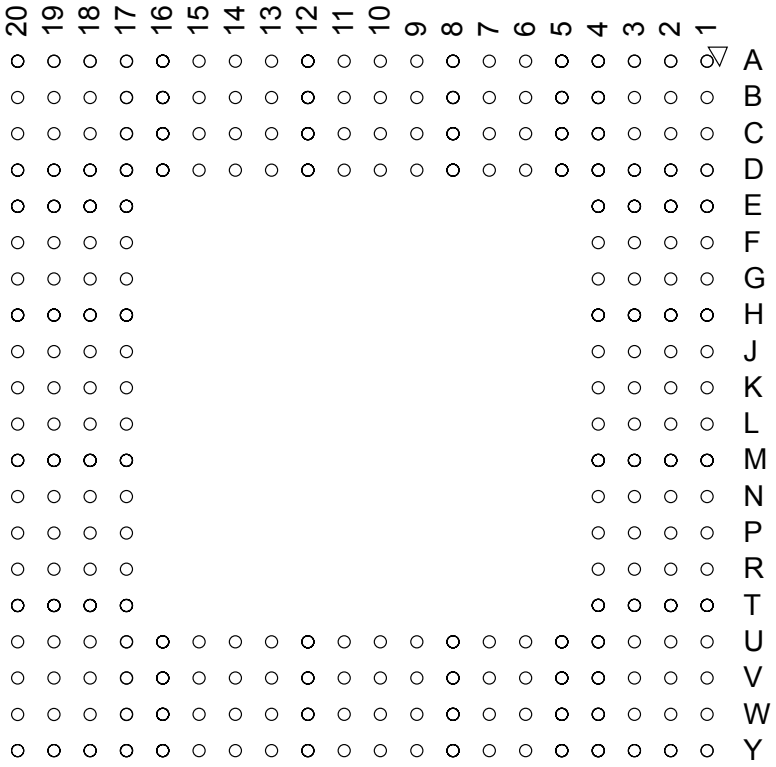
Order the MC92501 with the order number PC92501GC (PBGA).

9.5 Mechanical Data

Mechanical data includes:

- Pin assignments
- Package mechanical drawing

9.5.1 Pin Assignments



256 OMPAC PBGA (BOTTOM VIEW)

Figure 9-15. 256-pin OMPAC Pin Diagram

Table 9-11. Power Pin Assignment

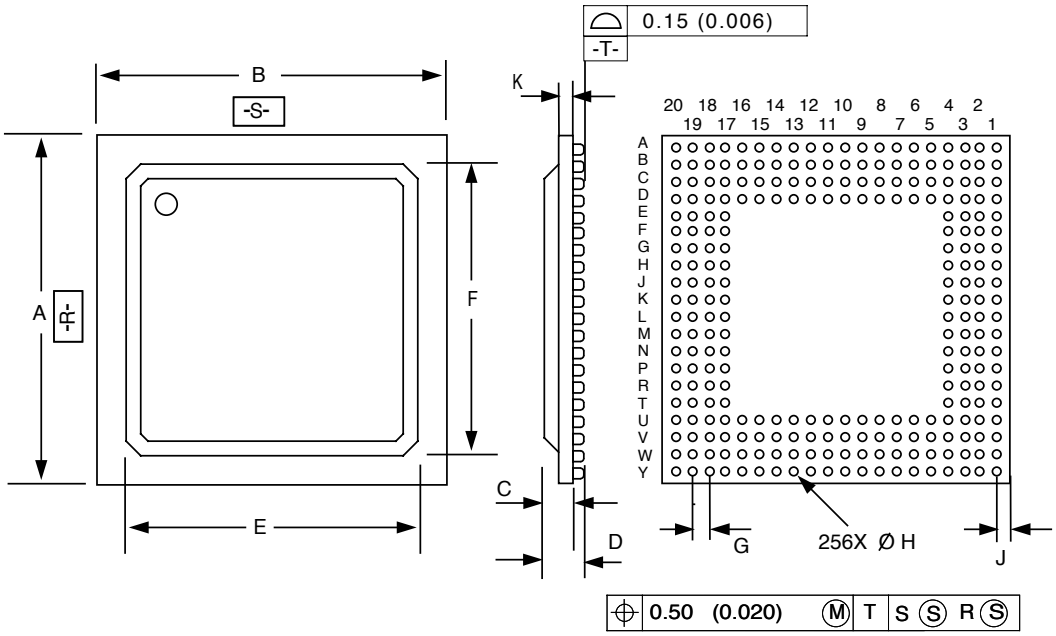
Signal Name	Pin Assignment
V_{DD}	D6, D11, D15, F4, F17, K4, L17, R4, R17, U6, U10, U15, B19, B18, C4, D3, W2, Y2, V19, W19, P18, V14
V_{SS}	A1, D4, D8, D13, D17, H4, H17, N4, N17, U4, U8, U13, U17, A19, A20, B3, E4, Y1, W3, Y20, W20, T18, V16
AV_{DD}	C2
AV_{SS}	B1

NOTE: To eliminate coupling of digital switching noise into the PLL through pins AV_{DD} and AV_{SS}, connect these pins to isolated power and ground.

Table 9-12. MC92501 Functional Pin Assignment

Pin	Signal Name	Pin	Signal Name	Pin	Signal Name	Pin	Signal Name	Pin	Signal Name
C3	TESTOUT	B15	SRXPRTY	L19	EMDATA10	W12	EMBSL3	P4	MDATA31
A2	ACLK	D14	MDTACK1	M20	EMDATA9	Y12	N/C	T1	MDATA30
B2	TESTSEL	C15	RXADDR4	M19	EMDATA8	U11	AMODE1	R2	MDATA29
D5	MADD17	A16	RXSOC	M18	EMDATA7	V11	AMODE0	P3	MDATA28
A3	MADD16	B16	RXENB	M17	EMDATA6	W11	ARST	R1	MDATA27
B4	MADD15	C16	RXEMPTY	N20	EMDATA5	Y11	TCK	P2	MDATA26
C5	MADD14	A17	RXPHYID3	N19	EMDATA4	Y10	TRST	P1	MDATA25
A4	MADD13	A18	RXPHYID2	N18	EMDATA3	V10	TMS	N3	MDATA24
B5	MADD12	D16	RXPHYID1	P20	EMDATA2	W10	TDO	N2	MDATA23
C6	MADD11	C17	RXPHYID0	P19	EMDATA1	Y9	TDI	N1	MDATA22
D7	MADD10	B17	RXPRTY	R20	EMDATA0	W9	ENID	M4	MDATA21
A5	MADD9	C18	RXDATA7	R19	EMADD23	V9	STXCLK	M3	MDATA20
B6	MADD8	B20	RXDATA6	P17	EMADD22	U9	STXCLAV	M2	MDATA19
C7	MADD7	C19	RXDATA5	R18	EMADD21	Y8	STXSOC	M1	MDATA18
A6	MADD6	D18	RXDATA4	T20	EMADD20	W8	STXPRTY	L4	MDATA17
B7	MADD5	E17	RXDATA3	T19	EMADD19	V8	STXDATA7	L3	MDATA16
A7	MADD4	C20	RXDATA2	U20	EMADD18	Y7	STXDATA6	L2	MDATA15
C8	MADD3	D19	RXDATA1	V20	EMADD17	W7	STXDATA5	L1	MDATA14
B8	MADD2	E18	RXDATA0	T17	EMADD16	V7	STXDATA4	K1	MDATA13
A8	MSEL	D20	EMDATA31	U18	EMADD15	Y6	STXDATA3	K3	MDATA12
D9	MREQ0	E19	EMDATA30	U19	EMADD14	W6	STXDATA2	K2	MDATA11
C9	MREQ1	F18	EMDATA29	V18	EMADD13	U7	STXDATA1	J1	MDATA10
B9	MDTACK0	G17	EMDATA28	Y19	EMADD12	V6	STXDATA0	J2	MDATA9
A9	MINT	E20	EMDATA27	W18	EMADD11	Y5	STXENB	J3	MDATA8
D10	MREQ2	F19	EMDATA26	V17	EMADD10	W5	TXENB	J4	MDATA7
C10	MCLK	G18	EMDATA25	U16	EMADD9	V5	TXFULL	H1	MDATA6
B10	MWR	F20	EMDATA24	Y18	EMADD8	Y4	TXCCLR	H2	MDATA5
A10	MWSH	G19	EMDATA23	W17	EMADD7	Y3	TXPHYIDV	H3	MDATA4
A11	MWSL	G20	EMDATA22	Y17	EMADD6	U5	TXPRTY	G1	MDATA3
C11	MDS	H18	EMDATA21	W16	EMADD5	V4	TXSOC	G2	MDATA2
B11	SRXENB	H19	EMDATA20	V15	EMADD4	W4	TXDATA7	G3	MDATA1
A12	SRXDATA7	H20	EMDATA19	U14	EMADD3	V3	TXDATA6	F1	MDATA0
B12	SRXDATA6	J17	EACEN	Y16	EMADD2	W1	TXDATA5	F2	MADD25
C12	SRXDATA5	J18	EMWR	W15	N/C	V2	TXDATA4	G4	MADD24
D12	SRXDATA4	J19	EMDATA18	Y15	EMBSH0	U3	TXDATA3	F3	MADD23
A13	SRXDATA3	J20	EMDATA17	W14	EMBSH1	T4	TXDATA2	E1	MADD22
B13	SRXDATA2	K17	EMDATA16	Y14	EMBSH2	V1	TXDATA1	E2	MADD21
C13	SRXDATA1	K18	EMDATA15	V13	EMBSH3	U2	TXDATA0	E3	MADD20
A14	SRXDATA0	K19	EMDATA14	W13	N/C	T3	TXPHYID3	D1	MADD19
B14	SRXCLK	K20	EMDATA13	Y13	EMBSL0	U1	TXPHYID2	C1	MADD18
C14	SRXCLAV	L20	EMDATA12	U12	EMBSL1	T2	TXPHYID1	D2	VCOCTL
A15	SRXSOC	L18	EMDATA11	V12	EMBSL2	R3	TXPHYID0		

9.5.2 Package Dimensions



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	26.90	27.10	1.0590	1.0669
B	26.90	27.10	1.0590	1.0669
C	1.330	1.730	0.0523	0.0681
D	1.830	2.430	0.072	0.0956
E	23.80	24.20	0.9370	0.9527
F	23.80	24.20	0.9370	0.9527
G	1.27 BSC		.050 BSC	
H	0.690	0.810	0.0271	0.0318
J	1.335	1.535	.0526	.0604
K	0.310	0.410	0.012	0.016

Figure 9-16. 256-pin OMPAC PBGA (Preliminary Drawing)

A

UPC/NPC Design

A.1 Time-Stamped Leaky Time Bucket

A UPC design is normally conceptualized as a bucket with a hole in the bottom that allows water to drain out at a steady rate. Each admitted cell is considered a fixed quantity of water added to the bucket. Arriving cells are admitted at the rate at which the bucket drains. Any cell that would cause the bucket to overflow is not admitted. The water drainage rate represents the average data flow bandwidth. Cell processing at each update is defined by the following equation:

new_contents = old_contents – [elapsed_time / avg_time_between_cells] + 1

where

- new_contents is the new contents of the bucket in cells
- old_contents is the previous contents of the bucket in cells
- elapsed_time is the amount of time that has elapsed since the previous cell arrived
- avg_time_between_cells is the expected elapsed time between consecutive cells, in units of time per cell.

NOTE: The term [elapsed_time / avg_time_between_cells] represents the number of cells that drained from the bucket since the previous update.

The same flow control can also be achieved by re-dimensioning the quantities involved. The size of the hole (representing the bandwidth) may be held constant, while the quantity of water added with each cell may be varied. In effect, this changes the dimension of the bucket from cells to time. The amount of water in the bucket (the current bucket contents) now represents the amount of time that should pass without any cells being admitted if we wish to produce the stated average bandwidth. The processing of the bucket can now be obtained by:

new_contents = old_contents – elapsed_time + avg_time_between_cells

where

- new_contents is the new contents of the bucket in units of time
- old_contents is the previous contents of the bucket in units of time
- elapsed_time is the amount of time elapsed since the previous cell arrived
- avg_time_between_cells is the average amount of time that should elapse between consecutive cells.

NOTE: This equation is obtained from the previous one by multiplying each term by the avg_time_between_cells. The primary advantage of this proposal, as compared to the equivalent design in which the bucket is dimensioned in units of cells, is to eliminate a complicated floating point divider circuit that would be needed to divide the elapsed_time by the avg_time_between_cells.

The following synopsis of the UPC design uses the method described above. The bucket content value indicates how much time must elapse before the bucket becomes empty. A time-stamp indicates the arrival time of the most recently admitted cell. The next cell arrival subtracts the time-stamp from the current time to compute the elapsed time. It subtracts this elapsed time value from the bucket content value indicating that less time is now required to deplete the bucket fully. As long as the bucket contents value is less than a defined threshold, the arriving cell is admitted to the cell flow, and the bucket value is incremented by a programmable increment value (avg_time_between_cells), indicating that more time is required to deplete the bucket.

A.2 Multi-Enforcer UPC/NPC

The MC92501 UPC/NPC design provides up to four enforcers per connection. These enforcers may be configured to check compliance with the Peak Cell Rate and/or the Sustainable Cell Rate for combinations of CLP = 0, CLP = 1, and CLP = 0 + 1 cell streams. The enforcers are applied to the cell stream in a serial manner, such that a “tag” decision by one enforcer results in the cell being treated as belonging to the CLP = 1 cell stream by subsequent enforcers.

Each enforcer produces one of four results:

1. **Bypass**—The cell is outside the scope of this enforcer
2. **Pass**—The cell is conforming
3. **Tag**—The cell is non-conforming and the tag option is in effect
4. **Discard**—The cell is non-conforming and the discard option is not in effect

The combined result of the enforcers is a single decision regarding the cell:

- Admit as is,
- Admit and tag, or
- Discard.

As each cell is processed, the bucket parameters of all of the enforcers are updated as follows:

- The time-stamp is set to the current time value
- The elapsed time is subtracted from the bucket contents of each enforcer.
- If the cell is admitted, each enforcer that produced a “pass” is updated by adding the avg_time_between_cells to the bucket contents.

A.3 Data Structure

The UPC/NPC enforcement process uses the data structure shown in **Figure A-1**. The Context Parameters Table contains a pointer (BKT_PTR) to a set of entries in the bucket table which hold the key parameters that customize the UPC/NPC function for a specific connection. See Section 7.3.3.4 Common Parameters and Section 7.3.11 Buckets Record for the physical structure of the External Memory tables.

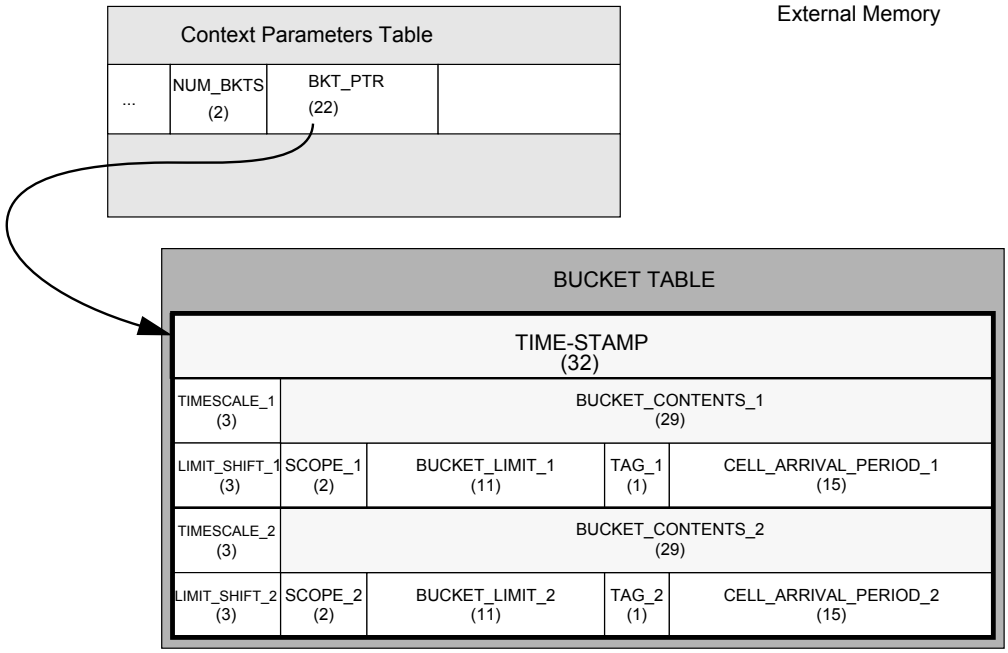


Figure A-1. UPC Data Structures

- The BKT_PTR field points to the Time-Stamp word. If the Bucket Pointer is all 1s, it is the NULL pointer, indicating NO buckets are applied to the connection.
- The NUM_BKTS (NBK) field in the Context Parameters Table indicates how many Leaky Bucket enforcers (1–4) are active on this connection. The buckets are sequential in the bucket memory address space following the Time-Stamp word. See Section 7.3.3.4 Common Parameters.
- The BUCKET_CONTENTS and TIME-STAMP fields are dynamic fields that are updated whenever a cell is admitted.
- All other entries in the bucket table are normally defined upon connection setup. They are considered to be static parameters in that they are not changed by the UPC/NPC enforcement process.

A.4 Detailed Flowchart

A flow chart of the multi-enforcer UPC design is shown in **Figure A-2**.

NOTE: If the cell is discarded, no write back occurs.

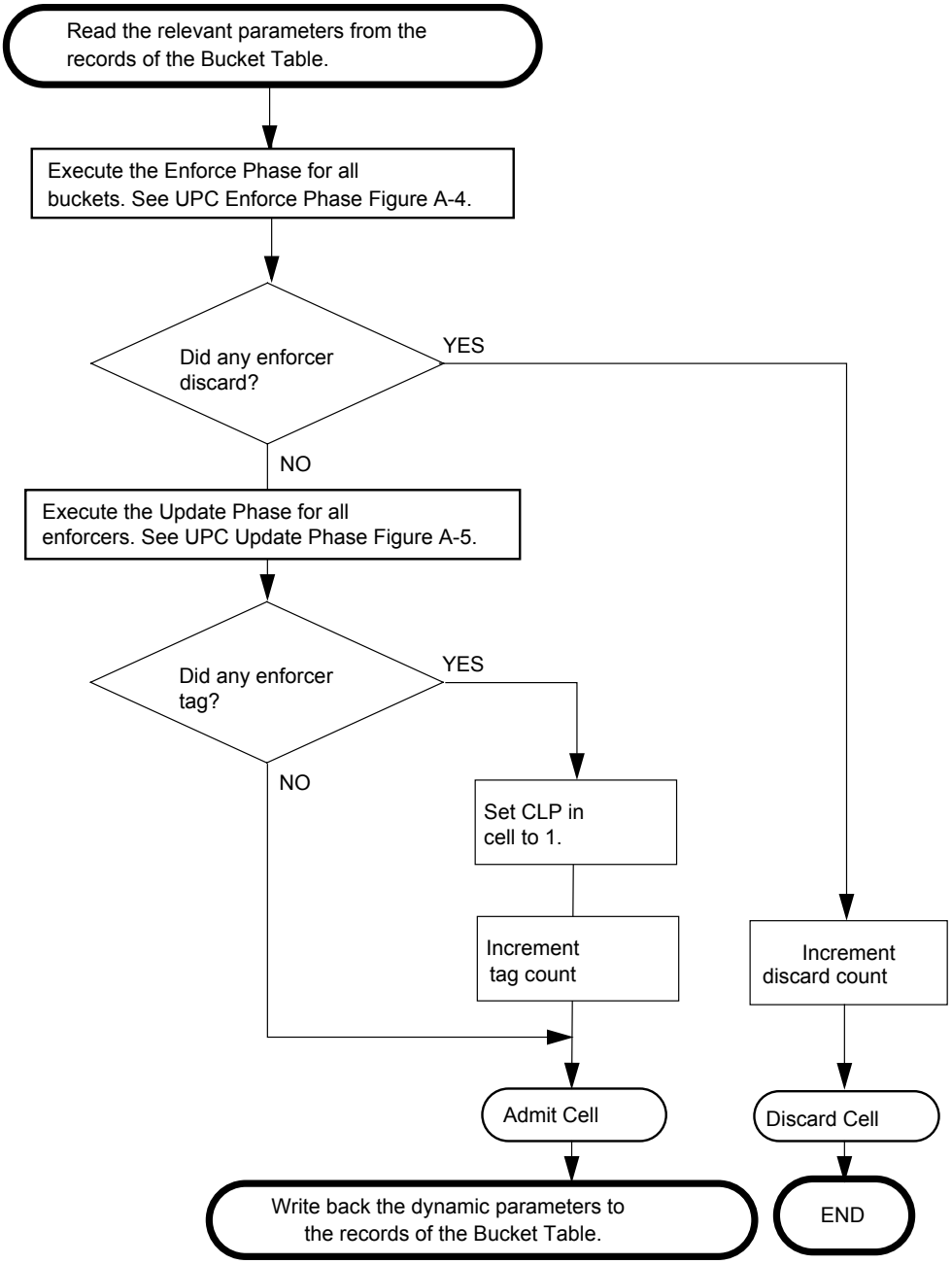


Figure A-2. Detailed UPC Flowchart

A.5 Control/Data Flowcharts

The control/data flowchart shown in **Figure A-3** shows the functional operation of the UPC design. A global timer computes the current time. The end result of the process is to update the Time-Stamp and the Bucket Contents values in the Bucket Table, to make an admit or discard judgement for every submitted cell, and to determine the CLP of the cell to be admitted.

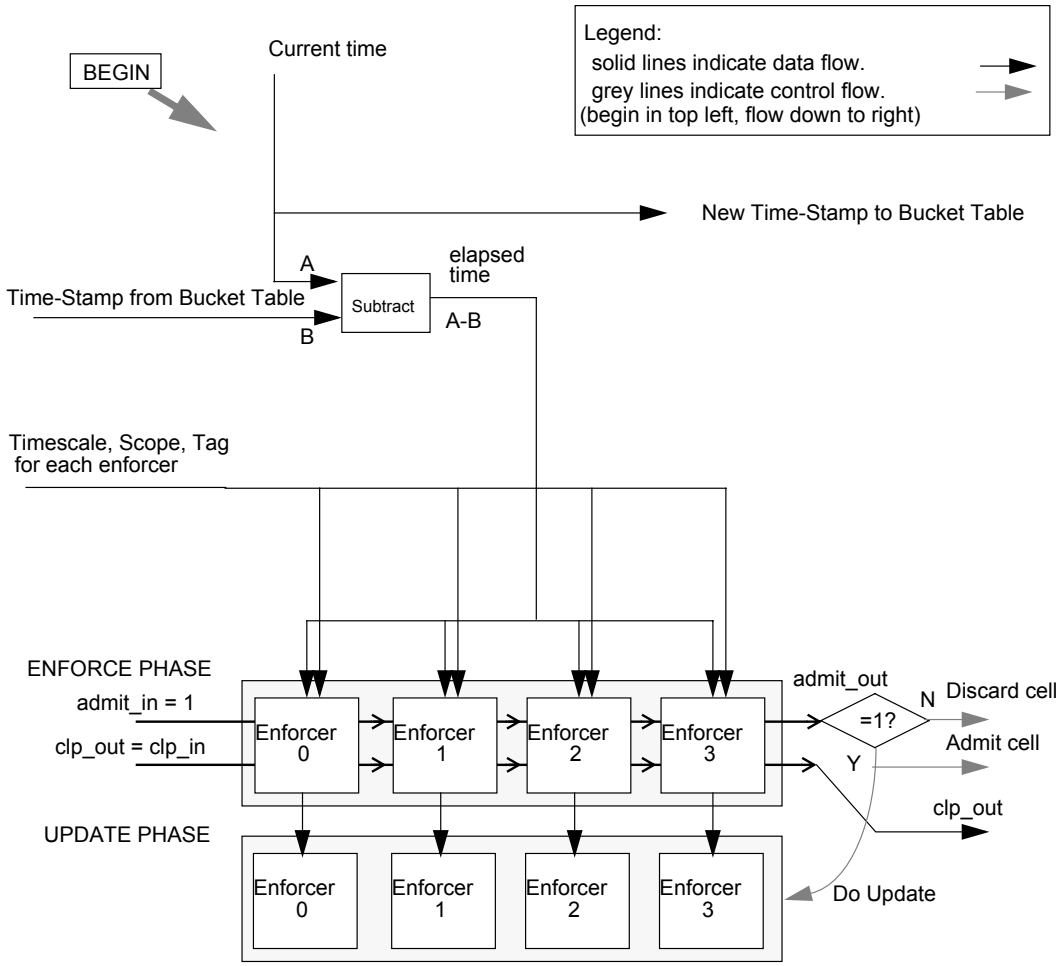


Figure A-3. UPC Control and Data Flowchart

Since the Time-Stamp stored in the Bucket record is 32 bits, the elapsed time is also limited to 32 bits. If a connection is silent for more than 2^{32} cell times (> 3 hours), the current time wraps, and there is a small probability that a cell is discarded unnecessarily. This unlikely occurrence can be prevented if the microprocessor clears the bucket contents of any connection that is silent for long periods of time. Another possible solution is to ensure that OAM Continuity Check cells are occasionally transmitted on such a connection. The CC cells trigger updates of the bucket record.

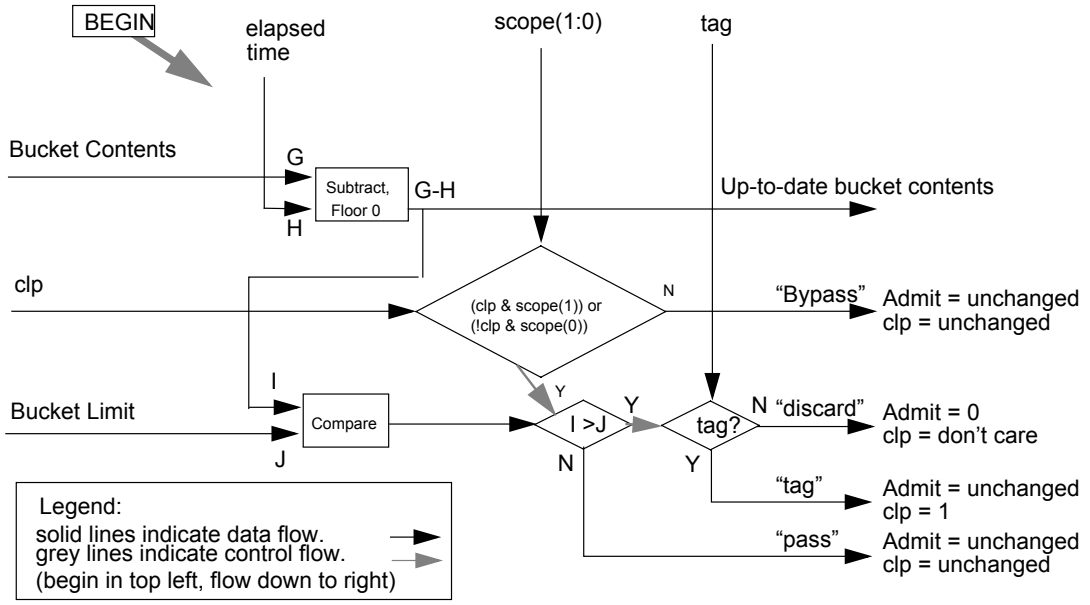


Figure A-4. UPC Enforce Phase

In the Enforce Phase each enforcer block executes the operations shown in **Figure A-4**. The elapsed time is subtracted from the previous Bucket Contents to compute the up-to-date bucket contents. The bucket contents value has a floor of zero applied such that it never goes negative. The CLP is checked against the Scope field from the Bucket Record for the enforcer according to **Table A-1**. If it is not in the scope, the enforcer is bypassed, i.e. further processing is aborted, and no changes are made to ADMIT or CLP. If the CLP is in the scope, the up-to-date bucket contents value is compared to the Bucket Limit to determine if the enforcer should pass the cell. If the bucket contents value is larger than the Bucket Limit, the result of the enforcement is either "tag" or "discard" depending on the setting of the Tag bit from the Bucket Record.

Table A-1 Enforcer SCOPE Field

SCOPE(1:0)	CLP = 1	CLP = 0
00	Bypass	Bypass
01	Bypass	Operate
10	Operate	Bypass
11	Operate	Operate

If the enforcer passed the cell, no changes are made to ADMIT or CLP. If the enforcement result is "discard", ADMIT is reset so that the cell is discarded. If the enforcement result is "tag", CLP is set. Note that the tag option is only relevant if the Scope field is 01

(operate only on CLP = 0 cells). If an enforcer produces a “tag” result, subsequent enforcers operates on this cell only if SCOPE(1) is set. Therefore, if the tag option is used in any of the enforcers, the order of the Bucket Records becomes significant.

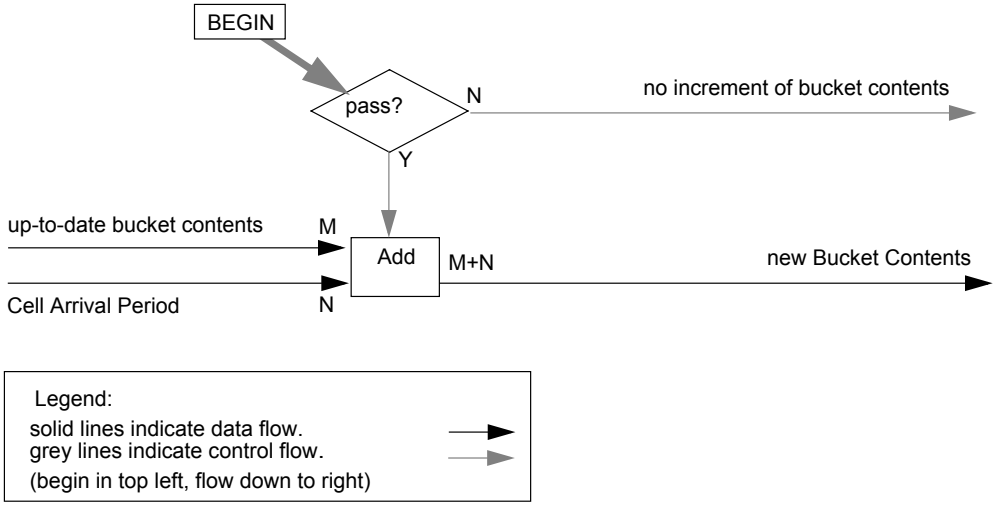


Figure A-5. UPC Update Phase

The Update Phase takes place only if the processed cell is actually admitted, i.e. no enforcer produced a “discard” result. In the Update Phase each enforcer that passed the cell increments the bucket contents as shown in **Figure A-5**.

A.6 Bucket Parameter Encoding

The choice of an encoding format for the Cell Arrival Period (CAP) is driven by the need to keep enforcement errors small. With insufficient bits allocated to this field, the granularity of enforcement becomes coarse, and at high bandwidths, the difference between two settings 64 Kb/s apart becomes indistinguishable. The enforcement errors result from rounding the cell arrival period to a finite number of bits. The cell arrival period should always be rounded down, not up, since errors in the user’s favor are preferable as they do not affect the Quality of Service of the connection. Using 15 bits with timescales that shift the decimal point 2 bits at a time produces a worst case error of 2^{-13} of the enforced bandwidth. This results in a maximum error smaller than 32 Kb/s for bandwidths up to STS-3c.

The Bucket Contents (BKC) field needs the same precision as the CAP, and it must contain larger values in order to accommodate a burst. 29 bits are used for the contents value which means that bursts of at least 16,000 cells can be accommodated. If larger bursts are needed, they can be obtained by reducing the precision of the CAP and BKC values. This is done by changing the value of the Timescale field and filling the MSBs of the CAP with 0s. The justification for this trade-off is that the precision of the average bandwidth (as specified by the CAP) becomes less meaningful as we allow greater variations from this value over long periods of time (i.e. larger bursts).

The CAP and BKC fields are dimensioned in units of time. The basic unit is referred to as a cell time. It is defined as the period of the rate at which the MC92501 processes the ATM cells. This number can be derived by multiplying the number of clock cycles used to process a cell (64) by the period of ACLK, the clock signal provided to the MC92501. For example, using a 25 MHz clock results in a cell time of 2.56 µs.

Eight timescale values are used for encoding the CAP and BKC fields. The Timescale defines the units of these fields as a fraction of a cell time. **Table A-2** contains the units of the CAP and BKC fields as a function of the Timescale. In terms of implementation, the Timescale would indicate how much to shift the elapsed time (which is in whole cell time units) to the left before subtracting it from the BKC.

Table A-2 Cell Arrival Period and Bucket Contents Encoding

Timescale	Units of Cell Arrival Period (cell times)	Number of Bits the Elapsed Time is Shifted
0	1	0
1	1/4	2
2	1/16	4
3	1/64	6
4	1/256	8
5	1/1024	10
6	1/4096	12
7	1/16384	14

To enforce a bandwidth of N cells per second, the cell arrival period needed is 1/N seconds. This must be normalized to units of MC92501 cell times by dividing by the duration of a cell time. This leads to the following equation:

$$CAP = \frac{1}{N}/(CellTime) = [N \cdot CellTime]^{-1}$$

A.6.1 Cell Arrival Computation Example

Here is a quick example of the computation of the Cell Arrival Period. We wish to enforce a bandwidth of 5003 cells per second. A cell time is 2.56 µs, as defined above. Therefore, the cell arrival period is 78.0782. The best precision is obtained by expressing this value in smaller units. Checking **Table A-2**, we find that using Timescale 4 where the units are 1/256 of a cell time is the best we can do, since Timescale 5 would require multiplying the cell arrival period by 1024 which would overflow the 15 bits allocated to the CAP. Multiplying 78.0782 by 256 to convert it to Timescale 4 yields 19,988.01. We

round this down to 19,988 and encode this value in the CAP field.

A.6.2 Bucket Limit Encoding

The size of the bucket determines the amount that the cells can get ahead of the bandwidth. The criterion for conformance is that adding a cell arrival period to the bucket would not cause it to overflow. In order to simplify the calculations, the value used by the MC92501 to specify the bucket size is the Bucket Limit (BKL) which is defined as one cell arrival period less than the required bucket size. By comparing the up-to-date bucket contents to the BKL, the MC92501 can determine if there is room in the bucket for a cell arrival period without performing the addition.

The BKL is defined in units of cell times. Its value may range from a few cell times (constant bit rate with little jitter) up to tens of thousands of cell times (large bursts). However, the precision of this field does not need to be extraordinarily high. Therefore, we can save space by providing a limited number of bits in which to encode the BKL, along with a shift factor. The BKL is chosen to be an 11-bit field. The shift factor indicates how much to shift the BKL to the left before the bucket contents value is compared to it. **Table A-3** defines the shift of the BKL field in terms of the Limit Shift (LMS) field.

Table A-3 Bucket Limit Encoding

Limit Shift (LMS)	Number of Bits the Bucket Limit (BKL) is Shifted
0	0
1	3
2	6
3	9
4	12
5	15
6	18
7	Reserved

A.7 UPC Parameter Calculations

The following examples demonstrate how the static UPC parameters may be calculated based on the traffic parameters of the connection.

A.7.1 Example A

This example involves a 64 Kb/s data connection providing circuit emulation using AAL1, allowing a 10% jitter. A connection that provides 64 Kbps using AAL1 needs 170.21 cells per second. Using the formula given above with a cell time of 2.56 µs, the cell arrival period should be 2294.9 cell times which can be approximated by encoding the CAP as $23DB using Timescale 1 (9179 × 1/4 = 2294.75). A jitter of 10% means that a cell

can arrive up to 2294.9/10 = 229.49 cell times early without being discarded. Therefore, the bucket size used by this connection should be 1.10 × 2294.9 = 2524.4. Since Timescale 1 is being used, the bucket size should be rounded up to the nearest 1/4 cell time (2524.5 = 10,098 × 1/4). The Bucket Limit is calculated by subtracting one CAP value (9179) from the bucket size (10,098), yielding 919. Thus, the value to be placed in the BKL field is $397, and the LMS value is 0.

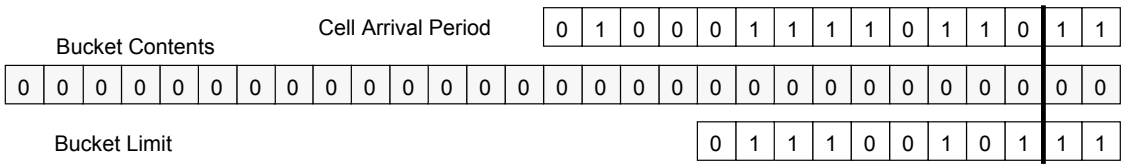


Figure A-6. Encoding of Bucket Parameters for 64 Kbps Circuit Emulation

Figure A-6 shows the encoding of the bucket parameters for this example. The actual bandwidth being enforced is 64.0048 Kb/s with a jitter of 10.01%. Note that the errors are small and in the user's favor.

A.7.2 Example B

This example involves a connection with a Sustainable Cell Rate of STS-1 with a burst of twice the bandwidth for up to 400 milliseconds and an all-out burst of up to 200 microseconds. To enforce two different burst sizes, we use two buckets. One bucket uses the average connection bandwidth with a burst of twice the bandwidth for 400 ms. The other bucket takes the double bandwidth as its base bandwidth and allows an all-out burst for 200 μs.

An STS-1 connection (51.840 Mb/s at the physical layer) uses 116,830.19 cells per second. This is equivalent to a cell arrival period of 3.343528 cell times. This value is approximated by encoding the CAP as $357F using Timescale 6 (13695/4096 = 3.343506). The burst of twice the bandwidth for 400 ms is 2 × 0.400 × 116,830.19 = 93,464 cells. When this is multiplied by the cell arrival period, the resulting bucket size is 312,497 cell times. Expressed in terms of Timescale 6, the bucket size is 312,497 × 4096 = 1,279,989,533. The Bucket Contents (BKC) field has 29 bits which do not suffice to contain this value. Therefore, we choose to compromise precision in order to increase the bucket size. We now encode the CAP as $0D5F using Timescale 5 (3423/1024 = 3.3428). Recalculating the bucket size using this value produces 93,464 × 3.3428 = 312,431.5 Converting the bucket size to Timescale 5 involves multiplying by 1024 and rounding up, yielding 319,929,815. The Bucket Limit is obtained by subtracting the Cell Arrival Period from the bucket size, yielding 319,926,392. Since the Bucket Limit is large, this example uses Limit Shift = 6. To determine the BKL after the shift, we must divide by 2^{18} (since LMS=6 provides a shift of 18 bits). The result is rounded up to yield 1221, or $4C5. These results are illustrated in **Figure A-7**.

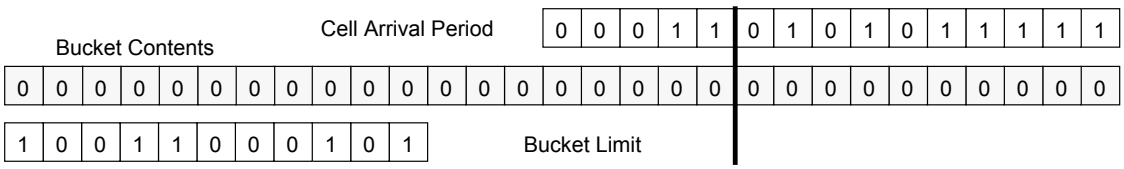


Figure A-7. Encoding of Bucket 1 Parameters

The second bucket uses a base rate of 2 × 116,830.19 = 233,660.38 cells per second. This is equivalent to a cell arrival period of 1.671764. This value is approximated by encoding the CAP as $6AFE using Timescale 7 (27,390 / 16,384 = 1.671753). The continuous burst for 200 µs is 70.6415 cells. When this is multiplied by the cell arrival period, the resulting bucket size is 118.095 cell times, or in Timescale 7 terms, 1,934,871. The Bucket Limit is 1,934,871 – 27,390 = 1,907,481. This value requires only 21 bits, so LMS = 4 can be used. Dividing by 2^{12} and rounding up produces a BKL value of 466, or $1D2. The encoding of these values is shown in **Figure A-6**.

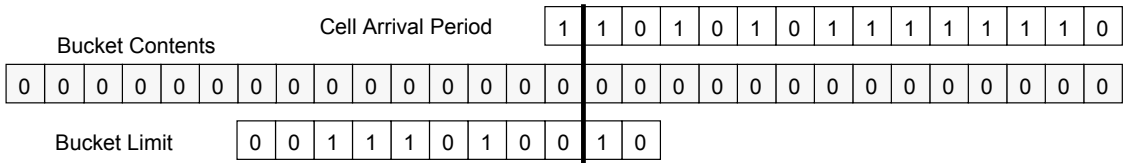


Figure A-8. Encoding of Bucket 2 Parameters

A.8 Bellcore Cell Relay Service Parameters

Bellcore [14] lists supported values of Peak Cell Rate (PCR) and Cell Delay Variation (CDV) for Constant Bit Rate (CBR) connections. For Variable BIt Rate (VBR) connections, supported values of PCR, CDV, Sustainable Cell Rate (SCR), and Maximum Burst Size (MBS) are provided.

All of the computed values are based on the ACLK frequency being 25 MHz. The only CDV specified is 250 µs which is about 98 cell times. The only MBS specified is 210 cells. For CBR connections the increment (CAP) is 1/PCR and the limit is CDV. For VBR connections the increment is 1/SCR and the limit is (MBS – 1)(1/SCR – 1/PCR) + CDV.

Table A-4 Bucket Parameters for CBR Connections

PCR (cells/s)	CAP (Hex)	TSC	BKL (Hex)	LMS
0[a]	7FFF	0	062	0
173	2347	1	188	0

Table A-4 Bucket Parameters for CBR Connections (Continued)

PCR (cells/s)	CAP (Hex)	TSC	BKL (Hex)	LMS
346	468F	2	61C	0
1038	5E14	3	30E	1
2076	2F0A	3	30E	1
4140	5E5A	4	187	2
8280	2F2D	4	187	2
16,560	5E5A	5	61B	2
119,910	341F	6	30E	3

a. The minimum enforceable cell rate is approximately 11.9 cells/s.

Table A-5 Bucket Parameters for VBR Connections

PCR (cells/s)	CAP (Hex)	TSC	BKL (Hex)	LMS	SCR (cells/s)	CAP (Hex)	TSC	BKL (Hex)	LMS
173	2347	1	188	0	18	54C5	0	3E1	4
					87	4627	1	1C8	4
346	468F	2	61C	0	35	2B98	0	200	4
					173	2347	1	735	3
1038	5E14	3	30E	1	104	3AB0	1	2B2	4
					519	2F0A	2	134	4
2076	2F0A	3	30E	1	208	7560	2	564	4
					1038	5E14	3	268	4
3622	6BD9	4	187	2	363	4341	2	317	4
					1811	35EC	3	162	4
4140	5E5A	4	187	2	414	3AF8	2	2B6	4
					2070	2F2D	3	136	4
12,420	7DCE	5	61B	2	1242	4EA0	3	39E	4
					6210	3EE7	4	1A1	4

Table A-5 Bucket Parameters for VBR Connections (Continued)

PCR (cells/s)	CAP (Hex)	TSC	BKL (Hex)	LMS	SCR (cells/s)	CAP (Hex)	TSC	BKL (Hex)	LMS
28,980	35EA	5	61B	2	2898	21B2	3	18E	4
					14,490	6BD5	5	2D9	4
33,120	2F2D	5	61B	2	3312	75F1	4	571	4
					16,560	5E5A	5	281	4
45,540	224F	5	61B	2	4554	55C6	4	3F7	4
					22,770	449E	5	1D9	4
96,000	411A	6	30E	3	9600	28B0	4	1E5	4
					48,000	208D	5	769	3
120,060	340E	6	30E	3	12,006	2089	4	185	4
					60,030	681D	6	30A	4
273,240	5B7E	7	187	4	27,324	392F	5	2B9	4
					136,620	2DBF	6	18D	4
353,207[a]	46C7	7	187	4	35,321	2C3C	5	221	4
					176,604	2363	6	149	4

a. The benefit derived from enforcing the PCR of a connection using the full link rate is somewhat questionable.

B

Maintenance Slot Calculations

B.1 Maintenance Slot Equations

The following variables are used in the Maintenance Slot calculations:

- $\mathbf{R_L}$ — Link Rate measured in cells per second = Link Rate in bps divided by 53 B/cell and 8 b/B
- **C** — Number of clocks per cell slot is a design constant (64)
- **P** — Period of a cell slot
- **f** — ACLK frequency, a system design parameter
- $\mathbf{I_S}$ — Interval of maintenance slots measured in cell slots, a programmed value chosen by the user
- $\mathbf{I_T}$ — Interval of maintenance slots measured in time
- $\mathbf{N_{CS}}$ — Number of cell slots per second
- N_{FS} — Number of free cell slots second
- $\mathbf{N_{MS}}$ — Number of maintenance slots per second
- $\mathbf{N_{ES}}$ — Number of empty cell slots per second

The equations used in the Maintenance Slot calculations include:

- The period of an MC92501 cell slot is 64 ACLK periods:
  - $P = C / f$
- The number of cell slots per second is the inverse of a cell slot period:
  - $N_{CS} = 1 / P = f / C$
- Free slots are cell slots that are not used for processing cells arriving from the PHY layer. The number of free slots per second is computed by subtracting the link cell rate from the total number of MC92501 cell slots:
  - $N_{FS} = N_{CS} - R_L$

- Free slots can be used for one of two purposes. Each free slot is either declared a Maintenance Slot, or it is left empty to be used for processing inserted cells if necessary. The fraction of cell slots used as maintenance slots is determined by the value of the *Maintenance Period Length (MPL)* field (see Section 7.2.5.2 Maintenance Control Register (MACTLR)) programmed by the user: I_S = MPL + 1. The remaining free slots are left empty.
  - $N_{MS} = N_{CS} / I_S$
  - $N_{ES} = N_{FS} - N_{MS}$
- The time between maintenance slots is calculated by multiplying the number of cell slots between maintenance slots (Maintenance Slot Interval) by the period of a cell slot:
  - $I_T = I_S * P$

B.2 Maintenance Time Slot Example

We take as an example a configuration in which ACLK is 25 MHz, and the link rate is STS-3c (149.76 Mb/s net bit rate after the SONET overhead is removed). Therefore,

- R_L = 149.76 E + 6 / (53 * 8) = 353,207 cells/sec
- f = 25 E + 6 Hz
- P = C / f = 64 / 25 E + 6 = 2.560 µs
- N_{CS} = f / C = 25 E + 6 / 64 = 390,625 cells/s

The MC92501, when running at 25 MHz, has 390,625 cell processing slots available per second. Of these, 353,207 are used for processing the cells received from the STS-3c link. Therefore, the number of free cell slots is:

- $N_{FS} = N_{CS} - R_L$ = 390,625 – 353,207 = 37,418 cell slots per second

At this point the user must calculate how many maintenance slots are needed for the maintenance tasks to be performed by the microprocessor (or DMA device). Note that each Maintenance Slot is the same length as a cell processing slot (P = 2.560 µs). Let us assume that at least 25,000 maintenance slots are needed per second. We must now calculate the Maintenance Slot Interval which is the fraction of the cell processing slots used as maintenance slots.

- $N_{MS} = N_{CS} / I_S$

so

- $I_S = N_{CS} / N_{MS}$ = 390,625 / 25,000 = 15.63

For a minimum of 25,000 maintenance slots, round I_S down to 15 and

- $N_{MS} = N_{CS} / I_S$ = 390,625 / 15 = 26,042 maintenance slots
- $I_T = I_S * P$ = 15 * 2.560 = 38.40 µs

The remaining free slots are left empty for insertion:

- $N_{ES} = N_{FS} - N_{MS} = 37,418 - 26,042 = 11,376$ cell slots

If this number is not large enough for the expected number of inserted cells per second, we might reconsider the system design with the goal of reducing the number of maintenance slots needed.

B.3 Maintenance Slot Parameters

Table B-1 presents I_T, N_{MS}, and N_{ES} as a function of the ACLK frequency and I_S for an STS-3c physical link in order to assist the user in choosing the operating frequency and the value of the Maintenance Period Length.

Table B-1 Maintenance Slot Parameters

Freq		25 MHz P = 2.56 μs 37418 free slots			24 MHz P= 2.67 μs 21792 free slots		
MPL	I_S	I_T (μs)	N_{MS}	N_{ES}	I_T (μs)	N_{MS}	N_{ES}
10	11	28.2	35512	1906	—	—	—
11	12	30.7	32553	4865	—	—	—
12	13	33.3	30049	7369	—	—	—
13	14	35.8	27902	9516	—	—	—
14	15	38.4	26042	11376	—	—	—
15	16	41.0	24415	13003	—	—	—
16	17	43.5	22979	14439	—	—	—
17	18	46.1	21702	15716	—	—	—
18	19	48.6	20560	16858	50.7	19737	2055
19	20	51.2	19532	17886	53.3	18751	3041
24	25	64.0	15626	21792	66.7	15001	6791
29	30	76.8	13021	24397	80.0	12501	9291
34	35	89.6	11161	26257	93.3	10714	11078
39	40	102.4	9766	27652	106.7	9375	12417
44	45	115.2	8681	28737	120.0	8333	13459
49	50	128.0	7813	29605	133.3	7500	14292
54	55	140.8	7102	30316	146.7	6818	14974
59	60	153.6	6510	30908	160.0	6250	15542
63	64	166.4	6104	31314	173.3	5859	15933

C

External Memory Calculations

C.1 External Memory Allocation Program

The command line parsing routines were not included, but their function is obvious.

```
#include <stdio.h>
#include <string.h>
#include <math.h>

usage()
{
          printf("Usage: em [-l links] -n kconns [-p vpibits] [-c kvccs -f vceff] -r klw\n");
          printf("          [-cadd] [-sp swtprms] [-iat] [-cprm] [-eprm] [-iprm] [-extprm]\n");
          printf("          [-eb ebctrs] [-ib ibctrs] [-ip ipctrs]\n");
          printf("          [-ft] [-m midbits [-z]] [-oam] [-dv dvsize] [-b bkts]\n");
          printf("          [-elctrs] [-ilctrs]\n");
}


/*****************************************************************************
This c program calculates the RAM allocation for the External Memory associated
with the MC92500.

The options are:
        -l links        links is the number of physical links supported (1-16).
                        Default is 1.

        -n kconns       kconns is the number (in K) of active connections.

        -p vpibits      vpibits is the number of allocated bits in the VPI.
                        If VP Lookup is not performed, this option should be
                        omitted.

        -c kvccs        kvccs is the number (in K) of active VC connections.
                        If VC Lookup is not performed, this option should be
                        omitted.

        -f vceff        vceff is the percent of active connections in the VC
                        Table (0-100).  If VC Lookup is not performed,
                        this option should be omitted.

          -cadd               Connection Address word exists.

        -sp swtprms     swtprms is the number of longwords of switch parameters
                        (0-3).  Default is 0.

          -iat                Ingress Address Translation word does not exist.

          -cprm               Common Parameters word exists.

          -eprm               Egress Parameters word exists.

          -iprm               Ingress Parameters word exists.

          -extprm             Extension Parameters table exists.
```

```
        -eb ebctrs      ebctrs is the number of Egress billing counters per
                               connection (0-4).  Default is 4.

        -ib ibctrs      ibctrs is the number of Ingress billing counters per
                               connection (0-4).  Default is 4.

        -ip ipctrs      ipctrs is the number of Ingress policing counters per
                               connection (0-4).  Default is 4.

          -ft                  Flags Table exists.

        -m midbits       midbits is the number of allocated bits of the
                         multicast identifier.  If multicast translation is not
                         performed, this option should be omitted.

        -z               Multicast translation is performed on a link basis.

          -oam                 OAM Table exists.

          -pm_on_allPerformance monitoring is done on all connections

          -dv dvsizedvsize is the size of the Dump Vector Table (in K LW).
                               If the Dump Table is not used, this option should be
                               omitted.

        -b bkts          bkts is the number of UPC/NPC buckets per connection
                         (1-4).  If this option is not specified, the maximum
                         number of buckets that fit are calculated.

          -elctrs              Egress link counters table exists

          -ilctrs              Ingress link counters table exists

        -r klw           klw is the number (in K) of longwords of RAM being used
                         for the External Memory.

*****************************************************************************/

#include "cmd_line_parse.h"

main(argc, argv)
int argc;
char *argv[];
{
    int i;
    int kconns, kvccs, klw;
    int links, num_conn, vpibits, num_vc, vceff_pct, ibctrs, ipctrs, ebctrs, swtprms;
    int vp_lookup, vc_lookup, multicast, mcst_per_link, oam_table, dv_table;
    int cadd, iat, cprm, eprm, iprm, extprm, ft;
    int pm_on_all;
    int def_bkts, midbits;
    int ram_size;
    int elctrs, ilctrs;
    int context, vp, vc, mc, tot_bkts, sum, words;
    float vceff, bkts_per;
    double bkts, kdv;

/*****************************************************************************/
/* Initialize variables including default input values. */

    vp_lookup = 0;
    vc_lookup = 0;
    cadd = 0;
    iat = 0;
```

```
    cprm = 0;
    eprm = 0;
    iprm = 0;
    extprm = 0;
    ft = 0;
    pm_on_all = 0;
    multicast = 0;
    mcst_per_link = 0;
    oam_table = 0;
    dv_table = 0;
    def_bkts = 0;
    links = 1;
    ebctrs = 4;
    ibctrs = 4;
    ipctrs = 4;
    swtprms = 0;
    elctrs = 0;
    ilctrs = 0;

/*****************************************************************************/
/* Parse command line options. */

    if (argc == 1) {
          usage();
          exit(0);
    }
    i = 1;
    while (i < argc) {
              OPT_INT("-l", links);
              OPT_INT("-n", kconns);
              OPT_BOOL_INT("-p", vp_lookup, vpibits);
              OPT_BOOL_INT("-c", vc_lookup, kvccs);
              OPT_INT("-f", vceff_pct);
              OPT_BOOL_INT("-m", multicast, midbits);
              OPT_BOOL("-z", mcst_per_link);
              OPT_INT("-eb", ebctrs);
              OPT_INT("-ib", ibctrs);
              OPT_INT("-ip", ipctrs);
              OPT_BOOL("-cadd", cadd);
              OPT_INT("-sp", swtprms);
              OPT_BOOL("-iat", iat);
              OPT_BOOL("-cprm", cprm);
              OPT_BOOL("-eprm", eprm);
              OPT_BOOL("-iprm", iprm);
              OPT_BOOL("-extprm", extprm);
              OPT_BOOL("-ft", ft);
              OPT_BOOL_FLOAT("-b", def_bkts, bkts);
              OPT_INT("-r", klw);
              OPT_BOOL("-oam", oam_table);
              OPT_BOOL("-pm_on_all", pm_on_all);
              OPT_BOOL_FLOAT("-dv", dv_table, kdv);
              OPT_BOOL("-elctrs", elctrs);
              OPT_BOOL("-ilctrs", ilctrs);
              /* If we got to here, the option was not recognized. */
            printf("illegal option %s\n", argv[i]);
              usage();
            exit (-1);
            }
    num_conn = kconns << 10;
    num_vc = kvccs << 10;
    vceff = vceff_pct / 100.;
    ram_size = klw << 10;

/*****************************************************************************/
/* Echo inputs. */
```

```
    printf("\n\nExternal Memory allocation for the following configuration:  ");
    printf("%dK x 32 SRAM\n", ram_size >> 10);
    printf("\tNumber of links: %d\n", links);
    printf("\tNumber of connections: %d\n", num_conn);
    if (vp_lookup)
        printf("\tVP address bits: %d\n", vpibits);
    if (vc_lookup) {
        printf("\tNumber of VC connections: %d\n", num_vc);
        printf("\tVC efficiency: %2.0f %%\n", 100 * vceff);
    }
    printf("\tNumber of Egress Billing counters: %d\n", ebctrs);
    printf("\tNumber of Ingress Billing counters: %d\n", ibctrs);
    printf("\tNumber of Ingress Policing counters: %d\n", ipctrs);
    printf("\tConnection Address word: %s", cadd ? "yes\n" : "no\n");
    printf("\tNumber of switch parameter words: %d\n", swtprms);
    printf("\tIngress Address Translation word: %s", iat ? "yes\n" : "no\n");
    printf("\tCommon Parameters word: %s", cprm ? "yes\n" : "no\n");
    printf("\tEgress Parameters word: %s", eprm ? "yes\n" : "no\n");
    printf("\tIngress Parameters word: %s", iprm ? "yes\n" : "no\n");
    printf("\tExtension Parameters word: %s", extprm ? "yes\n" : "no\n");
    printf("\tFlags Table: %s", ft ? "yes\n" : "no\n");
    printf("\tOAM Table: %s", oam_table ? "yes" : "no");
    printf( pm_on_all ? " (PM on all)\n" : "\n");
    printf("\tEgress link counters: %s", elctrs ? "yes\n" : "no\n");
    printf("\tIngress link counters: %s", ilctrs ? "yes\n" : "no\n");
    if (def_bkts)
        printf("\tBuckets per connection: %5.2f\n", bkts);
    printf("\n");

/*****************************************************************************/
/* Calculate and print results. */

    sum = 0;
    context = num_conn * (cadd + swtprms + iat + cprm + eprm + iprm);
    if (context) {
        sum += context;
        printf("Context Parameters Table:\t%3d    K longwords\n", context >> 10);
    }

    if (extprm) {
          words = num_conn;
          sum += words;
        printf("Extension Parameters Table :\t%3d    K longwords\n", words >> 10);
    }

    if (ebctrs) {
          words = num_conn * ebctrs;
          sum += words;
        printf("Egress Billing Cntrs Table:\t%3d    K longwords\n", words >> 10);
    }

    if (ibctrs) {
          words = num_conn * ibctrs;
          sum += words;
        printf("Ingress Billing Cntrs Table:\t%3d    K longwords\n", words >> 10);
    }

    if (ipctrs) {
          words = num_conn * ipctrs;
          sum += words;
        printf("Ingress Policing Cntrs Table:\t%3d    K longwords\n", words >> 10);
    }

    if (ft) {
```

```
      sum += num_conn;
    printf("Flags Table:\t\t\t%3d    K longwords\n", num_conn >> 10);
}

if (vp_lookup) {
    vp = links * (1 << vpibits);
    if (!vc_lookup)
        vp /= 2;
    sum += vp;
    printf("VP Table:\t\t\t%6.2f K longwords\n", vp / 1024.);
}

if (vc_lookup) {
    vc = ceil((num_vc / vceff) / 2);
    sum += vc;
    printf("VC Table:\t\t\t%6.2f K longwords\n", vc / 1024.);
}

if (multicast) {
    mc = (1 << midbits) / 2;
    if (mcst_per_link)
        mc *= links;
    sum += mc;
    printf("Multicast Table:\t\t%6.2f K longwords\n", mc / 1024.);
}

if (oam_table) {
    if (pm_on_all)
          words = num_conn * 8;
      else
          words = 64 * 8;
      sum += words;
    printf("OAM Table:\t\t\t%6.2f K longwords\n", words / 1024.);
}

if (dv_table) {
    sum += (kdv * 1024);
    printf("Dump Vector Table:\t\t%6.2f K longwords\n", kdv);
}

if (elctrs) {
      words = links * 4;
      sum += words;
    printf("Egress Link Counters Table:\t%6.2f K longwords\n", words / 1024.);
}

if (ilctrs) {
      words = links * 5;
      sum += words;
    printf("Ingress Link Counters Table:\t%6.2f K longwords\n", words / 1024.);
}

if (def_bkts) {
    tot_bkts = num_conn * (1 + bkts * 2);
    printf("Buckets Table:\t\t\t%6.2f K longwords\n", tot_bkts / 1024.);
}
else {
    if (sum > ram_size) {
        printf("ERROR: RAM is not large enough for this configuration.\n");
        exit (-1);
    }
    else {
        tot_bkts = ram_size - sum;
        bkts_per = (((float)tot_bkts / num_conn) - 1) / 2;
        if (bkts_per > 4.0) {
```

```
                tot_bkts = num_conn * 9;
                bkts_per = 4.0;
            }
              if (bkts_per < 1.0) {
                bkts_per = ((float)tot_bkts / num_conn) / 3;
            }
            printf("Buckets Table:\t\t\t%6.2f K longwords", tot_bkts / 1024.);
            printf("\t\t\t%4.2f buckets/conn\n", bkts_per);
        }
    }
    sum += tot_bkts;

    printf("================================================\n");
    printf("Total:\t\t\t%14.2f K longwords\n", sum / 1024.);

    if (!def_bkts && (bkts_per < 1.0))
        printf("\nWARNING: Less than 1 bucket per connection.\n");
    if (sum > ram_size) {
        printf("\nERROR: RAM is not large enough for this configuration.\n\n");
    }
    printf("\n\n\n");
    usage();
}

}
```

C.2 External Memory Allocation Examples

The following sections describe various memory allocation strategies.

C.2.1 16K Connections—2 MB RAM

This example provides for 16K VC connections with VP and VC lookup and a full complement of counters and switch parameters.

```
External Memory allocation for the following configuration:  512K x 32 SRAM
          Number of links: 1
          Number of connections: 16384
          VP address bits: 12
          Number of VC connections: 16384
          VC efficiency: 50 %
          Number of Egress Billing counters: 4
          Number of Ingress Billing counters: 4
          Number of Ingress Policing counters: 4
          Connection Address word: yes
          Number of switch parameter words: 3
          Ingress Address Translation word: yes
          Common Parameters word: yes
          Egress Parameters word: yes
          Ingress Parameters word: yes
          Extension Parameters word: no
          Flags Table: yes
          OAM Table: yes
          Egress link counters: yes
          Ingress link counters: yes

Context Parameters Table:      128    K longwords
Egress Billing Cntrs Table:     64    K longwords
Ingress Billing Cntrs Table:    64    K longwords
Ingress Policing Cntrs Table:   64    K longwords
Flags Table:                    16    K longwords
VP Table:                     4.00 K longwords
VC Table:                    16.00 K longwords
OAM Table:                    0.50 K longwords
Egress Link Counters Table:   0.00 K longwords
Ingress Link Counters Table:  0.00 K longwords
Buckets Table:              144.00 K longwords              4.00 buckets/conn
================================================
Total:                      500.51 K longwords
```

C.2.2 8K Connections—512 KB RAM

In this example only two Ingress Billing counters, two Egress Billing counters, one Ingress Policing counter, and one long word of switch parameters are provided in order to increase the allocation of UPC/NPC enforcers.

```
External Memory allocation for the following configuration:  128K x 32 SRAM
          Number of links: 1
          Number of connections: 8192
          VP address bits: 12
          Number of VC connections: 8192
          VC efficiency: 50 %
          Number of Egress Billing counters: 2
          Number of Ingress Billing counters: 2
          Number of Ingress Policing counters: 1
          Connection Address word: yes
          Number of switch parameter words: 1
          Ingress Address Translation word: no
          Common Parameters word: yes
          Egress Parameters word: yes
          Ingress Parameters word: yes
          Extension Parameters word: no
          Flags Table: yes
          OAM Table: yes
          Egress link counters: yes
          Ingress link counters: yes

Context Parameters Table:               40    K longwords
Egress Billing Cntrs Table:             16    K longwords
Ingress Billing Cntrs Table:            16    K longwords
Ingress Policing Cntrs Table:            8    K longwords
Flags Table:                                    8    K longwords
VP Table:                                       4.00 K longwords
VC Table:                                       8.00 K longwords
OAM Table:                                      0.50 K longwords
Egress Link Counters Table:              0.00 K longwords
Ingress Link Counters Table:             0.00 K longwords
Buckets Table:                                 27.49 K longwords   1.22 buckets/conn
================================================
Total:                                        128.00 K longwords
```

C.2.3 4K Connections—256 KB RAM

```
External Memory allocation for the following configuration:  64K x 32 SRAM
          Number of links: 1
          Number of connections: 4096
          VP address bits: 12
          Number of Egress Billing counters: 2
          Number of Ingress Billing counters: 2
          Number of Ingress Policing counters: 1
          Connection Address word: yes
          Number of switch parameter words: 0
          Ingress Address Translation word: yes
          Common Parameters word: yes
          Egress Parameters word: yes
          Ingress Parameters word: yes
          Extension Parameters word: yes
          Flags Table: yes
          OAM Table: yes
          Egress link counters: yes
          Ingress link counters: yes

Context Parameters Table:                20    K longwords
Extension Parameters Table :              4    K longwords
Egress Billing Cntrs Table:               8    K longwords
Ingress Billing Cntrs Table:              8    K longwords
Ingress Policing Cntrs Table:             4    K longwords
Flags Table:                                4    K longwords
VP Table:                                   2.00 K longwords
OAM Table:                                  0.50 K longwords
Egress Link Counters Table:              0.00 K longwords
Ingress Link Counters Table:             0.00 K longwords
Buckets Table:                             13.49 K longwords     1.19 buckets/conn
================================================
Total:                                     64.00 K longwords
```

C.2.4 64K Connections—16 MB of RAM

This configuration uses Multicast Translation and PM on all connections.

```
External Memory allocation for the following configuration:  4096K x 32 SRAM
          Number of links: 1
          Number of connections: 65536
          VP address bits: 12
          Number of VC connections: 65536
          VC efficiency: 100 %
          Number of Egress Billing counters: 4
          Number of Ingress Billing counters: 4
          Number of Ingress Policing counters: 4
          Connection Address word: yes
          Number of switch parameter words: 3
          Ingress Address Translation word: yes
          Common Parameters word: yes
          Egress Parameters word: yes
          Ingress Parameters word: yes
          Extension Parameters word: yes
          Flags Table: yes
          OAM Table: yes (PM on all)
          Egress link counters: yes
          Ingress link counters: yes

Context Parameters Table:5               12    K longwords
Extension Parameters Table :             64    K longwords
Egress Billing Cntrs Table:             256    K longwords
Ingress Billing Cntrs Table:            256    K longwords
Ingress Policing Cntrs Table:           256    K longwords
Flags Table:                             64    K longwords
VP Table:                                 4.00 K longwords
VC Table:                                32.00 K longwords
Multicast Table:                         32.00 K longwords
OAM Table:                              512.00 K longwords
Egress Link Counters Table:               0.00 K longwords
Ingress Link Counters Table:              0.00 K longwords
Buckets Table:                          576.00 K longwords     4.00 buckets/conn
================================================
Total:                                 2564.01 K longwords
```

D

Tutorial

D.1 VC Bundling

This section describes the use of the MC92501 for VC bundling. This involves bundling several VCCs that are routed identically through a series of switches into a single VPC in order to reduce the processing complexity at the intermediate switches. The MC92501 fully supports VC bundling. However, the programming of the connections at the bundling point is not totally obvious and is described here in some detail.

D.1.1 VP-VC Boundary

This section describes the treatment of the boundaries between the Virtual Channel region (where the VCCs are treated individually) and the Virtual Path region (where the bundle of VCCs is treated as one VPC.) This configuration is illustrated in **Figure D-1**.

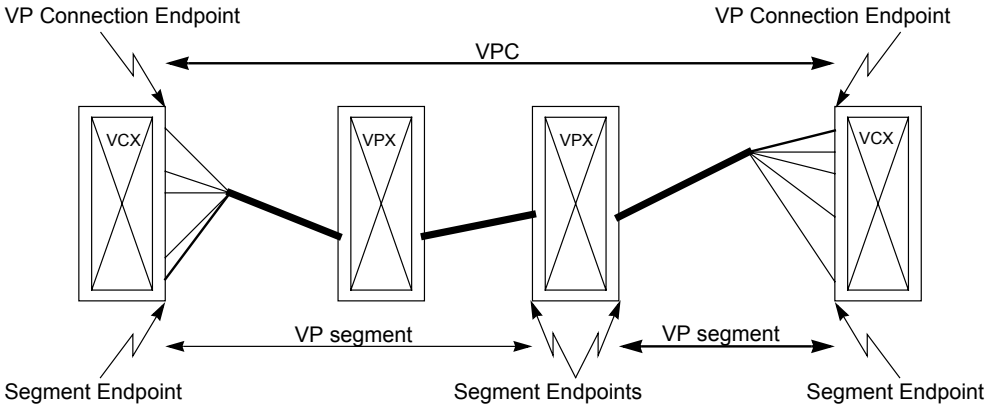


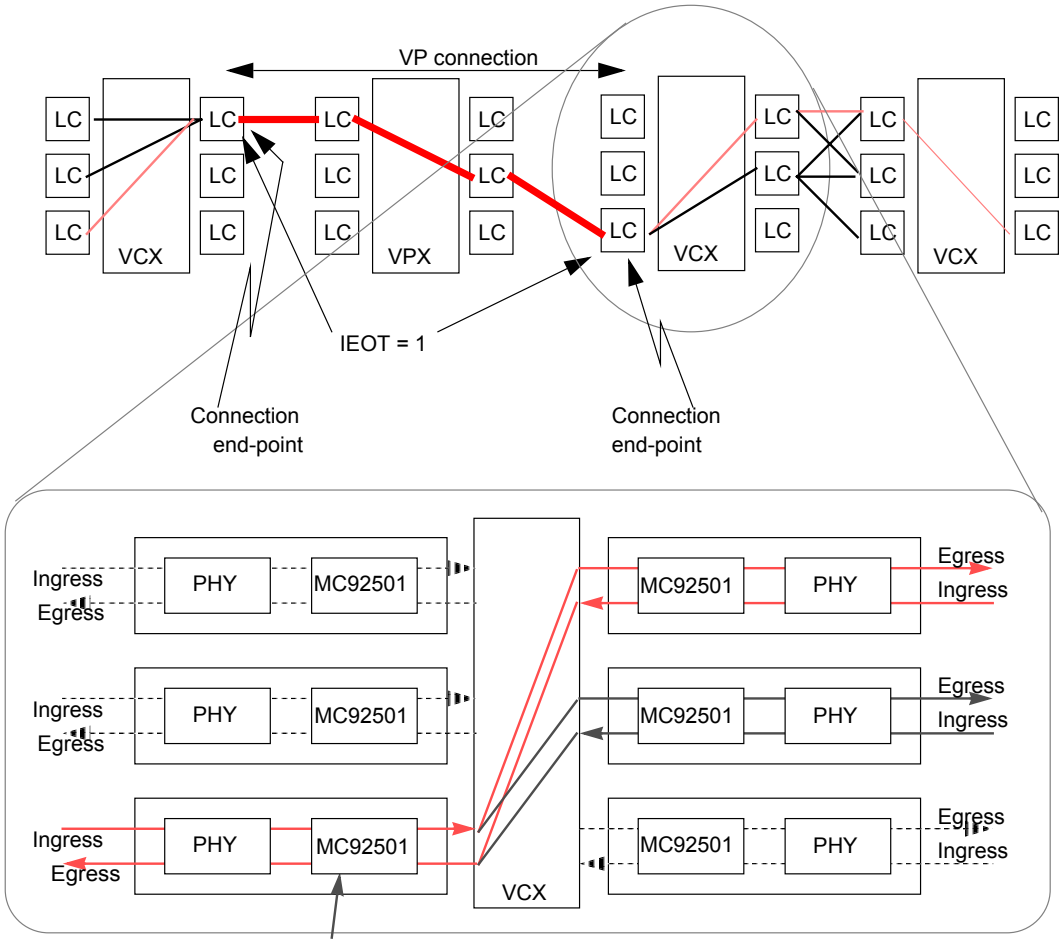
Figure D-1. Visibility of VCCs at the Endpoints of VPCs

D.1.1.1 Bundling VCs into a VP

On the Ingress side of the switch, all cells undergo a normal VC-level address-compression. Each VC connection is treated individually with its own entry in the External Memory connection tables and a distinct CI pointing to its entry. The switch parameters must direct all the cells to the same line card after passing through the switch. On the Egress side, the VCCs are still treated as separate connections, each with its own connection table entry. To provide full address translation, the *Egress Virtual Path Connection (EVPC)* bit must be cleared for all VCCs. The VPI for all connections must be identical, and the VCIs must be distinct. The VP switches along the way do not change the VCI values, which remain constant until the opposite endpoint of the VPC.

D.1.1.2 Separating a VP into VCs

As shown in **Figure D-2**, the explosion of a VP into its component VCs must take place on the Ingress side of the switch so that the virtual connections from which the virtual path is gathered can be routed to different line cards. Each VCC has its own entry in the connection tables in the External Memory and a distinct CI to point to its entry. Therefore, to produce a distinct CI for each VCC, the address compression must be done on the VPI/VCI.



Notes:
1. VPI_VCI address compression on INGRESS for each Link where ACM ≠ '01'
2. For F4-OAM traffic (i.e., VCI = 4 or VCI = 3):
   — The INGRESS is the connection/segment end-point, (i.e., IEOT=1 and (possibly) ISOT=1).
   — EVPC=1 (only VPI address translation on EGRESS)
3. For all other connections (user + F5-OAM): VPI_VCI address translation on EGRESS: EVPC=0.

Figure D-2. VP/VC Boundary Point

D.1.2 F4-Level OAM Processing at a VP/VC Boundary

As stated above, the VCCs are treated individually at the VP/VC boundary. However, the F4 flow of OAM cells refers to the VPC as a whole. We must treat these OAM cells as belonging to a VPC connection. For this purpose two additional entries are provided in the connection tables. The address compression on the F4-level OAM cells (VCI = 3 or 4) of the VPC must produce a CI that points to the special connection entry for the VCI value. The *Egress Virtual Path Connection (EVPC)* and *Ingress Virtual Path Connection (IVPC)* bits of these entries are set to indicate that VP treatment is necessary. The *Ingress End-to-end OAM Termination (IEOT)* bit of this entry should be set so that all F4-level OAM cells are removed from the cell flow.

D.1.2.1 Continuity Check

When performing an F4-level Continuity Check at the VP/VC boundary point, a user cell arriving on one of the VCCs causes the Receive Traffic bits of that VCC's entry in the Flag Table to be set. In order to determine if continuity was lost, the Receive Traffic bits from all of the VCCs should be collected by reading the entries of the Flag Table. Performing a logical OR on all of the bits consolidates them into a receive traffic indicator of the VPC as a whole.

D.1.2.2 OAM Block Test

This section describes how a block test at the VP/VC boundary point is performed. In order to perform an F4-level Performance Monitoring block test that originates or terminates at the VP/VC boundary point, a single entry in the OAM Table must be used since the block test covers all of the cells belonging to the VPC. Since each VCC has its own entry in the connection tables, the OAM_Ptr fields of all of the VCCs (as well as the special VPC entry—VCI = 3 for segment or VCI = 4 for end-to-end) must be identical and point to the common entry in the OAM Table. The CI field of the OAM Table entry points to the special VPC entry. At the originating end of the block test, the *Forward Monitoring Cell Generation (FMCG)* bit in the OAM Table entry should be set to indicate that Forward Monitoring Cells should be generated at this point. At the terminating end of the block test, the *Forward Monitoring Cell Generation (FMCG)* bit in the OAM Table entry should be reset to prevent generation of FMCs.

D.1.3 F5-Level OAM Processing at a VP/VC Boundary

At a VP/VC boundary point, the F4-level OAM block test requires that, on the VP side of the switch, the OAM pointers of all of the VCC entries point to a common entry in the OAM Table, as described above. In this situation it is not possible to run an F5-level block test on one of the VCCs while an F4-level block test is being run on the VPC as a whole since the F5-level block test would require a separate OAM Table entry for the VCC. This is not a severe limitation since VCC visibility is not required on the VP side of the switch. Instead, the F5-level OAM block test may be performed from the VC side of the switch. This situation is illustrated in **Figure D-3**.

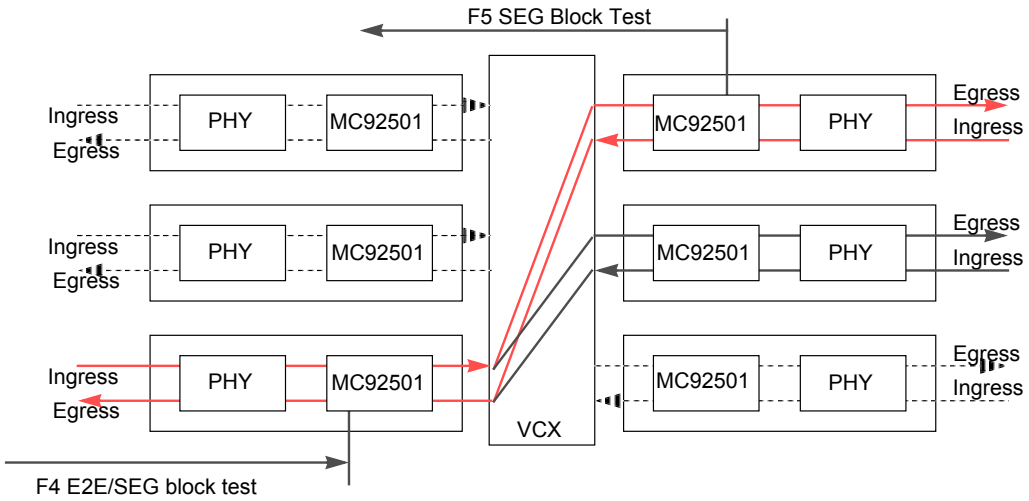


Figure D-3. F4- and F5-Level Block Tests at the Same Switch

D.1.4 Reserved VCI Values

At the VP/VC boundary, the MC92501 provides significant flexibility in copying and/or removing cells with reserved VCI values. Since the individual VCIs are visible at this point, the cells with each specific VCI value can be directed to a separate entry in the connection tables. In the connection entry the *Ingress Copy All (ICA) cells* and/or *Ingress Remove All (IRA) cells* bits may be set. This arrangement provides the ability to specify the treatment of the reserved VCI values of each VPC independently. If cells containing any one of a group of VCI values are to be treated in an identical manner, they may be routed to a single shared connection entry in which the *Ingress Virtual Path Connection (IVPC)* bit is set, thereby reducing the memory needed for this function.

NOTE: Cells with reserved VCI values are not intended to be routed past the end of the VPC, so they should most likely be removed at this point.

E

MC92501 Applications

E.1 Introduction

The following sections describe several applications that use the MC92501 in an ATM system.

E.2 Standard Line Card Architecture

Figure E-1 describes Motorola's proposed architecture for a line card implementation.

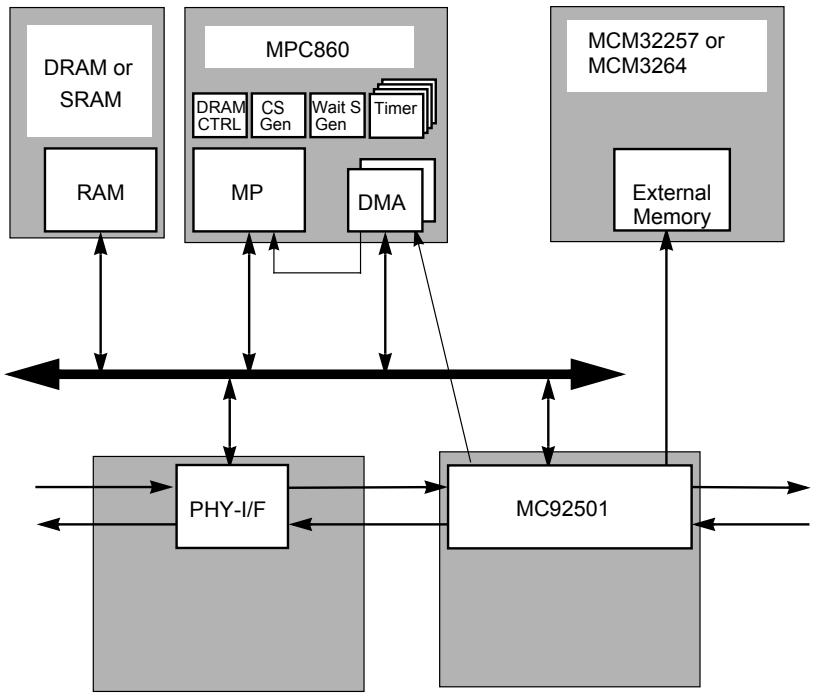


Figure E-1. Motorola's ATM Line Card Architecture

E.2.1 MC92501

The MC92501 supports PHY chips up to 155.52 Mbps.

E.2.2 PHY

For details, call your local Motorola representatives.

E.2.3 Microprocessor

The MPC860 may be used as the Line Card Processor. The MPC860 contains a powerful CPU and many additional features such as:

- DRAM Controller
- Chip Select generator
- Wait State generator
- Timers
- Two external DMA channels (IDMA)
- Communication controller for many standards

Another recommended part is the MPC860SAR which is an MPC860 + SAR functionality and can be used for automating the SAR functionality. For more details see MPC860 User's Manual.

NOTE: Any microprocessor can be used as the line card processor.

E.2.4 External Memory

The External Memory implementation can use MCM32257—256K x 32 SRAM Modules, MCM3264—64K x 32 SRAM Modules, or other memory architectures.

NOTE: Refer to Section 4.4 External Memory Interface for additional information.

E.2.5 Microprocessor RAM

This application can use any DRAM or SRAM that meets the application design requirements.

E.3 Multiple PHY Line Card Architecture

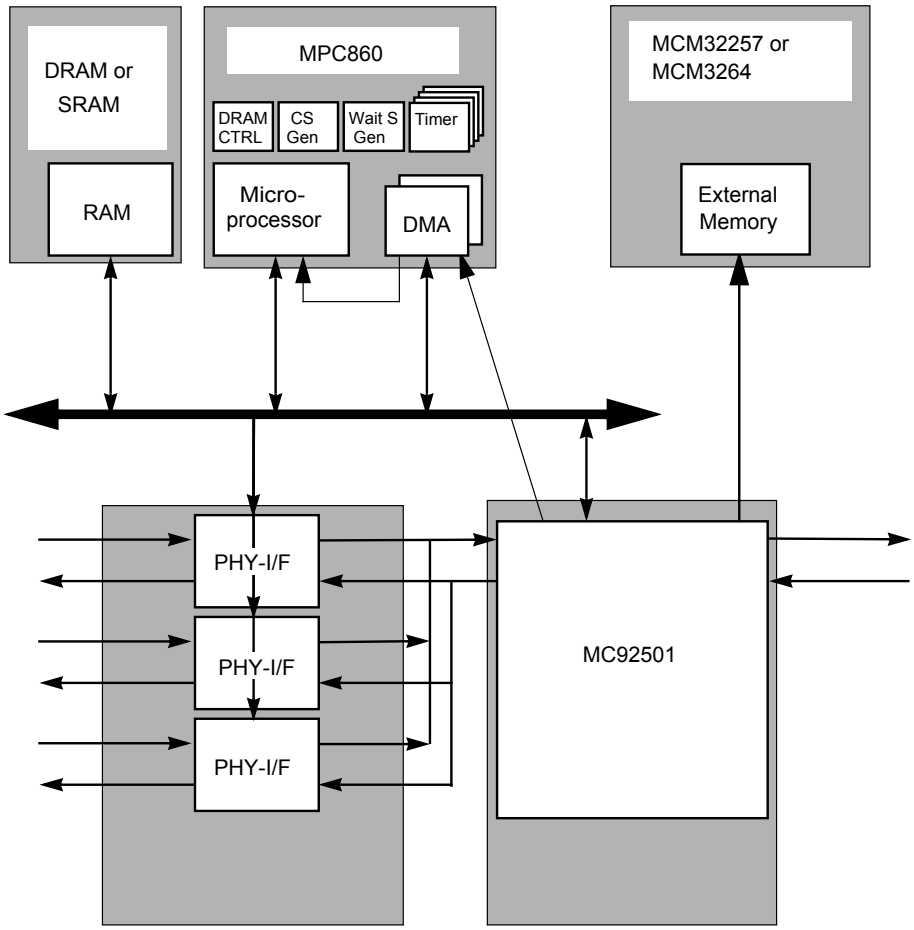


Figure E-2. Motorola's ATM Line Card Architecture for Multiple PHY Devices

Figure E-2 shows a possible line card architecture where multiple (low-speed) PHY devices are connected to a single MC92501.

E.4 DSLAM Access Network Architectures

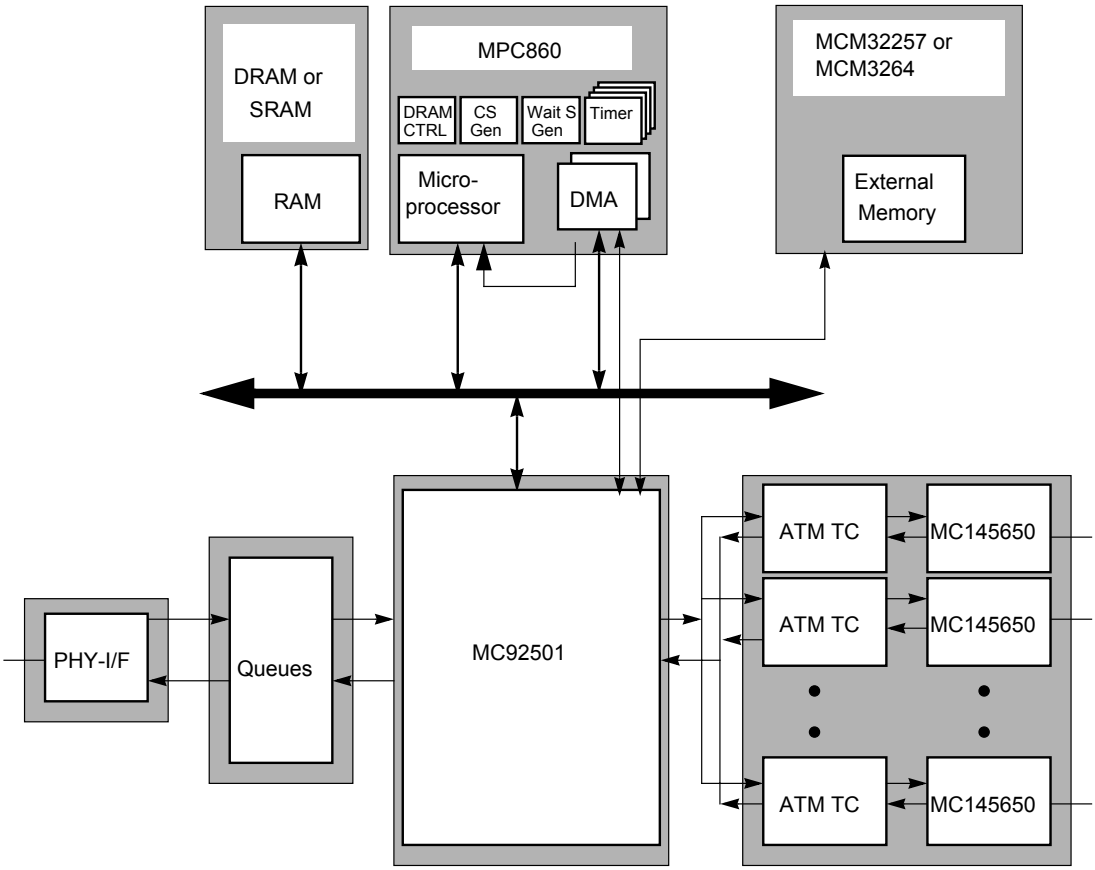


Figure E-3. Motorola's DSLAM Solution (Ingress Upstream/Egress Downstream)

Figure E-3 shows a possible access network architecture on which one MC92501 device interfaces a SONET PHY layer device on the network side and multiple PHY layer devices on the subscriber side. In this case the Ingress of the MC92501 corresponds to the upstream while the Egress corresponds to the downstream. The switch interface of the MC92501 is interfaced to the queues.

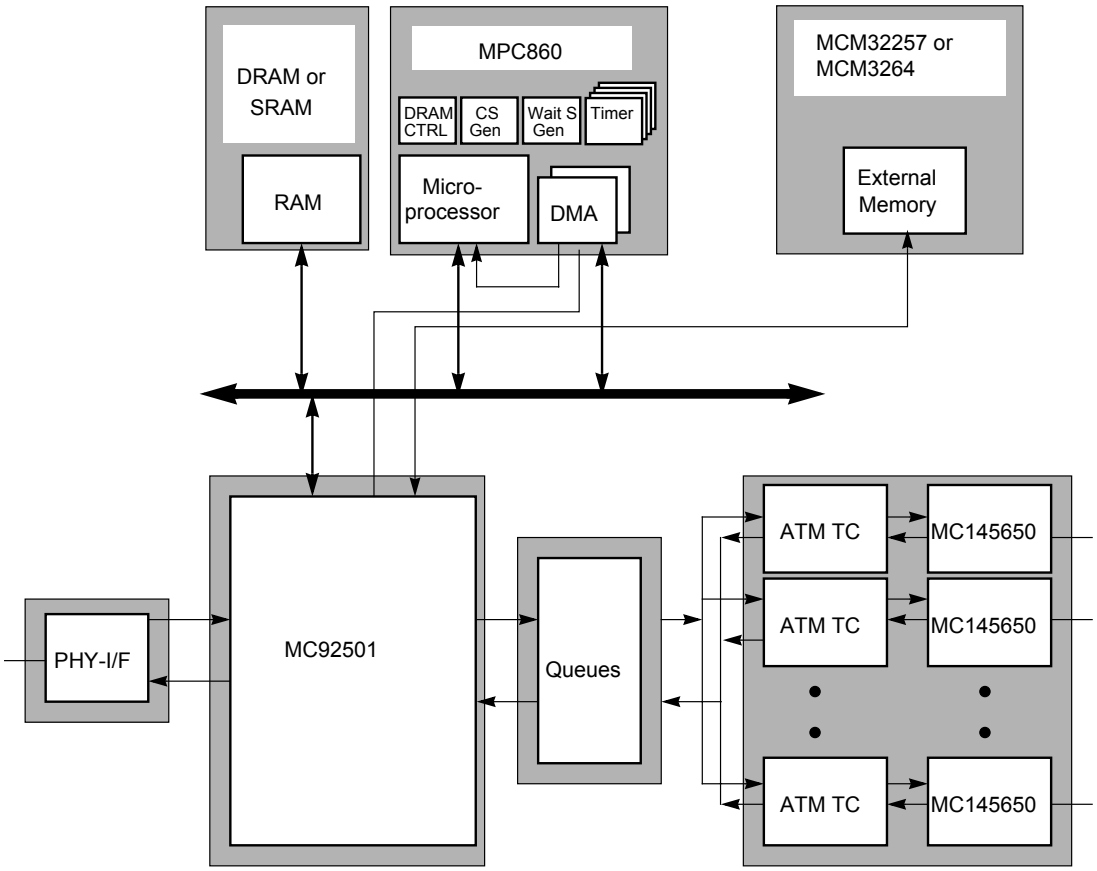


Figure E-4. Motorola's DSLAM Solution (Ingress Downstream/Egress Upstream)

Figure E-4 shows a possible access network architecture on which one MC92501 device interfaces a SONET PHY layer device on the network side and multiple PHY layer devices on the subscriber side, but in this case the Ingress of the MC92501 corresponds to the downstream while the Egress corresponds to the upstream. The switch interface of the MC92501 is again interfaced to the queues.

F

BSDL Code

```
-- M O T O R O L A   S S D T   J T A G   S O F T W A R E
-- BSDL File Generated: Sun Aug 17 15:15:16 1997
--
-- Revision History:
--


-------- Seperate This User Package File From The BSDL File --------

-- User Package File --

 package USER_PACKAGE_92501 is
    constant UC_8: CELL_INFO;
    constant UC_9: CELL_INFO;
 end USER_PACKAGE_92501;

 package body USER_PACKAGE_92501 is
    constant UC_8: CELL_INFO :=
        ((INTERNAL, EXTEST, X), (INTERNAL, SAMPLE, X),
         (OUTPUT2, EXTEST, X), (OUTPUT2, SAMPLE, X),
         (OUTPUT3, EXTEST, X), (OUTPUT3, SAMPLE, X));
    constant UC_9: CELL_INFO :=
        ((OUTPUT2, EXTEST, ZERO), (OUTPUT2, SAMPLE, PI),
         (OUTPUT3, EXTEST, ZERO), (OUTPUT3, SAMPLE, PI));
 end USER_PACKAGE_92501;


----------------------- Cut From Here ---------------------------------------
-

-- Bsdl File --


entity MC92501 is
        generic (PHYSICAL_PIN_MAP : string := "PBGA");

        port (   TRST:inbit;
                  TCK:inbit;
                  TMS:inbit;
                  TDI:inbit;
                  TDO:outbit;
                STXCLK:inbit;
               STXCLAV:bufferbit;
                STXSOC:inbit;
               STXPRTY:inbit;
              STXDATA0:inbit;
```

```
 STXDATA1:inbit;
 STXDATA2:inbit;
 STXDATA3:inbit;
 STXDATA4:inbit;
 STXDATA5:inbit;
 STXDATA6:inbit;
 STXDATA7:inbit;
  STXENB_:inbit;
   TXENB_:bufferbit;
  TXFULL_:inbit;
  TXCCLR_:inbit;
TXPHYIDV_:bufferbit;
   TXPRTY:bufferbit;
    TXSOC:bufferbit;
  TXDATA0:bufferbit;
  TXDATA1:bufferbit;
  TXDATA2:bufferbit;
  TXDATA3:bufferbit;
  TXDATA4:bufferbit;
  TXDATA5:bufferbit;
  TXDATA6:bufferbit;
  TXDATA7:bufferbit;
 TXPHYID0:bufferbit;
 TXPHYID1:bufferbit;
 TXPHYID2:bufferbit;
 TXPHYID3:bufferbit;
   MDATA0:inoutbit;
   MDATA1:inoutbit;
   MDATA2:inoutbit;
   MDATA3:inoutbit;
   MDATA4:inoutbit;
   MDATA5:inoutbit;
   MDATA6:inoutbit;
   MDATA7:inoutbit;
   MDATA8:inoutbit;
   MDATA9:inoutbit;
  MDATA10:inoutbit;
  MDATA11:inoutbit;
  MDATA12:inoutbit;
  MDATA13:inoutbit;
  MDATA14:inoutbit;
  MDATA15:inoutbit;
  MDATA16:inoutbit;
  MDATA17:inoutbit;
  MDATA18:inoutbit;
  MDATA19:inoutbit;
  MDATA20:inoutbit;
  MDATA21:inoutbit;
  MDATA22:inoutbit;
  MDATA23:inoutbit;
  MDATA24:inoutbit;
  MDATA25:inoutbit;
  MDATA26:inoutbit;
  MDATA27:inoutbit;
```

```
 MDATA28:inoutbit;
 MDATA29:inoutbit;
 MDATA30:inoutbit;
 MDATA31:inoutbit;
   MADD2:inbit;
   MADD3:inbit;
   MADD4:inbit;
   MADD5:inbit;
   MADD6:inbit;
   MADD7:inbit;
   MADD8:inbit;
   MADD9:inbit;
  MADD10:inbit;
  MADD11:inbit;
  MADD12:inbit;
  MADD13:inbit;
  MADD14:inbit;
  MADD15:inbit;
  MADD16:inbit;
  MADD17:inbit;
  MADD18:inbit;
  MADD19:inbit;
  MADD20:inbit;
  MADD21:inbit;
  MADD22:inbit;
  MADD23:inbit;
  MADD24:inbit;
  MADD25:inbit;
   MSEL_:inbit;
  MREQ0_:bufferbit;
  MREQ1_:bufferbit;
MDTACK0_:outbit;
   MINT_:bufferbit;
  MREQ2_:bufferbit;
    MCLK:inbit;
    MWR_:inbit;
   MWSH_:inbit;
   MWSL_:inbit;
    MDS_:inbit;
 SRXENB_:inbit;
SRXDATA0:outbit;
SRXDATA1:outbit;
SRXDATA2:outbit;
SRXDATA3:outbit;
SRXDATA4:outbit;
SRXDATA5:outbit;
SRXDATA6:outbit;
SRXDATA7:outbit;
  SRXCLK:inbit;
 SRXCLAV:outbit;
  SRXSOC:outbit;
 SRXPRTY:outbit;
MDTACK1_:outbit;
 RXADDR4:outbit;
```

```
   RXSOC:inbit;
  RXENB_:bufferbit;
RXPHYID0:inoutbit;
RXPHYID1:inoutbit;
RXPHYID2:inoutbit;
RXPHYID3:inoutbit;
  RXPRTY:inbit;
 RXDATA0:inbit;
 RXDATA1:inbit;
 RXDATA2:inbit;
 RXDATA3:inbit;
 RXDATA4:inbit;
 RXDATA5:inbit;
 RXDATA6:inbit;
 RXDATA7:inbit;
 EMDATA0:inoutbit;
 EMDATA1:inoutbit;
 EMDATA2:inoutbit;
 EMDATA3:inoutbit;
 EMDATA4:inoutbit;
 EMDATA5:inoutbit;
 EMDATA6:inoutbit;
 EMDATA7:inoutbit;
 EMDATA8:inoutbit;
 EMDATA9:inoutbit;
EMDATA10:inoutbit;
EMDATA11:inoutbit;
EMDATA12:inoutbit;
EMDATA13:inoutbit;
EMDATA14:inoutbit;
EMDATA15:inoutbit;
EMDATA16:inoutbit;
EMDATA17:inoutbit;
EMDATA18:inoutbit;
EMDATA19:inoutbit;
EMDATA20:inoutbit;
EMDATA21:inoutbit;
EMDATA22:inoutbit;
EMDATA23:inoutbit;
EMDATA24:inoutbit;
EMDATA25:inoutbit;
EMDATA26:inoutbit;
EMDATA27:inoutbit;
EMDATA28:inoutbit;
EMDATA29:inoutbit;
EMDATA30:inoutbit;
EMDATA31:inoutbit;
  EACEN_:bufferbit;
   EMWR_:bufferbit;
  EMADD2:bufferbit;
  EMADD3:bufferbit;
  EMADD4:bufferbit;
  EMADD5:bufferbit;
  EMADD6:bufferbit;
```

```
        EMADD7:bufferbit;
        EMADD8:bufferbit;
        EMADD9:bufferbit;
       EMADD10:bufferbit;
       EMADD11:bufferbit;
       EMADD12:bufferbit;
       EMADD13:bufferbit;
       EMADD14:bufferbit;
       EMADD15:bufferbit;
       EMADD16:bufferbit;
       EMADD17:bufferbit;
       EMADD18:bufferbit;
       EMADD19:bufferbit;
       EMADD20:bufferbit;
       EMADD21:bufferbit;
       EMADD22:bufferbit;
       EMADD23:bufferbit;
       EMBSH0_:bufferbit;
       EMBSH1_:bufferbit;
       EMBSH2_:bufferbit;
       EMBSH3_:bufferbit;
       EMBSL0_:bufferbit;
       EMBSL1_:bufferbit;
       EMBSL2_:bufferbit;
       EMBSL3_:bufferbit;
        AMODE0:linkagebit;
        AMODE1:linkagebit;
         ARST_:inbit;
     RXEMPTY_:inbit;
          VDD:linkagebit_vector(0 to 21);
          GND:linkagebit_vector(0 to 22);
         AVDD:linkagebit;
         AVSS:linkagebit;
         ACLK:linkagebit;
      TESTSEL:linkagebit;
       VCOCTL:linkagebit;
      TESTOUT:linkagebit;
         ENID:linkagebit;
           NC:linkagebit_vector(0 to 2));

use STD_1149_1_1994.all;
use USER_PACKAGE_92501.all;

attribute PIN_MAP of MC92501 : entity is PHYSICAL_PIN_MAP;

constant PBGA : PIN_MAP_STRING :=
 "GND:       (A1, D4, D8, D13, D17, H4, H17, N4, N17, U4, U8, U13, U17,"
              "A19, A20, B3, Y1, E4, W3, Y20, W20, T18, V16), " &
 "ACLK:        A2, " &
 "MADD16:      A3, " &
 "MADD13:      A4, " &
 "MADD9:       A5, " &
 "MADD6:       A6, " &
 "MADD4:       A7, " &
```

```
"MSEL_:        A8, " &
"MINT_:        A9, " &
"MWSH_:        A10, " &
"MWSL_:        A11, " &
"SRXDATA7:     A12, " &
"SRXDATA3:     A13, " &
"SRXDATA0:     A14, " &
"SRXSOC:       A15, " &
"RXSOC:        A16, " &
"RXPHYID3:     A17, " &
"RXPHYID2:     A18, " &
"AVSS:         B1, " &
"TESTSEL:      B2, " &
"MADD15:       B4, " &
"MADD12:       B5, " &
"MADD8:        B6, " &
"MADD5:        B7, " &
"MADD2:        B8, " &
"MDTACK0_:     B9, " &
"MWR_:         B10, " &
"SRXENB_:      B11, " &
"SRXDATA6:     B12, " &
"SRXDATA2:     B13, " &
"SRXCLK:       B14, " &
"SRXPRTY:      B15, " &
"RXENB_:       B16, " &
"RXPRTY:       B17, " &
"VDD:         (V14, D11, D15, F4, F17, K4, L17, R4, R17, U6, U10, D6,"
               "U15, B19, B18, C4, D3, W2, Y2, V19, W19, P18), " &
"RXDATA6:      B20, " &
"MADD18:       C1, " &
"AVDD:         C2, " &
"TESTOUT:      C3, " &
"MADD14:       C5, " &
"MADD11:       C6, " &
"MADD7:        C7, " &
"MADD3:        C8, " &
"MREQ1_:       C9, " &
"MCLK:         C10, " &
"MDS_:         C11, " &
"SRXDATA5:     C12, " &
"SRXDATA1:     C13, " &
"SRXCLAV:      C14, " &
"RXADDR4:      C15, " &
"RXEMPTY_:     C16, " &
"RXPHYID0:     C17, " &
"RXDATA7:      C18, " &
"RXDATA5:      C19, " &
"RXDATA2:      C20, " &
"MADD19:       D1, " &
"VCOCTL:       D2, " &
"MADD17:       D5, " &
"MADD10:       D7, " &
"MREQ0_:       D9, " &
```

```
"MREQ2_:      D10, " &
"SRXDATA4:    D12, " &
"MDTACK1_:    D14, " &
"RXPHYID1:    D16, " &
"RXDATA4:     D18, " &
"RXDATA1:     D19, " &
"EMDATA31:    D20, " &
"MADD22:      E1, " &
"MADD21:      E2, " &
"MADD20:      E3, " &
"RXDATA3:     E17, " &
"RXDATA0:     E18, " &
"EMDATA30:    E19, " &
"EMDATA27:    E20, " &
"MDATA0:      F1, " &
"MADD25:      F2, " &
"MADD23:      F3, " &
"EMDATA29:    F18, " &
"EMDATA26:    F19, " &
"EMDATA24:    F20, " &
"MDATA3:      G1, " &
"MDATA2:      G2, " &
"MDATA1:      G3, " &
"MADD24:      G4, " &
"EMDATA28:    G17, " &
"EMDATA25:    G18, " &
"EMDATA23:    G19, " &
"EMDATA22:    G20, " &
"MDATA6:      H1, " &
"MDATA5:      H2, " &
"MDATA4:      H3, " &
"EMDATA21:    H18, " &
"EMDATA20:    H19, " &
"EMDATA19:    H20, " &
"MDATA10:     J1, " &
"MDATA9:      J2, " &
"MDATA8:      J3, " &
"MDATA7:      J4, " &
"EACEN_:      J17, " &
"EMWR_:       J18, " &
"EMDATA18:    J19, " &
"EMDATA17:    J20, " &
"MDATA13:     K1, " &
"MDATA11:     K2, " &
"MDATA12:     K3, " &
"EMDATA16:    K17, " &
"EMDATA15:    K18, " &
"EMDATA14:    K19, " &
"EMDATA13:    K20, " &
"MDATA14:     L1, " &
"MDATA15:     L2, " &
"MDATA16:     L3, " &
"MDATA17:     L4, " &
"EMDATA11:    L18, " &
```

```
"EMDATA10:    L19, " &
"EMDATA12:    L20, " &
"MDATA18:     M1, " &
"MDATA19:     M2, " &
"MDATA20:     M3, " &
"MDATA21:     M4, " &
"EMDATA6:     M17, " &
"EMDATA7:     M18, " &
"EMDATA8:     M19, " &
"EMDATA9:     M20, " &
"MDATA22:     N1, " &
"MDATA23:     N2, " &
"MDATA24:     N3, " &
"EMDATA3:     N18, " &
"EMDATA4:     N19, " &
"EMDATA5:     N20, " &
"MDATA25:     P1, " &
"MDATA26:     P2, " &
"MDATA28:     P3, " &
"MDATA31:     P4, " &
"EMADD22:     P17, " &
"EMDATA1:     P19, " &
"EMDATA2:     P20, " &
"MDATA27:     R1, " &
"MDATA29:     R2, " &
"TXPHYID0:    R3, " &
"EMADD21:     R18, " &
"EMADD23:     R19, " &
"EMDATA0:     R20, " &
"MDATA30:     T1, " &
"TXPHYID1:    T2, " &
"TXPHYID3:    T3, " &
"TXDATA2:     T4, " &
"EMADD16:     T17, " &
"EMADD19:     T19, " &
"EMADD20:     T20, " &
"TXPHYID2:    U1, " &
"TXDATA0:     U2, " &
"TXDATA3:     U3, " &
"TXPRTY:      U5, " &
"STXDATA1:    U7, " &
"STXCLAV:     U9, " &
"AMODE1:      U11, " &
"EMBSL1_:     U12, " &
"EMADD3:      U14, " &
"EMADD9:      U16, " &
"EMADD15:     U18, " &
"EMADD14:     U19, " &
"EMADD18:     U20, " &
"TXDATA1:     V1, " &
"TXDATA4:     V2, " &
"TXDATA6:     V3, " &
"TXSOC:       V4, " &
"TXFULL_:     V5, " &
```

```
    "STXDATA0:     V6, " &
    "STXDATA4:     V7, " &
    "STXDATA7:     V8, " &
    "STXCLK:       V9, " &
    "TMS:          V10, " &
    "AMODE0:       V11, " &
    "EMBSL2_:      V12, " &
    "EMBSH3_:      V13, " &
    "EMADD4:       V15, " &
    "EMADD10:      V17, " &
    "EMADD13:      V18, " &
    "EMADD17:      V20, " &
    "TXDATA5:      W1, " &
    "TXDATA7:      W4, " &
    "TXENB_:       W5, " &
    "STXDATA2:     W6, " &
    "STXDATA5:     W7, " &
    "STXPRTY:      W8, " &
    "ENID:         W9, " &
    "TDO:          W10, " &
    "ARST_:        W11, " &
    "EMBSL3_:      W12, " &
    "NC:          (W13, W15, Y12), " &
    "EMBSH1_:      W14, " &
    "EMADD5:       W16, " &
    "EMADD7:       W17, " &
    "EMADD11:      W18, " &
    "TXPHYIDV_:    Y3, " &
    "TXCCLR_:      Y4, " &
    "STXENB_:      Y5, " &
    "STXDATA3:     Y6, " &
    "STXDATA6:     Y7, " &
    "STXSOC:       Y8, " &
    "TDI:          Y9, " &
    "TRST:         Y10, " &
    "TCK:          Y11, " &
    "EMBSL0_:      Y13, " &
    "EMBSH2_:      Y14, " &
    "EMBSH0_:      Y15, " &
    "EMADD2:       Y16, " &
    "EMADD6:       Y17, " &
    "EMADD8:       Y18, " &
    "EMADD12:      Y19 ";

attribute TAP_SCAN_IN    of   TDI : signal is true;
attribute TAP_SCAN_OUT   of   TDO : signal is true;
attribute TAP_SCAN_MODE  of   TMS : signal is true;
attribute TAP_SCAN_RESET of  TRST : signal is true;
attribute TAP_SCAN_CLOCK of   TCK : signal is (20.0e6, BOTH);

attribute INSTRUCTION_LENGTH of MC92501 : entity is 4;

attribute INSTRUCTION_OPCODE of MC92501 : entity is
   "EXTEST  (0000)," &
```

```
            "SAMPLE   (0010)," &
            "IDCODE   (0001)," &
            "CLAMP    (0100)," &
            "HIGHZ    (0011)," &
            "EXTEST1  (1000)," &
            "IDCODE1  (1001)," &
            "SAMPLE1  (1010)," &
            "HIGHZ1   (1011)," &
            "CLAMP1   (1100)," &
            "BYPASS   (1111)";

        attribute INSTRUCTION_CAPTURE of MC92501 : entity is "0001";
        attribute  INSTRUCTION_PRIVATE  of  MC92501  :  entity  is  "EXTEST1,
IDCODE1,"
                  "SAMPLE1, HIGHZ1, CLAMP1";

        attribute IDCODE_REGISTER   of MC92501 : entity is
            "0000"          & -- version
            "010100"        & -- manufacturer's use
            "0000111011"    & -- sequence number
            "00000001110"   & -- manufacturer identity
            "1";              -- 1149.1 requirement


        attribute REGISTER_ACCESS of MC92501 : entity is
            "BOUNDARY   (EXTEST1,SAMPLE1)," &
            "BYPASS    (HIGHZ1,CLAMP1)," &
            "IDCODE    (IDCODE1)" ;

        attribute BOUNDARY_LENGTH of MC92501 : entity is 361;

        attribute BOUNDARY_REGISTER of MC92501 : entity is
        -- num    cell   port          func     safe [ccell dis rslt]
            "0     (BC_1, *,          control,  0)," &
            "1     (BC_1, *,          control,  0)," &
            "2     (BC_1, *,          control,  0)," &
            "3     (BC_1, *,          control,  0)," &
            "4     (BC_1, *,          control,  0)," &
            "5     (BC_1, *,          control,  0)," &
            "6     (BC_4, ARST_,      input,    X)," &
            "7     (UC_9, EMBSL3_,    output2,  X)," &
            "8     (UC_9, EMBSL2_,    output2,  X)," &
            "9     (UC_9, EMBSL1_,    output2,  X)," &
            "10    (UC_9, EMBSL0_,    output2,  X)," &
            "11    (UC_9, EMBSH3_,    output2,  X)," &
            "12    (UC_9, EMBSH2_,    output2,  X)," &
            "13    (UC_9, EMBSH1_,    output2,  X)," &
            "14    (UC_9, EMBSH0_,    output2,  X)," &
            "15    (UC_8, EMADD2,     output2,  X)," &
            "16    (BC_2, *,          internal, X)," &
            "17    (UC_8, EMADD3,     output2,  X)," &
            "18    (BC_2, *,          internal, X)," &
            "19    (UC_8, EMADD4,     output2,  X)," &
        -- num    cell   port          func     safe [ccell dis rslt]
```

```
   "20    (BC_2, *,          internal, X)," &
   "21    (UC_8, EMADD5,     output2,  X)," &
   "22    (BC_2, *,          internal, X)," &
   "23    (UC_8, EMADD6,     output2,  X)," &
   "24    (BC_2, *,          internal, X)," &
   "25    (UC_8, EMADD7,     output2,  X)," &
   "26    (BC_2, *,          internal, X)," &
   "27    (UC_8, EMADD8,     output2,  X)," &
   "28    (BC_2, *,          internal, X)," &
   "29    (UC_8, EMADD9,     output2,  X)," &
   "30    (BC_2, *,          internal, X)," &
   "31    (UC_8, EMADD10,    output2,  X)," &
   "32    (BC_2, *,          internal, X)," &
   "33    (UC_8, EMADD11,    output2,  X)," &
   "34    (BC_2, *,          internal, X)," &
   "35    (UC_8, EMADD12,    output2,  X)," &
   "36    (BC_2, *,          internal, X)," &
   "37    (UC_8, EMADD13,    output2,  X)," &
   "38    (BC_2, *,          internal, X)," &
   "39    (UC_8, EMADD14,    output2,  X)," &
-- num    cell   port         func     safe [ccell dis rslt]
   "40    (BC_2, *,          internal, X)," &
   "41    (UC_8, EMADD15,    output2,  X)," &
   "42    (BC_2, *,          internal, X)," &
   "43    (UC_8, EMADD16,    output2,  X)," &
   "44    (BC_2, *,          internal, X)," &
   "45    (UC_8, EMADD17,    output2,  X)," &
   "46    (BC_2, *,          internal, X)," &
   "47    (UC_8, EMADD18,    output2,  X)," &
   "48    (BC_2, *,          internal, X)," &
   "49    (UC_8, EMADD19,    output2,  X)," &
   "50    (BC_2, *,          internal, X)," &
   "51    (UC_8, EMADD20,    output2,  X)," &
   "52    (BC_2, *,          internal, X)," &
   "53    (UC_8, EMADD21,    output2,  X)," &
   "54    (BC_2, *,          internal, X)," &
   "55    (UC_8, EMADD22,    output2,  X)," &
   "56    (BC_2, *,          internal, X)," &
   "57    (UC_8, EMADD23,    output2,  X)," &
   "58    (BC_2, *,          internal, X)," &
   "59    (UC_8, EMDATA0,    output3,  X,   1,   0,   Z)," &
-- num    cell   port         func     safe [ccell dis rslt]
   "60    (BC_2, EMDATA0,    input,    X)," &
   "61    (UC_8, EMDATA1,    output3,  X,   1,   0,   Z)," &
   "62    (BC_2, EMDATA1,    input,    X)," &
   "63    (UC_8, EMDATA2,    output3,  X,   1,   0,   Z)," &
   "64    (BC_2, EMDATA2,    input,    X)," &
   "65    (UC_8, EMDATA3,    output3,  X,   1,   0,   Z)," &
   "66    (BC_2, EMDATA3,    input,    X)," &
   "67    (UC_8, EMDATA4,    output3,  X,   1,   0,   Z)," &
   "68    (BC_2, EMDATA4,    input,    X)," &
   "69    (UC_8, EMDATA5,    output3,  X,   1,   0,   Z)," &
   "70    (BC_2, EMDATA5,    input,    X)," &
   "71    (UC_8, EMDATA6,    output3,  X,   1,   0,   Z)," &
```

```
    "72    (BC_2, EMDATA6,    input,    X)," &
    "73    (UC_8, EMDATA7,    output3,  X,   1,   0,   Z)," &
    "74    (BC_2, EMDATA7,    input,    X)," &
    "75    (UC_8, EMDATA8,    output3,  X,   1,   0,   Z)," &
    "76    (BC_2, EMDATA8,    input,    X)," &
    "77    (UC_8, EMDATA9,    output3,  X,   1,   0,   Z)," &
    "78    (BC_2, EMDATA9,    input,    X)," &
    "79    (UC_8, EMDATA10,   output3,  X,   1,   0,   Z)," &
 -- num    cell   port          func     safe [ccell dis rslt]
    "80    (BC_2, EMDATA10,   input,    X)," &
    "81    (UC_8, EMDATA11,   output3,  X,   1,   0,   Z)," &
    "82    (BC_2, EMDATA11,   input,    X)," &
    "83    (UC_8, EMDATA12,   output3,  X,   1,   0,   Z)," &
    "84    (BC_2, EMDATA12,   input,    X)," &
    "85    (UC_8, EMDATA13,   output3,  X,   1,   0,   Z)," &
    "86    (BC_2, EMDATA13,   input,    X)," &
    "87    (UC_8, EMDATA14,   output3,  X,   1,   0,   Z)," &
    "88    (BC_2, EMDATA14,   input,    X)," &
    "89    (UC_8, EMDATA15,   output3,  X,   1,   0,   Z)," &
    "90    (BC_2, EMDATA15,   input,    X)," &
    "91    (UC_8, EMDATA16,   output3,  X,   1,   0,   Z)," &
    "92    (BC_2, EMDATA16,   input,    X)," &
    "93    (UC_8, EMDATA17,   output3,  X,   1,   0,   Z)," &
    "94    (BC_2, EMDATA17,   input,    X)," &
    "95    (UC_8, EMDATA18,   output3,  X,   1,   0,   Z)," &
    "96    (BC_2, EMDATA18,   input,    X)," &
    "97    (UC_9, EMWR_,      output2,  X)," &
    "98    (UC_9, EACEN_,     output2,  X)," &
    "99    (UC_8, EMDATA19,   output3,  X,   1,   0,   Z)," &
 -- num    cell   port          func     safe [ccell dis rslt]
    "100   (BC_2, EMDATA19,   input,    X)," &
    "101   (UC_8, EMDATA20,   output3,  X,   1,   0,   Z)," &
    "102   (BC_2, EMDATA20,   input,    X)," &
    "103   (UC_8, EMDATA21,   output3,  X,   1,   0,   Z)," &
    "104   (BC_2, EMDATA21,   input,    X)," &
    "105   (UC_8, EMDATA22,   output3,  X,   1,   0,   Z)," &
    "106   (BC_2, EMDATA22,   input,    X)," &
    "107   (UC_8, EMDATA23,   output3,  X,   1,   0,   Z)," &
    "108   (BC_2, EMDATA23,   input,    X)," &
    "109   (UC_8, EMDATA24,   output3,  X,   1,   0,   Z)," &
    "110   (BC_2, EMDATA24,   input,    X)," &
    "111   (UC_8, EMDATA25,   output3,  X,   1,   0,   Z)," &
    "112   (BC_2, EMDATA25,   input,    X)," &
    "113   (UC_8, EMDATA26,   output3,  X,   1,   0,   Z)," &
    "114   (BC_2, EMDATA26,   input,    X)," &
    "115   (UC_8, EMDATA27,   output3,  X,   1,   0,   Z)," &
    "116   (BC_2, EMDATA27,   input,    X)," &
    "117   (UC_8, EMDATA28,   output3,  X,   1,   0,   Z)," &
    "118   (BC_2, EMDATA28,   input,    X)," &
    "119   (UC_8, EMDATA29,   output3,  X,   1,   0,   Z)," &
 -- num    cell   port          func     safe [ccell dis rslt]
    "120   (BC_2, EMDATA29,   input,    X)," &
    "121   (UC_8, EMDATA30,   output3,  X,   1,   0,   Z)," &
    "122   (BC_2, EMDATA30,   input,    X)," &
```

```
    "123   (UC_8, EMDATA31,   output3,  X,   1,   0,   Z)," &
    "124   (BC_2, EMDATA31,   input,    X)," &
    "125   (UC_8, *,          internal, 0)," &
    "126   (BC_2, RXDATA0,    input,    X)," &
    "127   (UC_8, *,          internal, 0)," &
    "128   (BC_2, RXDATA1,    input,    X)," &
    "129   (UC_8, *,          internal, 0)," &
    "130   (BC_2, RXDATA2,    input,    X)," &
    "131   (UC_8, *,          internal, 0)," &
    "132   (BC_2, RXDATA3,    input,    X)," &
    "133   (UC_8, *,          internal, 0)," &
    "134   (BC_2, RXDATA4,    input,    X)," &
    "135   (UC_8, *,          internal, 0)," &
    "136   (BC_2, RXDATA5,    input,    X)," &
    "137   (UC_8, *,          internal, 0)," &
    "138   (BC_2, RXDATA6,    input,    X)," &
    "139   (UC_8, *,          internal, 0)," &
--  num    cell   port         func     safe [ccell dis rslt]
    "140   (BC_2, RXDATA7,    input,    X)," &
    "141   (UC_8, *,          internal, 0)," &
    "142   (BC_2, RXPRTY,     input,    X)," &
    "143   (UC_8, RXPHYID0,   output3,  X,   3,   0,   Z)," &
    "144   (BC_2, RXPHYID0,   input,    X)," &
    "145   (UC_8, RXPHYID1,   output3,  X,   3,   0,   Z)," &
    "146   (BC_2, RXPHYID1,   input,    X)," &
    "147   (UC_8, RXPHYID2,   output3,  X,   3,   0,   Z)," &
    "148   (BC_2, RXPHYID2,   input,    X)," &
    "149   (UC_8, RXPHYID3,   output3,  X,   3,   0,   Z)," &
    "150   (BC_2, RXPHYID3,   input,    X)," &
    "151   (UC_8, *,          internal, 0)," &
    "152   (BC_2, RXEMPTY_,   input,    X)," &
    "153   (UC_8, RXENB_,     output2,  X)," &
    "154   (BC_2, *,          internal, X)," &
    "155   (UC_8, *,          internal, 0)," &
    "156   (BC_2, RXSOC,      input,    X)," &
    "157   (UC_9, RXADDR4,    output3,  X,   3,   0,   Z)," &
    "158   (UC_9, MDTACK1_,   output3,  X,   0,   0,   Z)," &
    "159   (UC_8, SRXPRTY,    output3,  X,   2,   0,   Z)," &
--  num    cell   port         func     safe [ccell dis rslt]
    "160   (UC_8, *,          internal, 0)," &
    "161   (UC_8, SRXSOC,     output3,  X,   2,   0,   Z)," &
    "162   (UC_8, *,          internal, 0)," &
    "163   (UC_8, SRXCLAV,    output3,  X,   2,   0,   Z)," &
    "164   (UC_8, *,          internal, 0)," &
    "165   (BC_4, SRXCLK,     input,    X)," &
    "166   (UC_8, SRXDATA0,   output3,  X,   2,   0,   Z)," &
    "167   (UC_8, *,          internal, 0)," &
    "168   (UC_8, SRXDATA1,   output3,  X,   2,   0,   Z)," &
    "169   (UC_8, *,          internal, 0)," &
    "170   (UC_8, SRXDATA2,   output3,  X,   2,   0,   Z)," &
    "171   (UC_8, *,          internal, 0)," &
    "172   (UC_8, SRXDATA3,   output3,  X,   2,   0,   Z)," &
    "173   (UC_8, *,          internal, 0)," &
    "174   (UC_8, SRXDATA4,   output3,  X,   2,   0,   Z)," &
```

```
   "175   (UC_8, *,          internal, 0)," &
   "176   (UC_8, SRXDATA5,   output3,  X,   2,   0,   Z)," &
   "177   (UC_8, *,          internal, 0)," &
   "178   (UC_8, SRXDATA6,   output3,  X,   2,   0,   Z)," &
   "179   (UC_8, *,          internal, 0)," &
-- num    cell   port          func     safe [ccell dis rslt]
   "180   (UC_8, SRXDATA7,   output3,  X,   2,   0,   Z)," &
   "181   (UC_8, *,          internal, 0)," &
   "182   (BC_2, SRXENB_,    input,    X)," &
   "183   (BC_2, MDS_,       input,    X)," &
   "184   (BC_2, MWSL_,      input,    X)," &
   "185   (BC_2, MWSH_,      input,    X)," &
   "186   (BC_2, MWR_,       input,    X)," &
   "187   (BC_4, MCLK,       input,    X)," &
   "188   (UC_9, MREQ2_,     output2,  X)," &
   "189   (UC_9, MINT_,      output2,  X)," &
   "190   (UC_9, MDTACK0_,   output3,  X,   4,   0,   Z)," &
   "191   (UC_9, MREQ1_,     output2,  X)," &
   "192   (UC_9, MREQ0_,     output2,  X)," &
   "193   (BC_2, MSEL_,      input,    X)," &
   "194   (UC_8, *,          internal, 0)," &
   "195   (BC_2, MADD2,      input,    X)," &
   "196   (UC_8, *,          internal, 0)," &
   "197   (BC_2, MADD3,      input,    X)," &
   "198   (UC_8, *,          internal, 0)," &
   "199   (BC_2, MADD4,      input,    X)," &
-- num    cell   port          func     safe [ccell dis rslt]
   "200   (UC_8, *,          internal, 0)," &
   "201   (BC_2, MADD5,      input,    X)," &
   "202   (UC_8, *,          internal, 0)," &
   "203   (BC_2, MADD6,      input,    X)," &
   "204   (UC_8, *,          internal, 0)," &
   "205   (BC_2, MADD7,      input,    X)," &
   "206   (UC_8, *,          internal, 0)," &
   "207   (BC_2, MADD8,      input,    X)," &
   "208   (UC_8, *,          internal, 0)," &
   "209   (BC_2, MADD9,      input,    X)," &
   "210   (UC_8, *,          internal, 0)," &
   "211   (BC_2, MADD10,     input,    X)," &
   "212   (UC_8, *,          internal, 0)," &
   "213   (BC_2, MADD11,     input,    X)," &
   "214   (UC_8, *,          internal, 0)," &
   "215   (BC_2, MADD12,     input,    X)," &
   "216   (UC_8, *,          internal, 0)," &
   "217   (BC_2, MADD13,     input,    X)," &
   "218   (UC_8, *,          internal, 0)," &
   "219   (BC_2, MADD14,     input,    X)," &
-- num    cell   port          func     safe [ccell dis rslt]
   "220   (UC_8, *,          internal, 0)," &
   "221   (BC_2, MADD15,     input,    X)," &
   "222   (UC_8, *,          internal, 0)," &
   "223   (BC_2, MADD16,     input,    X)," &
   "224   (UC_8, *,          internal, 0)," &
   "225   (BC_2, MADD17,     input,    X)," &
```

```
   "226   (UC_8, *,          internal, 0)," &
   "227   (BC_2, MADD18,     input,    X)," &
   "228   (UC_8, *,          internal, 0)," &
   "229   (BC_2, MADD19,     input,    X)," &
   "230   (UC_8, *,          internal, 0)," &
   "231   (BC_2, MADD20,     input,    X)," &
   "232   (UC_8, *,          internal, 0)," &
   "233   (BC_2, MADD21,     input,    X)," &
   "234   (UC_8, *,          internal, 0)," &
   "235   (BC_2, MADD22,     input,    X)," &
   "236   (UC_8, *,          internal, 0)," &
   "237   (BC_2, MADD23,     input,    X)," &
   "238   (UC_8, *,          internal, 0)," &
   "239   (BC_2, MADD24,     input,    X)," &
-- num    cell   port         func     safe [ccell dis rslt]
   "240   (UC_8, *,          internal, 0)," &
   "241   (BC_2, MADD25,     input,    X)," &
   "242   (UC_8, MDATA0,     output3,  X,   5,   0,   Z)," &
   "243   (BC_2, MDATA0,     input,    X)," &
   "244   (UC_8, MDATA1,     output3,  X,   5,   0,   Z)," &
   "245   (BC_2, MDATA1,     input,    X)," &
   "246   (UC_8, MDATA2,     output3,  X,   5,   0,   Z)," &
   "247   (BC_2, MDATA2,     input,    X)," &
   "248   (UC_8, MDATA3,     output3,  X,   5,   0,   Z)," &
   "249   (BC_2, MDATA3,     input,    X)," &
   "250   (UC_8, MDATA4,     output3,  X,   5,   0,   Z)," &
   "251   (BC_2, MDATA4,     input,    X)," &
   "252   (UC_8, MDATA5,     output3,  X,   5,   0,   Z)," &
   "253   (BC_2, MDATA5,     input,    X)," &
   "254   (UC_8, MDATA6,     output3,  X,   5,   0,   Z)," &
   "255   (BC_2, MDATA6,     input,    X)," &
   "256   (UC_8, MDATA7,     output3,  X,   5,   0,   Z)," &
   "257   (BC_2, MDATA7,     input,    X)," &
   "258   (UC_8, MDATA8,     output3,  X,   5,   0,   Z)," &
   "259   (BC_2, MDATA8,     input,    X)," &
-- num    cell   port         func     safe [ccell dis rslt]
   "260   (UC_8, MDATA9,     output3,  X,   5,   0,   Z)," &
   "261   (BC_2, MDATA9,     input,    X)," &
   "262   (UC_8, MDATA10,    output3,  X,   5,   0,   Z)," &
   "263   (BC_2, MDATA10,    input,    X)," &
   "264   (UC_8, MDATA11,    output3,  X,   5,   0,   Z)," &
   "265   (BC_2, MDATA11,    input,    X)," &
   "266   (UC_8, MDATA12,    output3,  X,   5,   0,   Z)," &
   "267   (BC_2, MDATA12,    input,    X)," &
   "268   (UC_8, MDATA13,    output3,  X,   5,   0,   Z)," &
   "269   (BC_2, MDATA13,    input,    X)," &
   "270   (UC_8, MDATA14,    output3,  X,   5,   0,   Z)," &
   "271   (BC_2, MDATA14,    input,    X)," &
   "272   (UC_8, MDATA15,    output3,  X,   5,   0,   Z)," &
   "273   (BC_2, MDATA15,    input,    X)," &
   "274   (UC_8, MDATA16,    output3,  X,   5,   0,   Z)," &
   "275   (BC_2, MDATA16,    input,    X)," &
   "276   (UC_8, MDATA17,    output3,  X,   5,   0,   Z)," &
   "277   (BC_2, MDATA17,    input,    X)," &
```

```
   "278   (UC_8, MDATA18,    output3,  X,   5,   0,   Z)," &
   "279   (BC_2, MDATA18,    input,    X)," &
-- num    cell   port          func     safe [ccell dis rslt]
   "280   (UC_8, MDATA19,    output3,  X,   5,   0,   Z)," &
   "281   (BC_2, MDATA19,    input,    X)," &
   "282   (UC_8, MDATA20,    output3,  X,   5,   0,   Z)," &
   "283   (BC_2, MDATA20,    input,    X)," &
   "284   (UC_8, MDATA21,    output3,  X,   5,   0,   Z)," &
   "285   (BC_2, MDATA21,    input,    X)," &
   "286   (UC_8, MDATA22,    output3,  X,   5,   0,   Z)," &
   "287   (BC_2, MDATA22,    input,    X)," &
   "288   (UC_8, MDATA23,    output3,  X,   5,   0,   Z)," &
   "289   (BC_2, MDATA23,    input,    X)," &
   "290   (UC_8, MDATA24,    output3,  X,   5,   0,   Z)," &
   "291   (BC_2, MDATA24,    input,    X)," &
   "292   (UC_8, MDATA25,    output3,  X,   5,   0,   Z)," &
   "293   (BC_2, MDATA25,    input,    X)," &
   "294   (UC_8, MDATA26,    output3,  X,   5,   0,   Z)," &
   "295   (BC_2, MDATA26,    input,    X)," &
   "296   (UC_8, MDATA27,    output3,  X,   5,   0,   Z)," &
   "297   (BC_2, MDATA27,    input,    X)," &
   "298   (UC_8, MDATA28,    output3,  X,   5,   0,   Z)," &
   "299   (BC_2, MDATA28,    input,    X)," &
-- num    cell   port          func     safe [ccell dis rslt]
   "300   (UC_8, MDATA29,    output3,  X,   5,   0,   Z)," &
   "301   (BC_2, MDATA29,    input,    X)," &
   "302   (UC_8, MDATA30,    output3,  X,   5,   0,   Z)," &
   "303   (BC_2, MDATA30,    input,    X)," &
   "304   (UC_8, MDATA31,    output3,  X,   5,   0,   Z)," &
   "305   (BC_2, MDATA31,    input,    X)," &
   "306   (UC_9, TXPHYID0,   output2,  X)," &
   "307   (UC_9, TXPHYID1,   output2,  X)," &
   "308   (UC_9, TXPHYID2,   output2,  X)," &
   "309   (UC_9, TXPHYID3,   output2,  X)," &
   "310   (UC_8, TXDATA0,    output2,  X)," &
   "311   (BC_2, *,          internal, X)," &
   "312   (UC_8, TXDATA1,    output2,  X)," &
   "313   (BC_2, *,          internal, X)," &
   "314   (UC_8, TXDATA2,    output2,  X)," &
   "315   (BC_2, *,          internal, X)," &
   "316   (UC_8, TXDATA3,    output2,  X)," &
   "317   (BC_2, *,          internal, X)," &
   "318   (UC_8, TXDATA4,    output2,  X)," &
   "319   (BC_2, *,          internal, X)," &
-- num    cell   port          func     safe [ccell dis rslt]
   "320   (UC_8, TXDATA5,    output2,  X)," &
   "321   (BC_2, *,          internal, X)," &
   "322   (UC_8, TXDATA6,    output2,  X)," &
   "323   (BC_2, *,          internal, X)," &
   "324   (UC_8, TXDATA7,    output2,  X)," &
   "325   (BC_2, *,          internal, X)," &
   "326   (UC_8, TXSOC,      output2,  X)," &
   "327   (BC_2, *,          internal, X)," &
   "328   (UC_8, TXPRTY,     output2,  X)," &
```

```
            "329   (BC_2, *,          internal, X)," &
            "330   (UC_9, TXPHYIDV_,  output2,  X)," &
            "331   (UC_8, *,          internal, 0)," &
            "332   (BC_2, TXCCLR_,    input,    X)," &
            "333   (BC_2, TXFULL_,    input,    X)," &
            "334   (UC_8, TXENB_,     output2,  X)," &
            "335   (UC_8, *,          internal, 0)," &
            "336   (UC_8, *,          internal, 0)," &
            "337   (BC_2, STXENB_,    input,    X)," &
            "338   (UC_8, *,          internal, 0)," &
            "339   (BC_2, STXDATA0,   input,    X)," &
         -- num    cell   port         func     safe [ccell dis rslt]
            "340   (UC_8, *,          internal, 0)," &
            "341   (BC_2, STXDATA1,   input,    X)," &
            "342   (UC_8, *,          internal, 0)," &
            "343   (BC_2, STXDATA2,   input,    X)," &
            "344   (UC_8, *,          internal, 0)," &
            "345   (BC_2, STXDATA3,   input,    X)," &
            "346   (UC_8, *,          internal, 0)," &
            "347   (BC_2, STXDATA4,   input,    X)," &
            "348   (UC_8, *,          internal, 0)," &
            "349   (BC_2, STXDATA5,   input,    X)," &
            "350   (UC_8, *,          internal, 0)," &
            "351   (BC_2, STXDATA6,   input,    X)," &
            "352   (UC_8, *,          internal, 0)," &
            "353   (BC_2, STXDATA7,   input,    X)," &
            "354   (UC_8, *,          internal, 0)," &
            "355   (BC_2, STXPRTY,    input,    X)," &
            "356   (UC_8, *,          internal, 0)," &
            "357   (BC_2, STXSOC,     input,    X)," &
            "358   (UC_8, STXCLAV,    output2,  X)," &
            "359   (UC_8, *,          internal, 0)," &
         -- num    cell   port         func     safe [ccell dis rslt]
            "360   (BC_4, STXCLK,     input,    X)";

end MC92501;
```

G

References

1. ANSI Recommendation I.361, “B-ISDN ATM Layer Specification”, November 1995
2. ITU-T Recommendation I.610, “B-ISDN Operation and Maintenance Principles and Functions”, November, 1995
3. ATM Forum, “ATM User-Network Interface Specification”, Version 3.1, September, 1994
4. ATM Forum, “BISDN Inter Carrier Interface (B-ICI) Specification”, Version 1.1, September, 1994
5. ATM Forum, “BISDN Inter Carrier Interface (B-ICI) Specification”, Version 2.0 Letter Ballot, November, 1995
6. ATM Forum, “Traffic Management Specification”, Version 4.0, April, 1996.
7. ANSI T1S1.5/93-004R2, “Broadband ISDN - Operations and Maintenance Principles and Functions”, January, 1994
8. Bellcore TA-NWT-01110, “Broadband ISDN Switching Systems Generic requirements”, Issue 1, August, 1992
9. Bellcore GR-1113-CORE, “Asynchronous Transfer Mode (ATM) and ATM Adaptation Layer (AAL) Protocols”, Issue 1, July, 1994
10. Bellcore GR-1248-CORE, “Generic Requirements for Operations of ATM Network Elements”, Issue 1, August, 1994
11. Motorola, “MC68360 Quad Integrated Communications Controller User’s Manual”, 1993
12. “UTOPIA, An ATM-PHY Interface Specification, Level 1, Version 2.01”, March 21, 1994.
13. “UTOPIA Level 2 Specification, Version 1.0”, June, 1995
14. Bellcore TA-TSV-001408, “Generic Requirements for Exchange PVC Cell Relay Service”, Issue 1, August, 1993
15. P. Bardell, W. McAnney, and J. Savir, *Built-In Test for VLSI: Pseudorandom Techniques*, John Wiley & Sons, 1987.

Index

D

D

E

E

F

F

G

H

I

J

L

W

W