CN8237

ATM OC-12 ServiceSAR Plus with xBR Traffic Management

The CN8237 Service Segmentation and Reassembly (*Service*SAR) Controller integrates ATM terminal functions, PCI Bus Master and Slave controllers, and a UTOPIA 1 or 2 interface with service-specific functions in a single package for AAL0 and AAL5 operations. The *Service*SAR Controller generates and terminates ATM traffic and automatically schedules cells for transmission. The CN8237 is targeted at 622 Mbps throughput systems where the number of VCCs is relatively large, or the performance of the overall system is critical. Networking equipment it supports ranges from routers and Ethernet switches to ATM Edge switches and Frame Relay switches.

Service-Specific Performance Accelerators

The CN8237 incorporates numerous service-specific features designed to accelerate and enhance system performance. For instance, the CN8237 implements Echo Suppression of LAN traffic via LECID filtering and supports Frame Relay DE to CLP interworking.

Advanced xBR Traffic Management

The xBR Traffic Manager in the CN8237 supports multiple ATM service categories. Categories include CBR, VBR (both single and dual leaky bucket), UBR, GFR (Guaranteed Frame Rate), and ABR (explicit rate, relative rate, or EFCI marking). The CN8237 manages each VCC independently. It dynamically schedules segmentation traffic to comply with up to 16+ CBR user-configured scheduling priorities for the various traffic classes. Scheduling control is based on a user-specified time reference. ABR channels are managed in hardware according to user-programmable ABR templates. These templates tune the performance of the CN8237's ABR algorithms to a specific system's or network's requirements (user-defined granularity).

–Continued–

Distinguishing Features

Service-Specific Performance Accelerators

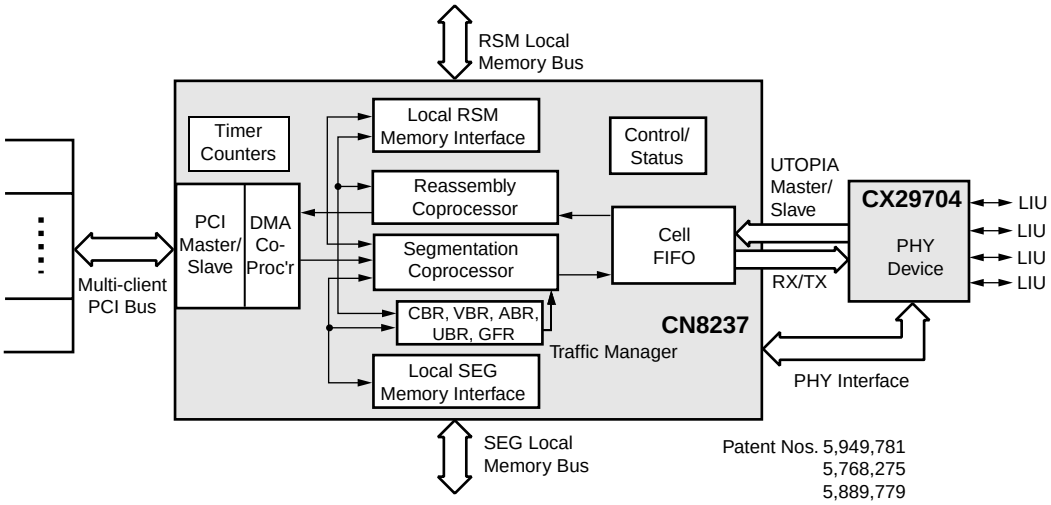
- LECID filtering and echo suppression
- Dual leaky bucket based on CLP (frame relay)
- Frame relay DE interworking
- Internal SNMP MIB counters
- IP over ATM; supports both CLP0+1 and ABR shaping

Flexible Architectures

- Multi-peer host
- Direct switch attachment via reverse UTOPIA
- ATM terminal
  - Host control
  - Local bus control

–Continued–

Functional Block Diagram



Patent Nos. 5,949,781
5,768,275
5,889,779

Ordering Information

Model Number	Manufacturing Part Number	Product Revision	Package	Operating Temperature
CN8237EBGB	28237-12	B	456-pin BGA	–40 °C to 85 °C

Document Revision History

Revision	Level	Date	Description
100454A	Advanced	April 1999	Created
100454B	Advanced	March 2000	Revisions made.
100454C	Advanced	February 2001	Revisions made.
500376A	Advanced	July 2002	Revisions made. Changed format from Conexant to Mindspeed.
28237-DSH-001-B	Advanced	May 2003	Revisions made denoted by change bars.
28237-DSH-001-C	Advanced	August 2003	Added performance warning note regarding PCI optimization.

–Continued from Front–

Multi-Queue Segmentation Processing

The CN8237's segmentation coprocessor generates ATM cells for up to 64 K VCCs at a line rate of up to 600 Mbps for simplex connections. The segmentation coprocessor formats cells on each channel according to segmentation VCC tables, utilizing up to 32 independent transmit queues and reporting segmentation status on a parallel set of up to 32 segmentation status queues. The segmentation coprocessor gathers client data from the host, formats ATM cells while generating and appending protocol overhead, and forwards these to the UTOPIA port. The segmentation coprocessor operates as a slave to the xBR Traffic Manager which schedules VCCs for transmission.

Multi-Queue Reassembly Processing

The CN8237's reassembly coprocessor stores the payload data from the cell stream received by the UTOPIA port into host data buffers. Using a dynamic lookup method which supports NNI or UNI addressing, the reassembly coprocessor processes up to 64 K VCCs simultaneously at a line rate up to 600 Mbps. The host supplies free buffers on up to 32 independent free buffer queues. The reassembly coprocessor performs all CPCS protocol checks and reports the results of these checks and other status data on one of 32 independent reassembly status queues.

High Performance Host Architecture with Buffer Isolation

The CN8237 host interface architecture maximizes performance and system flexibility. The device's control and status queues enable host/SAR communication via write operations alone. This "write only" architecture lowers latency and PCI bus occupancy. Flexibility is achieved by supporting a scalable peer-to-peer architecture. Multiple host clients can be addressed by the segmentation and reassembly (SAR) as separate physical or logical PCI peers. Segmentation and reassembly data buffers on the host system are identified by buffer descriptors in SAR-local (or host) memory which contain pointers to buffers. The use of buffer descriptors in this way allows isolation of data buffers from the mechanisms that handle buffer allocation and linking. This provides a layer of indirection in buffer assignment and management that maximizes system architecture flexibility.

Important System Performance Requirements:

The PCI bus is the management, control and data plane for the CN8237 SAR. In order for the CN8237 SAR to achieve OC-12 line rate, the PCI bus performance MUST be optimized to the maximum extent by ensuring the following:

1) There can be NO retries issued by the target slave device when the SAR is requesting a read transaction for packet segmentation. The address to data latency [PCI 2.2 Specification section 3.5.1.1] for a SAR segmentation read MUST be 9 PCI clock cycles or less. All subsequent word transfers should require one clock cycle each.
2) The PCI bus arbiter should implement the following signaling protocols:
    a) the arbiter should "park" the bus with the SAR [PCI 2.2 Spec. section 3.4.3]
    b) one GNT# may be deasserted coincident with another GNT# being asserted if the bus is not in the Idle state.
3) Refer to the Mindspeed web site for additional performance related material

Designer Toolkit

Mindspeed supplies a toolkit designers can use to establish an evaluation environment for the CN8237. The toolkit, (CN8237EVM) includes a working reference design, an example of a software driver, and facilities for generating and terminating all service categories of ATM traffic. In addition, because the CN8237 buffer management and control architecture is based on Mindspeed's 155 Mbps *Service*SAR family (Bt8233, RS8234, RS8235, and CN8236), it allows for straight-forward migration of preexisting software. Together the toolkit and architecture enable rapid prototyping and accelerate ATM system development.

–Continued Distinguishing Features–

New Features

- 3.3 V, 456 BGA lowers power and eases PCB assembly
- 64-bit/66 MHz PCI 2.1, including support for serial EEPROM
- Enhancements to xBR Traffic Manager
  - fewer ABR templates
  - improved CBR tunneling
- Reduced memory size for VCC lookup tables
- Increased addressing flexibility
- Additional byte lane swappers for increased system flexibility
- Programmable size routing tags up to 64 B cells
- Selectable single/separate UTOPIA clocks
- Updated PM-OAM processing per i.610
- SECBC calculated per GR-1248
- Compact PCI Hot Swap capabilities
- Master PCI write over read arbitration control
- Multi-PHY UTOPIA Level 2
- Head of line blocking protection for multi-PHY operation

xBR Traffic Management

- *TM4.1* Service Classes
  - CBR
  - VBR (single, dual and CLP-based leaky buckets)
  - Real time VBR
  - ABR (ER, RR, EFCI)
  - UBR
  - GFC (controlled and uncontrolled flows)
  - Guaranteed Frame Rate (GFR) (guaranteed MCR on UBR VCCs)
- 16 levels of priorities (16 + CBR)
- Dynamic per-VCC scheduling
- Multiple programmable ABR templates (supplied by Mindspeed or user)
- Scheduler driven by selectable clock
  - Local system clock
  - External reference clock
- Internal RM OAM cell feedback path
- Virtual FIFO buffer rate matching (Source Rate Matching)
- Per-VCC MCR and ICR
- Tunneling
  - VP tunnels (VCI interleaving on PDU boundaries)
  - CBR tunnels (cells interleaved as UBR, VBR, or ABR with an aggregate CBR limit)

Multi-Queue Segmentation Processing

- 32 transmit queues with optional priority levels
- 64 K VCCs maximum
- AAL5 CPCS generation
- AAL0 Null CPCS (optional use of PTI for PDU demarcation)
- ATM cell header generation
- Raw cell mode (52 octet)
- 800 Mbps half duplex
- 622 Mbps full duplex (w/ 2-cell PDUs)
- Variable length transmit FIFO buffer - CDV - host latency matching (1 to 9 cells)
- Symmetric Tx and Rx architecture
  - buffer descriptors
  - queues
- User defined field circulates back to the host (32 bits)
- Distributed host or SAR-shared memory segmentation
- Simultaneous segmentation and reassembly
- Per-PDU control of CLP/PTI (UBR)
- Per-PDU control of AAL5 UU field
- Message and streaming status modes
- Virtual Tx FIFO buffer (PCI host)

Multi-Queue Reassembly Processing

- 32 reassembly queues
- 64 K VCCs maximum
- AAL5 CPCS checking
- AAL0
  - PTI termination
  - Cell count termination
- Early Packet Discard, based on:
  - Receive buffer underflow
  - Receive status overflow
  - CLP with priority threshold
  - AAL5 max PDU length
  - Rx FIFO buffer full
  - Frame relay DE with priority threshold
  - LECID filtering and echo suppression
  - Per-VCC firewalls
- Dynamic channel lookup (NNI or UNI addressing)
  - Supports full address space
  - Deterministic
  - Flexible VCI count per VPI
  - Optimized for signalling address assignment
- Message and streaming status modes
- Raw cell mode (52 octet)
- 800 Mbps half duplex
- 622 Mbps full duplex (with 2-cell PDUs)
- Distributed host or SAR-shared memory reassembly
- 8 programmable reassembly hardware time-outs (per-VCC assignable)
- Global max PDU length for AAL5
- Per-VCC buffer firewall (memory usage limit)
- Simultaneous reassembly and segmentation
- Idle cell filtering
- 64 K duplex VCCs

High Performance Host Architecture with Buffer Isolation

- Write-only control and status
- Read multiple command for data transfer
- Up to 32 host clients control and status queues
- Physical or logical clients
  - Enables peer-to-peer architecture
- Descriptor-based buffer chaining
- Scatter/gather DMA
- Endian neutral (allows data word and control word byte swapping, for both big and little endian systems)
- Non-word (byte) aligned host buffer addresses
- Automatically detects presence of Tx data or Rx free buffers
- Virtual FIFO buffers (PCI bursts treated as a single address)
- Hardware indication of BOM
- Allows isolation of system resources
- Status queue interrupt delay

Designer Toolkit

- Evaluation hardware and software
- Reference schematics
- Hardware Programming Interface-RS823xHPI reference source code (C)

Generous Implementation of OAM-PM Protocols

- Detection of all F4/F5 OAM flows
- Internal PM monitoring and generation for up to 128 VCCs
- Optional global OAM Rx/Tx queues
- In-line OAM insertion and generation

–Continued–

–Continued Distinguishing Features–

Standards-Based I/O

- 64-bit, 66 MHz PCI 2.1
- Serial EEPROM to store PCI configuration information
- PHY interfaces
  - UTOPIA master (Level 2)
  - UTOPIA slave (Level 2)
- Flexible SAR-shared memory architecture
- Boundary scan for board-level testing
- Source loopback, for diagnostics
- Glueless connection to Mindspeed's ATM physical layer devices, the RS8254/5

Standards Compliance

- *UNI/NNI 3.1*
- *TM 4.1*
- *Bellcore GR-1248*
- *ATM Forum B-ICI V2.0*

Table of Contents

Table of Contents . . . . . i-i

List of Figures . . . . . i-xiii

List of Tables . . . . . i-xvii

1.0 CN8237 Product Overview 1-1

1.1 Introduction 1-1

1.2 Service-Specific Performance Accelerators 1-3

1.3 Designer Toolkit 1-7

2.0 Architecture Overview 2-1

2.1 Introduction 2-1

2.2 High Performance Host Architecture with Buffer Isolation 2-1

2.2.1 Multiple ATM Clients 2-2

2.2.2 CN8237 Queue Structure 2-3

2.2.3 Buffer Isolation Utilizing Descriptor-Based Buffer Chaining 2-6

2.2.4 Status Queue Relation to Buffers and Descriptors 2-8

2.2.5 Write-only Control/Status 2-10

2.2.6 Scatter/Gather DMA 2-11

2.2.7 Interrupts 2-11

2.3 Automated Segmentation Engine 2-11

2.4 Automated Reassembly Engine 2-13

2.5 Advanced xBR Traffic Management 2-15

2.5.1 CBR Traffic 2-17

2.5.2 VBR Traffic 2-17

2.5.3 ABR Traffic 2-18

2.5.4 UBR Traffic 2-18

2.5.5 GFR Traffic 2-18

2.5.6 xBR Cell Scheduler 2-19

2.5.7 ABR Flow Control Manager 2-20

2.6 Burst FIFO Buffers 2-20

2.7 Implementation of OAM-PM Protocols 2-21

2.8 Standards-Based I/O 2-22

2.9 Electrical/Mechanical 2-23

2.10 Logic Diagram and Pin Descriptions . . . 2-23

3.0 Host Interface . . . 3-1

3.1 Overview . . . 3-1

3.2 Multiple Client Architecture . . . 3-2

3.2.1 Logical Clients . . . 3-2
3.2.2 Resource Allocation . . . 3-3
3.2.3 Resource Isolation . . . 3-3
3.2.4 Peer-to-Peer Transfers . . . 3-3

3.3 Write-only Control and Status . . . 3-4

3.3.1 Write-only Control Queues . . . 3-5
3.3.1.1 Control Variables . . . 3-5
3.3.1.2 Queue Management . . . 3-5
3.3.1.3 Underflow Conditions . . . 3-6
3.3.2 Write-only Status Queues . . . 3-7
3.3.2.1 Control Variables . . . 3-7
3.3.2.2 Queue Management . . . 3-7
3.3.2.3 Overflow Conditions . . . 3-8
3.3.2.4 Status Queue Interrupt Delay . . . 3-9

4.0 Segmentation Coprocessor . . . 4-1

4.1 Overview . . . 4-1

4.2 Segmentation Functional Description . . . 4-1

4.2.1 Segmentation VCCs . . . 4-1
4.2.1.1 Segmentation VCC Table . . . 4-2
4.2.1.2 VCC Identification . . . 4-3
4.2.2 Submitting Segmentation Data . . . 4-4
4.2.2.1 User Data Format . . . 4-4
4.2.2.2 Buffer Descriptors . . . 4-4
4.2.2.3 Host Linked Segmentation Buffer Descriptors . . . 4-5
4.2.2.4 Transmit Queues . . . 4-5
4.2.2.5 Partial PDUs . . . 4-7
4.2.2.6 Virtual Paths . . . 4-7
4.2.3 CPCS-PDU Processing . . . 4-8
4.2.3.1 AAL5 . . . 4-8
4.2.3.2 AAL0 . . . 4-9
4.2.4 ATM PHY Layer Interface . . . 4-9

4.2.5 Status Reporting . . . 4-9
4.2.6 Virtual FIFO Buffers . . . 4-10
4.3 Segmentation Control and Data Structures . . . 4-10
4.3.1 Segmentation VCC Table Entry . . . 4-10
4.3.2 Data Buffers . . . 4-15
4.3.3 Segmentation Buffer Descriptors . . . 4-15
4.3.4 Transmit Queues . . . 4-18
4.3.4.1 Entry Format . . . 4-18
4.3.4.2 Transmit Queue Management . . . 4-19
4.3.5 Routing Tags . . . 4-20
4.3.6 Segmentation Status Queues . . . 4-23
4.3.6.1 Entry Format . . . 4-23
4.3.6.2 Status Queue Management . . . 4-26
4.3.6.3 Status Queue Overflow . . . 4-27
4.3.7 Segmentation Internal SRAM Memory Map . . . 4-28
5.0 Reassembly Coprocessor . . . 5-1
5.1 Overview . . . 5-1
5.2 Reassembly Functional Description . . . 5-1
5.2.1 Reassembly VCCs . . . 5-2
5.2.1.1 Relation to Segmentation VCCs . . . 5-3
5.2.2 Channel Lookup . . . 5-4
5.2.2.1 Programmable Block Size for VCC Table/ VCI Index Table . . . 5-4
5.2.2.2 Setup . . . 5-5
5.2.2.3 Operation . . . 5-6
5.2.2.4 Variable VPI/PORT_ID Lookup (Multiphy Support) . . . 5-7
5.3 CPCS-PDU Processing . . . 5-7
5.3.1 AAL5 Processing . . . 5-9
5.3.1.1 AAL5 COM Processing . . . 5-9
5.3.1.2 AAL5 EOM Processing . . . 5-9
5.3.1.3 AAL5 Error Conditions . . . 5-10
5.3.2 AAL0 Processing . . . 5-11
5.3.2.1 Termination Methods . . . 5-11
5.3.2.2 AAL0 Error Conditions . . . 5-11
5.3.3 ATM Header Processing . . . 5-12

5.3.4 BOM Synchronization Signal . . . 5-12
5.3.4.1 Prepend Index . . . 5-12
5.4 Buffer Management . . . 5-13
5.4.1 Scatter Method . . . 5-13
5.4.2 Free Buffer Queues . . . 5-14
5.4.3 Linked Data Buffers . . . 5-16
5.4.4 Initialization of Buffer Structures . . . 5-17
5.4.4.1 Buffer Descriptors . . . 5-17
5.4.4.2 Free Buffer Queue Base Table . . . 5-17
5.4.4.3 Free Buffer Queue Entries . . . 5-17
5.4.4.4 Other Initialization . . . 5-17
5.4.5 Buffer Allocation . . . 5-18
5.4.6 Error Conditions . . . 5-18
5.4.7 Early Packet Discard . . . 5-19
5.4.7.1 General Description . . . 5-19
5.4.7.2 Frame Relay Packet Discard . . . 5-19
5.4.7.3 CLP Packet Discard . . . 5-19
5.4.7.4 LANE-LECID Packet Discard—Echo Suppression on Multicast Data Frames . . . 5-19
5.4.7.5 DMA FIFO Buffer Full . . . 5-20
5.4.7.6 Error Conditions . . . 5-20
5.4.8 Hardware PDU Time-Out . . . 5-21
5.4.8.1 Reassembly Time-Out Process . . . 5-21
5.4.8.2 Halting Time-Out Processing . . . 5-21
5.4.8.3 Timer Reset . . . 5-21
5.4.8.4 Reassembly Time-Out Condition . . . 5-22
5.4.8.5 Time-Out Period Calculation . . . 5-22
5.4.9 Virtual FIFO Buffer Mode . . . 5-22
5.4.9.1 Setup . . . 5-22
5.4.9.2 Operation . . . 5-22
5.4.9.3 Errors . . . 5-22
5.4.10 Firewall Functions . . . 5-23
5.4.10.1 Setup . . . 5-23
5.4.10.2 Operation . . . 5-23
5.4.10.3 Credit Return . . . 5-24
5.5 Global Statistics . . . 5-24
5.6 Status Queue Operation . . . 5-25
5.6.1 Structure . . . 5-25
5.6.1.1 Setup . . . 5-27
5.6.1.2 Operation . . . 5-27
5.6.1.3 Errors . . . 5-28

5.6.1.4 Host Detection of Status Queue Entries . . . 5-28
5.6.2 Status Queue Overflow or Full Condition . . . 5-29
5.7 Reassembly Control and Data Structures . . . 5-29
5.7.1 Channel Lookup Structures . . . 5-29
5.7.2 Reassembly VCC Table . . . 5-32
5.7.2.1 AAL5, AAL0 and VCC Table Entries . . . 5-33
5.7.3 Reassembly Buffer Descriptor Structure . . . 5-37
5.7.4 Free Buffer Queues . . . 5-39
5.7.5 Reassembly Status Queues . . . 5-41
5.7.6 LECID Table . . . 5-46
5.7.7 Global Time-Out Table . . . 5-47
5.7.8 Reassembly Internal SRAM Memory Map . . . 5-48

6.0 Traffic Management . . . 6-1
6.1 CN8237 Overview . . . 6-1
6.1.1 xBR Cell Scheduler . . . 6-3
6.1.2 ABR Flow Control Manager . . . 6-5
6.2 xBR Cell Scheduler Functional Description . . . 6-6
6.2.1 Scheduling Priority . . . 6-6
6.2.1.1 16 Priority Levels + CBR . . . 6-6
6.2.1.2 VCC Priority Assignment . . . 6-6
6.2.2 Dynamic Schedule Table . . . 6-6
6.2.2.1 Overview . . . 6-6
6.2.2.2 Schedule Table Slots . . . 6-9
6.2.2.3 Schedule Slot Formats without USE_SCH_CTRL Asserted . . . 6-10
6.2.2.4 Schedule Slot Formats with USE_SCH_CTRL Asserted . . . 6-11
6.2.2.5 Some Scheduling Scenarios . . . 6-12
6.2.3 CBR Traffic . . . 6-13
6.2.3.1 CBR Rate Selection . . . 6-13
6.2.3.2 Available Rates . . . 6-14
6.2.3.3 CBR Cell Delay Variation (CDV) . . . 6-15
6.2.3.4 CBR Channel Management . . . 6-17
6.2.4 VBR Traffic . . . 6-18
6.2.4.1 Mapping CN8237 VBR Service Categories to *TM 4.1* VBR Service Categories . . . 6-18
6.2.4.2 Rate-Shaping vs. Policing . . . 6-18
6.2.4.3 Single Leaky Bucket . . . 6-18
6.2.4.4 Dual Leaky Bucket . . . 6-18
6.2.4.5 CLP-Based Buckets . . . 6-19
6.2.4.6 Rate Selection . . . 6-19
6.2.4.7 Real-Time VBR and CDV . . . 6-19
6.2.5 UBR Traffic . . . 6-19
6.2.6 xBR Tunnels (Pipes) . . . 6-20

6.2.7 Guaranteed Frame Rate . . . 6-22
6.2.8 PCR Control for Priority Queues . . . 6-23
6.3 ABR Flow Control Manager . . . 6-24
6.3.1 A Brief Overview of *TM 4.1* . . . 6-24
6.3.2 Internal ABR Feedback Control Loop . . . 6-24
6.3.2.1 Source Flow Control Feedback . . . 6-25
6.3.2.2 Destination Behavior . . . 6-26
6.3.2.3 Out-of-Rate Cells . . . 6-26
6.3.3 Source and Destination Behaviors . . . 6-27
6.3.4 ABR VCC Parameters . . . 6-27
6.3.5 ABR Templates . . . 6-27
6.3.6 Cell Type Decisions . . . 6-28
6.3.6.1 In-rate Cell Streams . . . 6-28
6.3.6.2 ABR Cell Decisions . . . 6-29
6.3.7 Rate Decisions and Updates . . . 6-32
6.3.7.1 ABR Traffic Shaping . . . 6-32
6.3.7.2 Rate Adjustment Overview . . . 6-32
6.3.7.3 Backward RM Cell Flow Control . . . 6-33
6.3.7.4 Forward RM Cell Transmission Decisions . . . 6-37
6.3.7.5 ACR Change Notification . . . 6-37
6.3.7.6 Rate Adjustment in Turnaround RM Cells . . . 6-38
6.3.7.7 Optional Rate Adjustment Due to Use-It-Or-Lose-It Behavior . . . 6-40
6.3.8 Boundary Conditions and Out-of-Rate RM Cells . . . 6-41
6.3.8.1 Calculated Rate Boundaries . . . 6-41
6.3.8.2 Out-of-Rate Forward RM Cell Generation . . . 6-41
6.3.8.3 Out-of-Rate Backward RM Cells . . . 6-41
6.4 GFC Flow Control Manager . . . 6-41
6.4.1 A Brief Overview of GFC . . . 6-41
6.4.2 The CN8237's Implementation of GFC . . . 6-42
6.4.2.1 Configuring the Link for GFC Operation . . . 6-43
6.5 Traffic Management Control and Status Structures . . . 6-43
6.5.1 Schedule Table . . . 6-43
6.5.2 CBR-Specific Structures . . . 6-44
6.5.2.1 CBR Traffic . . . 6-44
6.5.2.2 Tunnel Traffic . . . 6-44
6.5.2.3 SCH_STATE Fields For CBR . . . 6-45
6.5.3 VBR-Specific Structure . . . 6-46
6.5.3.1 VBR SCH_STATE . . . 6-46
6.5.3.2 VBR1 or VBR2 Schedule State Table . . . 6-46
6.5.3.3 Bucket Table for VBR2 and VBRC . . . 6-47
6.5.4 GFR-Specific Structures . . . 6-48
6.5.4.1 GFR Schedule State Table . . . 6-48
6.5.4.2 GFR MCR Limit Bucket Table . . . 6-49
6.5.5 ABR-Specific Structures . . . 6-50
6.5.5.1 ABR Schedule State Table . . . 6-50

6.5.6 RS_QUEUE 6-53
6.5.7 Scheduler Internal SRAM Registers 6-54

7.0 OAM Functions 7-1

7.1 OAM Overview 7-1
7.1.1 OAM Functions Supported 7-2
7.1.2 OAM Flows Supported 7-3
7.1.2.1 F4 OAM Flow 7-3
7.1.2.2 F5 OAM Flow 7-3
7.1.2.3 Performance Monitoring (PM) 7-4
7.1.3 OAM Cell Format 7-5
7.1.4 Global Queue Processing of OAM 7-6

7.2 Segmentation of OAM Cells 7-6
7.2.1 Key OAM-Related Fields for OAM Segmentation 7-7
7.2.1.1 Segmentation Buffer Descriptors 7-7
7.2.1.2 Low Latency Transmission 7-7
7.2.1.3 Segmentation Status Queue 7-7
7.2.1.4 F4 Flow 7-7
7.2.2 Error Condition During OAM Segmentation 7-7

7.3 Reassembly of OAM Cells 7-8
7.3.1 Key OAM-Related Fields for OAM Reassembly 7-8
7.3.1.1 Reassembly VCC State Table 7-8
7.3.1.2 Reassembly Status Queue 7-8
7.3.1.3 F4 Flow 7-8
7.3.2 OAM Reassembly Operation 7-9
7.3.3 Error Conditions During OAM Reassembly 7-9

7.4 PM Processing 7-9
7.4.1 Initializing PM Operation 7-11
7.4.2 Setting Up Channels for PM Operation 7-12
7.4.3 PM Operation 7-12
7.4.3.1 Generation of Forward Monitoring PM Cells 7-12
7.4.3.2 Reassembly of Forward Monitoring PM Cells 7-13
7.4.3.3 Reassembly of Backward Reporting PM Cells 7-13
7.4.3.4 Turnaround and Segmentation of Backward Reporting PM Cells 7-13
7.4.3.5 Turnaround of Backward Reporting PM Cells ONLY 7-13
7.4.4 Error Conditions During PM Processing 7-13
7.4.5 PASS_OAM Function 7-13

7.5 OAM Control and Status Structures 7-14

7.5.1 SEG_PM Structure . . . 7-14
7.5.2 RSM_PM Table . . . 7-16

8.0 DMA Coprocessor . . . 8-1

8.1 Overview . . . 8-1

8.2 DMA Read . . . 8-1

8.3 DMA Write . . . 8-1

8.4 Master Transactions . . . 8-2
8.4.1 Data Transfers . . . 8-2
8.4.2 Control Word Transfers . . . 8-2

8.5 Slave Transactions . . . 8-2
8.5.1 Control Word Transfers . . . 8-2

8.6 Control Bit by Bit Endian Transfers . . . 8-3
8.6.1 Master Control Swap . . . 8-3
8.6.2 Slave Swap . . . 8-4
8.6.3 Slave Dword Swap . . . 8-5
8.6.4 Master Data Dword Swap . . . 8-6
8.6.5 Master Control Dword Swap . . . 8-8
8.6.6 Master Data Byte Swap . . . 8-9

8.7 Little Endian Mode . . . 8-10
8.7.1 Little Endian 64-bit Master Data Transfers . . . 8-11
8.7.2 Little Endian 32-bit Master Data Transfers . . . 8-12
8.7.3 Little Endian SAR as Master Control Transfers . . . 8-13
8.7.4 Little Endian Host Slave Interface Control Transfers . . . 8-14

8.8 Big Endian Mode . . . 8-14
8.8.1 Big Endian 64-bit Master Data Transfers . . . 8-14
8.8.2 Big Endian 32-bit Master Data Transfers . . . 8-16
8.8.3 Big Endian SAR as Master Control Transfers . . . 8-17
8.8.4 Big Endian Host Slave Interface Control Transfers . . . 8-18

9.0 System Interface . . . 9-1

9.1 System Clocking . . . 9-1

9.2 Real-Time Clock Alarm . . . 9-1

9.3 CN8237 Reset . . . 9-2

10.0 Local Memory Interface . . . 10-1

10.1 Overview . . . 10-1

10.2 Memory Bank Characteristics . . . 10-3
10.3 Memory Size Analysis . . . 10-4

11.0 PHY Interface . . . 11-1
11.1 Overview . . . 11-1
11.2 Microprocessor Interface to PHY Device(s) . . . 11-1
11.3 Microprocessor Interface for Multiple Physical Devices . . . 11-5
11.4 Interface Control . . . 11-6
11.4.1 Strobe Mode . . . 11-7
11.4.1.1 Wait Mode . . . 11-7

12.0 PCI Bus Interface . . . 12-1
12.1 Overview . . . 12-1
12.2 Unimplemented PCI Bus Interface Functions . . . 12-3
12.3 PCI Configuration Space . . . 12-3
12.4 PCI Bus Master Logic . . . 12-4
12.5 Burst FIFO Buffers . . . 12-6
12.6 PCI Bus Slave Logic . . . 12-6
12.7 Byte Swapping and Dword Conversion of Control Structures . . . 12-7
12.8 Power Management . . . 12-7
12.9 Interface Module to Serial EEPROM . . . 12-8
12.9.1 EEPROM Format . . . 12-9
12.9.2 Loading the EEPROM Data at Reset . . . 12-10
12.9.3 Accessing the EEPROM . . . 12-10
12.9.4 Using the Subsystem ID Without an EEPROM . . . 12-11
12.10 PCI Host Address Map . . . 12-11

13.0 ATM UTOPIA Interface . . . 13-1
13.1 Overview of ATM UTOPIA Interface . . . 13-1
13.2 ATM UTOPIA Interface Logic . . . 13-2
13.3 ATM Physical Interface I/O Pins . . . 13-2
13.3.1 UTOPIA Interface . . . 13-5
13.4 UTOPIA Level 2 Interface . . . 13-5
13.4.1 Cell Tagging . . . 13-6
13.4.2 UTOPIA Configuration Control . . . 13-8
13.4.3 UTOPIA Level 2 Multi-Port Operation . . . 13-8
13.5 UTOPIA Level 1 Mode Cell Handshake Timing . . . 13-10
13.6 UTOPIA Level 1 Mode Octet Handshake Timing . . . 13-11
13.7 Slave Level 1 UTOPIA Mode . . . 13-13
13.8 Loopback Mode . . . 13-14

13.9 Receive Cell Synchronization Logic . . . 13-16
13.10 Transmit Cell Synchronization Logic . . . 13-16
13.10.1Head of Line Flushing (HoLF) . . . 13-17

14.0 CN8237 Registers . . . 14-1
14.1 Control and Status Registers . . . 14-1
14.1.1 Register Terminology . . . 14-2
14.2 System Registers . . . 14-4
14.3 Segmentation Registers . . . 14-9
14.4 Scheduler Registers . . . 14-13
14.5 Reassembly Registers . . . 14-19
14.6 Counters and Status Registers . . . 14-26
14.6.1 Host Interrupt Status Registers . . . 14-29
14.7 PCI Bus Interface Registers . . . 14-32

15.0 SAR Initialization—Example Tables . . . 15-1
15.1 Segmentation Initialization . . . 15-1
15.1.1 Segmentation Control Registers . . . 15-1
15.1.2 Segmentation Internal Memory Control Structures . . . 15-2
15.1.3 Segmentation SAR Shared Memory Control Structures . . . 15-4
15.2 Scheduler Initialization . . . 15-6
15.2.1 Scheduler Control Registers . . . 15-6
15.2.2 Scheduler Internal Memory Control Structures . . . 15-7
15.2.3 Scheduler SAR Shared Memory Control Structures . . . 15-7
15.3 Reassembly Initialization . . . 15-9
15.3.1 Reassembly Control Registers . . . 15-9
15.3.2 Reassembly Internal Memory Control Structures . . . 15-11
15.3.3 Reassembly SAR Shared Memory Control Structures . . . 15-12
15.4 General Initialization . . . 15-14
15.4.1 General Control Registers . . . 15-14

16.0 Electrical and Mechanical Specifications . . . 16-1
16.1 Timing . . . 16-1
16.1.1 PCI Bus Interface Timing . . . 16-1
16.1.2 ATM Physical Interface Timing—UTOPIA and Slave UTOPIA . . . 16-3
16.1.3 CN8237 Local Memory Interface Timing . . . 16-6
16.1.3.1 Local Memory Interface Design Details . . . 16-10
16.1.4 PHY Interface Timing . . . 16-15
16.2 Package I/O Description . . . 16-16
16.3 CN8223 PHY Device Connection . . . 16-17

16.4 Absolute Maximum Ratings 16-18
16.5 DC Characteristics 16-20
16.6 Mechanical Specifications 16-21

Appendix A: Boundary Scan A-1
A.1 Instruction Register A-3
A.2 BYPASS Register A-3
A.3 Boundary Scan Register A-3
A.4 Electrical Characteristics A-4
A.5 Boundary Scan Description Language (BSDL) File A-6

Appendix B: List of Acronyms B-1

List of Figures

Figure 1-1. CN8237 Functional Block Diagram 1-3
Figure 2-1. Multiple Client Architecture Supports Up To 32 Clients (Peers). 2-3
Figure 2-2. CN8237 Queue Architecture 2-4
Figure 2-3. Interaction of Queues with CN8237 Functional Blocks 2-5
Figure 2-4. Reassembly Buffer Isolation—Data Buffers Separated from Descriptors 2-6
Figure 2-5. Segmentation Buffer Isolation—Data Buffers Separated from Descriptors 2-7
Figure 2-6. Segmentation Status Queues Related to Data Buffers and Descriptors 2-8
Figure 2-7. Reassembly Status Queues Related to Data Buffers and Descriptors 2-9
Figure 2-8. Write-only Control and Status Architecture 2-10
Figure 2-9. Multi-Level Model for Prioritizing Segmentation Traffic. 2-16
Figure 2-10. Data FIFO Buffers 2-21
Figure 2-11. CN8237 Logic Diagram (1 of 6) 2-24
Figure 3-1. Client/Server Model of the CN8237 3-2
Figure 3-2. Peer-to-Peer vs. Centralized Memory Data Transfers 3-4
Figure 3-3. Write-only Control Queue 3-6
Figure 3-4. Write-only Status Queue 3-8
Figure 4-1. Segmentation VCC Table 4-2
Figure 4-2. Segmentation Buffer Descriptor Chaining 4-5
Figure 4-3. Before SAR Transmit Queue Entry Processing 4-6
Figure 4-4. After SAR Transmit Queue Entry Processing 4-7
Figure 4-5. AAL5 CPCS-PDU Generation 4-8
Figure 4-6. Segmentation Master Control Word 4-19
Figure 4-7. Route Tag Table for tag_size = 2 and 4 4-21
Figure 4-8. Route Tag Table for tag_size = 6 and 8 4-22
Figure 4-9. Route Tag Table for tag_size = 10 4-22
Figure 4-10. Segmentation Master Control Word 4-24
Figure 5-1. Reassembly—Basic Process Flow 5-2
Figure 5-2. Reassembly VCC Table 5-3
Figure 5-3. Direct Index Method for VPI/VCI Channel Lookup. 5-4
Figure 5-4. Programmable Block Size Alternate Direct Index Method 5-5
Figure 5-5. VPI Index Table with Multiple Ports. 5-7
Figure 5-6. CPCS-PDU Reassembly 5-8
Figure 5-7. AAL5 EOM Cell Processing—Fields to Status Queue 5-9
Figure 5-8. AAL5 Processing—CRC and PDU Length Checks. 5-10
Figure 5-9. AAL0 PTI PDU Termination 5-11
Figure 5-10. Host and SAR RSM-Shared Memory Data Structures for Scatter Method 5-13

Figure 5-11. Free Buffer Queue Structure . . . 5-15
Figure 5-12. Data Buffer Structures . . . 5-16
Figure 5-13. Data Structure Locations for Status Queues . . . 5-25
Figure 5-14. Status Queue Structure Format . . . 5-26
Figure 5-15. VPI/VCI Channel Lookup Structure . . . 5-30
Figure 5-16. Reassembly VCC Table Entry Lookup Mechanism . . . 5-32
Figure 5-17. Reassembly Master Control Word . . . 5-38
Figure 5-18. Reassembly Master Control Word . . . 5-39
Figure 5-19. Reassembly Master Control Word . . . 5-41
Figure 5-20. LECID Table, Illustrated . . . 5-46
Figure 6-1. Non-ABR Cell Scheduling . . . 6-4
Figure 6-2. ABR Flow Control . . . 6-5
Figure 6-3. Schedule Table with Size = 100 . . . 6-8
Figure 6-4. Schedule Slot Formats without USE_SCH_CTRL Asserted . . . 6-10
Figure 6-5. Schedule Slot Formats with USE_SCH_CTRL Asserted . . . 6-11
Figure 6-6. One Possible Scheduling Priority Scheme with the CN8237 . . . 6-12
Figure 6-7. Assigning CBR Cell Slots . . . 6-14
Figure 6-8. Introduction of CDV at the ATM/PHY Layer Interface . . . 6-15
Figure 6-9. Schedule Table with Slot Conflicts at Different CBR Rates . . . 6-15
Figure 6-10. CDV Caused by Schedule Table Size at Certain CBR Rates . . . 6-16
Figure 6-11. Another Possible Scheduling Priority Scheme with the CN8237 . . . 6-22
Figure 6-12. ABR Service Category Feedback Control . . . 6-24
Figure 6-13. CN8237 ABR-ER Feedback Loop (Source Behavior) . . . 6-25
Figure 6-14. CN8237 ABR-ER Feedback Generation (Destination Behavior) . . . 6-26
Figure 6-15. Steady State ABR-ER Cell Stream . . . 6-28
Figure 6-16. Cell Type Interleaving on ABR-ER Cell Stream . . . 6-29
Figure 6-17. Cell Decision Table for Nrm = 32 . . . 6-31
Figure 6-18. Backward_RM Flow Control, Block Diagram . . . 6-33
Figure 6-19. RR RATE_INDEX Candidate Selection . . . 6-34
Figure 6-20. ER RATE_INDEX Candidate Selection . . . 6-35
Figure 6-21. Dynamic Traffic Shaping from RM Cell Feedback . . . 6-36
Figure 6-22. ER Reduction Mapping . . . 6-39
Figure 6-23. Head and Tail Pointers . . . 6-54
Figure 7-1. OAM Cell Format . . . 7-5
Figure 7-2. Functional Blocks for PM Segmentation and Reassembly . . . 7-10
Figure 8-1. Master Control Swap . . . 8-3
Figure 8-2. Slave Swap . . . 8-4
Figure 8-3. Slave Dword Swap . . . 8-5
Figure 8-4. Master Data Dword Swap—64-bit Transfer . . . 8-6
Figure 8-5. Master Data Dword Swap—32-bit Transfer . . . 8-7
Figure 8-6. Master Control Dword Swap . . . 8-8
Figure 8-7. Master Data Byte Swap—64-bit Transfer . . . 8-9
Figure 8-8. Master Data Byte Swap—32-bit Transfer . . . 8-10
Figure 8-9. Little Endian 64-bit Master Data Transfers . . . 8-11
Figure 8-10. Little Endian 32-bit Master Data Transfers . . . 8-12
Figure 8-11. Little Endian SAR as Master Control Transfers . . . 8-13

Figure 8-12. Little Endian Host Slave Interface Control Transfers 8-14
Figure 8-13. Big Endian 64-bit Master Data Transfers 8-15
Figure 8-14. Big Endian 32-bit Master Data Transfers 8-16
Figure 8-15. Big Endian SAR as Master Control Transfers. 8-17
Figure 8-16. Big Endian Host Slave Interface Control Transfers 8-18
Figure 10-1. ZBT or Syncburst Synchronous SRAM Bank Utilizing By_16 Devices. 10-4
Figure 11-1. SAR (CN8237) and OC-3 PHY (RS825x) Interface 11-2
Figure 11-2. CN8237/PHY Functional Timing with Inserted Wait States (Normal Mode). 11-3
Figure 11-3. CN8237/RS825x Read/Write Functional Timing (Normal Mode) 11-4
Figure 11-4. Typical PHY Connection to RS825x. 11-5
Figure 11-5. Typical PHY Connection to RS8228. 11-6
Figure 12-1. EEPROM Connection. 12-8
Figure 13-1. UTOPIA Level 2 Receive Timing 13-9
Figure 13-2. Receive Timing in UTOPIA Level 1 Mode with Cell Handshake 13-10
Figure 13-3. Transmit Timing in UTOPIA Level 1 Mode with Cell Handshake 13-11
Figure 13-4. Receive Timing in UTOPIA Level 1 Mode with Octet Handshake 13-12
Figure 13-5. Transmit Timing in UTOPIA Level 1 Mode with Octet Handshake 13-12
Figure 13-6. Receive Timing in Slave UTOPIA Level 1 Mode. 13-13
Figure 13-7. Transmit Timing in Slave UTOPIA Level 1 Mode 13-14
Figure 13-8. Source Loopback Mode Diagram 13-15
Figure 16-1. PCI Bus Input Timing Measurement Conditions 16-2
Figure 16-2. PCI Bus Output Timing Measurement Conditions 16-3
Figure 16-3. UTOPIA and Slave UTOPIA Input Timing Measurement Conditions 16-5
Figure 16-4. UTOPIA and Slave UTOPIA Output Timing Measurement Conditions 16-5
Figure 16-5. ZBT Flow Through SRAM Timing Diagram—One Cycle Memory 16-6
Figure 16-6. ZBT Flow Through SRAM Memory Timing Diagram—Two Cycle Memory 16-7
Figure 16-7. Memory Interface to Syncburst Flow Through SRAM Timing—One Cycle Memory 16-8
Figure 16-8. Memory Interface to Syncburst Flow Through SRAM Timing—Two Cycle Memory 16-9
Figure 16-9. SAR's Local Memory Interface 16-10
Figure 16-10. Add a Delay Line 16-11
Figure 16-11. Flight Time. 16-12
Figure 16-12. Recommended PCB Trace Layout Scheme 16-13
Figure 16-13. Recommended PCB Layout. 16-14
Figure 16-14. CN8237/PHY Functional Timing with Inserted Wait States (Normal Mode). 16-15
Figure 16-15. Interface Timing Diagram—Strobed Mode 16-16
Figure 16-16. RS8223 PHY Device Connection. 16-18
Figure 16-17. 456-Pin Ball Grid Array Package (BGA). 16-22
Figure 16-18. CN8237 Package Dimensions 16-23
Figure A-1. Test Circuitry Block Diagram A-2
Figure A-2. Timing Diagram. A-5

List of Tables

Table 2-1. Hardware Signal Definitions 2-30
Table 3-1. CN8237 Control and Status Queues 3-4
Table 3-2. Write-only Control Queue Variables 3-5
Table 3-3. Write-only Status Queue Variables 3-7
Table 4-1. Segmentation PDU Delineation 4-4
Table 4-2. Segmentation VCC Table Entry—AAL5-AAL0 Format 4-10
Table 4-3. Segmentation VCC Table Entry—AAL5 and 0 Field Descriptions. 4-12
Table 4-4. Segmentation VCC Table Entry—Virtual FIFO Buffers 4-14
Table 4-5. Segmentation VCC Table Entry—Virtual FIFO Buffer Format Field Descriptions 4-14
Table 4-6. Segmentation Buffer Descriptor Entry Format. 4-15
Table 4-7. MISC_DATA Field Bit Definitions with HEADER_MOD Bit Set 4-15
Table 4-8. MISC_DATA Field Bit Definitions with RPL_VCI BIt Set. 4-15
Table 4-9. Segmentation Buffer Descriptor Field Descriptions. 4-16
Table 4-10. Transmit Queue Entry Format (64-bit). 4-18
Table 4-11. Transmit Queue Entry Format (32-bit). 4-18
Table 4-12. Transmit Queue Entry Field Descriptions. 4-18
Table 4-13. Transmit Queue Base Table Entry (64-bit) 4-20
Table 4-14. Transmit Queue Base Table Entry (32-bit) 4-20
Table 4-15. Transmit Queue Base Table Entry Field Descriptions 4-20
Table 4-16. Maximum TxFIFO Size with Routing Tags 4-21
Table 4-17. Routing Tag Cross-Reference 4-23
Table 4-18. Segmentation Status Queue Entry (64-bit) 4-24
Table 4-19. Segmentation Status Queue Entry (32-bit) 4-24
Table 4-20. Segmentation Status Queue Entry Field Descriptions 4-25
Table 4-21. Segmentation Status Queue Format for ACR/ER (64-bit) 4-25
Table 4-22. Segmentation Status Queue Format for ACR/ER (32-bit) 4-26
Table 4-23. Status Queue Entry Field Descriptions for ACR/ER 4-26
Table 4-24. Segmentation Status Queue Base Table Entry (64-bit) 4-27
Table 4-25. Segmentation Status Queue Base Table Entry (32-bit) 4-27
Table 4-26. Segmentation Status Queue Base Table Entry Field Descriptions 4-27
Table 4-27. Segmentation Internal SRAM Memory Map 4-28
Table 5-1. Programmable Block Size Values for Direct Index Lookup 5-5
Table 5-2. STAT Output Pin Values for BOM Synchronization 5-12
Table 5-3. Prepend Index Table Format 5-12
Table 5-4. Normal VPI Index Table Entry Format 5-30
Table 5-5. VPI Index Table Entry Format with EN_PROG_BLK_SZ(RSM_CTRL1) Enabled 5-30

Table 5-6. VPI Index Table Entry Descriptions . . . 5-30
Table 5-7. Normal VCI Index Table Format . . . 5-31
Table 5-8. VCI Index Table Format with EN_PROG_BLK_SZ (RSM_CTRL1) Enabled . . . 5-31
Table 5-9. VCI Index Table Descriptions . . . 5-31
Table 5-10. Reassembly VCC Table Entry Format—AAL5 . . . 5-33
Table 5-11. Reassembly VCC Table Entry Format—AAL0 . . . 5-34
Table 5-12. Reassembly VCC Table Descriptions . . . 5-35
Table 5-13. Reassembly Buffer Descriptor Structure . . . 5-38
Table 5-14. Reassembly Buffer Descriptor Structure Definitions . . . 5-38
Table 5-15. Free Buffer Queue Base Table Entry Format . . . 5-39
Table 5-16. Free Buffer Queue Base Table Entry Descriptions . . . 5-40
Table 5-17. Free Buffer Queue Entry Format . . . 5-40
Table 5-18. Free Buffer Queue Entry Descriptions . . . 5-40
Table 5-19. Reassembly Status Queue Base Table Entry Format . . . 5-41
Table 5-20. Reassembly Status Queue Base Table Entry Descriptions . . . 5-42
Table 5-21. Reassembly Status Queue Entry Format with FWD_PM = 0 . . . 5-42
Table 5-22. Reassembly Status Queue Entry Format with FWD_PM = 1 . . . 5-42
Table 5-23. Reassembly Status Queue Entry Format with FWD_PM = 1 and NEW_PMOAM = 1 . . . 5-43
Table 5-24. PDU_CHECKS Field Bits . . . 5-43
Table 5-25. PDU_CHECKS Field Bits with CNT_ROVR = 1 . . . 5-43
Table 5-26. STATUS Field Bits . . . 5-43
Table 5-27. STM Field Bits . . . 5-43
Table 5-28. Reassembly Status Queue Entry Descriptions . . . 5-44
Table 5-29. LECID Table Entries . . . 5-46
Table 5-30. LECID Table Field Definition . . . 5-46
Table 5-31. Global Time-Out Table Entry Format . . . 5-47
Table 5-32. Global Time-Out Table Entry Descriptions . . . 5-47
Table 5-33. Reassembly Internal SRAM Memory Map . . . 5-48
Table 6-1. ATM Service Category Parameters and Attributes . . . 6-2
Table 6-2. Scheduler Clock Selection . . . 6-7
Table 6-3. Selection of Schedule Table Slot Size by System Requirements . . . 6-9
Table 6-4. CN8237 VBR to TM 4.1 VBR Mapping . . . 6-18
Table 6-5. ABR Cell Type Decision Vector (ACDV) . . . 6-30
Table 6-6. Schedule Slot Entry—CBR/Tunnel Traffic . . . 6-43
Table 6-7. CBR_TUN_ID Field, Bit Definitions—CBR Slot . . . 6-44
Table 6-8. CBR_TUN_ID Field, Bit Definitions—Tunnel Slot . . . 6-44
Table 6-9. Schedule Slot Field Descriptions—CBR Traffic . . . 6-44
Table 6-10. SCH_STATE for SCH_MODE = CBR . . . 6-45
Table 6-11. CBR SCH_STATE Field Descriptions . . . 6-45
Table 6-12. SCH_STATE for SCH_MODE = VBR1 or VBR2 . . . 6-46
Table 6-13. VBR1 and VBR2 SCH_STATE Field Descriptions . . . 6-46
Table 6-14. Bucket Table Entry . . . 6-47
Table 6-15. Bucket Table Entry Field Descriptions . . . 6-47
Table 6-16. SCH_STATE for SCH_MODE = GFR . . . 6-48
Table 6-17. SCH_STATE Field Descriptions for SCH_MODE = GFR . . . 6-48
Table 6-18. GFR MCR Limit Bucket Table Entry . . . 6-49

Table 6-19. GFR MCR Bucket Table Entry Field Descriptions . . . 6-49
Table 6-20. SCH_STATE for SCH_MODE = ABR . . . 6-50
Table 6-21. ABR SCH_STATE Field Descriptions . . . 6-51
Table 6-22. RS_QUEUE Entry—OAM-PM Reporting Information Ready for Transmission . . . 6-53
Table 6-23. RS_QUEUE Entry—Forward ER RM Cell Received . . . 6-53
Table 6-24. RS_QUEUE Entry—Backward ER RM Cell Received . . . 6-53
Table 6-25. RS_QUEUE Field Descriptions . . . 6-53
Table 6-26. Scheduler Internal SRAM Memory Map (Head/Tail Pointers) . . . 6-54
Table 7-1. OAM Functions of the ATM Layer . . . 7-2
Table 7-2. VCI Values for F4 OAM Flows . . . 7-3
Table 7-3. PTI Values for F5 OAM Flows . . . 7-3
Table 7-4. OAM Type and Function Type Identifiers . . . 7-6
Table 7-5. PM-OAM Field Initialization For Any PM_INDEX . . . 7-11
Table 7-6. SEG_PM Structure . . . 7-14
Table 7-7. SEG_PM Field Descriptions . . . 7-15
Table 7-8. RSM_PM Table Entry . . . 7-16
Table 7-9. RSM_PM Table Field Descriptions . . . 7-16
Table 10-1. Memory Map . . . 10-2
Table 10-2. Example of Memory Map . . . 10-2
Table 10-3. BANKSIZE Selection Number . . . 10-3
Table 10-4. Memory Size in Bytes . . . 10-6
Table 11-1. Microprocessor Interface to PHY Devices Interface Pins . . . 11-2
Table 12-1. EEPROM Fields . . . 12-9
Table 13-1. ATM Physical Interface Mode Select (FRCFG) . . . 13-2
Table 13-2. ATM Physical Interface Mode Select (UTOPIA1) . . . 13-2
Table 13-3. UTOPIA Mode Signals . . . 13-3
Table 13-4. Slave UTOPIA Mode Interface Signals . . . 13-4
Table 13-5. Cell Format 8-bit Mode . . . 13-6
Table 13-6. Cell Format 16-bit Mode . . . 13-6
Table 13-7. Cell Format, Tagging Enabled, 8-bit Mode . . . 13-7
Table 13-8. Cell Format, Tagging Enabled, 16-bit Mode . . . 13-7
Table 14-1. Access Type Abbreviation Description . . . 14-1
Table 14-2. CN8237 Control and Status Registers . . . 14-2
Table 14-3. PCI Configuration Register . . . 14-33
Table 14-4. PCI Configuration Field Descriptions . . . 14-34
Table 14-5. PCI Status Register (0x04) . . . 14-36
Table 14-6. PCI Command Register (0x04) . . . 14-36
Table 14-7. PCI Special Status Register (0x40) . . . 14-37
Table 14-8. EEPROM Register (0x4C) . . . 14-39
Table 14-9. Power Management Capabilities Register . . . 14-39
Table 14-10. Power Management Control/Status Register . . . 14-40
Table 14-11. Hot Swap Control/Status Register . . . 14-41
Table 15-1. Table of Values for Segmentation Control Register Initialization . . . 15-2
Table 15-2. Table of Values for Segmentation Internal Memory Initialization . . . 15-3
Table 15-3. Table of Values for Segmentation SAR Shared Memory Initialization . . . 15-4
Table 15-4. Table of Values for Scheduler Control Register Initialization . . . 15-6

Table 15-5. Table of Values for Sch SAR Shared Memory Initialization 15-7
Table 15-6. Table of Values for Reassembly Control Register Initialization 15-9
Table 15-7. Table of Values for Reassembly Internal Memory Initialization 15-11
Table 15-8. Table of Values for Reassembly SAR Shared Memory Initialization. 15-12
Table 15-9. Table of Values for General Control Register Initialization 15-14
Table 16-1. Clock Specifications 16-1
Table 16-2. 66 MHz and 33 MHz Timing Parameters 16-2
Table 16-3. UTOPIA Interface Timing Parameters 16-3
Table 16-4. Slave UTOPIA Interface Timing Parameters 16-4
Table 16-5. One Cycle Memory Timing Parameter Characteristics. 16-6
Table 16-6. Two Cycle Memory Timing Parameter Characteristics 16-7
Table 16-7. Interface Timing—Normal Mode. 16-15
Table 16-8. Interface Timing—Strobed Mode 16-16
Table 16-9. Package Interface Pins 16-17
Table 16-10. Absolute Maximum Ratings 16-19
Table 16-11. DC Characteristics. 16-20
Table 16-12. Listing of Pin Numbers and Labels (Numeric Order) (1 of 5) 16-24
Table 16-13. Listing of Pin Numbers and Labels (Alphabetic Order) (1 of 5). 16-29
Table A-1. Boundary Scan Signals A-1
Table A-2. IEEE Std. 1149.1 Instructions A-3
Table A-3. Timing Specifications A-4

1.0 CN8237 Product Overview

1.1 Introduction

The CN8237 Service Segmentation and Reassembly Controller (*Service*SAR) delivers, at OC-12 rates, a wide range of advanced ATM and AAL functions in a single highly-integrated CMOS package. The CN8237 *Service*SAR also delivers service-specific features that enable system designers to accelerate specific protocol interworking functions. These features include:

- Frame Relay Early Packet Discard (EPD)
- Generic Flow Control (GFC)
- Echo suppression of multicast data frames on Emulated LAN (ELAN)

The CN8237 meets the requirements of *UNI 3.1* and complies with *ATM Forum Traffic Management Specification, TM 4.1*. The CN8237 provides traffic-shaping for all service categories, including:

- Constant Bit Rate (CBR)
- Variable Bit Rate (VBR)—both single and dual leaky bucket
- Unspecified Bit Rate (UBR)
- Available Bit Rate (ABR)
- GFC—both controlled and uncontrolled flows
- Guaranteed Frame Rate (GFR), that is, guaranteed Minimum Cell Rate (MCR) on UBR Virtual Channel Connections (VCCs)

The internal xBR Traffic Manager automatically schedules each VCC according to user-assigned parameters.

The CN8237 architecture minimizes and controls host traffic congestion. The host manages the CN8237 terminal with an efficient architecture that uses write-only control and status queues. For example, the host submits data for transmit by writing buffer descriptor pointers to one of 32 transmit queues. These entries form lists of tasks for the *Service*SAR to perform. The CN8237 reports segmentation and reassembly status to the host by writing entries to segmentation and reassembly status queues, which the host processes. This architecture lessens the control burden on the host and minimizes Peripheral Component Interconnect (PCI) bus utilization by eliminating host control activities across the PCI bus.

The host interface controls host congestion by having each peer maintain separate control and status queues. Each VCC in a peer group can be limited to a specific maximum receive buffer utilization, further controlling congestion. EPD is supported for VCCs exceeding their resource allotments. On transmit, peers are assigned fixed or round-robin priority to ensure predictable servicing. The host implements a congestion notification algorithm for ABR with a simple one-word write to a SAR control register. The SAR either reduces the Explicit Rate (ER) field or sets the Congestion Indication (CI) bit in Turnaround Resource Management (RM) cells, based on user configuration.

The CN8237 consists of five separate coprocessors:

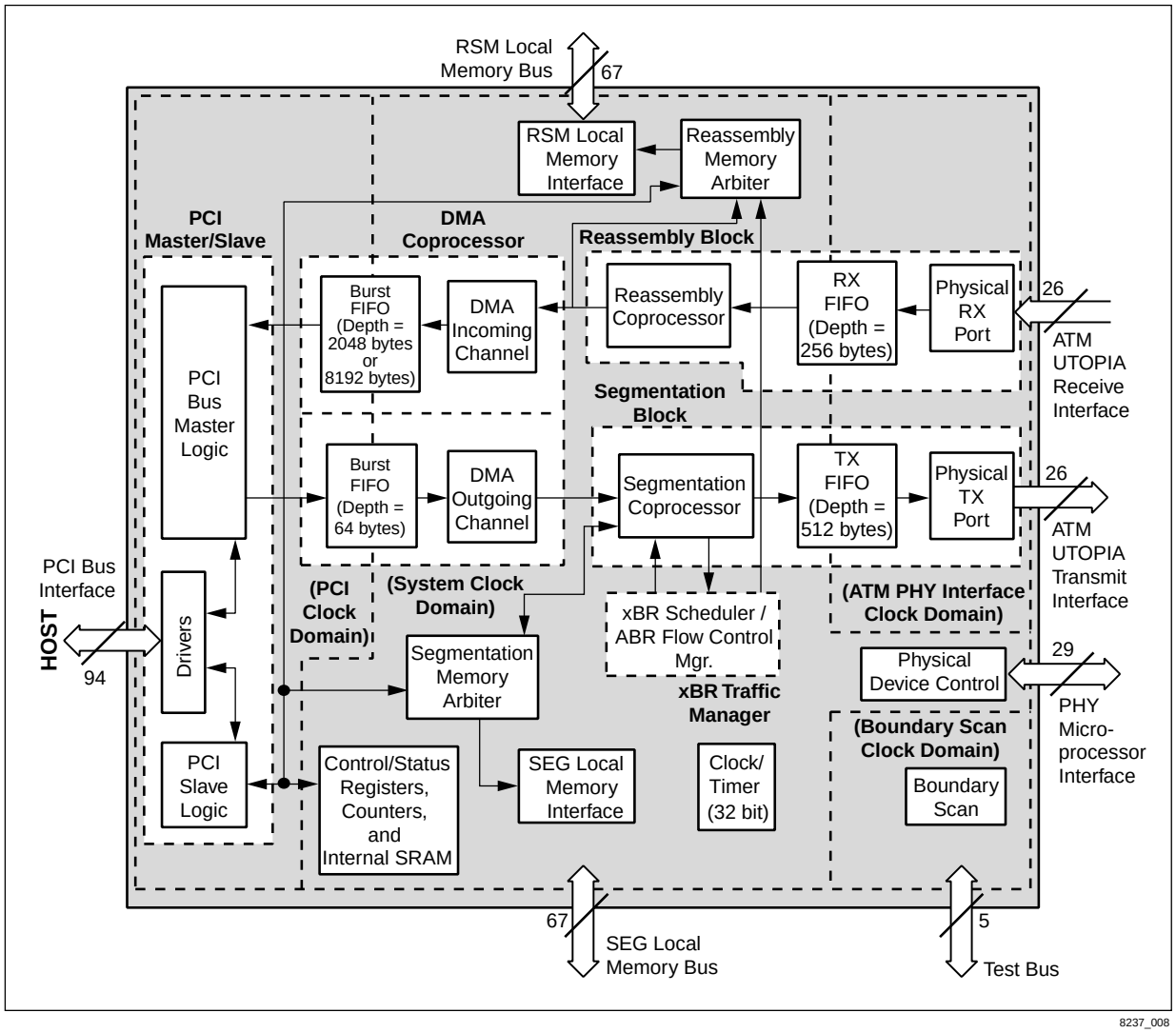
- Incoming DMA
- Outgoing DMA
- Reassembly
- Segmentation
- xBR Traffic Manager

Each coprocessor maintains state information in local memory. This memory is SAR-controlled through local bus interfaces and arbitrates access to the busses between the users. The coprocessors run off the same system clock but operate asynchronously from each other. Communication between the coprocessors takes place through on-chip FIFO buffers or through queues in SAR-shared memory (that is, memory local to the SAR and accessible to both the SAR and the host).

The CN8237 on-chip coprocessors are surrounded by high performance PCI and UTOPIA ports that provide glueless interfaces to a variety of systems. The interfaces have full line rate throughput and low bus occupancy.

Figure 1-1 illustrates CN8237's coprocessors and interfaces.

Figure 1-1. CN8237 Functional Block Diagram



1.2 Service-Specific Performance Accelerators

The CN8237 incorporates several service-specific features, which accelerate system performance. Some of these service level features provide the possibility for designers to accelerate specific protocol interworking functions. Other service level features enable network level functionality. These features are outlined in Chapter 2.0, and are fully described in succeeding chapters.

UNI or NNI Addressing

The CN8237 handles both User-Network Interface (UNI) addresses, which use an 8-bit Virtual Path Identifier (VPI) field, and Network-to-Network Interface (NNI) addresses, which use a 12-bit VPI field.

Frame Relay Interworking

The VBR traffic category includes rate-shaping via the dual leaky bucket Generic Cell Rate Algorithm (GCRA) based on the Cell Loss Priority (CLP) bit, for use in Frame Relay. The CN8237 also implements the Frame Relay discard attribute by performing early packet discard based on the frame's DE field and assigned discard priority.

IP Interworking

The CN8237 facilitates ATM call control signalling procedures as defined in ATM Forum's UNI Signalling 4.0 Specification (SIG 4.0), to support IP over ATM environments. Some of the SIG 4.0 capabilities that are of interest to IP over ATM and which the CN8237 allows for are as follows:

- ABR signalling for point-to-point calls
- Traffic parameter negotiation
- Frame discard support

Guaranteed Frame Rate

The CN8237 can rate-shape ATM Adaptation Layer Type 5 (AAL5) Common Part Convergence Sublayer Protocol Data Units (CPCS-PDUs, that is, frames) in the UBR service category, by providing a guaranteed Minimal Cell Rate (MCR) for UBR VCCs.

Early Packet Discard

The EPD feature provides a mechanism to discard complete or partial CPCS-PDUs based upon service discard attributes or error conditions. The reassembly coprocessor performs EPD functions under the following conditions:

- Frame Relay packet discard based on the DE field in the received frame and the channel exceeding a user-defined priority threshold.
- Packet discard based on the CLP bit.
- LANE-LECID packet discard to implement echo suppression on multicast data frames on ELAN channels.
- Packet discard when a firewall condition occurs on a VCC or group.
- Receive FIFO buffer full condition/threshold.

CBR Traffic Handling

The segmentation coprocessor includes an internal rate-matching mechanism to match the internal rate (the local reference rate) of CBR segmentation to an external rate (the host rate).

The user can direct the CN8237 to segment traffic from a fixed PCI address (that is, a Virtual FIFO buffer) for circuit-based CBR traffic.

The user can delineate up to 16 CBR pipes (or tunnels) in which to transmit multiple UBR, VBR, or ABR channels. In addition, the bandwidth of any single tunnel can be shared by up to four different priorities of traffic, establishing a multi-service tunnel. This allows proprietary management schemes to operate under preallocated CBR bandwidths.

ABR Traffic Management

The ABR Flow Control Manager dynamically rate-shapes ABR traffic independently per VCC, based upon network feedback. One or more ABR templates are used to govern the behavior of traffic.

- Both Relative Rate (RR) and Explicit Rate (ER) algorithms are used when computing a rate adjustment on an ABR VCC.
- Programmable ABR templates allow rate-shaping on groups of VCCs to be tuned for different network policies.
- New per-VCC MCR and ICR fields reduce the number of ABR templates needed in local memory.
- The CN8237 allows rate adjustments on Turnaround RM cells, based on congestion in the host.
- The CN8237 allows rate adjustments due to use-it-or-lose-it behavior.
- The CN8237 generates out-of-rate Forward RM cell(s) to restart scheduling of a VCC whose rate has dropped below the Schedule table minimum rate.
- The CN8237 optionally posts the current Allowed Cell Rate (ACR) on the segmentation status queue for the host monitoring functions.

VBR Traffic Management

The CN8237 schedules each VBR VCC according to GCRA parameters stored in the individual VCC control tables. The internal xBR Traffic Manager schedules the transmitted data to maximize the permitted link utilization. The actual rate sent is accurate to within 0.15% of the negotiated rates over a range from 10 cells per second to full line rate of 600 Mbps.

Three VBR modes are supported:

- Sustained cell rate (one leaky bucket)
- Peak and sustained cell rate (dual leaky bucket)
- CLP 0+1 shaping (supports committed/best effort services)
  (This is the mode recommended by the Internet Engineering Task Force [IETF] as the most convenient model for IP over ATM interworking.)

Virtual Path Networking

The CN8237 can interleave segmentation of numerous VCCs (that is, separate VC channels) as members of one Virtual Path (VP). VP-based traffic shaping is supported. The entire VP is scheduled according to parameters for one VCC.

AAL for Proprietary Traffic

The CN8237 incorporates an AAL0 traffic class for both segmentation and reassembly, which acts as an AAL level for proprietary use. Several options for packetization are implemented.

Internal SNMP MIB Counters

CN8237 has three internal counters that measure cells received, cells discarded, and AAL5 PDUs discarded (to meet ILMI and RFC1695 requirements).

CompactPCI Hot Swap

Circuity and I/O have been added in to functionally comply with the *CompactPCI Hot Swap Specification*. Electrical compliance is pending analysis. Refer to Sections 7.2 and 3.1.8 of the *PICMG Hot Swap Specification (PICMG 2.1 R1.0*) for details of the Hot Swap operation. The HSWITCH* input indicates the state of the handle switch. A logic low indicates that the handle is locked, whereas a logic high indicates that the handle is unlocked. The HLED* output is a 12 mA open drain capable of driving an LED directly. A logic low illuminates the LED. The HENUM* output is an 8 mA open drain in compliance with the ENUM# signal defined in the *CompactPCI* specification.

NOTE: If HSWITCH* is not used, it must be tied to ground.

1.3 Designer Toolkit

The CN8237 ATM evaluation environment provides evaluation capability for the CN8237 *Service*SAR. This environment serves as a hardware and software reference design for development of customer-specific ATM applications. The evaluation hardware and software was designed to provide a rapid prototyping environment to assist and speed customer development of new ATM products, thereby reducing product time to market.

This environment facilitates the following:

- rapid customer product development
- hardware reference design
- software reference design (based on VxWorks)
- traffic generation and checking capability

Comprising part of this development environment is the CN8237/RS8254 EVM, a PCI card specifically designed to be a full-featured ATM controller implementing the full functionality of the CN8237 *Service*SAR. The CN8237 resides at the heart of this PCI card.

The PCI interface between the host processor and the local system is controlled by Mindspeed's Hardware Programming Interface (CN823x HPI), a software driver to the CN8237, on top of which a system designer can develop and place proprietary driver software. This interface allows users to easily port their applications to the CN8237. This software is written in C, and Source code is available under license agreement.

The evaluation environment also includes a full set of design schematics, and artwork for the CN8237/RS8254 EVM PCI card.

2.0 Architecture Overview

2.1 Introduction

The CN8237 *Service*SAR architecture efficiently handles high bandwidth throughput across the spectrum of two different ATM Adaptation Layers (AAL0 and AAL5) and all ATM service categories. This chapter provides an overview of this architecture.

The first section describes the queue and data buffer system. The second section describes the segmentation and reassembly functions from within the context of the queue structures. The third section covers the xBR Traffic Manager. The remaining sections cover Operation and Maintenance and Performance Monitoring (OAM PM), the various device inputs and outputs, and the logic diagram with pin descriptions.

This architectural overview serves as a solid foundation for understanding the complete functionality of the CN8237.

2.2 High Performance Host Architecture with Buffer Isolation

Once initialized and given a segmentation or reassembly task, the CN8237 operates autonomously. Because the CN8237 is a high performance subsystem, the host/*Service*SAR architecture and the algorithms for task submission and status reporting have been optimized to minimize the control burden on the host system.

2.2.1 Multiple ATM Clients

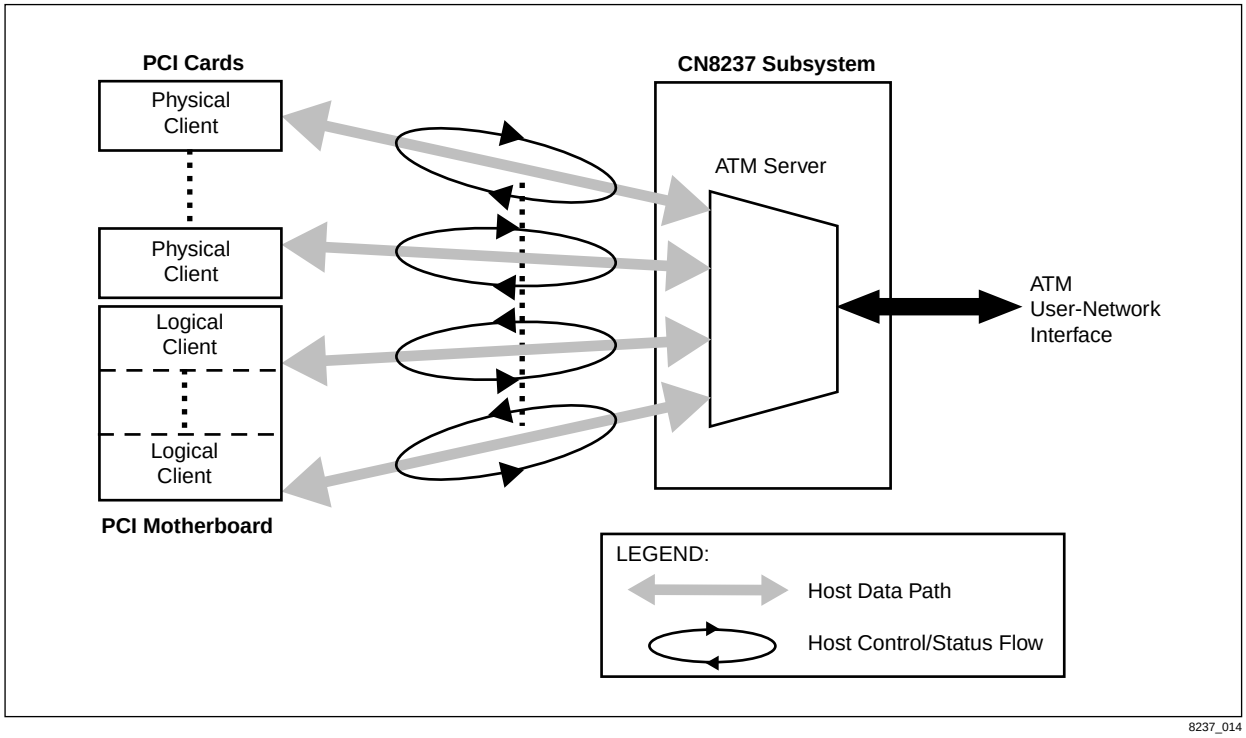
The CN8237, functioning as an ATM UNI, provides a high throughput uplink to a broadband network. Most individual ATM service users (or clients) do not have the bandwidth requirements to equal the throughput capability of the CN8237. As an application example, ATM clients can be Ethernet or Frame Relay ports. Therefore, many service clients are typically aggregated onto one ATM UNI. These clients may have very different needs and/or isolation requirements.

In order to fully capitalize on the high bandwidth of this service while meeting per-VCC Quality of Service (QoS) needs, the CN8237 functions as an ATM server for up to 32 clients. In this way, the bandwidth requirements of the service user (the client) can be balanced with the service throughput capability of the CN8237 and the rest of the specific system.

The CN8237 provides multiple independent control and status communication paths. Each communication path, or flow, consists of a control queue and a status queue for both segmentation and reassembly. The host assigns each of these independent flows to system clients, or peers. These can be either physical or logical entities. As throughput requirements escalate, the host system can add processing power in the form of additional peers. This degree of freedom creates a scalable host environment. Multiple VCCs can be assigned to each client (peer).

Each client interfaces to the CN8237 independently. Due to its server architecture, the CN8237 supplies the synchronization between asynchronous tasks requiring ATM services. Figure 2-1 illustrates this client/server model. The figure shows that clients can be multiple applications in a shared memory, or separate physical entities. All communicate directly with the CN8237.

Figure 2-1. Multiple Client Architecture Supports Up To 32 Clients (Peers)



2.2.2 CN8237 Queue Structure

The flow of the reassembly, scheduling, and segmentation processes in the CN8237 is monitored, coordinated, and controlled through the use of a full array of circular queues, serviced by the CN8237 or the host.

The following queues exist in local memory:

- Transmit queues (up to 32 queues)
- Reassembly/segmentation queue
- Free buffer queues (up to 32 queues)—includes the Global OAM free buffer queue.

Figure 2-2 illustrates the location of each queue.

Transmit queues are used by the host to submit chains of segmentation buffer descriptors to the CN8237 for segmentation. The segmentation coprocessor then processes these transmit queue entries as part of the segmentation function.

The reassembly/segmentation queue is written to by the reassembly coprocessor and read by the segmentation coprocessor. The queue includes data on OAM PM cells to be transmitted and as data on ABR-class received Backward_RM and Forward_RM cells. The RSM/SEG queue is a private queue for the SAR that the host cannot read or write.

The host furnishes data buffers to the reassembly processor by posting their location and availability to the free buffer queues, one of which can be designated as the Global OAM free buffer queue. The reassembly coprocessor uses the free buffer queue entries to allocate data buffers for received ATM cells during reassembly.

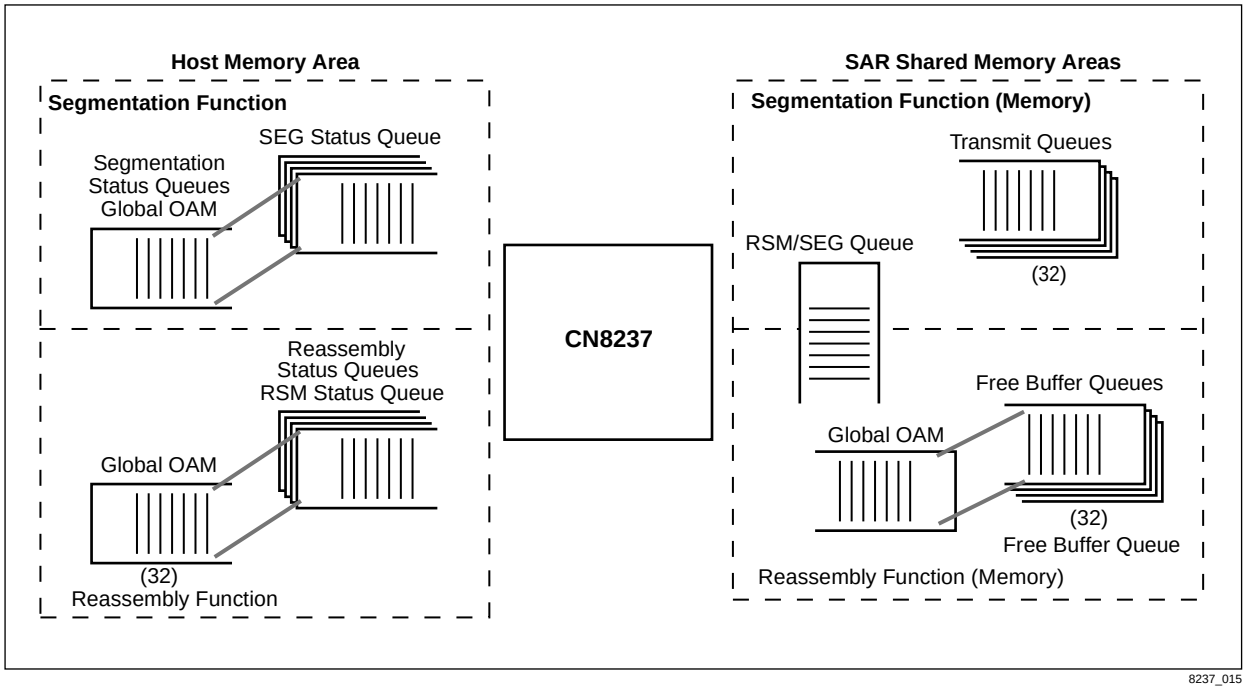
The following queues exist in host memory:

- Segmentation status queues (up to 32 queues)—includes the Global OAM segmentation status queue
- Reassembly status queues (up to 32 queues)—includes the Global OAM reassembly status queue

The CN8237 reports segmentation status to the segmentation status queues. One of these can be designated as the Global OAM segmentation status queue. The host further processes these segmentation status queue entries.

The CN8237 reports reassembly status to the reassembly status queues. One of these can be designated as the Global OAM reassembly status queue. The host further processes these reassembly status queue entries.

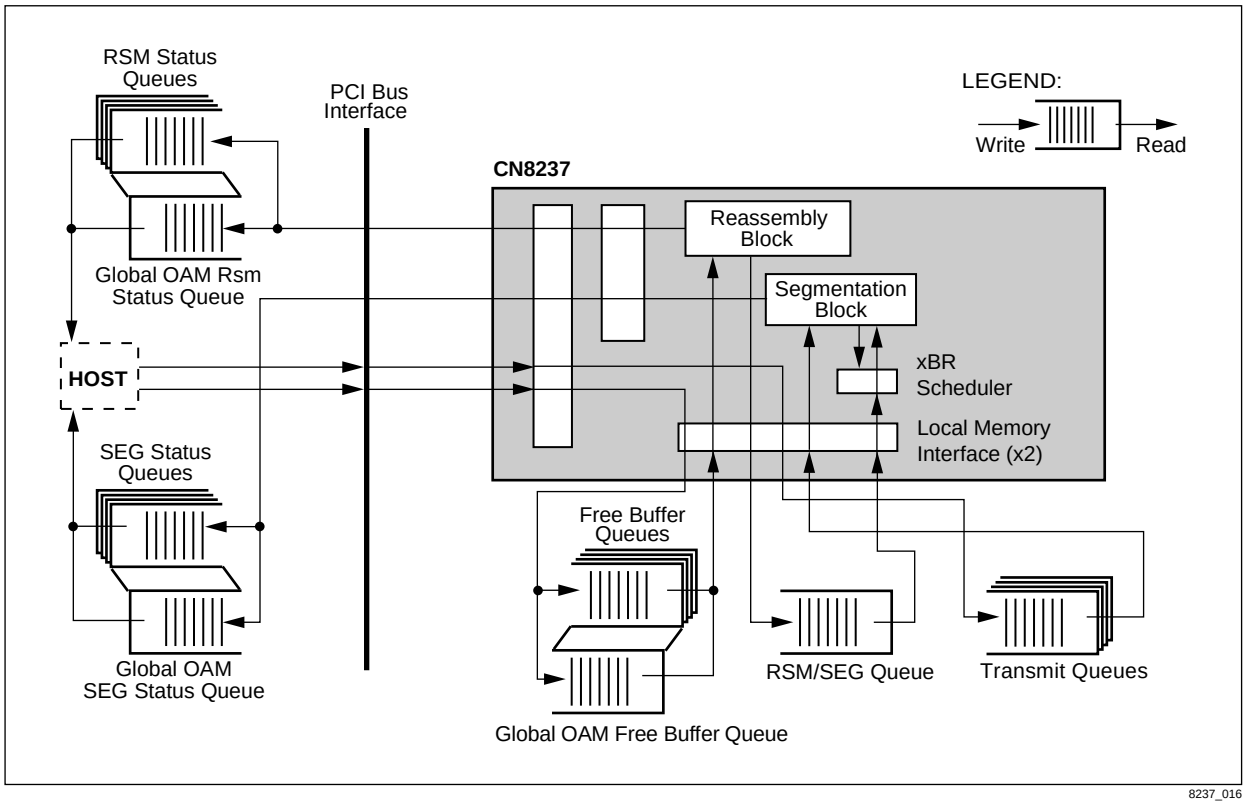
Figure 2-2. CN8237 Queue Architecture



The queues described above provide the control information that directs the reassembly and segmentation functions.

These queues, placed on asynchronous communication paths, directly associate the host with the CN8237 during processing and associate each of the major functional blocks of the CN8237 with each other. Figure 2-3 illustrates these interactions. The arrows indicate which system entity writes to each queue, and which entity reads each queue.

Figure 2-3. Interaction of Queues with CN8237 Functional Blocks



2.2.3 Buffer Isolation Utilizing Descriptor-Based Buffer Chaining

The CN8237 uses buffer structures for reassembly and segmentation. The buffer structures maximize the flexibility of the system architecture by isolating the data buffers from the mechanisms that handle buffer allocation and linking. This allows the data buffers to contain only payload data, and no control fields or other user fields. The user can store the data buffers separately from the buffer descriptors and implement a minimum data copy architecture.

Figure 2-4 illustrates how reassembly data buffers and buffer descriptors are chained together and manipulated by the *Service*SAR.

1. The host creates a link between a reassembly data buffer and a buffer descriptor by writing in the buffer descriptor entry, a pointer to the data buffer.
2. The host then formats a free buffer queue entry, which includes pointers to both the data buffer and buffer descriptor, and writes this message to the free buffer queue.
3. The reassembly coprocessor reads this free buffer queue entry and uses the pointer to the reassembly data buffer as the memory location to write the PDU being reassembled.
4. As reassembly of that PDU progresses, the reassembly coprocessor chains together the necessary number of additional buffer descriptors (and thus their associated reassembly data buffers) to complete reassembly of the PDU.

Figure 2-4. Reassembly Buffer Isolation—Data Buffers Separated from Descriptors

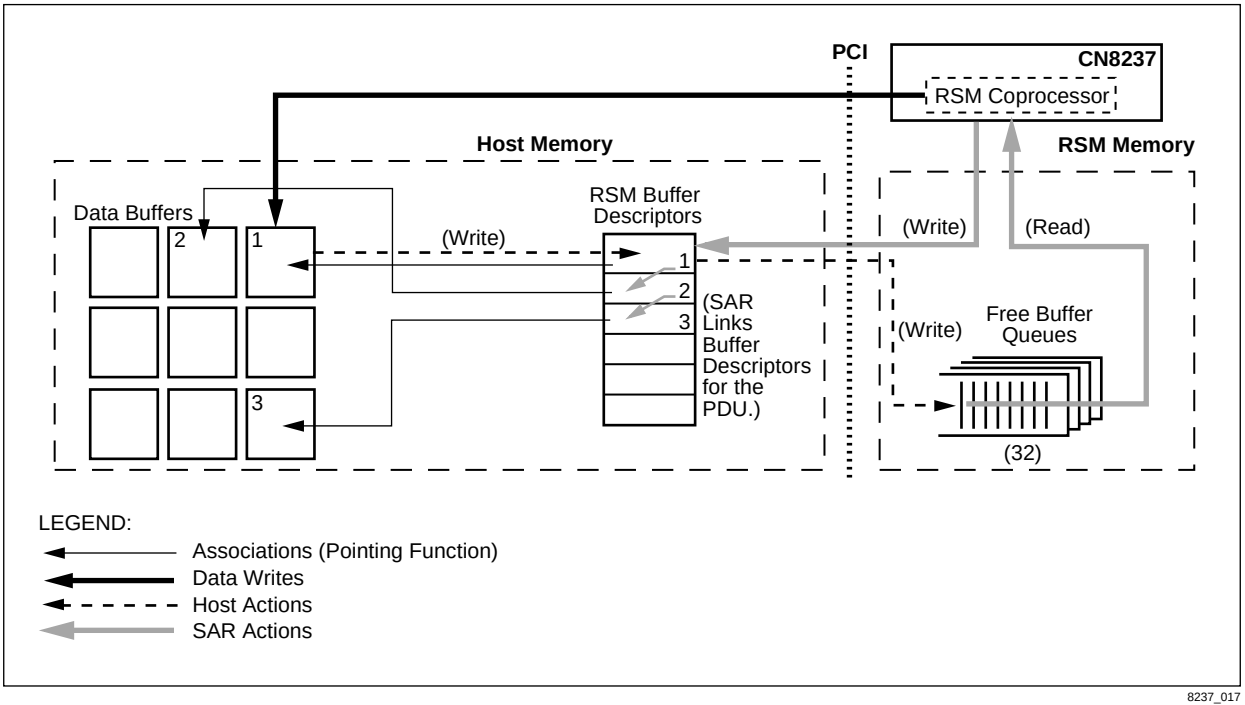
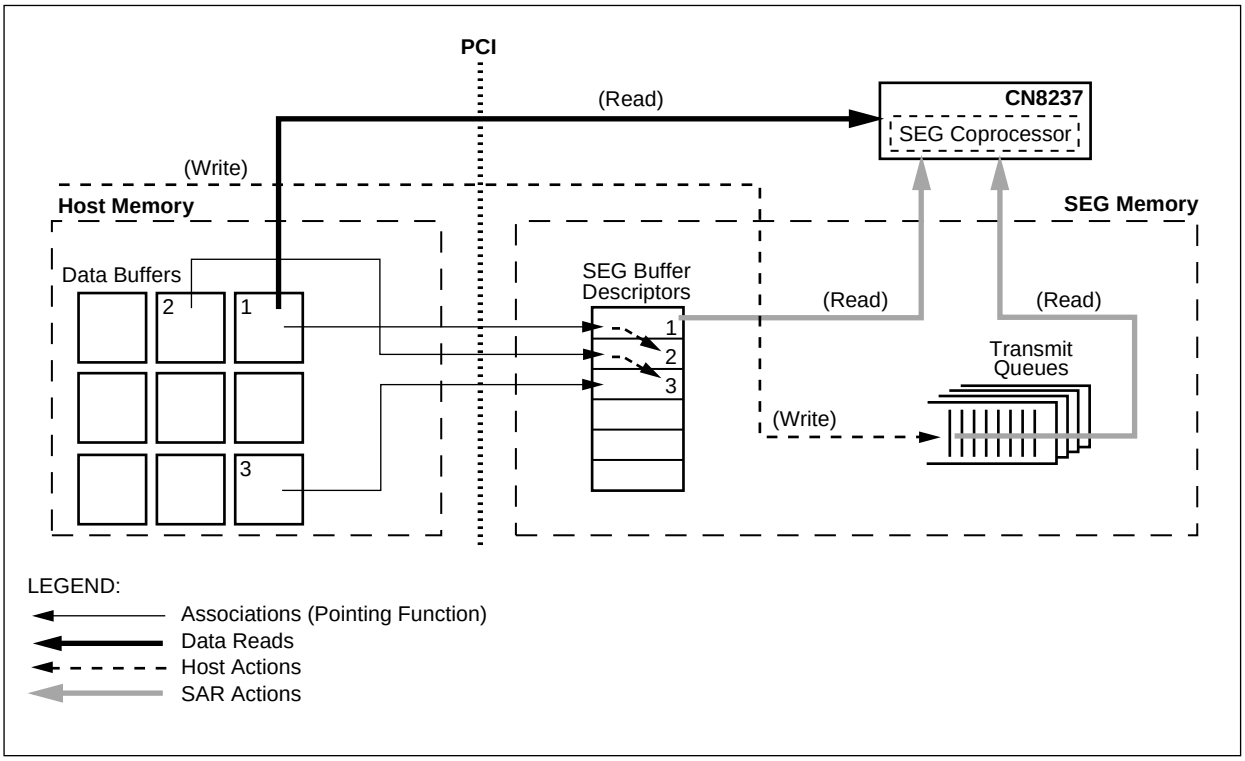


Figure 2-5 illustrates how the host submits linked data buffers (that is, PDUs) to the transmit queue for segmentation. The process is as follows:

1. The host links the buffer descriptors (in SEG memory) for the associated data buffers (in host memory) containing the PDU to be segmented.
2. The host then formats a 2-word transmit queue entry and writes this entry to the transmit queue. The location of the first buffer descriptor in the linked chain is contained in the transmit queue entry.
3. The segmentation coprocessor automatically senses the presence of new transmit queue entries, reads them, and schedules the new data for transmission. The transmit queue acts as a FIFO buffer for segmentation task pointers.

Figure 2-5. Segmentation Buffer Isolation—Data Buffers Separated from Descriptors



2.2.4 Status Queue Relation to Buffers and Descriptors

The status queues employed by the CN8237 are written by the SAR and read by the host. These status queue entries provide the data needed by the host in order to further process the segmentation and reassembly data flow in progress or just completed. Each status queue entry thus includes data (such as error flags and status bits), which the host uses in its succeeding process steps. The SAR also includes, in each status queue entry, a pointer to the first buffer descriptor of the segmented or reassembled data buffer(s) which comprise a single PDU. This accomplishes the other principal function of a status queue entry—to establish the association from the SAR to the host of the successful or unsuccessful segmentation or reassembly of a PDU.

Figure 2-6 illustrates the association between the segmentation status queues and segmentation data buffers and descriptors. The figure shows one three-buffer PDU on a single virtual channel, represented by a single entry in one of the transmit queues and a single entry in one of the SEG status queues. The host links the buffer descriptors pointing to the data buffers containing the PDU, makes a host-only copy of the buffer descriptor, then writes the transmit queue entry. The SAR performs segmentation processing on the PDU and writes a SEG status queue entry informing the host of the status of the segmentation process.

Figure 2-6. Segmentation Status Queues Related to Data Buffers and Descriptors

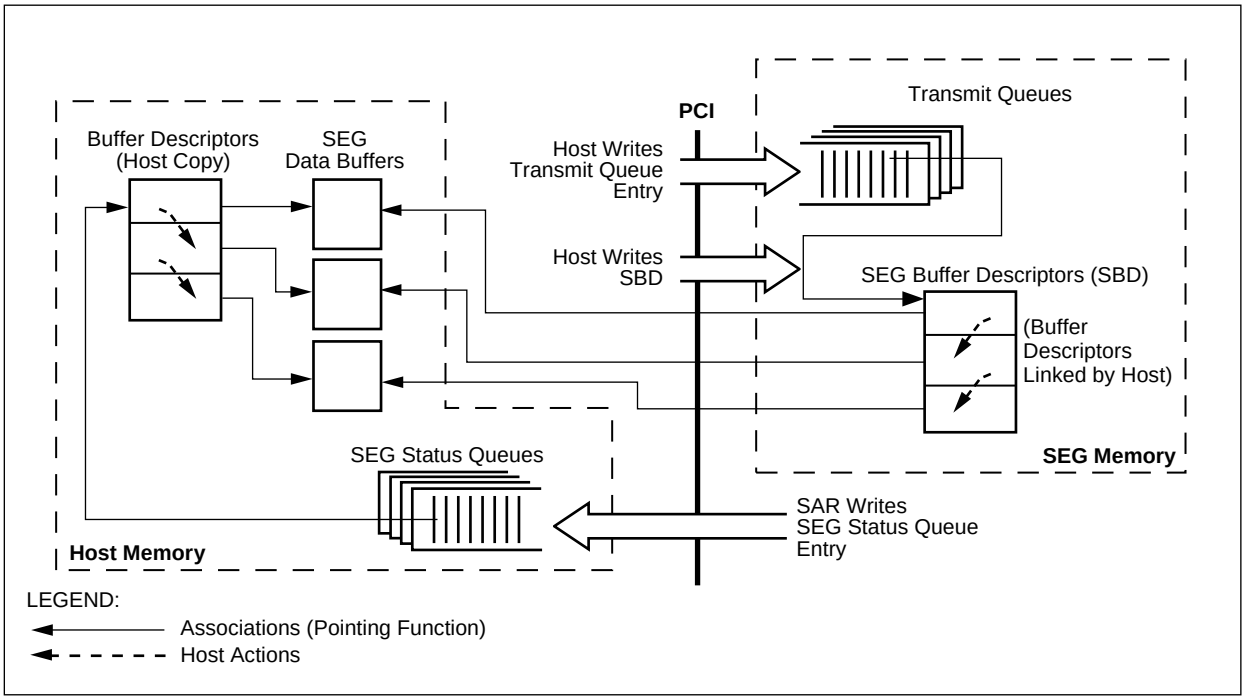
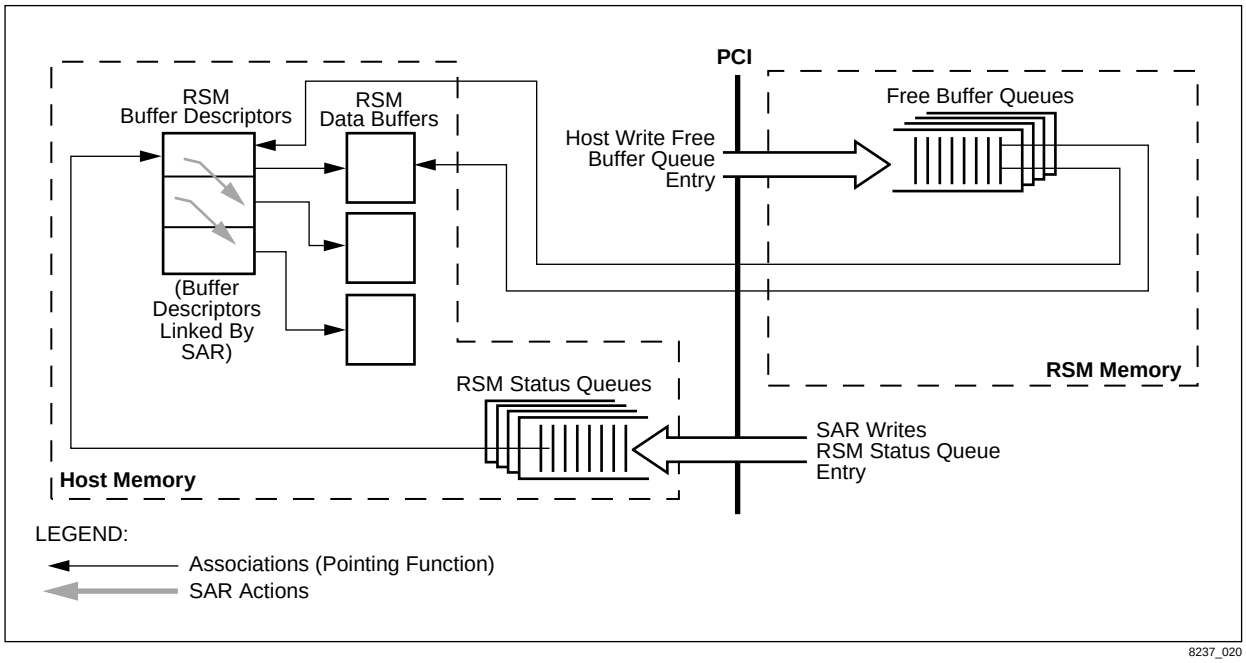


Figure 2-7 illustrates the association between the reassembly status queues and reassembly data buffers and descriptors. The host submits free buffers to the SAR by writing pointers to them in the free buffer queue entries. The SAR links the buffer descriptors pointing to the three data buffers containing the reassembled PDU, and writes the RSM status queue entry containing the pointer to the first buffer descriptor for that PDU. The host further processes the PDU utilizing that data.

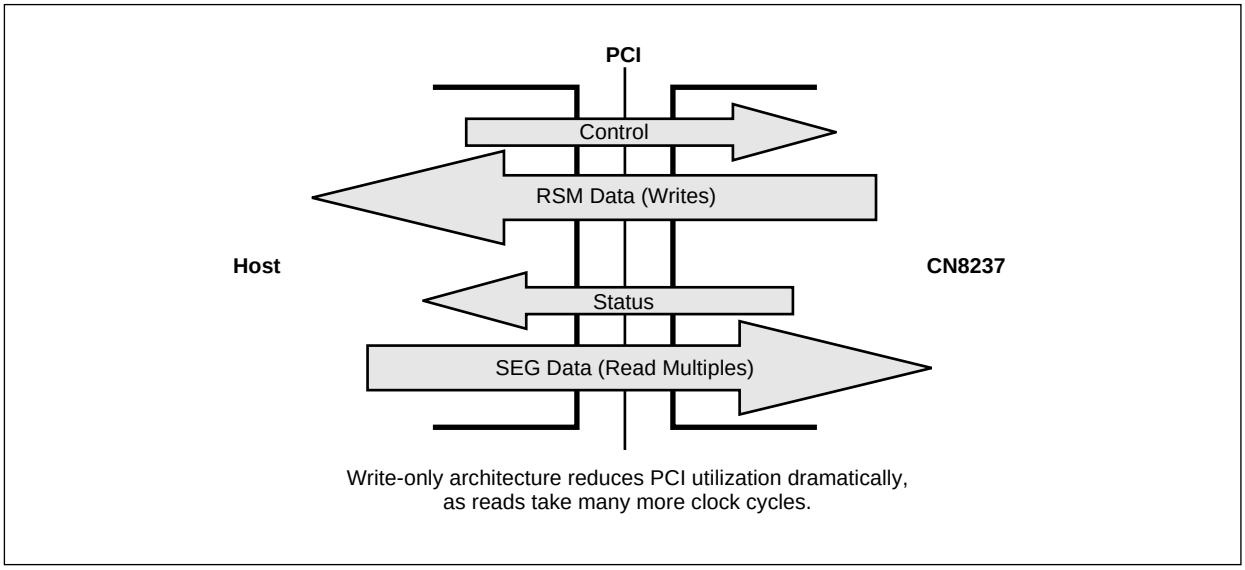
Figure 2-7. Reassembly Status Queues Related to Data Buffers and Descriptors



2.2.5 Write-only Control/Status

Figure 2-8 illustrates the CN8237's write-only PCI control architecture. The host manages the CN8237 ATM terminal using write-only control and status queues. This architecture minimizes PCI bus utilization by eliminating reads from control activities. PCI writes utilize the bus much more efficiently than PCI reads. During a PCI write, the Bus Master can post the write data to an internal FIFO buffer in the slave, terminate the transaction, and immediately release the bus. On the other hand, during PCI reads, the Bus Master retrieves the data from the slave while holding the bus. Since the data retrieval takes some time, reads increase the PCI bus utilization time for each transaction. The CN8237 eliminates read operations except for burst reads to gather segmentation data.

Figure 2-8. Write-only Control and Status Architecture



2.2.6 Scatter/Gather DMA

The CN8237's Direct Memory Access (DMA) coprocessor works in close conjunction with the segmentation and reassembly coprocessors to gain access to the PCI bus, transfer the requested data, and notify the segmentation or reassembly coprocessor that the transfer is complete. The DMA coprocessor transfers all data using the read and write burst buffers in the PCI Bus Interface.

In general, two types of transactions are processed:

- For 32-bit PCI
  - 12 or 14 word accesses for data
  - 1 to 4 word accesses for control and status messages
- For 64-bit PCI
  - 6 or 7 double word accesses for data
  - 1 or 2 double word accesses for control and status messages

For outgoing messages, the DMA coprocessor moves data from host memory to the segmentation coprocessor using a gather DMA method. For incoming messages, the DMA coprocessor moves data from the reassembly coprocessor to host memory using a scatter DMA method.

The DMA coprocessor can handle transfers from the PCI bus with data that is not aligned on word boundaries. It also selectively transfers data to comply with either a big endian or little endian host data structure.

2.2.7 Interrupts

The CN8237 informs the host of segmentation and reassembly activity by means of maskable interrupts sent to the host processor, triggered by writes to the segmentation and reassembly status queues.

The user can configure the SAR to generate these status queue entries and interrupts at PDU boundaries (called Message Mode), or at data buffer boundaries (called Streaming Mode).

The CN8237 can also be configured with a status queue interrupt delay, which can be enabled in order to reduce the interrupt processing load on the host. This has value when the SAR resides in an environment in which the host is not dedicated to data communications processing.

2.3 Automated Segmentation Engine

The CN8237 can segment up to 64 K VCCs simultaneously. The segmentation coprocessor block independently segments each channel and multiplexes the VCCs onto the line with cell level interleaving. For each cell transmission opportunity, the xBR Traffic Manager tells the segmentation coprocessor which VCC to send.

The CN8237 provides full support of the AAL5 protocol and a transparent or NULL adaptation layer, AAL0.

Each segmentation channel is specified as a single entry in the segmentation VCC table located in SEG memory. A VCC specifies a single VC or VP in the ATM network. These VCC table entries define the negotiated or contracted characteristics of the traffic for that channel, and are initialized by the host either during system initialization or on-the-fly during operation. An initialized segmentation VCC table entry effectively establishes a connection on which data can be segmented.

NOTE: ABR VCCs occupy two table entries.

The host submits data for segmentation by first linking buffer descriptors that point to the buffers containing the PDU to be transmitted, and then submitting that chained message to the SAR by writing to one of 32 independent circular transmit queues.

The segmentation coprocessor then operates autonomously, formatting the cells on each channel according to the host-defined segmentation VCC table entries for each channel. The formatting functions include the following:

- The segmentation coprocessor formats the ATM cell header for each cell, based on the settings in the segmentation VCC table entry for that VCC.
- The segmentation coprocessor also generates the CPCS-PDU header and trailer fields in the first and last cell of the segmented PDU.
- For AAL5 traffic, the SEG coprocessor also generates the PDU-specific fields in the trailer of the CPCS-PDU, and places these in the last cell (the End of Message [EOM] cell) for the PDU.
- AAL0 is intended for client-proprietary use. For AAL0, the segmentation coprocessor segments the Service Data Unit (SDU) to ATM cell payload boundaries and generates ATM cell headers, but generates no other overhead fields.
- The user has per-channel, per-PDU control of Raw Cell Mode segmentation, wherein the segmentation coprocessor reads the entire 52-octet ATM cell from the segmentation buffer and does not generate the ATM headers for the cells.
- The formatted cells are passed through the transmit FIFO buffer to the PHY interface for transmission.

The system designer can set the depth of the transmit FIFO buffer from one to nine cells deep in order to optimize the balance between Cell Delay Variation (CDV), which increases with longer transmit FIFO buffer depth, and PCI latency protection, which decreases with shorter transmit FIFO buffer depth.

The CN8237 provides a method to segment traffic from a fixed PCI address (or Virtual FIFO buffer). This is intended for circuit-based CBR traffic such as voice channel(s).

The CN8237 reports segmentation status to the host on one of a set of 32 independent parallel segmentation status queues. The CN8237 writes segmentation status queue entries on either PDU boundaries or buffer boundaries, selectable on a per-VCC basis. PDU boundary status reporting is called Message Mode, while buffer status reporting is called Streaming Mode.

2.4 Automated Reassembly Engine

The reassembly coprocessor processes cells received from the ATM PHY Interface block. The coprocessor extracts the AAL SDU payload from the received cell stream and reassembles this information into buffers supplied by the host system.

Each active reassembly Channel is specified as a single entry in the reassembly VCC table located in RSM memory. One RSM VCC table entry defines the negotiated or contracted characteristics of the reassembly traffic for a particular channel. Each table entry is initialized by the host during system initialization, or on-the-fly. The SAR uses the RSM VCC table to store temporary information to assist the reassembly process. An initialized Reassembly VCC table entry effectively establishes a connection on which the CN8237 can reassemble data.

Using a dynamic Channel Directory lookup method, the CN8237 reassembles up to 64 K VCCs simultaneously at a maximum rate of 600 Mbps on simplex and full-duplex connections. The Channel Directory mechanism allows flexible preallocation of resources and provides deterministic channel identification over the full UNI or NNI Virtual Path Identifier/Virtual Channel Identifier (VPI/VCI) address space. The total number of VCCs supported is limited by the memory allocated to the RSM VCC table and the Channel Directory.

The reassembly coprocessor extracts the AAL SDU payload from the received cell stream and reassembles this information into system buffers allocated per-VCC. The CN8237 supports AAL5 and AAL0 reassembly, as well as 52-octet Raw Cell mode.

For AAL5, the reassembly coprocessor extracts and checks all PDU protocol overhead.

The CN8237 provides two methods of terminating an AAL0 PDU:

1. Payload Type Identifier (PTI) termination, wherein the PTI bit in the cell header is monitored for the End of Message (EOM) cell indication.
2. Cell Count termination, wherein the CN8237 terminates the PDU when a user-defined number of cells have been received on that channel.

The AAL0 PDU termination method is selectable on a per-VCC basis.

The user can, on a per-channel basis, establish Raw Cell mode reassembly. In this mode the Header Error Check (HEC) octet is deleted to align the 53-octet cell to 32-bit boundaries, and the RSM coprocessor reassembles the entire 52-octet ATM cell into the reassembly buffer.

The CN8237 provides the user with generous per-channel control of the reassembly process, including the following:

- Assignment of priorities for reassembly buffer return processing.
- Cell filtering on inactive channels.
- Mechanisms to establish per-VCC firewalling by allocating buffer credits on a per-channel basis. (This limits the possibility of one VCC consuming all of the memory resources.)
- Per-VCC activation and control of a background hardware time-out function where the user selects one of eight programmable time-out periods. (The background function then automatically detects partially reassembled PDUs and reports this status to the host so that these buffers can be recovered and re-allocated.)
- Per-VCC monitoring of the length of the reassembled PDU, with status reporting if the length exceeds a set maximum length for that channel.

The CN8237 implements an early packet discard feature to enable discarding of complete or partial CPCS-PDUs based upon service discard attributes or error conditions. The early packet discard function halts reassembly of the CPCS-PDU marked for discard until the next Beginning of Message (BOM) cell and/or the error condition has cleared. The SAR writes a status queue entry with the appropriate status flags set, which indicate the reason for the discard. This function can be enabled for the following conditions:

- Frame Relay discard based on the frame's DE setting and the channel exceeding a user-defined priority threshold.
- CLP packet discard based on the received cell's CLP bit setting and exceeding channel priority threshold.
- LANE-LECID packet discard on ELAN channels, which implements echo suppression on multicast data frames.
- Early packet discard on AAL5 channels when the reassembled PDU length exceeds the user-defined maximum PDU length for that VCC.
- Early packet discard on channels encountering a free buffer queue empty (underflow) condition (meaning there are no available buffers in the free buffer queue that channel is assigned to).
- Early packet discard on PDUs when a DMA Incoming FIFO buffer Full condition occurs.
- Early packet discard on channels encountering a reassembly status queue full (overflow) condition.

The system designer can set the reassembly status reporting for any channel to either Message Mode or Streaming Mode. In Message Mode, a status entry is written only when the last buffer in a message completes reassembly. In Streaming Mode, a status entry is written for each buffer as it completes reassembly.

2.5 Advanced xBR Traffic Management

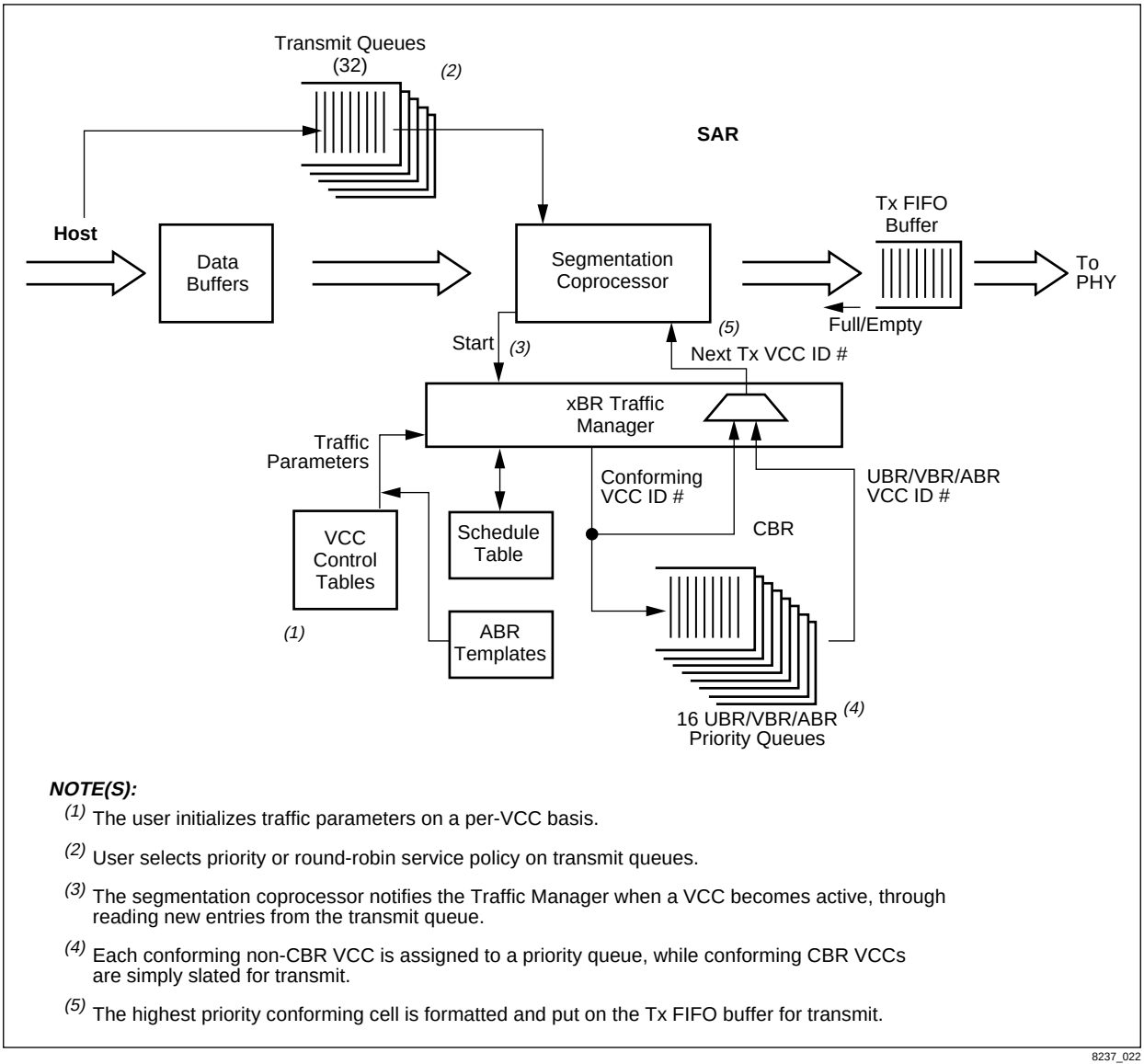
The CN8237 implements ATM's inherent robust traffic management capabilities for CBR, VBR, ABR, UBR, GFR, and GFC. The CN8237 manages each VCC independently and dynamically.

- The user assigns each connection a service class, a priority level, and a rate if applicable. The on-chip traffic controller (xBR Traffic Manager) optimizes use of the line bandwidth according to the VCC's traffic parameters and control information stored in SEG memory. The xBR Traffic Manager guarantees the compliance of each VCC to its service contract with the ATM network at the UNI ingress point. It schedules all data traffic by acting as a master to the segmentation coprocessor.

  One of the functional components of the xBR Traffic Manager is the xBR Scheduler. The xBR Traffic Manager assigns segmentation traffic from active VCCs to schedule slots, which the segmentation coprocessor then complies to by segmenting VCC traffic in the sequence/schedule dictated by the xBR Scheduler.
- In addition to reserved CBR bandwidth, the CN8237 provides 16 segmentation priorities. The user configures these priorities for the remaining service categories, including the *TM 4.1* defined ABR class. The CN8237's xBR Traffic Manager implements multiple functional levels of traffic prioritizing. This is illustrated in Figure 2-9.
- The host submits data to be sent by writing entries to the transmit queues for segmentation. The SAR processes these transmit queues either in round–robin order (transmit queue 0–31, looped back to 0), or in priority order (with transmit queue 31 having highest priority). This scheme gives the user or system designer some control of the delay between the host submitting traffic and the SAR starting to process that traffic. For instance, the user could assign CBR traffic to the highest priority transmit queue in order to minimize any delay in processing and scheduling that traffic.
- The CN8237 then submits this traffic demand to the xBR Traffic Manager for scheduling. Traffic is scheduled based on the traffic class plus certain parameters from the segmentation VCC table entries (primarily the GCRA *I* and *L* parameters). And if the service category is ABR, the SAR also uses certain parameters from the ABR templates to help determine that traffic's placement on the Schedule table.
- The conforming traffic to be transmitted is further groomed in internal priority queues. Each virtual channel is prioritized according to its assigned scheduling priority. CBR channels are given pre-assigned segmentation bandwidth, and channels for the remaining service categories scheduled according to their priority number (priority 0 being the lowest priority and priority 15 being highest).

Figure 2-9 shows this prioritization as a global set of queues, but it is actually maintained on a per-transmit opportunity basis. In this way, high priority traffic is transmitted up to its GCRA limits but does not block lower priority traffic when idle. The CN8237 asynchronously multiplexes traffic based on the above schemes, as the Tx FIFO buffer empties.

Figure 2-9. Multi-Level Model for Prioritizing Segmentation Traffic



NOTE(S):

(1) The user initializes traffic parameters on a per-VCC basis.

(2) User selects priority or round-robin service policy on transmit queues.

(3) The segmentation coprocessor notifies the Traffic Manager when a VCC becomes active, through reading new entries from the transmit queue.

(4) Each conforming non-CBR VCC is assigned to a priority queue, while conforming CBR VCCs are simply slated for transmit.

(5) The highest priority conforming cell is formatted and put on the Tx FIFO buffer for transmit.

8237_022

2.5.1 CBR Traffic

The CBR service category requires guaranteed transmission rates, and constrained CDV. The CN8237 facilitates these needs when generating CBR traffic by pre-assigning specific schedule slots to CBR VCCs. For each CBR-assigned cell slot, the CN8237 generates a cell for that specific VCC unless data is not available. The CN8237 also minimizes CDV by basing all traffic management on a local reference clock.

The CN8237 provides a mechanism to exactly match the scheduled rate of a CBR channel to the rate of its data source. To accomplish this rate-matching, the host can occasionally instruct the xBR Scheduler to skip one transmit opportunity on a channel.

The CN8237 manages CBR tunnels in the same manner as a CBR VCC. However, instead of one VCC, several UBR, VBR or ABR VCCs can be scheduled within this CBR tunnel, in round-robin order.

Unused CBR and Tunnel time slots are automatically made available to VCCs of other service categories by the xBR Scheduler.

2.5.2 VBR Traffic

The CN8237 takes advantage of the asynchronous nature of ATM by reserving bandwidth for VBR channels at average cell transmission rates without pre-assigning hardcoded schedule slots, as with CBR traffic. This dynamic scheduling allows VBR traffic to be statistically multiplexed onto the ATM line, resulting in better utilization of the shared bandwidth resources. The xBR Scheduler supports multiple priority levels for VBR traffic. Through the combination of VBR parameters and priorities, it is possible to support real-time VBR services.

The outgoing cell stream for each VBR VCC is scheduled according to the GCRA algorithm. The GCRA *I* and *L* parameters control the per-VCC Peak Cell Rate (PCR) and Cell Delay Variation Tolerance (CDVT) of the outgoing cell stream on any channel. This guarantees compliance to policing algorithms applied at the network ingress point. The user can control the granularity of rate by dictating the number of schedule slots in the schedule table.

Channels can be rate-shaped as VCs or VPs according to one of three VBR definitions. VBR1 controls PCR and CDVT. VBR2 controls PCR and CDVT, as well as Sustained Cell Rate (SCR) and Burst Tolerance (BT). VBRC (also called VBR3) controls PCR and CDVT on all cells, but controls SCR on only CLP = 0 (that is, high priority) cells.

2.5.3 ABR Traffic

The CN8237 implements the ATM Forum ABR flow control algorithms. The CN8237 acts as a fully compliant ABR Source and Destination, as defined in the *TM 4.1 ATM Source* specification. The ABR service category effectively allows low cell loss transmission through the ATM network by regulating transmission based upon network feedback. The ABR algorithms regulate the rate of each VCC independently.

The CN8237 employs an internal feedback control loop mechanism to enable the *TM 4.1* specification. The SAR utilizes the dynamic rate adjustment capability of the xBR Scheduler as the ATM Source's variable traffic rate-shaper. The CN8237 injects an in-rate stream of Forward RM cells for each ABR VCC. When these cells return to the CN8237's receive port as Backward RM cells after a round trip through the network, the CN8237 processes these cells and uses the data returned as feedback to dynamically adjust the rates on each ABR channel.

The CN8237 also responds to an incoming ABR cell stream as an ABR Destination. The reassembly coprocessor processes received Forward RM cells. It turns around this incoming information to the segmentation coprocessor, which formats Backward RM cells containing this information, and inserts these Turnaround RM cells into the transmit cell stream.

The exact performance of the rate-shaper is governed by one or more ABR Templates in SAR-shared memory. Each VCC is assigned to one of these templates. The templates control such behaviors as the size of the additive rate increase factors or multiplicative rate decrease steps. Each VCC's rate varies across the template independently.

2.5.4 UBR Traffic

The UBR service category is intended for non-real-time applications that do not require tightly constrained delay and delay variation, such as traditional computer communications applications like file transfer and e-mail.

Those VCCs which have not been assigned to one of the other service categories covered previously are scheduled as UBR traffic. All UBR channels within a priority are scheduled on a round-robin basis. To limit the bandwidth that a UBR priority consumes, the system designer should use a CBR tunnel in that priority level.

2.5.5 GFR Traffic

Guaranteed Frame Rate is a new service category defined by the ATM Forum to provide a MCR QoS guarantee for AAL5 CPCS-PDUs not exceeding a specified frame length. A GFR service connection is treated as UBR with a guaranteed MCR.

The CN8237 implements GFR by scheduling/shaping the connections using both the VBR1 scheduling procedure (for the MCR rate value) and a UBR priority queue, thereby providing fair sharing for all GFR connections to excess bandwidth.

2.5.6 xBR Cell Scheduler

The xBR Scheduler slates traffic for transmission according to a Dynamic Schedule table maintained in SEG memory. The table contains a user-programmable number of schedule slots. The duration of a single slot is a user-programmable number of system clock cycles. The xBR Scheduler sequences through this table in a circular fashion to schedule traffic. By configuring the number of slots in the table and the duration of each slot, the system designer chooses a range of available rates. A specific rate for any channel is determined by how many slots in the table to which that channel is assigned. Schedule slots not reserved for CBR during table setup are used for the rest of the service categories.

The xBR Scheduler implements Mindspeed's proprietary per-VCC rate-shaping algorithms. The predecessor to the CN8237, the Bt8230 SAR, proved the core algorithms. The CN8237 extends their use to other service classes. The CN8237 xBR Scheduler shapes all traffic classes, including CBR, single leaky bucket VBR, dual leaky bucket VBR, ABR, and UBR. The host configures the Dynamic Schedule table during system initialization, defining the table size in number of schedule slots and the length of each schedule slot in clock cycles. After setup, the CN8237 dynamically manages the entire table.

Some key features of the xBR Scheduler are as follows:

1. Per-VCC rate control guarantees conformance to GCRA UPC/policing
2. Dynamic reallocation of link bandwidth to active channels
3. Dynamic, fair sharing of bandwidth on oversubscribed lines
4. Multiple scheduling priorities
5. Fine grained rate control
6. Rate based on a user supplied reference clock

Dynamic management provides on-the-fly reallocation of link bandwidth without host intervention. The CN8237 fairly distributes the link bandwidth to channels based upon their QoS parameters and assigned transmission priority. As covered previously, the CN8237 supports 16 priorities in addition to preallocated CBR time slots.

The xBR Scheduler facilitates advanced network traffic management topologies. The CN8237 rate shapes VCs or VPs. Additionally, CBR tunneling allows UBR, VBR and/or ABR traffic management schemes to operate under a preallocated CBR limit.

2.5.7 ABR Flow Control Manager

The ABR Flow Control Manager operates in conjunction with the xBR Scheduler to control the rate of ABR channels. The CN8237 implements the *TM 4.1* specification in a template-controlled hardware state machine. Mindspeed provides an initial set of templates which reside in SEG memory. The information within these templates define conformant ABR behavioral responses to network and connection states. The CN8237 generates ABR Source traffic, including internally generated RM cells, according to the template instructions. The reassembly coprocessor and the Flow Control Manager collaboratively act as a fully compliant ABR Destination Terminal.

These templates provide three significant benefits to the user:

1. Since they control the Flow Control Manager state machine, they can be optimized for specific applications.
2. The programmability of the templates insulates the hardware from changes in the relatively stable, yet immature, *TM 4.1* specification.
3. Mindspeed provides the initial templates, which can be customized by the user later, shortening development time.

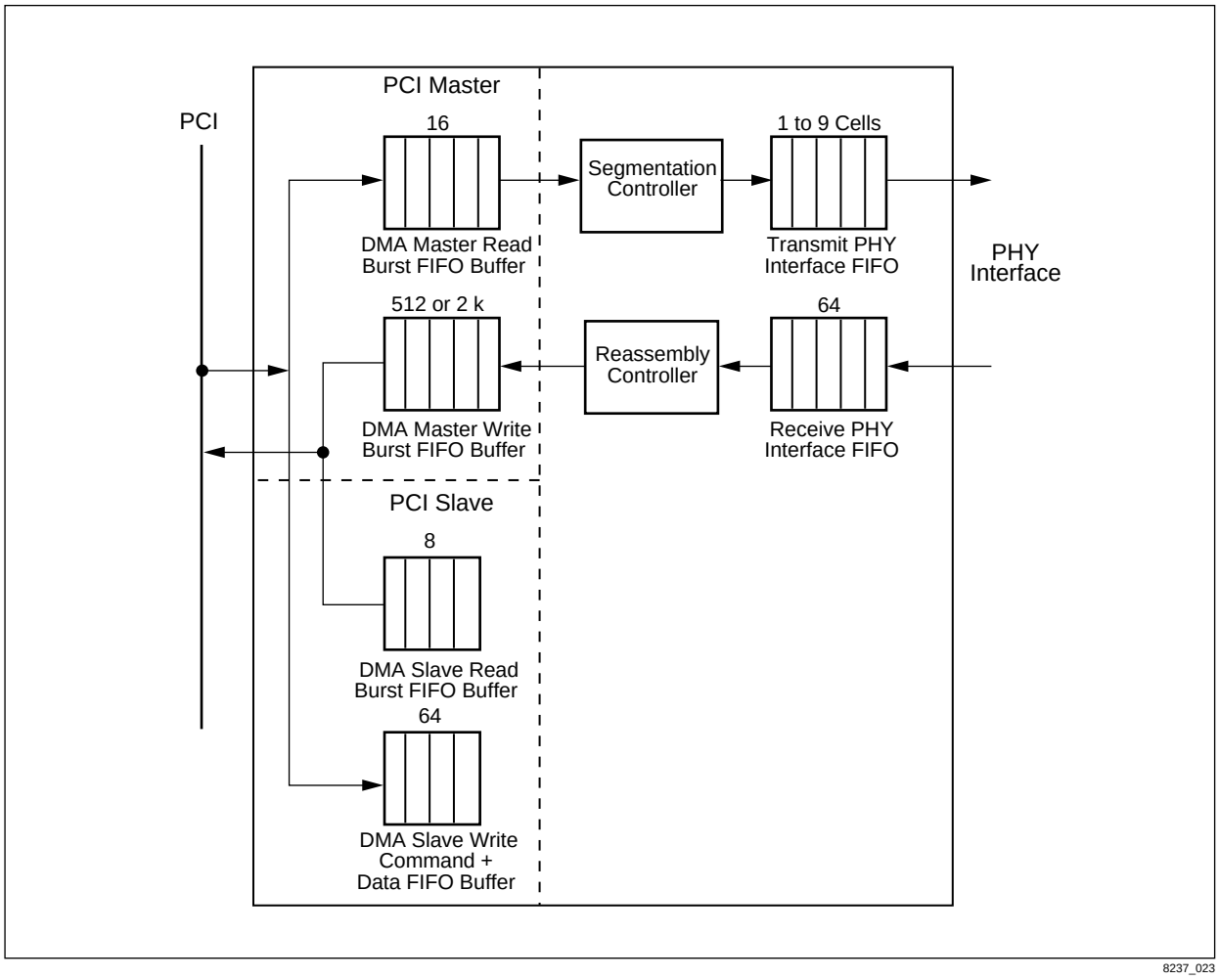
2.6 Burst FIFO Buffers

To conserve local memory bandwidth, the CN8237 does not use its local SRAM as a buffer for incoming or outgoing data. Instead, the CN8237 uses six dedicated internal FIFO buffers data as follows:

- Two DMA master burst FIFO buffers (read =16 words; write, 512 or 2 K words, programmable via CONFIG1 bit 1).
- Two DMA slave burst FIFO buffers, (read = 8 words and write = 64 words).
- One FIFO buffer between the PHY interface and the segmentation coprocessor (1 to 9 cells). See Section 4.2.4.
- One FIFO buffer between the PHY interface and the reassembly coprocessor (64 words).

Figure 2-10 illustrates the data FIFO buffer.

Figure 2-10. Data FIFO Buffers



2.7 Implementation of OAM-PM Protocols

The CN8237 provides internal support for the detection and generation of OAM traffic, including PM OAM.

The CN8237 supports the F4 and F5 OAM flows according to I.610. It monitors up to 128 channels and generates in-rate PM-OAM cells.

2.8 Standards-Based I/O

PCI Bus I/O

The PCI bus interface implements the full set of address, data, and control signals required to drive the PCI bus as a master, and contains the logic required to support arbitration for the PCI bus. This interface is PCI Version 2.1-compliant.

The PCI bus interface also includes an interface module that allows the PCI core to connect to a serial EEPROM. This 128-byte EEPROM is used to store specific PCI configuration information, loaded into the PCI configuration space at reset. This allows for several user-configurable features:

- User control of the size of the memory block for PCI addresses
- Enabling byte swapping of control words across the PCI bus
- Loading of Subsystem ID and Subsystem Vendor ID

ATM PHY I/O

The CN8237's ATM PHY interface communicates with and controls the ATM link interface device, which carries out all the transmission convergence and PHY media dependent functions defined by the ATM protocol. Two modes of operation are provided: standard UTOPIA and slave UTOPIA. Standard UTOPIA mode conforms to both UTOPIA Level 1 and Level 2 standards for ATM Layer devices. Slave UTOPIA mode reverses the control direction for use in place of a PHY on switch fabrics.

SAR Shared Memory I/O (SEG and RSM)

To simplify system implementations, the CN8237 integrates two complete memory controllers designed for direct interface to either ZBT or syncburst synchronous SRAM. There are two CN8237 memory controllers: one for reassembly and one for segmentation. They operates up to 66 MHz and each controller can access up to 4 MB of either ZBT or syncburst synchronous SRAM memory. The memory controller also arbitrates access to the internal control and status registers by the host processor. The memory banks can be configured to a variable number of sizes. All of this affords a wide degree of flexibility in SAR-shared memory architecture.

Physical Layer Device Microprocessor I/O

A physical microprocessor interface provides access to the microprocessor port of one physical device directly through the PCI interface. The interface consists of dedicated control signals and address/data signals. The PHY interface directly decodes accesses from the slave interface without using local memory bus bandwidth.

Boundary Scan I/O and Loopbacks

The CN8237 includes five pins for Joint Test Action Group (JTAG) Boundary Scan, for board-level testing. The CN8237 incorporates an internal loopback from the segmentation coprocessor to the reassembly coprocessor, to facilitate system diagnostics.

2.9 Electrical/Mechanical

The CN8237 is a CMOS device packaged in a 456 Ball Gate Array (BGA) format. It operates from a 3.3 V power supply and within the standard industrial temperature range.

The device inputs are tolerant of 5 V signal levels, so external 5 V devices can be used. Any I/O (except PCI) requiring a pullup must be tied through a resistor to 3.3 V.

2.10 Logic Diagram and Pin Descriptions

A functionally partitioned logic diagram of the CN8237 is illustrated in Figure 2-11. Pin descriptions, names, and input/output assignments are detailed in Table 2-1.

Figure 2-11. CN8237 Logic Diagram (1 of 6)

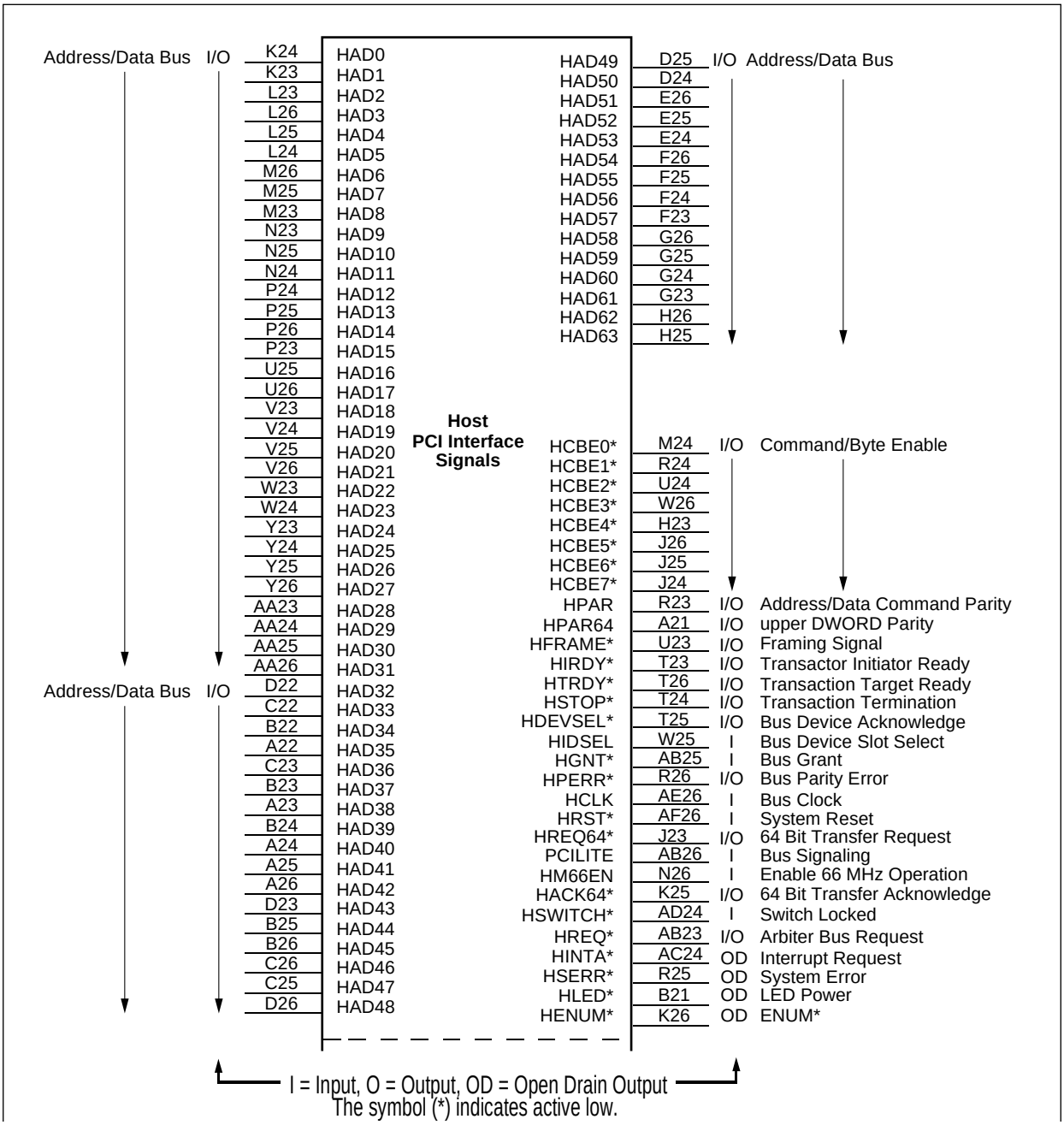


Figure 2-11. CN8237 Logic Diagram (2 of 6)

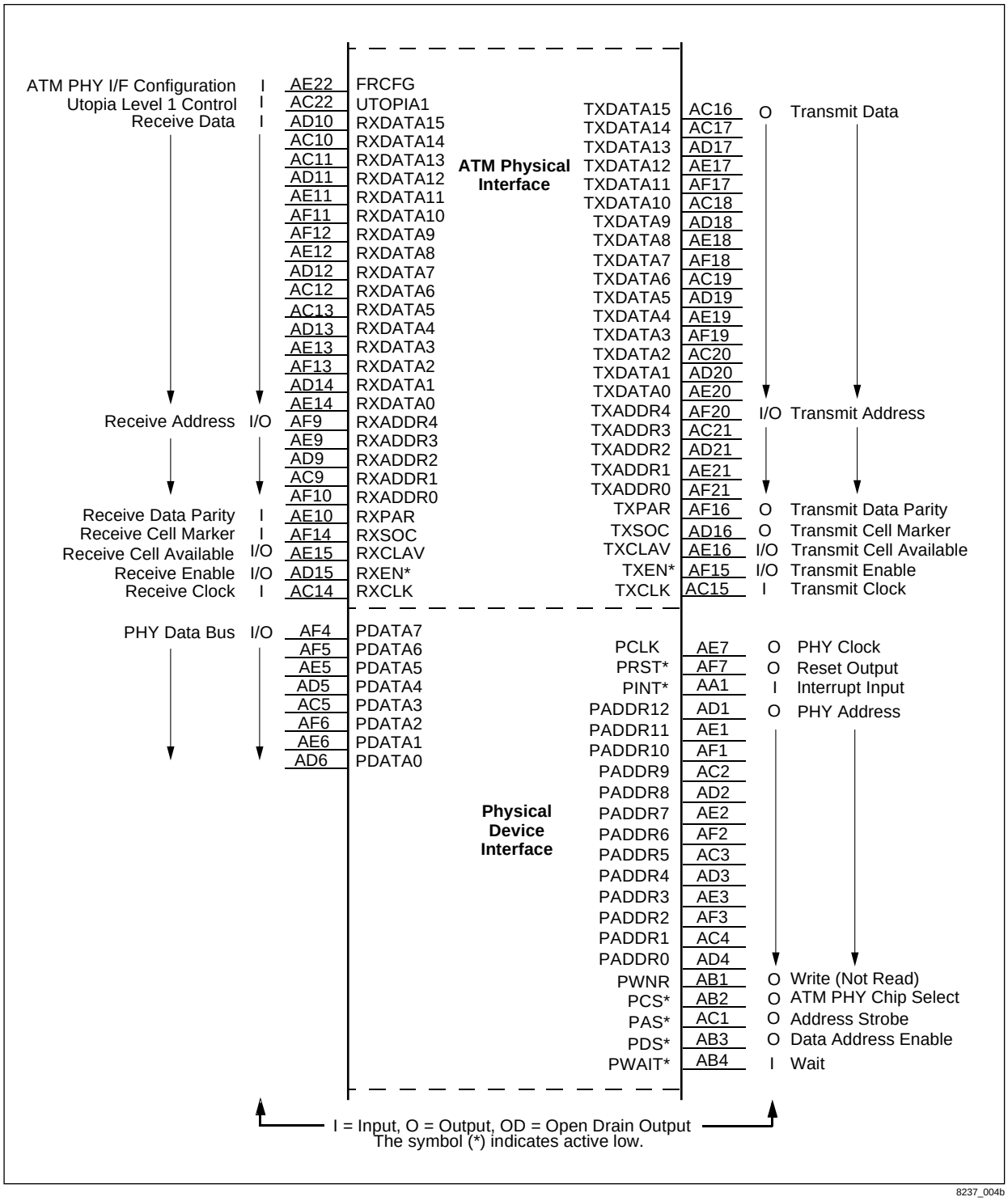


Figure 2-11. CN8237 Logic Diagram (3 of 6)

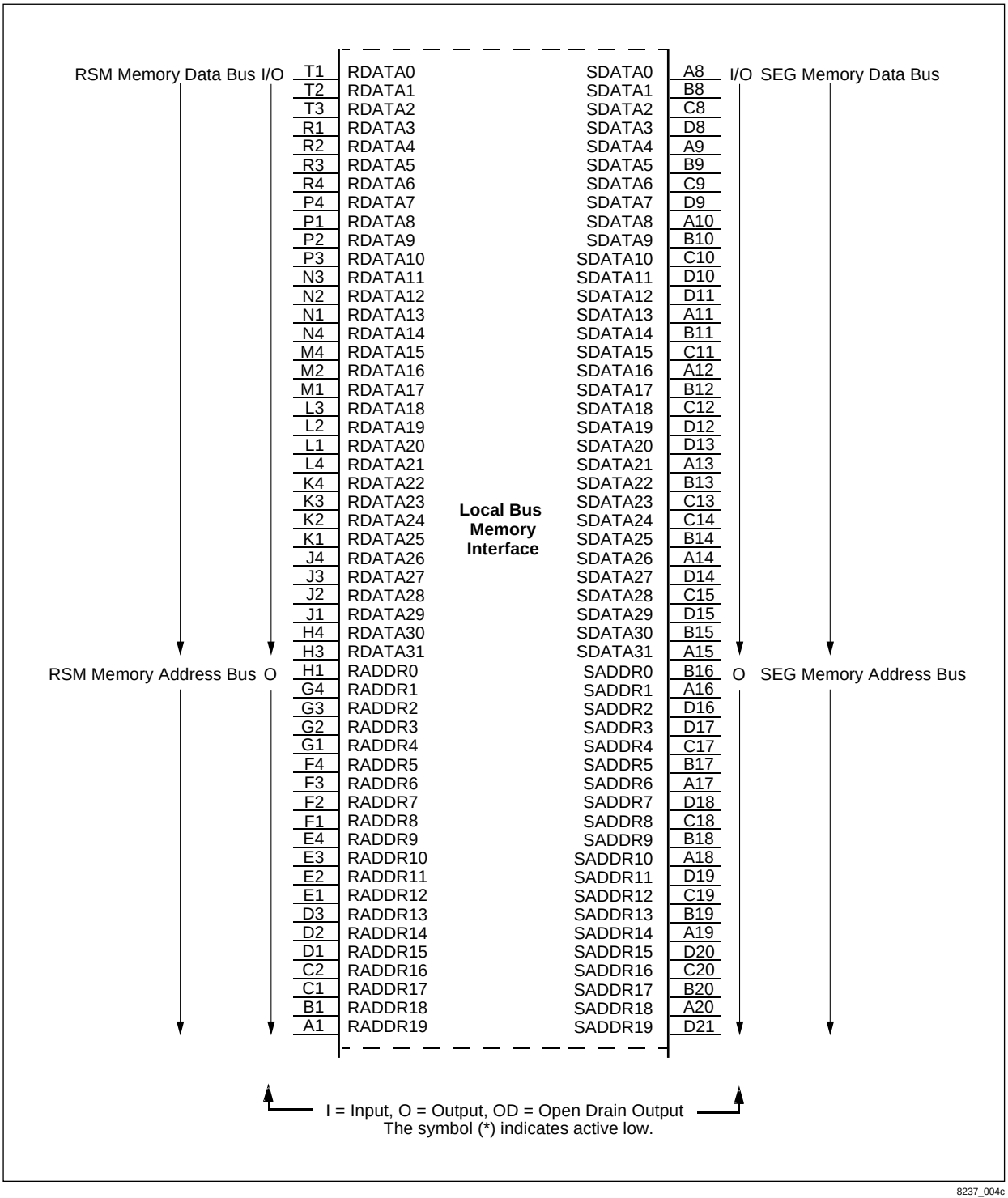


Figure 2-11. CN8237 Logic Diagram (4 of 6)

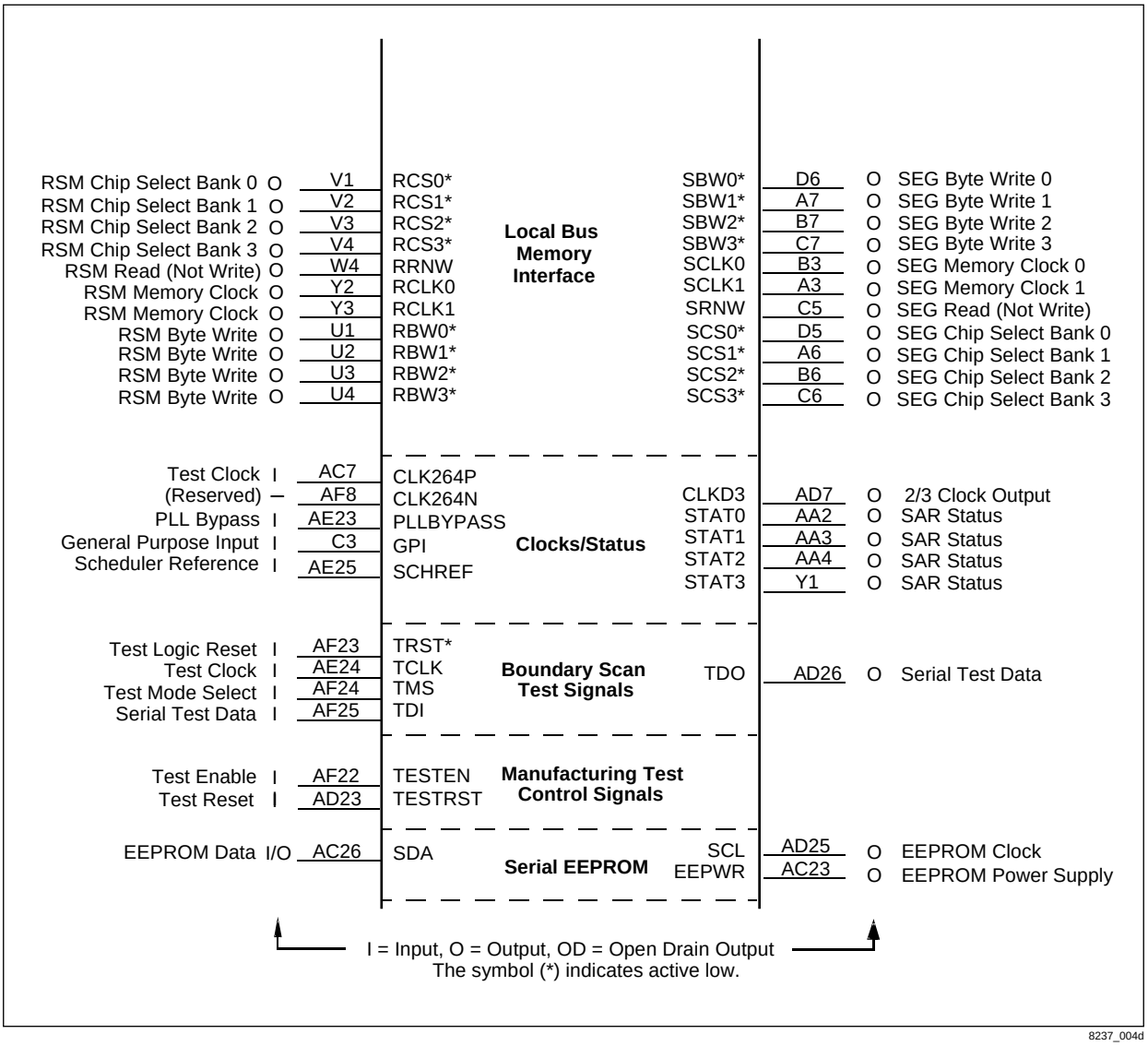


Figure 2-11. CN8237 Logic Diagram (5 of 6)

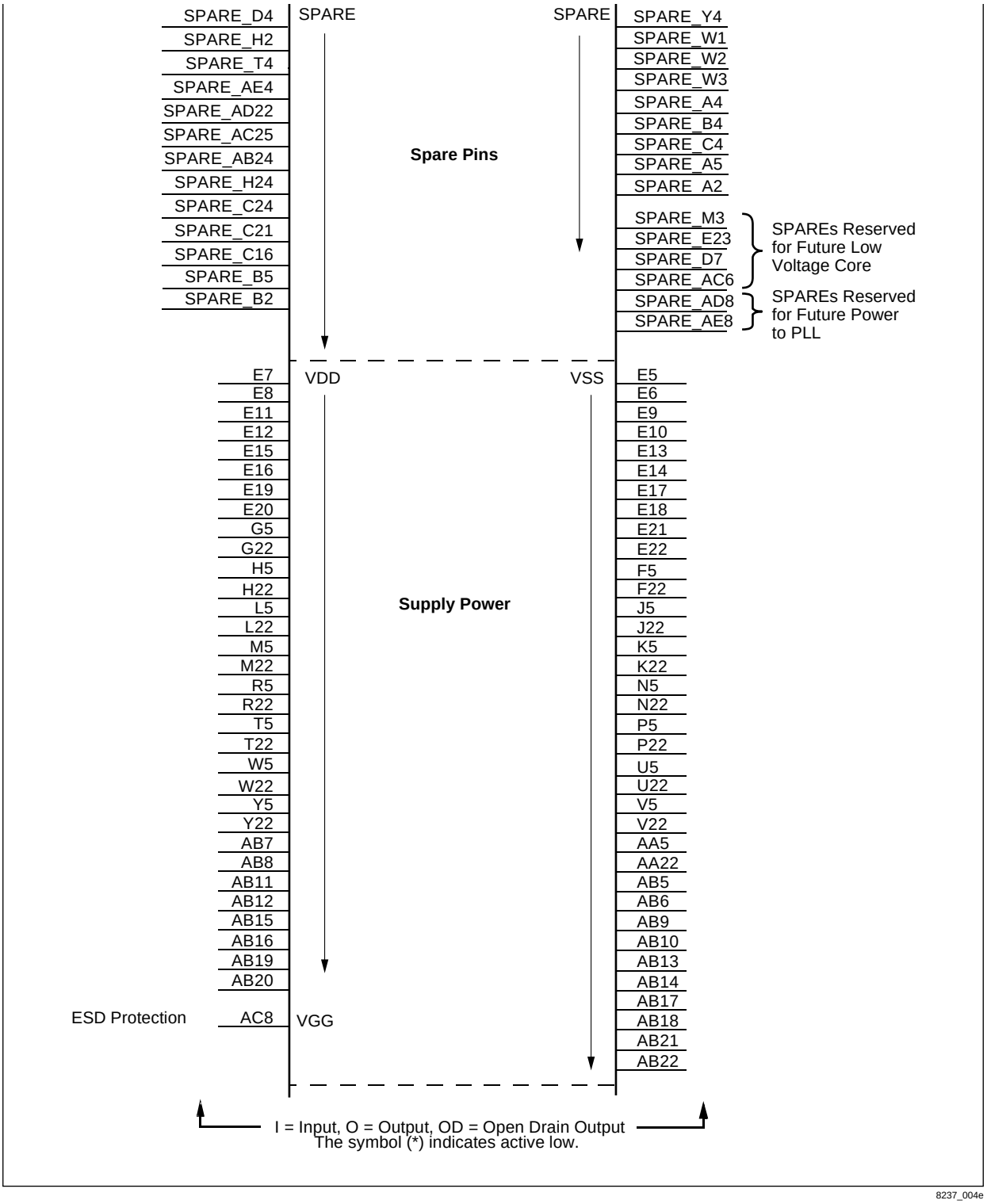


Figure 2-11. CN8237 Logic Diagram (6 of 6)

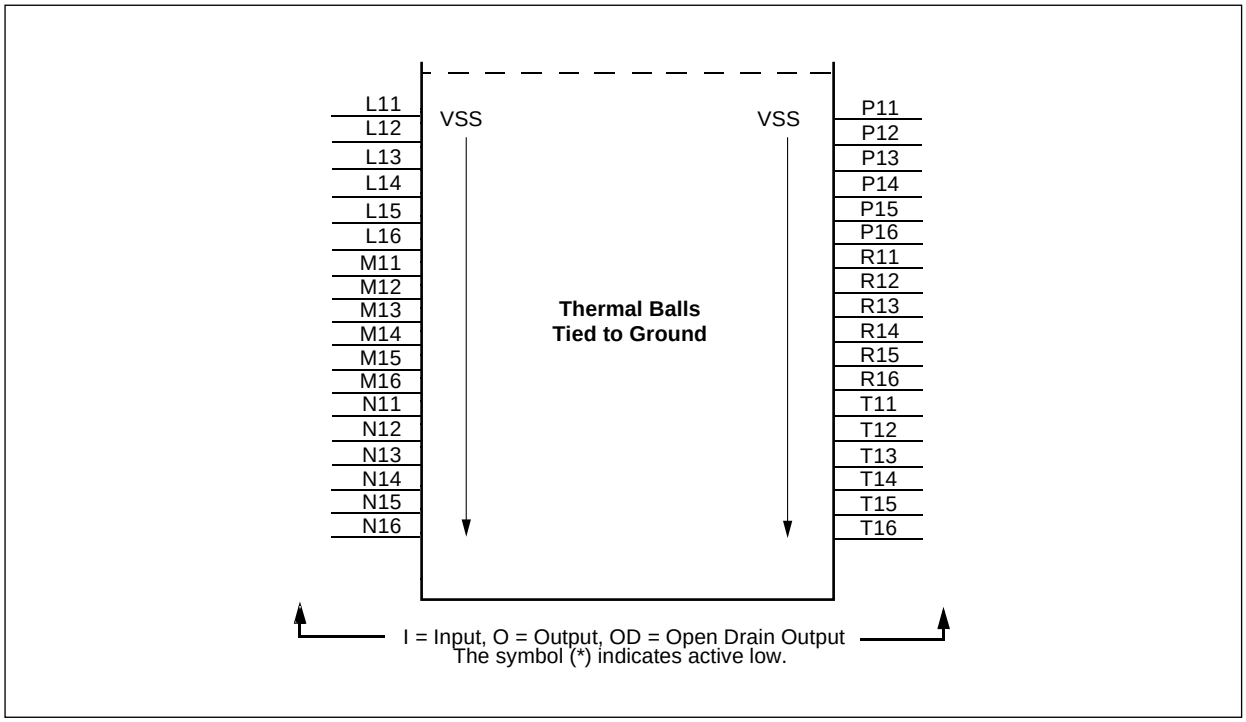


Table 2-1. Hardware Signal Definitions *(1 of 6)*

	Pin Label	Signal Name	I/O(1)	Definition
Host PCI Interface Signals	HAD[63:0]	Multiplexed Address/Data Bus	I/O	Used by the PCI host or CN8237 to transfer addresses or data over the PCI bus.
	HCBE[7:0]*	Command/Byte Enable	I/O	Outputs a command (during PCI address phases) or byte enables (during data phases) for each bus transaction.
	HPAR	Address/Data Command Parity	I/O	Supplies the even parity computed over the HAD[31:0] and HCBE[3:0]* lines during valid data phases. It is sampled (when the CN8237 is acting as a target) or driven (when the CN8237 acts as an initiator) one clock edge after the respective data phase.
	HPAR64	Address/Data Command Parity Upper DWORD	I/O	Supplies the even parity computed over the HAD[63:32] and HCBE[7:4]* lines during valid data phases. It is sampled (when the CN8237 is acting as a target) or driven (when the CN8237 acts as an initiator) one clock edge after the respective data phase.
	HFRAME*	Framing Signal	I/O	A high-to-low HFRAME* transition indicates that a new transaction is beginning (with an address phase). A low-to-high transition indicates that the next valid data phase will end the current transaction.
	HIRDY*	Transaction Initiator Ready	I/O	Used by the transaction initiator or bus master (either the CN8237 or the PCI host) to indicate ready for data transfer. A valid data transfer occurs when both HIRDY* and HTRDY* are active on the same clock edge.
	HTRDY*	Transaction Target Ready	I/O	Used by the transaction target or bus slave (either the CN8237 or the PCI bus memory) to indicate that it is ready for a data transfer. A valid data transfer occurs when both HIRDY* and HTRDY* are active on the same clock edge.
	HSTOP*	Transaction Termination	I/O	Driven by the current target or slave (either the CN8237 or the PCI bus memory) to abort, disconnect, or retry the current transfer. The HSTOP* line is used by the PCI master in conjunction with the HTRDY* and HDEVSEL* lines to determine the type of transaction termination.
	HDEVSEL*	Bus Device Acknowledge	I/O	Driven by a target to indicate to the initiator that the address placed on the HAD[31:0] lines (together with the command on the HC/BE[3:0]* lines) has been decoded and accepted as a valid reference to the target's address space. Once asserted, it is held by the CN8237 (when acting as a slave) until HFRAME* is deasserted; otherwise, it indicates (in conjunction with HSTOP* and HTRDY*) a target abort.

Table 2-1. Hardware Signal Definitions *(2 of 6)*

	Pin Label	Signal Name	I/O(1)	Definition
Host PCI Interface Signals	HIDSEL	Bus Device Slot Select	I	Signals the CN8237 that it is being selected for a configuration space access.
	HREQ*	Arbiter Bus Request	O	Asserted by the CN8237 to request control of the PCI bus.
	HGNT*	Bus Grant	I	Asserted to indicate to the CN8237 that it has been granted control of the PCI bus, and can begin driving the address/data and control lines after the current transaction has ended (indicated by HFRAME*, HIRDY*, and HTRDY*; all deasserted simultaneously).
	HINTA*	Interrupt Request	OD	Signals an interrupt request to the PCI host, and is tied to the INTA_ line on the PCI bus.
	HACK64*	64-bit transfer request acknowledged	I/O	Driven by target following successful address decode.
	HM66EN	PCI clock speed select	I	0 = 0 to 33 MHz, 1 = 33 to 66 MHz.
	HPERR*	Bus Parity Error	I/O	Driven asserted by the CN8237 (as a bus slave) or by a target addressed by the CN8237 when it acts as a bus master to indicate a parity error on the HAD[32:63] and HC/BE[3:0]* lines. It is asserted when the CN8237 is a bus slave or sampled when the CN8237 is a bus master on the second clock edge after a valid data phase. The CN8237 drives the HPERR* line only when acting as a slave.
	HSERR*	System Error	OD	Indicates a system error or a parity error on the HAD[32:64] and HC/BE[3:0]* lines during an address phase. This pin is handled in the same way as HPERR*, and is only driven by the CN8237 when it acts as a bus slave.
	HCLK	Bus Clock	I	PCI bus clock signal.
	HRST*	System Reset	I	Performs a hardware reset of the CN8237 and associated peripherals when asserted. Must be asserted for 16 cycles of HCLK.
	PCILITE	Bus Signalling	I	Selects PCILITE mode. High = PCILITE mode (disables retry disconnect).
	HSWITCH*	Switch indicator	I	Logic low means switch is locked, logic high means switch is unlocked. Compact PCI hot swap signal. Must be tied to ground if it is not used.
	HLED*	LED power	OD	12 mA open drain. Logic low turns on LED. Compact PCI hot swap signal.
	HENUM*	ENUM*	OD	8 mA open drain. Compact PCI hot swap signal.
	HREQ64*	64-bit transfer request	I/O	Driven by master requesting 64-bit transfer.

Table 2-1. Hardware Signal Definitions (3 of 6)

	Pin Label	Signal Name	I/O(1)	Definition
ATM PHY Interface	FRCFG	ATM Physical Interface Configuration	I	Configuration pin FRCFG determines what ATM physical interface the CN8237 supports. 1 = UTOPIA interface master. 0 = Slave UTOPIA interface.
	TxDATA[15:0]	Transmit Data	O	Carries outgoing data bytes to the PHY chip in all framer modes (8 mA drive).
	TxADDR[4:0]	Transmit Address	I/O	UTOPIA transmit address. 8 mA drive.
	TxPAR	Transmit Data Parity	O	Outputs the odd parity computed over the 8 or 16 data lines in all framer modes.
	TxSOC	Transmit Cell Marker	O	In both UTOPIA and slave UTOPIA modes, the TxSOC line is asserted by the CN8237 when the starting byte of a 53-byte cell is being output.
	TxCLAV (TxFLAG*)	Transmit Flag	I/O	In UTOPIA mode, TxCLAV (TxFLAG*) indicates that the transmit buffer in the downstream link interface chip is full and no more data can be accepted. In slave UTOPIA mode, this pin indicates to the link interface chip that the CN8237 transmit buffer is empty.
	TxEN*	Transmit Enable	I/O	Indicates that valid data has been placed on the TxDATA[15:0], TxPAR, and TxSOC lines in the current clock cycle when the CN8237 is in UTOPIA or slave UTOPIA mode. This pin is an output in UTOPIA mode and an input in slave UTOPIA mode.
	TxCLK	Transmit clock	I	UTOPIA transmit clock.
	RxADDR[4:0]	Receive Address	I/O	UTOPIA receive address.
	RxDATA[15:0]	Receive Data	I	Transfers incoming data bytes from the link interface or framer chip to the CN8237 in all framer modes.
	RxPAR	Receive Data Parity	I	Should be driven with the odd parity computed over the 8 or 16 data lines by the link interface or framer chip in all framer modes.
	RxSOC (RxMARK)	Receive Cell Marker	I	Indicates that the current byte being transferred on the RxDATA[7:0] lines is the starting byte of a 53-byte cell.
	RxCLAV (RxFLAG*)	Receive Flag	I/O	In UTOPIA mode, RxCLAV (RxFLAG*) indicates that the receive buffer in the downstream link interface chip is empty, no more data can be transferred, and the RxDATA[7:0], RxPAR, and RxSOC (RxMARK) lines are invalid. In slave UTOPIA mode, this pin indicates to the framer chip that the receive FIFO buffer in the CN8237 is full.
	RxEN*	Receive Enable	I/O	In UTOPIA mode, RxEN* indicates that the CN8237 is ready to receive data on the RxDATA[15:0], RxPAR, and RxSOC (RxMARK) lines in the next clock cycle. This pin is an output in UTOPIA mode and an input in slave UTOPIA mode.

Table 2-1. Hardware Signal Definitions *(4 of 6)*

	Pin Label	Signal Name	I/O(1)	Definition
ATM PHY Interface	RxCLK	Framer Control/Clock	I	UTOPIA receive clock.
	UTOPIA1	UTOPIA level select	I	Selects level 1/level 2 operation. In level 1 mode, address pins are forced to be outputs independent of master/slave mode. When UTOPIA1 input is a logic high, the UTOPIA address signals, TxADDR[4:0] and RxADDR[4:0], are forced as outputs. This pin must be pulled high or low.
Physical/Device	PADDR[12:0]	PHY Address	O	Address bus to PHY device.
	PCS*	ATM PHY Chip Select	O	The PCS* is connected to the chip select input of the Mindspeed PHY device.
	PAS*	Address Strobe	O	PAS* is used to drive the AS* pin of the Mindspeed PHY device.
	PWNR	Write/Not Read	O	This output provides the same function for the Mindspeed PHY device. Logic 1 writes; logic 0 reads.
	PDS*	Data/Address Strobe	O	PHY device data strobe.
	PCLK	PHY Bus CLK	O	Interface clock to PHY device.
	PDATA [7:0]	PHY Data Bus	I/O	PHY Device Data Bus.
	PINT*	Interrupt Input	I	PHY device interrupt signal.
	PRST*	Reset Output	O	Asserted by the CN8237 whenever the HRST* input is asserted.
	PWAIT*	PHY Interface Wait	I	Externally generated signal to extend PHY device accesses.

Table 2-1. Hardware Signal Definitions *(5 of 6)*

	Pin Label	Signal Name	I/O(1)	Definition
Local Bus Memory Interface	RDATA[31:0]	Memory Data Bus	I/O	RSM memory data I/O bus. Used for memory reads and writes.
	SDATA[31:0]	Memory Data Bus	I/O	SEG memory data I/O bus. Used for memory reads and writes.
	RADDR[19:0]	Memory Address Bus	O	RSM address bus. Used for RSM memory reads and writes.
	SADDR[19:0]	Memory Address Bus	O	SEG address bus. Used for SEG memory reads and writes.
	RCS[3:0]*	Memory Bank Chip Selects	O	Selects one of four addressable banks of RSM SRAM memory.
	SCS[3:0]*	Memory Bank Chip Selects	O	Selects one of four addressable banks of SEG SRAM memory.
	RRNW	RSM read not write	O	RSM read/write control line to synchronous SRAM.
	SRNW	SEG read not write	O	SEG read/write control line to synchronous SRAM.
	RCLK[1:0]	RSM synch SRAM clocks	O	RSM memory clock to synchronous SRAM.
	SCLK[1:0]	SEG synch SRAM clocks	O	SEG memory clock to synchronous SRAM.
	RBW[3:0]*	RSM byte writes	O	RSM byte write control line to synchronous SRAM.
	SBW[3:0]*	SEG byte writes	O	SEG byte write control line to synchronous SRAM.
Clocks/Status	CLK264P	264 MHz Clock	I	External clock input, for testability.
	CLK264N	(Reserved)	—	—
	PLLBYPASS	PLL BYPASS	I	When set high, external clock (CLK264P) is used by SAR.
	CLKD3	Divide by 2/3 Clock Output	O	This output clock is a 50% duty cycle, two-thirds divide of RCLK0; it can be used for the UTOPIA interface clock.
	STAT[3:0]	SAR Status	O	CN8237 internal status outputs. Internal status controlled by the STAT_MODE[4:0] field in the CONFIG0 Register.
	GPI	General Purpose Input	I	User-defined general purpose input. This value is reflected in HOST_ISTAT0 bit 31, GPI.
	SCHREF	Scheduler Reference	I	External Scheduler Reference Clock. If not used, must be grounded.

Table 2-1. Hardware Signal Definitions *(6 of 6)*

	Pin Label	Signal Name	I/O(1)	Definition
Boundary Scan Test Signals	TRST*	Test Logic Reset	I	When a logic low, this signal asynchronously resets the boundary scan test circuitry and puts the test controller into the reset state. This state allows normal system operation. Tie to ground when boundary scan is not implemented. This PAD has an internal pullup resister to VDD.
	TCLK	Test Clock	I	Generated externally by the system board or by the tester. Tie to ground when boundary scan is not implemented.
	TMS	Test Mode Select	I	Decoded to control test operations. This PAD has an internal pullup resister to VDD.
	TDO	Serial Test Data	O	Outputs serial test pattern data.
	TDI	Serial Test Data	I	Input for serial test pattern data. This PAD has an internal pullup resister to VDD.
Manufacturer Test Control	TESTEN	Test On	I	Internal factory test. ***NOTE:*** Must be tied to ground for proper operation.(2)
	TESTRST	Test Reset	I	Internal factory test.(2)
Serial EEPROM	SDA	EEPROM Data	I/O	Serial data.
	SCL	EEPROM Clock	O	Serial clock.
	EEPWR	EEPROM Power	O	12 mA drive. Direct buffer of HRST* input.
Supply Power	VDD	Power	—	VDD pins must be connected to 3.3 V.
	VSS	Ground	—	VSS pins must be connected to ground. (The 36 balls in the center help in heat dissipation.)
	VGG	ESD Protection – Voltage Clamp	I	Enables Electrostatic Discharge (ESD) protection. When the SAR is connected to 3.3 V power supply yet there are 5 V devices on the board, tie this pin to 5 V for ESD Protection. Otherwise, tie to 3.3 V.(3)

NOTE(S):

(1) KEY:
I = Input
O = Output
OD = Open Drain Output
The symbol (*) indicates active low.

(2) Must be tied to ground for proper operation.

(3) The 5 V must be applied before a coincident with 3.3 V.

3.0 Host Interface

3.1 Overview

The CN8237 segments and reassembles user data packets at over 600 Mbps full duplex. The actual segmentation and reassembly processes execute without run-time host control. However, the ATM host system supplies the data for transmission and buffers for received data. In addition to this control, the host processes status returned from the SAR. To take advantage of the CN8237's high throughput, the host must process control and status information at a comparable rate.

APPLICATION EXAMPLE

An Ethernet Switch uses a CN8237 based subsystem as an uplink to an OC-12 ATM backbone. Under worst case conditions, Ethernet packets (64 octets) map into two ATM cells. At OC-12 rates, the CN8237 converts 706.4 k packets/second (kpps) to cells in each direction. Therefore, the host must process control and status information for a total of 1412.8 kpps. This packet rate equates to a packet service time of 708 ns/packet.

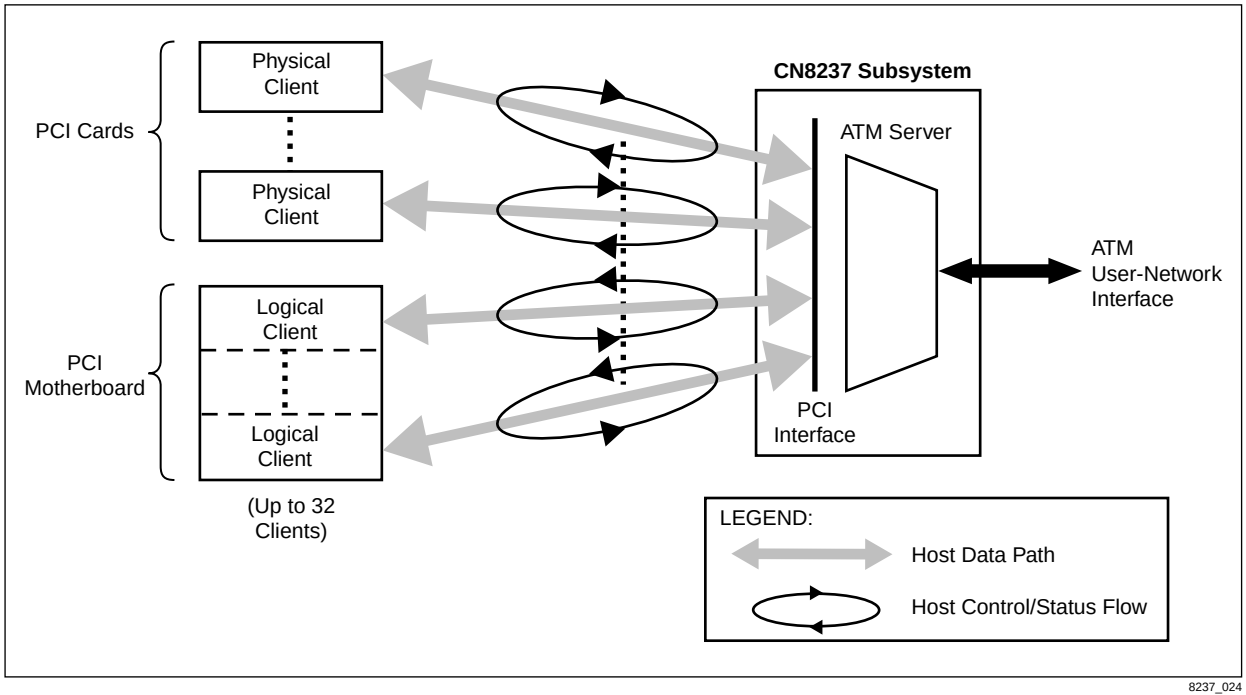
In cases such as the example above, the service rate places extraordinary performance requirements on the host system. High throughput systems require large numbers of processing cycles and efficient utilization of system buses.

The CN8237 provides a flexible, high performance host interface architecture. With this interface, the CN8237 facilitates a scalable, distributed host system. The interface also minimizes the impact of an ATM port on the host system's PCI bus.

3.2 Multiple Client Architecture

The CN8237 provides multiple independent control and status communication paths. Each communication path, or flow, consists of a control queue and a status queue for both segmentation and reassembly. The host assigns each of these independent flows to system clients or peers. As throughput requirements escalate, the host system can add processing power in the form of additional peers. This degree of freedom creates a scalable host environment. The CN8237 provides an ATM server for up to 32 clients. Figure 3-1 illustrates this client/server relationship.

Figure 3-1. Client/Server Model of the CN8237



3.2.1 Logical Clients

As shown in Figure 3-1, the clients need not be physically distinct PCI peers. The host can also assign control and status queues to system software tasks or logical clients. Since the queues offer individually distinct communication paths, each logical client interfaces to the CN8237 independently. Due to its server architecture, the CN8237 supplies the synchronization between asynchronous tasks requiring ATM services.

3.2.2 Resource Allocation

With either physical PCI peers, logical peers, or some combination of the two types, the CN8237 multiplexes each peer's transmitted packets onto the line and routes incoming packets to the appropriate peer. The host system allocates shared resources, such as host and SAR-shared memory, VPI/VCI address space, and CBR time slots to peers and clients arbitrarily.

3.2.3 Resource Isolation

Because each peer is assigned to an independent control and status path, the CN8237 isolates the resources of each peer. Not only does this simplify resource management, but queue error conditions caused by a single peer do not affect any other peers.

APPLICATION EXAMPLE

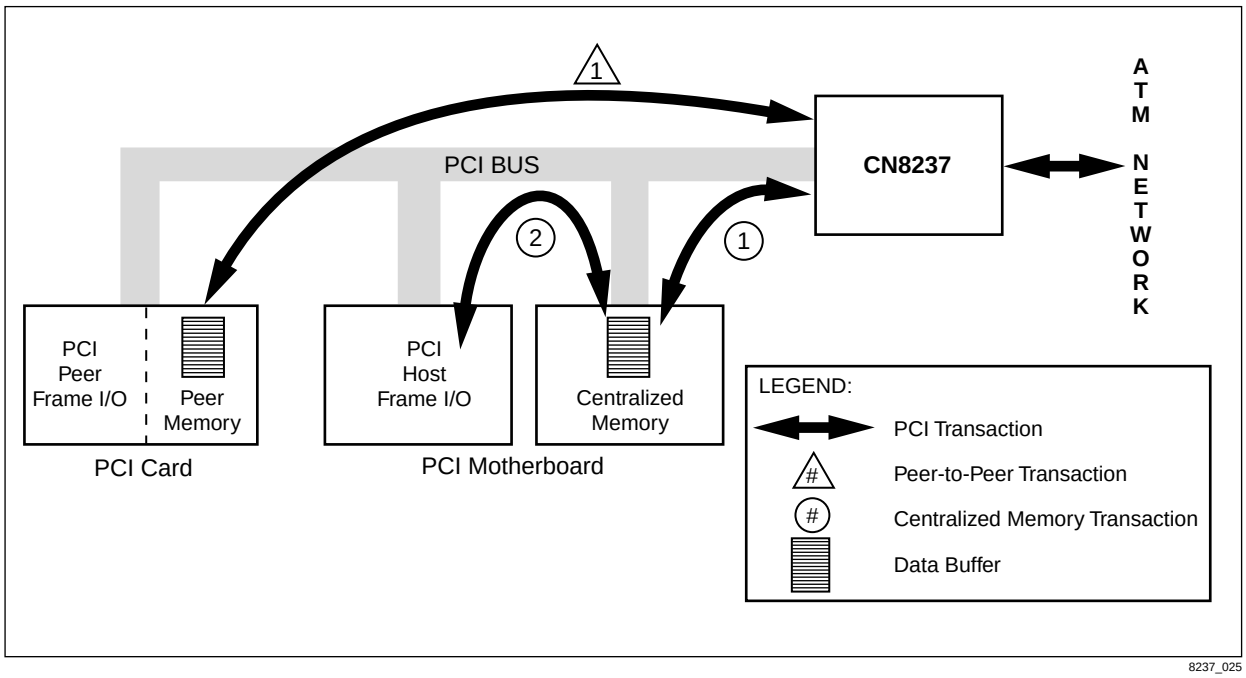
A system designer implements a CN8237 terminal as an ATM uplink for a Service Access Multiplexer (SAM). The SAM is comprised of the ATM card and several Frame Relay adapter cards. The host assigns each Frame Relay adapter card to a set of CN8237 control and status queues at initialization. During operation, one of the Frame Relay adapter cards experiences a hardware failure. The failure prevents the card's processor from servicing the CN8237's reassembly status queue. Eventually, the CN8237 fills the queue and is unable to proceed—this situation is referred to as queue overflow. The CN8237 shuts down reassembly on VCCs that are assigned to the overflowed queue only. Since the other cards in the system are assigned to other status queues, their VCCs remain unaffected by the failure.

3.2.4 Peer-to-Peer Transfers

The multiple queue architecture of the CN8237 also enables peer-to-peer PCI transfers. The CN8237 transfers ATM cells as a PCI master. Since the buffer control structures are independent for each peer, each identifies a unique address range in PCI memory space. The host defines the address range of each peer. The CN8237 transfers data within this address range. An address range corresponds either to a region of centralized host memory or to a set of peer resident buffers.

Figure 3-2 shows the difference between these two options. Centralized memory buffers require store-forward operations, while the peer buffers enable peer-to-peer transfers. Thus, peer-to-peer transfers reduce the use of the PCI bus.

Figure 3-2. Peer-to-Peer vs. Centralized Memory Data Transfers



3.3 Write-only Control and Status

For host-based applications, the host manages the CN8237 SAR using write-only control and status queues. This architecture minimizes PCI bus use by eliminating reads from the control path. PCI writes use the bus much more efficiently than PCI reads. During a PCI write, the target can post the write data in an internal FIFO buffer, terminate the transaction, and immediately release the bus. On the other hand, during reads, the target retrieves the data while holding the bus. Since the data retrieval takes some time, reads increase the PCI bus utilization.

The CN8237's write-only architecture uses reads only for segmentation data (PDU) fetches. All control and status transactions are writes. This section describes the management of write-only queues. The purpose and entries of each class of queue are described in later chapters.

Table 3-1 defines the CN8237 control and status queues.

Table 3-1. CN8237 Control and Status Queues

Type	Segmentation	Reassembly
Control	Transmit queue	Free buffer queue
Status	Segmentation status queue	Reassembly status queue

3.3.1 Write-only Control Queues

The host controls run-time segmentation and reassembly through write-only control queues. There are two types of control queues—the segmentation transmit queues and the reassembly free buffer queues. The host submits buffers of PDU data for segmentation on the transmit queues, and supplies empty buffers for received data on the free buffer queues. Each type of queue is managed as a write-only control queue.

These queues reside in SAR-shared memory at locations defined by base register pointers. To allow multiple clients, the CN8237 supports 32 queues of each type. The SAR and host manage each queue independently, through queue management variables. The SAR stores its variables in internal registers called base tables. The host maintains its own variables within its driver. Each queue contains a programmable number of queue entries.

3.3.1.1 Control Variables

Table 3-2 describes the variables for write-only control queues.

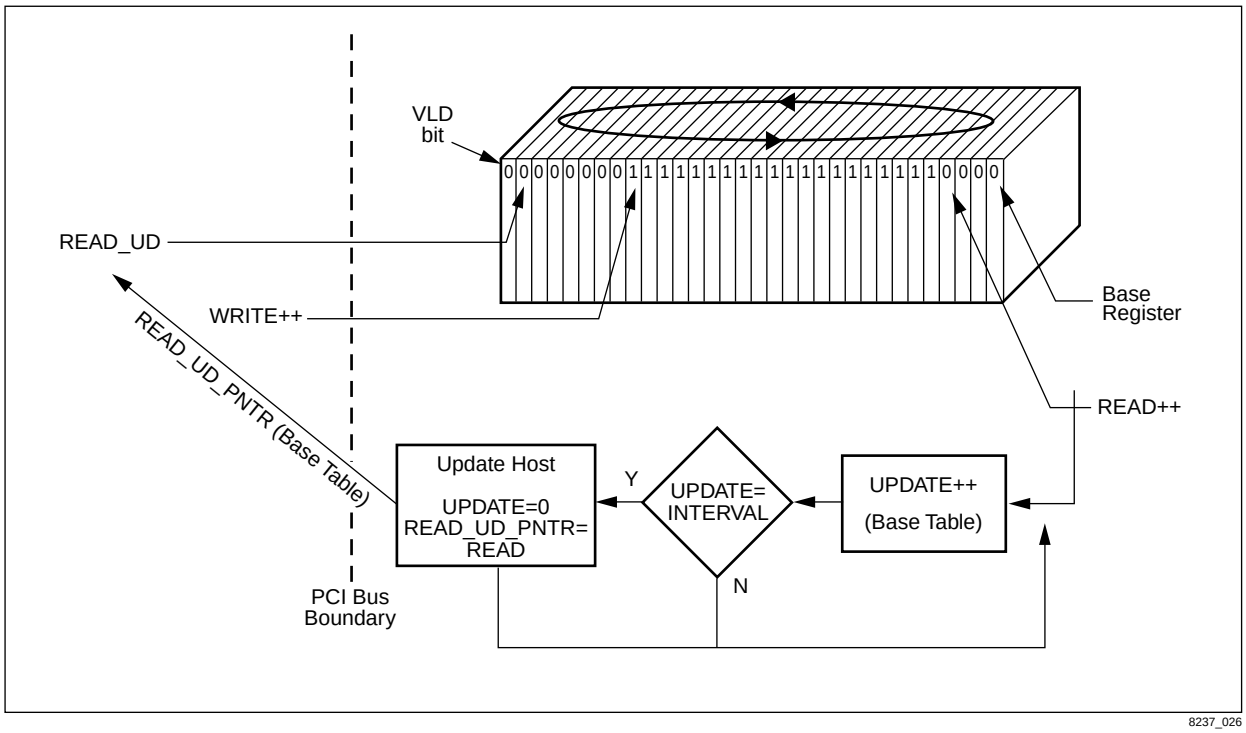
Table 3-2. Write-only Control Queue Variables

Variable	Definition	Location	Initialization
WRITE	Current host position in queue	Host variable	0
READ_UD	Last known SAR position in queue as seen by host	Host word aligned variable	0
READ	Current SAR position in queue	SAR Base table	0
INTERVAL	Number of queue entries processed by SAR before writing READ_UD	SAR register	Host defined
UPDATE	Number of queue entries since last write of READ_UD	SAR Base table	0
READ_UD_PNTR	SAR pointer to READ_UD	SAR Base table	&READ_UD

3.3.1.2 Queue Management

Figure 3-3 illustrates the control queue management algorithm. The host initializes all of the variables described in Table 3-2. Once the SAR is enabled, it maintains the READ pointer. When the SAR processes a queue entry, it advances the READ pointer (READ++). Since the queues are circular, the pointer eventually wraps back to 0. It also advances an internal counter, UPDATE (UPDATE++). When INTERVAL queue entries have been processed, the SAR writes its current position in the queue, READ, to a host variable, READ_UD. The host determines the magnitude of INTERVAL at initialization. Larger numbers result in fewer overhead PCI accesses, but also introduces larger latency between host updates.

Figure 3-3. Write-only Control Queue



The host also maintains a pointer into the queue, WRITE. When the host submits a new entry, it must first ensure that the SAR has already processed the entry location. The host compares WRITE to READ_UD. If (WRITE+1) modulo size_of_queue equals READ_UD, the host halts writing to the queue. This results in being able to use only N-1 queue entries. However, if this is not done, then a full condition cannot be distinguished from an empty condition. The host must wait until READ_UD is modified by the SAR before proceeding. This algorithm ensures that the host does not overflow the control queue, without reading the queue itself.

Once it has verified its ownership of the entry, the host writes the entry and increments its write pointer (WRITE++). During this write, the host sets the valid bit (VLD) in the entry to 1.

The CN8237 snoops the writes to the control queue areas. Once a write is detected to a specific queue, the SAR attempts to process the queue entry at READ. Before acting on the entry, the SAR checks for ownership of the entry, indicated by the VLD bit. Once the CN8237 has processed the entry, it resets the VLD bit to 0.

3.3.1.3 Underflow Conditions

An underflow condition occurs when the SAR attempts to retrieve a queue entry and the host has not yet supplied this entry. This condition happens only on the free buffer queues. The SAR detects this condition by checking the queue entry VLD bit. Once detected, the SAR enters an Underflow Detected state on this queue only. Since this signifies that no data buffers are available for reassembly, the SAR initiates EPD on all channels assigned to this queue. Chapter 5.0 describes SAR handling of free buffer queue underflow in detail.

3.3.2 Write-only Status Queues

The SAR reports status to the host through write-only status queues. Both the segmentation and reassembly coprocessors use their own format of status queue. However, the CN8237 manages all status queues with the same algorithm.

These queues reside in host memory. The host must assign double word-aligned (8-byte) status queue base addresses. To support multiple clients, the CN8237 provides 32 queues of each type. The SAR and host manage each queue independently through queue management variables. The SAR stores its variables in internal base table registers. The host maintains its variables in its driver. Each queue contains a programmable number of queue entries.

3.3.2.1 Control Variables

Table 3-3 describes the variables for write-only status queue management.

Table 3-3. Write-only Status Queue Variables

Variable	Definition	Location	Initialization
WRITE	Current SAR position in queue	SAR Base table	0
READ_UD	Last known host position in queue as seen by SAR	SAR Base table	0
READ	Current host position in queue	Host Variable	0
INTERVAL	Number of queue entries processed by host before writing READ_UD	Host Variable	Host defined
UPDATE	Number of queue entries since last write of READ_UD	Host Variable	0
READ_UD_PNTR	Host pointer to READ_UD	Host Variable	&READ_UD

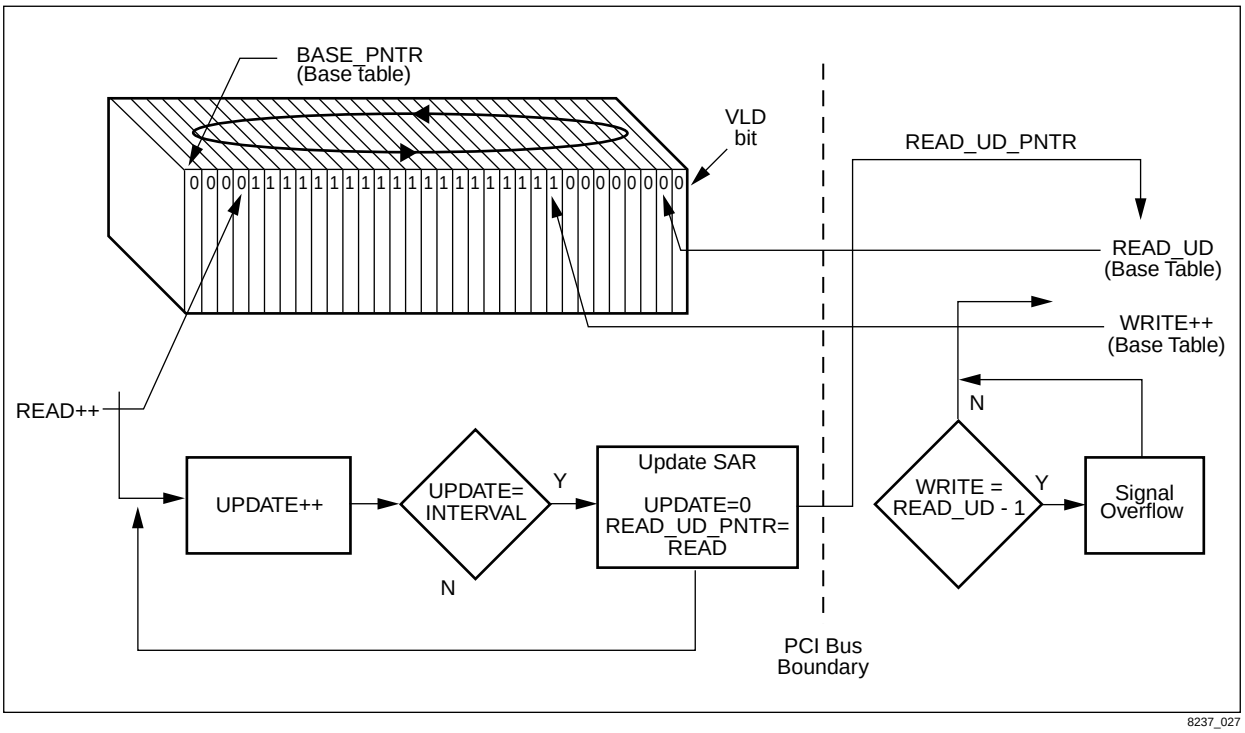
3.3.2.2 Queue Management

Figure 3-4 illustrates the status queue management algorithm. The host initializes all of the variables described in Table 3-3.

The SAR maintains its own write pointer, WRITE. The CN8237 reports status to the host by writing a status queue entry. After it writes the entry, the CN8237 increments its write pointer (WRITE++). This write also triggers a maskable interrupt.

The host either responds to this interrupt, or periodically polls the status queue. The VLD bit in each queue entry enables polling. The SAR sets the VLD bit equal to 1 when it writes a status queue entry. The host resets it to 0 after processing an entry.

Figure 3-4. Write-only Status Queue



The host also maintains a pointer (READ) into the status queue. Each time it services an entry, it increments a counter (UPDATE++). When this counter reaches a host-specified threshold (INTERVAL), the host informs the SAR of its current queue position by writing READ_UD in the queue's base table register.

3.3.2.3 Overflow Conditions

An overflow condition occurs when the SAR attempts to write a status queue entry, but the status queue entry is unavailable. This condition can happen for both the segmentation and reassembly status queues. Chapter 4.0 and Chapter 5.0 describe the handling of this event. In either case, the result is severe and therefore undesirable. The host control service rate of the status queue should match or exceed the status queue reporting rate of the CN8237.

The CN8237 detects an overflow condition by comparing its current WRITE pointer to the READ_UD pointer, that is, the last known host READ position. If WRITE points to the entry immediately before the READ_UD (WRITE = READ_UD -1), the SAR detects the imminent overflow condition.

To inform the host of the event, the SAR sets the overflow indication bit (OVFL) in the exhausted status queue. Since it cannot report status, the CN8237 segmentation and reassembly processing is temporarily halted for VCCs assigned to the overflowed status queue only. All other processes and queues remain operational.

3.3.2.4 Status Queue Interrupt Delay

Status Queue Interrupt Delay has been added in order to reduce the interrupt processing load on the host. This is valuable in a Network Interface Card (NIC)-based solution, where the SAR resides in an environment in which the host is not dedicated to datacom processing. Both a timer holdoff mechanism and an event counter mechanism are implemented and work in parallel. The timer holdoff mechanism uses the ALARM1 and CLOCK register resources to implement an interval timer. Interrupts due to status queue writes are delayed until the timer expires. The event counter mechanism delays the assertion of the interrupt due to status queue writes until a fixed number of status queue writes have occurred. Both mechanisms work in parallel (not in series) if enabled, so that either mechanism needs to expire before the interrupt propagates to the output pin. Interrupts due to conditions other than status queue writes are not delayed.

Timer Holdoff Mechanism

The timer holdoff mechanism is enabled by setting INT_DELAY (EN_TIMER) to a logic high. The ALARM1 register is set to a value that holds off the interrupt for a specified period of time. The user initializes the CLOCK register to 0. When the value in the CLOCK register is greater than the value in the ALARM1 register, status queue interrupts are allowed to propagate to the interrupt pin, HINT*.

Event Counter Mechanism

The event counter mechanism is enabled by setting INT_DELAY (EN_STAT_CNT) to a logic high. An internal counter is implemented that counts the number of status queue write events. The number of events before opening the interrupt window is programmable using the INT_DELAY(STAT_CNT) field. The window is closed for STAT_CNT number of events. When the internal counter has reached the value of STAT_CNT, the interrupt window is opened, which allows the interrupt to propagate to the output pin. The counter is reset when the status registers are read and the interrupt output goes inactive.

4.0 Segmentation Coprocessor

4.1 Overview

ATM's cell transport mechanism enables large numbers of virtual channels or VCCs to be multiplexed onto a single PHY interface. The segmentation process converts user data (typically Ethernet, Token Ring, or Frame Relay packets) into ATM cells.

The CN8237 can segment 64 K VCCs simultaneously. The segmentation coprocessor block independently segments each channel and multiplexes the VCCs onto the line with cell level interleaving. The CN8237 xBR Traffic Manager determines the order of execution of these independent processes to ensure the requested QoS for every channel. For each cell transmission opportunity, the xBR Traffic Manager tells the segmentation coprocessor which VCC to send. Therefore, the segmentation coprocessor acts as a slave to the xBR Traffic Manager.

In addition to converting blocked user data into ATM cells, the CN8237 generates all AAL overhead for AAL5, or optionally uses a null adaptation layer, AAL0. The CN8237 also generates the ATM cell header, as defined by the host, for each VCC. Furthermore, the segmentation coprocessor and xBR Traffic Manager together provide service-specific features to enhance the performance of Frame Relay internetworking and LAN Emulation.

4.2 Segmentation Functional Description

4.2.1 Segmentation VCCs

A VCC specifies a single VC or VP in the ATM network. The CN8237 supports up to 64 K segmentation VCCs, referenced by a unique index, VCC_INDEX. The VCC_INDEX identifies a location in the Segmentation VCC table, an array of 10-word segmentation VCC channel descriptors.

Except for ABR service category VCCs, each segmentation VCC occupies a single descriptor in the table (10 words). Due to the large number of specified parameters for ABR traffic, ABR VCCs occupy two descriptors (20 words) in the Segmentation VCC table. The VCC_INDEX for ABR VCCs points to the first of the two descriptors, and must be evenly divisible by two.

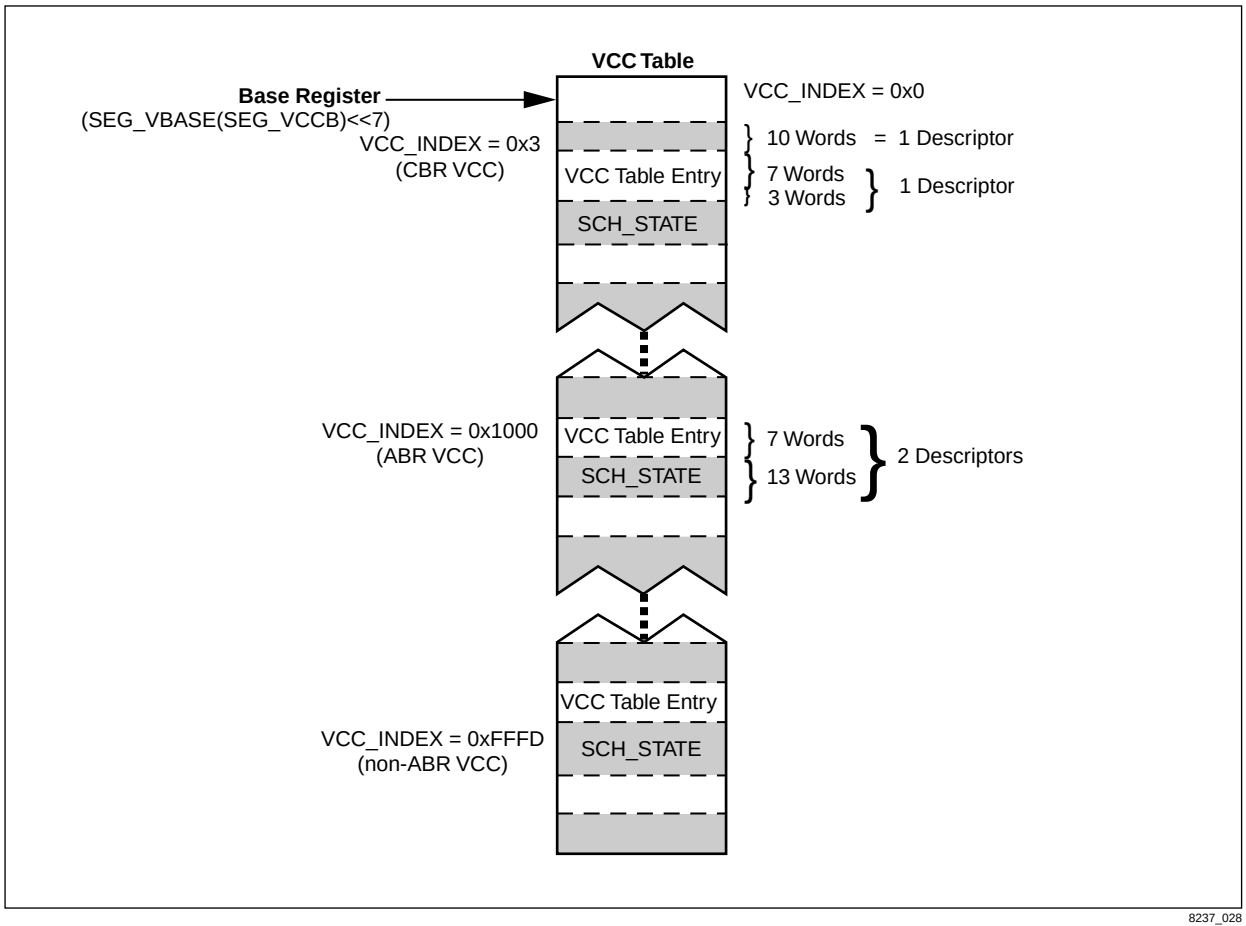
A Segmentation VCC table channel descriptor consists of two parts: an AAL specific VCC table entry (seven words) and a service class specific Schedule State (SCH_STATE). For non-ABR VCCs, the remaining three words of the channel descriptor form the SCH_STATE entry. For an ABR VCC, the SCH_STATE entry contains 13 words, which require an additional descriptor location.

4.2.1.1 Segmentation VCC Table

Figure 4-1 shows a VCC table with a CBR VCC at VCC_INDEX 3, an ABR VCC located at VCC_INDEX 0x1000, and a non-ABR VCC at VCC_INDEX 0xFFFD. The host allocates the VCC table in SAR SEG-shared memory and provides an address to the SAR in a base register field, SEG_VBASE(SEG_VCCB). For the most efficient ABR performance, all ABR VCCs should be placed in the upper half of the VCC table (that is, with smaller VCC indexes). The only reason not to put ABR VCCs at the lowest address range is to place CBR VCCs in the first 32 K of VCC addresses. Valid VCC indexes for CBR traffic range from 0x0000 to 0x7FFE, due to the limit imposed by a 15-bit address. A VCC index of 0xFFFF for non-CBR and 0x7FFF for CBR indicates a NULL VCC.

While the CN8237 will accept any VCC index within the range of 64 K VCC indexes, the actual number of segmentation VCCs allowed by the SAR is limited by the amount of SAR SEG-shared memory available in which to allocate and create segmentation VCC tables, transmit queues, etc.

Figure 4-1. Segmentation VCC Table



The VCC table entry contains generic information common to all traffic classes. This includes a default ATM header, which the host can modify during the segmentation process. See Section 4.3.1 for full details of the structure of a segmentation VCC table entry.

4.2.1.2 VCC Identification

The host allocates a region of SAR SEG-shared memory for the segmentation VCC table at system initialization, based on the maximum number of connections and the maximum number of ABR connections. The host informs the SAR of the location of the table through the internal base register, SEG_VBASE (SEG_VCCB).

Once a table has been established, the host assigns segmentation VCCs to entries in the table. The host describes the SEG VCC by initializing the SEG VCC table entry including the SCH_STATE portions of the assigned VCC. The VCC_INDEX, defined as the offset into the table in 10-word increments, uniquely identifies a segmentation channel. In all communication between the SAR and the host, a VCC_INDEX field specifies a VCC.

4.2.2 Submitting Segmentation Data

Once the host establishes a connection, it supplies data for segmentation. The host submits full or partial PDUs, either one at a time or in batches, for individual VCCs. The SAR accepts PDUs at any time, regardless of the state of the connection, and segments data on each VCC independently.

4.2.2.1 User Data Format

The CN8237 accepts user PDUs as sequences of data buffers. SAR SEG-shared memory resident segmentation buffer descriptors (SBDs) provide the address, length and control information for buffers. The host forms PDU buffer sequences by linking buffer descriptors. The data buffers themselves contain only user data and reside in host (or optionally SAR SEG-shared) memory. Host data buffers contain the bulk of segmentation data, and can begin on any byte-aligned address in the SAR's PCI address space.

NOTE: SAR SEG-shared memory data buffer segmentation should be limited to low bandwidth applications, such as Signalling, OAM, and ILMI.

4.2.2.2 Buffer Descriptors

The host submits data using the following process sequence. First, the host allocates an SBD for each buffer in a message. SBDs reside in SAR SEG-shared memory, and must begin on word-aligned addresses. Then, the host describes the buffers in the SBD. This description includes the address and length of the buffer, as well as control fields for the SAR during buffer segmentation. These control fields specify the VCC_INDEX, the AAL type, and header override information for the buffer.

The host then creates PDU message sequences by concatenating buffers. The host forms the buffer sequence by linking a list of buffer descriptors using the SBD's NEXT field. Two bits in the SBD control fields delineate the beginning and end of messages. One bit, BOM, specifies that the buffer contains the beginning of a message. The second bit, EOM, notifies the SAR that the buffer contains the end of the current message. The host identifies the end of the SBD chain by terminating the list with a NULL (0) NEXT pointer. Table 4-1 describes the appropriate utilization of these bits in detail.

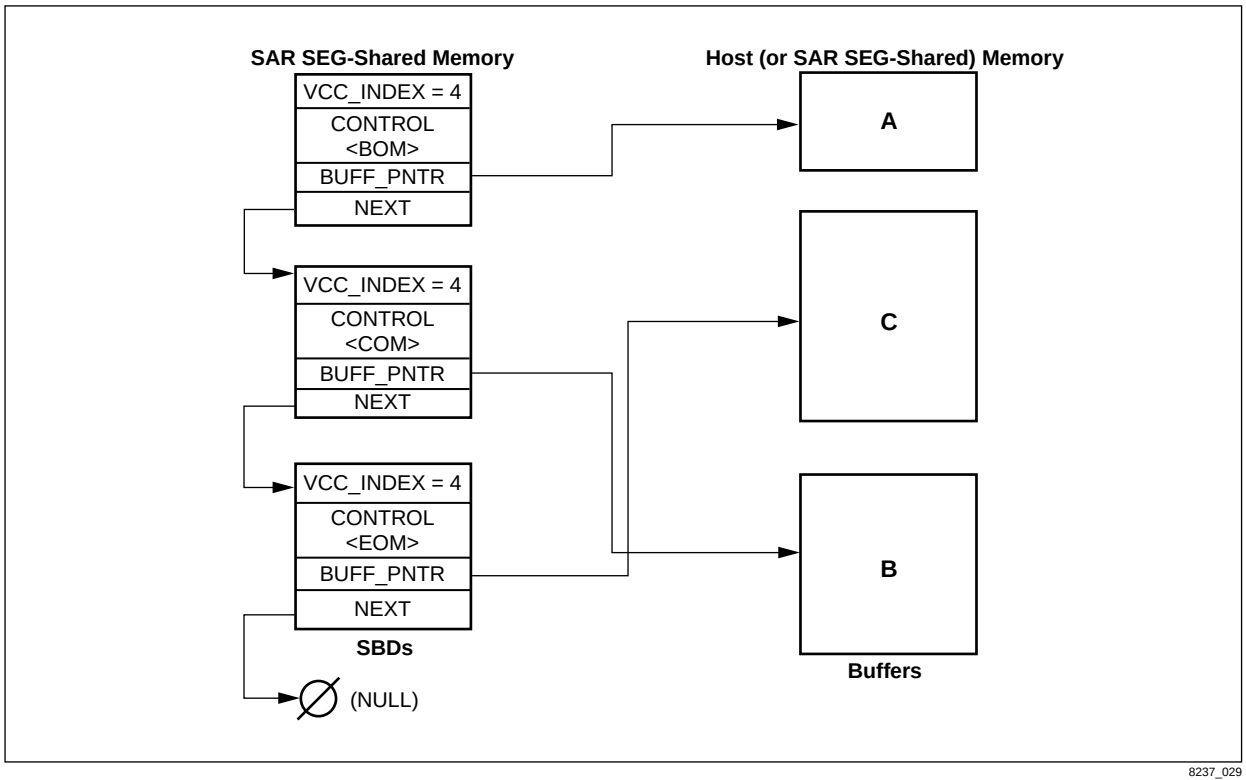
Table 4-1. Segmentation PDU Delineation

BOM	EOM	Buffer to Message Adaptation
0	0	Segment as Continuation of Message (COM).
0	1	Terminate current CPCS-PDU at end of buffer (EOM).
1	0	Restart CPCS-PDU generation (BOM). Wait for EOM for termination.
1	1	Buffer contains complete message. Restart/terminate CPCS-PDU.

4.2.2.3 Host Linked Segmentation Buffer Descriptors

Figure 4-2 illustrates an example of SBD chaining. The host has formed a three-buffer message by linking three SBDs. In this example, the SAR transmits a message sequence of buffer A, then buffer B, and finally buffer C as the end of message.

Figure 4-2. Segmentation Buffer Descriptor Chaining



4.2.2.4 Transmit Queues

Once the linked list of SBDs is complete, the host submits the chained message to the SAR for segmentation. The host uses one of the 32 transmit queues for this purpose. Each transmit queue is a circular queue of two-word entries. The host identifies the next available transmit queue entry according to the write-only host interface specification described in Section 3.3.1. The host processor writes a pointer to the SAR SEG-shared memory SBD onto the next available transmit queue entry. During this write, the host also sets the VLD bit to indicate the entry is valid.

The CN8237 detects this write by snooping SAR SEG-shared memory accesses. When a write occurs to any of the transmit queues, the CN8237 marks that queue as pending. Once every cell slot, the CN8237 services one queue entry from one Transmit Queue. The system designer selects one of two service order priority schemes. With the SEG_CTRL(TX_RND) bit set to 1, the CN8237 services queues in round-robin order (that is, one entry per queue in transmit queue sequence for all active queues). With the SEG_CTRL(TX_RND) bit set to 0, the CN8237 services the queues in fixed priority order (that is, entries from higher priority active queues are serviced before lower priority queue entries, with transmit queue 31 having highest priority).

In either case, the SAR services one transmit queue entry by linking the new buffer descriptor chain to the VCC table entry identified by the VCC_INDEX in the first SBD. The VCC table entry includes pointers to buffer descriptors for segmentation. The SAR will link the new SBDs to the current chain of SBDs on a VCC. The host can submit data to a VCC while the SAR is segmenting a previously submitted message. Once the chain has been linked, the CN8237 resets the transmit queue VLD bit to 0.

SAR Transmit Queue Processing

Figures 4-3 and 4-4 illustrate the process of linking SBDs to a VCC table entry.

NOTE: As the new buffers are submitted, the VCC is processing a single buffer PDU (BOM/EOM). The CN8237 accepts new PDUs while it is processing outstanding buffers.

Figure 4-3. Before SAR Transmit Queue Entry Processing

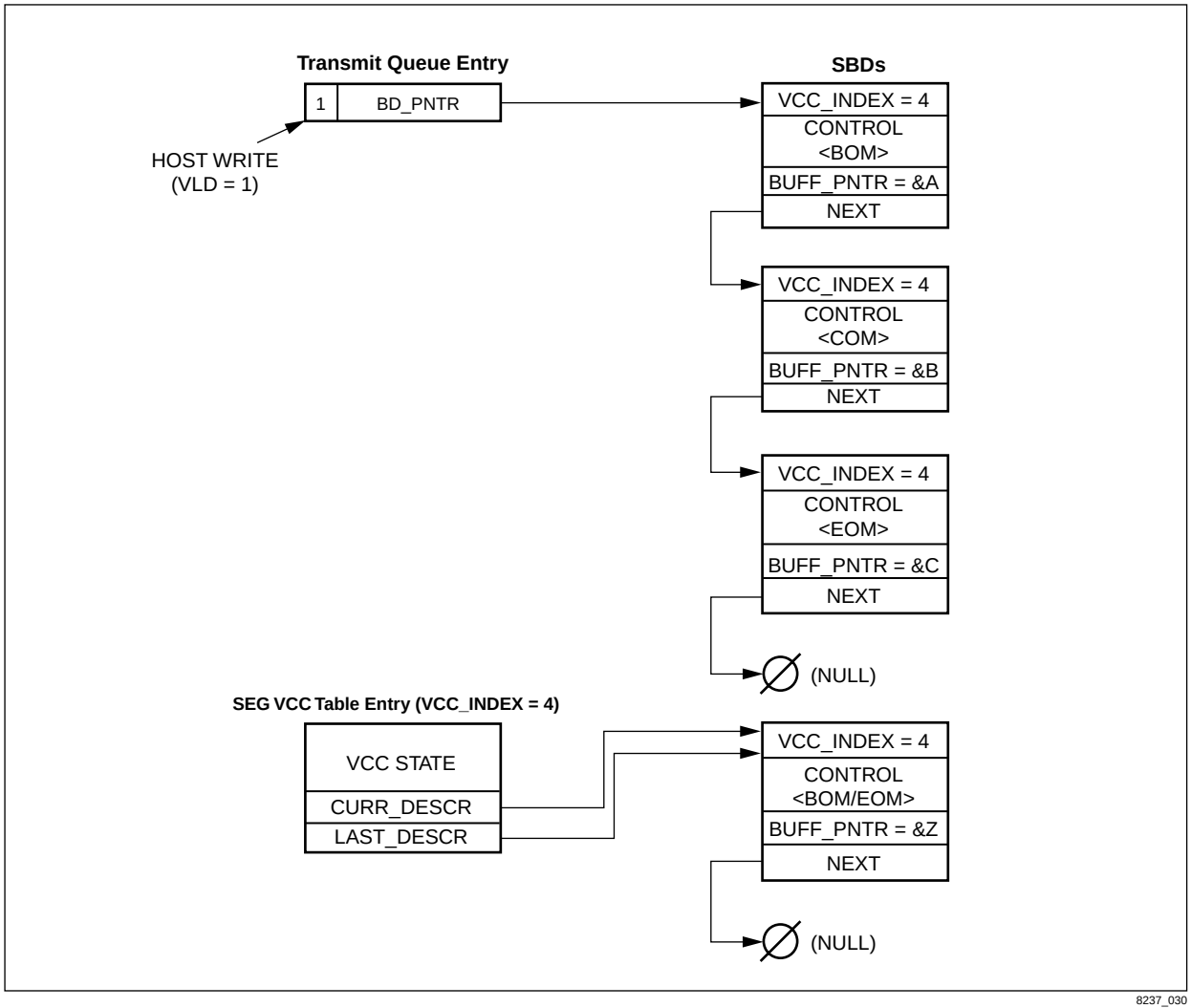
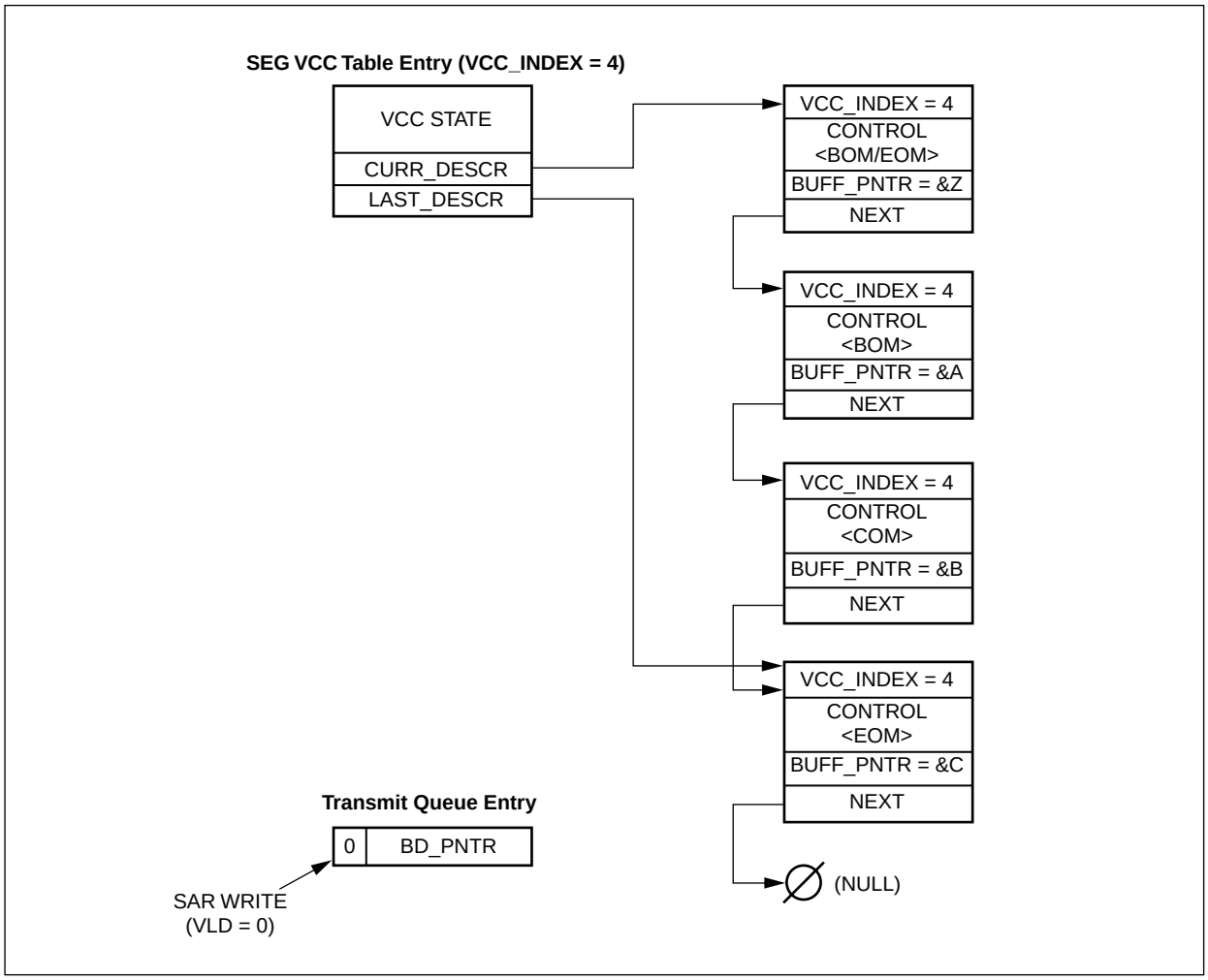


Figure 4-4. After SAR Transmit Queue Entry Processing



4.2.2.5 Partial PDUs

The host can submit partial PDUs to the CN8237. In this case, the SAR transmits the data and halts on a cell boundary. The partial PDU buffers are not required to be aligned to a cell boundary by the host. The CN8237 tracks the remaining segmentation data. If a partial cell remains, the CN8237 holds the buffer until it can complete a cell. Once the host submits an additional buffer, the CN8237 resumes segmentation.

4.2.2.6 Virtual Paths

For network management simplicity, the host can create Virtual Path VCCs (that is, it can segment many VCIs on one VP). The host supplies the VCI for the ATM header within each Segmentation Buffer Descriptor (SBD). When using this method, the CN8237 must be provided with contiguous linked SBDs that are of the same VCC_INDEX (that is, the same VCI) for the length of the PDU. This allows the CN8237 to multiplex VCI messages at the PDU level. For AAL5 segmentation, the host must guarantee that SBDs are linked with PDU multiplexing to preserve CPCS-PDU integrity.

4.2.3 CPCS-PDU Processing

The buffers submitted by the user contain only user data, the CPCS Service Data Unit (CPCS-SDU). The CN8237 adds the CPCS-PDU protocol fields to the CPCS-SDU. The SAR supports two AAL levels: AAL5 and a transparent adaptation layer (AAL0).

Specific features also allow the generation of OAM cells. Chapter 7.0 covers OAM generation in detail.

4.2.3.1 AAL5

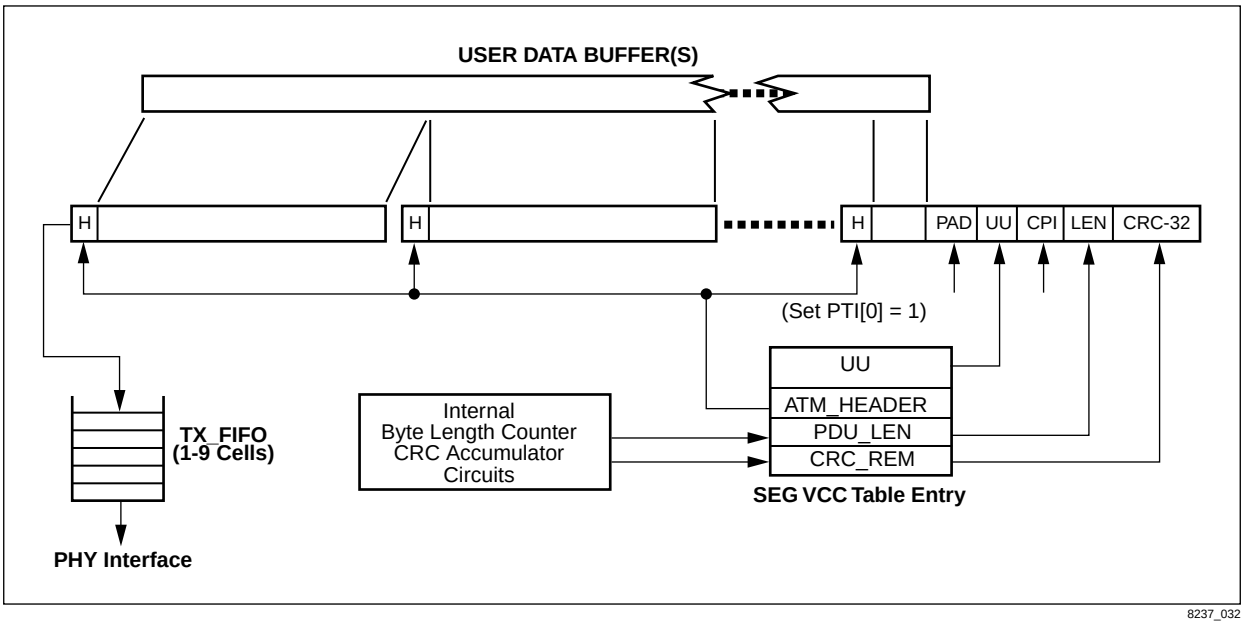
For AAL5, the SAR generates the CPCS-PDU trailer and pads the CPCS-SDU to align the PDU to a cell boundary. The CN8237 generates the PAD, Length (LEN), Common Part Indicator (CPI), and Cyclic Redundancy Check (CRC) fields according to I.363. The host supplies the CPCS User-to-User Indication (UU) field in the first buffer descriptor in a message, and the CN8237 transmits it following I.363.

The SAR generates the ATM header according to host-initialized settings in the VCC table entry. The CN8237 terminates AAL5 PDUs by setting bit 0 of the Payload Type Identifier field, PTI[0] = 1.

The host aborts PDUs by activating an EOM SBD abort option. The host activates this by setting the following fields in the SEG buffer descriptor entry to these values: AAL_MODE = AAL5 (b00), and AAL_OPT = ABORT (b01).

Figure 4-5 illustrates the CN8237's AAL5 PDU generation scheme. The SAR uses internal circuits to generate and store PDU length and CRC-32 in the SEG VCC table. The CN8237 transmits these fields within the EOM cell. The PAD and CPI fields are generated internally.

Figure 4-5. AAL5 CPCS-PDU Generation



4.2.3.2 AAL0

The CN8237 also supports a transparent or NULL adaptation layer, AAL0. AAL0 maps CPCS-SDUs directly to CPCS-PDU payloads. The SAR pads the SDU to a 48-byte cell payload boundary, but generates no other overhead. The SAR generates the ATM header and PDU termination indications in the same manner as it does with AAL5.

4.2.4 ATM PHY Layer Interface

Once the segmentation coprocessor has formed an ATM cell, the CN8237 transfers the cell to the transmit FIFO buffer. The user chooses the length of this FIFO buffer, with possible sizes from one to nine cells. The FIFO buffer depth is programmable, since there is a trade-off between absorbing PCI latency with a longer FIFO buffer, and introducing greater jitter. Section 6.2.3.3 discusses this trade-off in greater depth. Once sent to the transmit FIFO buffer, the cell passes through the FIFO buffer to the PHY layer interface circuits.

4.2.5 Status Reporting

The CN8237 informs the host of segmentation completion using segmentation status queues. The host assigns each VCC to one of 32 status queues, enabling a multiple peer architecture as described in Section 3.2. The CN8237 reports status entry on either PDU or buffer boundaries, selectable on a per-VCC basis by setting the STM_MODE bit. PDU boundary status is referred to as Message Mode, while buffer status reporting is called Streaming Mode. Error conditions also generate status queue entries, though this is a rare occurrence within a CN8237 subsystem's segmentation block. The segmentation status queues operate according to the write-only host interface, defined in Section 3.3.2.

The CN8237 returns a user-supplied field (USER_PNTR) from the first SBD associated with the status entry. The SAR does not use this field for any internal purpose; it simply circulates the information back to the host. The value of USER_PNTR must uniquely describe the segmented buffer associated with the SBD. USER_PNTR may contain the address of the buffer or of a host data structure describing the buffer. To simplify host management, the CN8237 also returns the VCC_INDEX of the VCC on which the buffer was transmitted.

4.2.6 Virtual FIFO Buffers

In addition to gathering PDU data from buffers, the CN8237 provides an optional method to segment from a fixed PCI address, or Virtual FIFO buffer. The CN8237 supports AAL0, CBR Virtual FIFO buffer segmentation.

The host configures the channel for Virtual FIFO buffer operation by setting the CURR_PNTR and RUN fields to 0 in the SEG VCC table entry. The host writes the FIFO buffer address to the FIFO_PNTR field in the VCC table entry. The host also initially sets the SCH_MODE field = CBR.

At this point the host can start writing cells to the external host transmit FIFO buffer. Once the FIFO buffer is almost full, the host sets the RUN bit to a logic high. The SAR will then start reading from the FIFO buffer. When the FIFO buffer gets below almost empty, the host will set the SCH_OPT bit to a logic high. The SAR will then skip a cell transmit opportunity in order to allow the FIFO buffer to refill. After the SAR skips a cell, it will reset the SCH_OPT bit to a logic low.

In this mode, the segmentation coprocessor reads 48 bytes of payload from the host FIFO buffer, and prepends (that is, attaches to the beginning) the ATM_HEADER value in the VCC table entry. The host does not use the transmit queue for Virtual FIFO buffers. The CN8237 transmits cell payloads from this location indefinitely, with no status reporting.

4.3 Segmentation Control and Data Structures

4.3.1 Segmentation VCC Table Entry

Each Segmentation VCC table entry occupies one 10-word descriptor of the Segmentation VCC table. The first 7 words are generic, independent of traffic class. The last 3 or 13 words provide additional parameters specific to service classes.

There are two basic formats for the generic part of the VCC table entry: AAL5-AAL0 and Virtual FIFO buffer. Each completely describe the state of the segmentation process for individual VCCs. Tables 4-2 and 4-3 describe the format of AAL5-AAL0 VCC table entries.

Table 4-2. Segmentation VCC Table Entry—AAL5-AAL0 Format *(1 of 2)*

Word	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	PM_INDEX							Rsvd	ACK_PM	FWD_PM	Rsvd	LAST_PNTR																				
1	UU								PORT_ID			BOM_PNTR																				
2	ATM_HEADER																															
3	PDU_LEN																BUFFER_LEN															
4	CRC_REM																															

Table 4-2. Segmentation VCC Table Entry—AAL5-AAL0 Format *(2 of 2)*

Word	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
5	STM_MODE	STAT					PM_EN	LAST_CLP	Run	NX_EOM	Rsvd	CURR_PNTR																				
6	SCH	BCK_PM	VPC	SCH_GFR/CBR_W_TUN	FLOW_ST	GFR_PRI			SCH_MODE			PRI				SCH_OPT	NEXT_VCC															
7-9/19	SCH_STATE																															

KEY:

= Written by host at VCC setup

= May be dynamically modified during active segmentation

Table 4-3. Segmentation VCC Table Entry—AAL5 and 0 Field Descriptions *(1 of 2)*

Field Name	Description
PM_INDEX	Performance Monitoring index. 128 VCCs can be selected for automatic OAM PM generation. Each monitored VCC has a unique performance monitoring index. If this field is changed while the VCC is active, only the byte containing the field should be written. (See Chapter 6.0.)
ACK_PM	Toggled after each backward reporting PM cell is sent. Used to prevent PM information in PM table from being overwritten before cell is sent.
FWD_PM	Indicates that next call should be a forward monitoring PM cell.
LAST_PNTR	Points to last buffer descriptor currently linked to the VCC. The two least significant bits of the pointer are assumed to be 0 (word-aligned). The address of the buffer descriptor is (LAST_PNTR << 2) or (LAST_PNTR x 4).
UU	AAL5 User-to-User indication. This field is copied from the buffer descriptor UU field. The CN8237 inserts the UU field in the CPCS-PDU trailer contained in the EOM cell.
PORT_ID	Identifies the Physical Device (Port) ID for the VCC.
BOM_PNTR	In Message mode (STM_MODE=0), it points to the first buffer descriptor of a message composed of more than one buffer descriptor. The CN8237 returns the corresponding USER_PNTR from this buffer descriptor in the status queue. In Streaming mode (STM_MODE=1) it is not used. The two least significant bits of the pointer are assumed to be zero (word-aligned). The address of the buffer descriptor is (BOM_PNTR << 2) or (BOM_PNTR * 4).
ATM_HEADER	Used for each ATM cell for the VCC. The transmitted header may be modified by option bits in the current buffer descriptor.
PDU_LEN	CPCS-PDU Trailer Length field. This field is generated by the SAR and inserted in the Length field of the EOM cell.
BUFFER_LEN	Buffer Length. The number of data bytes read from the current segmentation buffer.
CRC_REM	Accumulated value of AAL5 32-bit CRC, calculated on-the-fly by the SAR and appended to the PDU at EOM.
STM_MODE	Streaming Status Mode. 0 = Message Mode: a status entry is written (with a corresponding maskable interrupt) only when the last buffer in a message completes segmentation. The status entry written corresponds to the first buffer descriptor in the message. 1 = Streaming Mode: a status entry is written (with a corresponding maskable interrupt) for each buffer as it completes segmentation.
STAT	Identifies the Segmentation Status Queue used for all status entries.
PM_EN	Enables performance monitoring for the VCC. If this bit is changed while the VCC is active, only the byte containing the bit should be written. (See Chapter 7.0.)
LAST_CLP	Last transmitted CLP bit for VCC. This bit is updated by the SRC after each transmitted cell. This bit is in the same word as the PM_EN bit, and is copied by the SAR from the current buffer descriptor. The bit is used to select the correct VBR bucket for certain VBR VCCs.
RUN	Flag that indicates that segmentation is proceeding. This flag is set to one by the SAR when the host supplies data for the VCC; it is set to zero by the SAR when the data available for the VCC is not sufficient to send at entire ATM cell.
NX_EOM	Indicates that the next cell is the end of a CPCS-PDU.

Table 4-3. Segmentation VCC Table Entry—AAL5 and 0 Field Descriptions *(2 of 2)*

Field Name	Description
CURR_PNTR	Pointer to the current buffer descriptor for the VCC. This field is automatically updated by the SAR. The two least significant bits of the pointer are assumed to be 0 (word-aligned). The address of the descriptor is (CURR_PNTR << 2) or (CURR_PNTR x 4).
SCH	Indicates VCC is currently scheduled for segmentation.
BLK_PM	Toggled after each backward reporting PM cell is scheduled. Used to prevent PM information in PM table from being overwritten before the cell is sent.
VPC	On a VCC with SCH_MODE = ABR this indicates a VP (instead of a VCC) connection, and RM cells will be generated on VCI = 6.
SCH_GFR/ CBR_W_TUN	If SCH_MODE = GFR, this bit set to a logic high indicates that the VCC is currently scheduled on a GFR_PRI priority queue for segmentation. The SCH bit set to a logic high indicates that the VCC is currently scheduled on a VBR priority queue. This host does not have to set this bit—it is set by the SAR. If SCH_MODE = CBR, this bit set to a logic high specifies that an unused CBR schedule slot should be used as a tunnel slot, with tunnel priorities specified in word 7 of the VCC table entry.
FLOW_ST	Flow control state. This bit is active only in ER mode. Indicates that the priority of the connection is increased to insure MCR.
GFR_PRI	Specifies the UBR priority level for a GFR service connection. GFR_PRI must be < PRI.
SCH_MODE	Traffic class of VCC. (See Chapter 6.0 for details.) 000 UBR = Unspecified Bit Rate 001 CBR = Constant Bit Rate 010 = Reserved 011 GFR = Guaranteed Frame Rate 100 VBR1 = Single Leaky Bucket VBR 101 VBR2 = Dual Leaky Bucket VBR with both buckets always active 110 VBRC = Dual Leaky Bucket VBR with bucket 1 applied only to CLP = 0 cells 111 ABR = Available Bit Rate (as specified by TM 4.0)
PRI	Segmentation priority. The lowest priority is 0. The highest priority is 15. This field is not active when SCH_MODE is CBR. When SCH_MODE is GFR, this specifies the VBR priority.
SCH_OPT	Schedule option. The use of this bit depends on the setting of the SCH_MODE field. VBR1, VBR2, VBRC, ER: Initializes bucket state to send maximum burst. The SAR writes this bit to 0 after bucket state initialized. CBR: Indicates that the next cell slot opportunity should be skipped. This bit is written by the host and cleared by the SAR after the cell slot is skipped. If this bit is changed while the VCC is active, only the byte containing the bit should be written.
NEXT_VCC	Used by SAR to link VCCs in schedule chains.
SCH_STATE	Specific scheduling state information. The contents of this field depend on the setting of the SCH_MODE field. It is not used when SCH_MODE is set to UBR. (The contents of this field are detailed in Chapter 6.0.)
Reserved	Always set to 0.

Table 4-4 shows the format for Virtual FIFO buffer VCC table entries, including setup values for restricted fields. Table 4-5 describes the field that differs from the AAL5-AAL0 format.

Table 4-4. Segmentation VCC Table Entry—Virtual FIFO Buffers

Word	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	PM_INDEX							Reserved				LAST_PNTR																				
1	Reserved								Reserved			BOM_PNTR																			
2	ATM_HEADER																															
3	FIFO_PNTR																															
4	CRC_REM																															
5	STM_MODE	STAT						PM_EN	Reserved	RUN=1	Reserved	CURR_PNTR= 0x00000																				
6	Reserved								SCH_MODE (=001 - CBR)			PRI				SCH_OPT	Reserved														
7-9/19	SCH_STATE																															

Table 4-5. Segmentation VCC Table Entry—Virtual FIFO Buffer Format Field Descriptions

Field Name	Description
FIFO_PNTR	Pointer to the PCI space data FIFO buffer for CBR_FIFO scheduling mode.
Reserved	Always set to 0.

4.3.2 Data Buffers

Data buffers contain CPCS-SDUs for segmentation and reside in host or SAR SEG-shared memory. The CN8237 retrieves host data buffers from any byte aligned PCI address using the READ Multiple PCI command. SAR SEG-shared data buffers must be aligned on word boundaries. Buffers contain any number of bytes of only user data, up to a maximum of 64 KB.

4.3.3 Segmentation Buffer Descriptors

SBDs reside in SAR SEG-shared memory on word-aligned addresses. The host controls the allocation and management of SBDs. For each buffer to be segmented, the host utilizes one buffer descriptor from its pool of free descriptors.

Tables 4-6 through 4-9 describe the entry formats and field definitions for the SBDs.

Table 4-6. Segmentation Buffer Descriptor Entry Format

Word	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	UU								Rsvd	NEXT_PNTR																						Rsvd
1	USER_PNTR																															
2	BUFFER_PNTR																															
3	Reserved	LOCAL	SET_CI	SET_CLP	HEADER_MOD	RPL_VCI	OAM_STAT	GEN_PDU	CRC10	AAL_MODE		AAL_OPT		CELL	BOM	EOM	LENGTH															
4	MISC_DATA																SEG_VCC_INDEX															
5	Reserved																															

Table 4-7. MISC_DATA Field Bit Definitions with HEADER_MOD Bit Set

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Def.	GFC_DATA				Reserved		WR_GFC	WR_PTI	WR_VCI	PTI_DATA			VCI_DATA			
NOTE(S): This definition of bits 31:16, MISC_DATA field, applies when the HEADER_MOD bit is set and RPL_VCI=0; used when generating OAM cells.																

Table 4-8. MISC_DATA Field Bit Definitions with RPL_VCI BIt Set

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Def.	NEW_VCI															
NOTE(S): This definition of bits 31:16, MISC_DATA field, applies when the RPL_VCI bit is set; used when identifying virtual channels under a VP VCC.																

***Table 4-9. Segmentation Buffer Descriptor Field Descriptions** (1 of 2)*

Field Name	Description
UU	AAL5 User-to-User indication. This field is copied to the VCC table entry UU field when BOM is set and AAL_MODE is AAL5.
NEXT_PNTR	Pointer to next buffer descriptor for the VCC. The two least significant bits of the pointer are assumed to be 0 (word-aligned). The host links segmentation buffer descriptors by writing this field to [(ADDRESS of SDB)>>2] or [(ADDRESS of SBD)/4] before submitting the chain on the Transmit Queue. The NEXT_PNTR of the last buffer descriptor in a chain is set to NULL (=0).
USER_PNTR	User data field returned in status entry. This field may equal BUFFER_PNTR. The SAR circulates this field back to the host in the status entry without using it internally.
BUFFER_PNTR	Pointer to segmentation buffer in host- or SAR SEG-shared memory space. Host or SAR SEG-shared memory location is determined by the LOCAL bit.
LOCAL	0 = BUFFER_PNTR is a byte aligned PCI address. 1 = BUFFER_PNTR is a word-aligned address in SAR SEG-shared memory instead of host memory.
SET_CI	0 = The CN8237 generates bit 1 of PTI[2:0] from the VCC table entry ATM_HEADER field. 1 = Sets bit 1 of the ATM header PTI[2:0] field for all cells in buffer to 1.
SET_CLP	0 = The CN8237 generates the CLP bit from the VCC table entry ATM_HEADER field. 1 = Sets the ATM header CLP bit for all cells in buffer to 1. Also used to control VBR CLP Dual Leaky Bucket mode.
HEADER_MOD	0 = The CN8237 ignores the WR_GFC, WR_PTI, and WR_VCI bits in this buffer descriptor. 1 = Activates the WR_GFC, WR_PTI, and WR_VCI bits for this buffer descriptor.
RPL_VCI	0 = The CN8237 generates the VCI field from the VCC table entry ATM_HEADER field. 1 = The CN8237 generates the VCI field from the NEW_VCI for all cells in the buffer. NEW_VCI is also copied in to the VCI portion of the ATM_HEADER field in the VCC entry.
OAM_STAT	Buffer will reports status to the global OAM Status Queue SEG_CTRL(OAM_STAT_ID) instead of the STAT specified in the VCC table entry. For Message mode VCCs (VCC table entry STM_MODE = 0), the OAM_STAT bit of the last descriptor for the CPCS-PDU determines which queue to use (even though the first descriptor is returned in message mode). See Chapter 7.0—OAM for more details.
GEN_PDU	1 = Generates AAL5 trailer.
CRC10	Overwrite last ten bits of each cell with CRC10 calculation. Used for OAM cells.
AAL_MODE	Controls AAL segmentation mode. 00 = AAL5 01 = AAL0 Read 48-octet ATM cell payload from segmentation buffer. Only formatting is to set PTI[0] on last cell of an EOM buffer. 10 = Reserved 11 = Reserved

Table 4-9. Segmentation Buffer Descriptor Field Descriptions *(2 of 2)*

Field Name	Description
AAL_OPT	Options for AAL_MODE: For AAL_MODE = AAL0 00 = Normal operation 01 = SINGLE 10 = Reserved 11 = Reserved For AAL_MODE = AAL5 00 = Normal operation 01 = ABORT SINGLE: LENGTH is ignored and 48-octets are read from buffer to form the payload of a single ATM cell. VCC table entry CRC_REM, BUFF_LENGTH, and PDU_LENGTH are not affected. By using the LINK_HEAD bit in the transmit entry, the buffer descriptor is linked at start of buffer descriptor chain for VCC. This means that there are no concerns with mid-cell insertion and that the cell will have low latency. This is intended for OAM cells. NOTE: This option MUST be set in the buffer descriptor for any Tx Queue entry with LINK_HEAD set. To do otherwise may result in corrupted SEG data. ABORT: Send AAL5 ABORT cell (no data is read from segmentation buffer). A buffer that has both ABORT and BOM set is returned without sending an abort cell.
CELL	0 = CN8237 reads the 48-octet payload of ATM cells from memory and generates the ATM header internally. 1 = The CN8237 reads the entire 52-octet ATM cell from segmentation buffer. The ATM_HEADER stored in the VCC table entry is not used in this mode and AAL_MODE is ignored.
BOM	1 = Buffer contains the beginning of a message. (See Table 4-1.)
EOM	1 = Buffer contains the end of a message. (See Table 4-1.)
LENGTH	Number of bytes of data contained in the segmentation buffer. Local memory, non-EOM buffer lengths must be multiples of 4 bytes (mod 4). Maximum allowable size is 64 KB.
GFC_DATA	Data for WR_GFC option.
WR_GFC	0 = The CN8237 generates the GFC field from the VCC table entry ATM_HEADER field. 1 = The CN8237 overwrites the ATM header GFC field for all cells in the buffer with GFC_DATA. Global GFC changes (active for all buffers of VCC) can be set in the VCC table entry ATM header. This bit is active only when HEADER_MOD is set.
WR_PTI	0 = The CN8237 generates the PTI field from the VCC table entry ATM_HEADER field. 1 = The CN8237 overwrites the ATM header PTI field for all cells in the buffer with PTI_DATA. The host can use this feature to generate F5 and PM OAM cells. See Chapter 7.0. This bit is active only when HEADER_MOD is set. This bit disables PM TUC and BIP16 calculations.
WR_VCI	0 = The CN8237 generates the VCI field from the VCC table entry ATM_HEADER field. 1 = The CN8237 overwrites the ATM header VCI field for all cells in the buffer with (0x0000|VCI_DATA). (MSBs of VCI are set to 0.) Used to generate F4 OAM cells. (See Chapter 7.0.) This bit is active only when HEADER_MOD is set. This bit disables PM TUC and BIP16 calculations.
PTI_DATA	Data for WR_PTI option. Normally used to generate OAM cells.
VCI_DATA	Data for WR_VCI option. Normally used to generate OAM cells.
NEW_VCI	Data for the RPL_VCI. The CN8237 overwrites the VCC table entry ATM_HEADER VCI field with this data. Therefore, the effect is permanent until the next buffer descriptor with RPL_VCI is processed.
SEG_VCC_INDEX	Identifies the VCC entry in the VCC table. The CN8237 links this buffer descriptor to the identified VCC.
Reserved	Always set to 0.

4.3.4 Transmit Queues

4.3.4.1 Entry Format

The host submits chains of SBDs to the CN8237 by writing a double word transmit queue entry. Tables 4-10, 4-11, and 4-12 describe the format of these entries.

Table 4-10. Transmit Queue Entry Format (64-bit)

Word	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	VLD	LINK_HEAD	FIND_CHAIN	Reserved						SEG_BD_PNTR																					Rsvd	
	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
	Reserved																															

Table 4-11. Transmit Queue Entry Format (32-bit)

Word	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	VLD	LINK_HEAD	FIND_CHAIN	Reserved						SEG_BD_PNTR																					Rsvd	
1	Reserved (Must be Written)																															

Table 4-12. Transmit Queue Entry Field Descriptions

Field Name	Description
VLD	0 = Entry invalid. Waiting for the host to submit new data for segmentation. 1 = Entry valid. The SAR processes the entry when its read pointer into the queue advances to this entry. Written to 1 by the host when submitting a new entry. The SAR clears this bit to 0 when it has successfully linked the buffer descriptor chain to the VCC table.
LINK_HEAD	0 = The CN8237 links the new descriptor chain at the end of the existing chain on the VCC. 1 = The CN8237 links the new descriptor chain at the head of the existing chain. If this bit is set, the buffer must contain data for at least one cell. Only a single buffer descriptor can be linked to a transmit queue entry when this bit is set. This bit is intended for use with the buffer descriptor SINGLE option to send in line management cells with reduced latency. ***NOTE(S):*** It is mandatory that the SINGLE option is set in the buffer descriptor for any Tx Queue entry with LINK_HEAD set. To do otherwise may result in corrupted segmentation data.
FIND_CHAIN	Indicates the SAR is searching for the end of the buffer descriptor chain. The host always writes this bit to 0.
SEG_BD_PNTR	Points to the first buffer descriptor in the new buffer descriptor chain. Bits 22:2 of the address are specified; the two least significant bits of the pointer are assumed to be 0 (word-aligned).
Reserved	Always set to 0.

4.3.4.2 Transmit Queue Management

The transmit queues reside in a single continuous section of SAR SEG-shared memory. During initialization the host configures the number of entries per queue with the SEG_CTRL(TR_SIZE) field. The size ranges from 64 to 4096 entries per queue. The host also selects a priority scheme at initialization with the SEG_CTRL(TX_RND) bit. Both of these fields are static configurations and must not be changed during runtime operation.

By initializing the SEG_TXBASE register, the host determines the base address of all active transmit queues. This register contains the base address of the first queue, the number of active queues, and the write-only update interval for all queues. A set of other internal registers, the Transmit Queue Base table entries, track the current state of the queues. Tables 4-13, 4-14, and 4-15 below describe the fields of these queues.

The byte address of any transmit queue entry is given by:

(SEG_TXBASE(SEG_TXB) × 128) + <transmit queue number>
× <decoded TQ_SIZE value> + <entry number> × 8

The host manages each transmit queue as an independent write-only control queue. Chapter 3.0 describes the runtime management of a write-only control queue. The transmit queue base table contains all of the queue control variables except for INTERVAL, which is located in the SEG_TXBASE register.

Figure 4-6. Segmentation Master Control Word

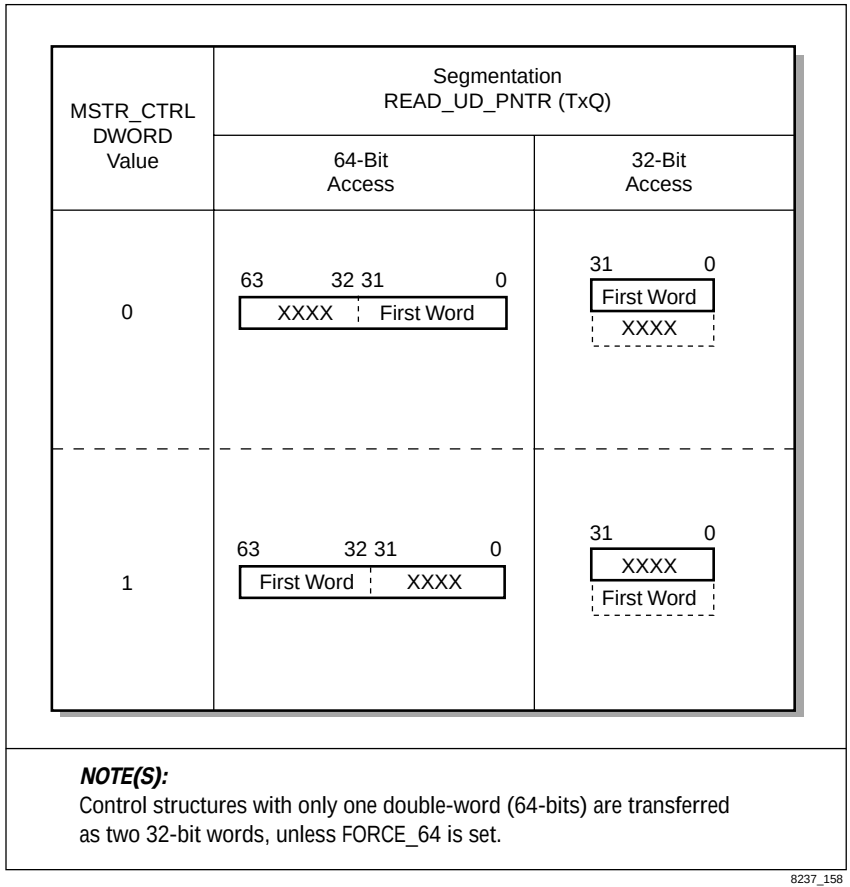


Table 4-13. Transmit Queue Base Table Entry (64-bit)

Word	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	READ_UD_PNTR																													Reserved		
	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
	Reserved								UPDATE								Reserved				READ											

Table 4-14. Transmit Queue Base Table Entry (32-bit)

Word	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	READ_UD_PNTR																													Rsvd		
1	Reserved								UPDATE								Reserved				READ											

Table 4-15. Transmit Queue Base Table Entry Field Descriptions

Field Name	Description
READ_UD_PNTR	Points to READ_UD used by host to prevent queue overflow. The SAR will write its read pointer into the queue to this address periodically. (See Chapter 2.0 for details.) Read Update will be written to host as 8-byte entity with lower 4-bytes disabled. ***NOTE(S):*** 1. MSTR_CTRL_DWORD must be set to 1. 2. Control structures (which SAR writes) with only double-word (64 bits) will be transferred as two 32-bit words, unless FORCE64 is set.
UPDATE	SAR position in update interval. Number of queue entries processed since last update of READ_UD.
READ	SAR read pointer. Represents the SAR's current position in the transmit queue.
Reserved	Always set to 0.

4.3.5 Routing Tags

Based upon the setting of CONFIG1(TAG_SIZE) field, routing tags need to be prepended to the cell written into the TxFIFO. Figures 4-7 through 4-9 show the routing tag tables and position in TxFIFO. The segmentation block writes either 4, 8, or 12 bytes into the TxFIFO, depending upon the size of tag required. The UTOPIA block strips off the appropriate number of bytes to form the desired size cell. Table 4-17 is a cross-reference between the routing tag table and the UTOPIA cell.

NOTE: The maximum TxFIFO size is reduced when using routing tags. See Table 4-16.

Table 4-16. Maximum TxFIFO Size with Routing Tags

Tag Size	TxFIFO (Maximum Number Of Cells)
2	9
4	
6	8
8	
10	7

Figure 4-7. Route Tag Table for tag_size = 2 and 4

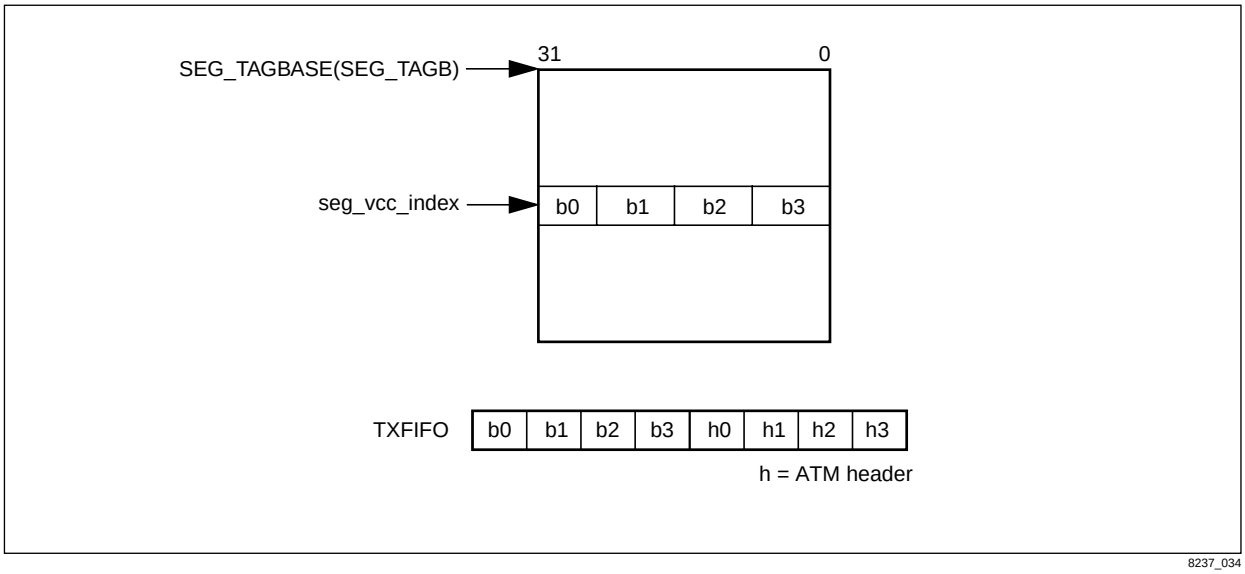


Figure 4-8. Route Tag Table for tag_size = 6 and 8

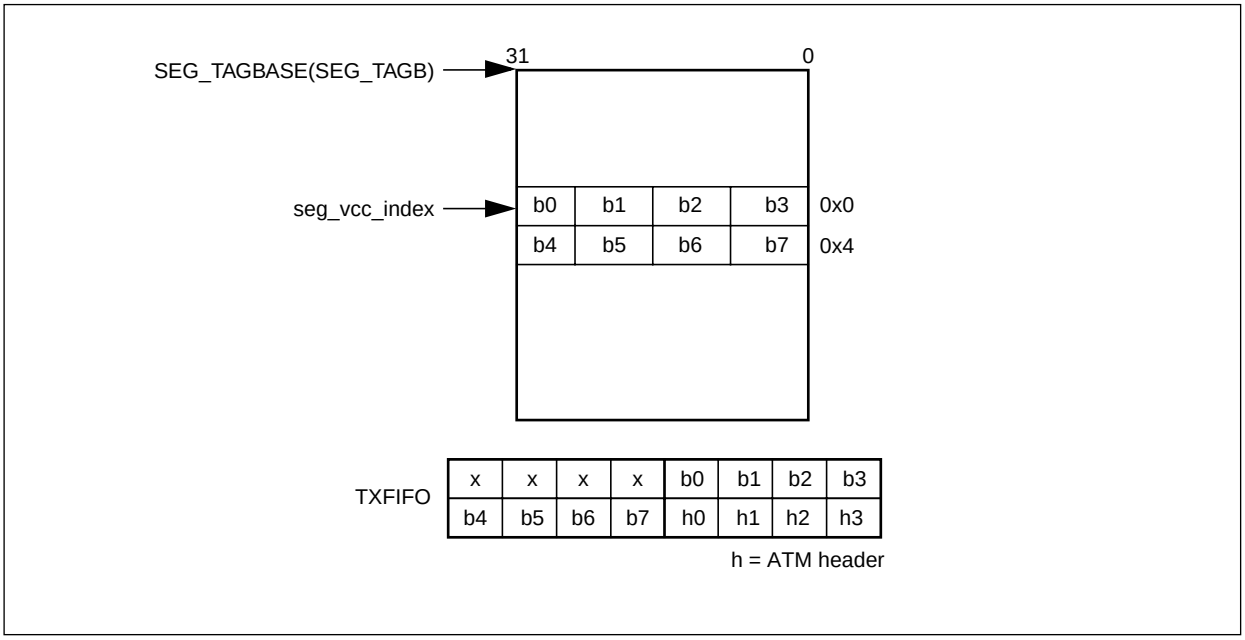


Figure 4-9. Route Tag Table for tag_size = 10

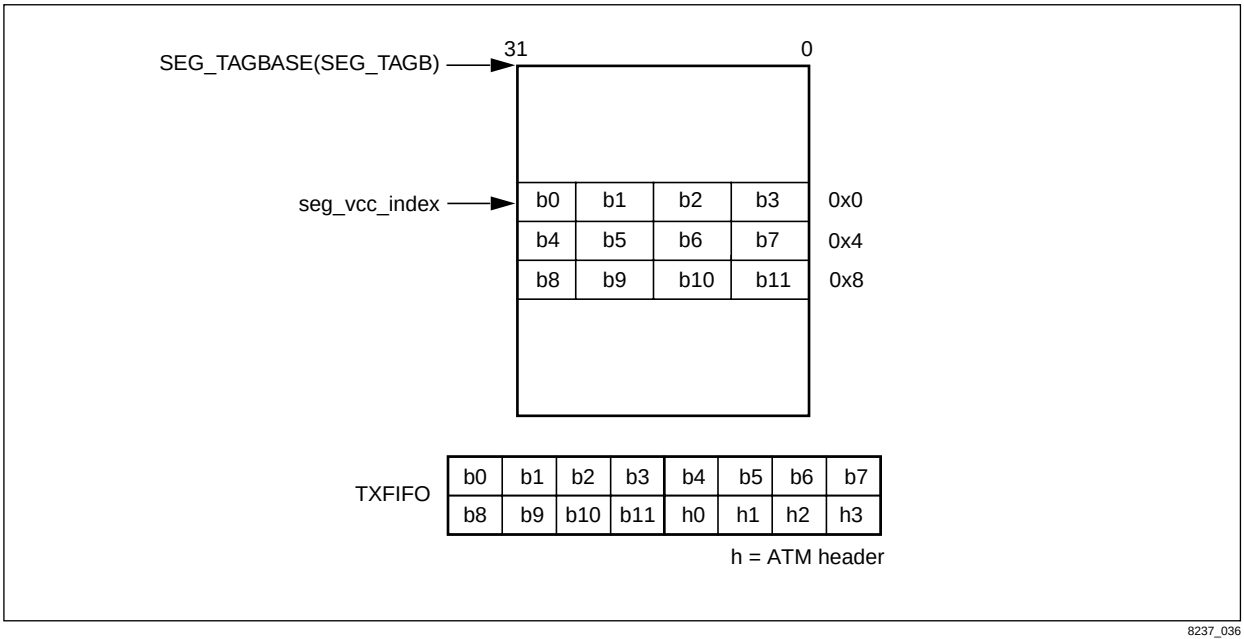


Table 4-17. Routing Tag Cross-Reference

Tag Size	b0	b1	b2	b3	b4	b5	b6	b7	b8	b9	b10	b11
2			t1	t2								
4	t1	t2	t3	t4								
6			t1	t2	t3	t4	t5	t6				
8	t1	t2	t3	t4	t5	t6	t7	t8				
10			t1	t2	t3	t4	t5	t6	t7	t8	t9	t10

4.3.6 Segmentation Status Queues

4.3.6.1 Entry Format

The CN8237 reports segmentation status to the host on one of 32 status queues. Each entry on the queue is two words. Table 4-18, 4-19, and 4-20 describe the format of a standard SEG status queue entry. A second special status queue format for ACR/ER change notification is described in Tables 4-21, 4-22, and Table 4-23. Refer to Section 6.3.7.5 for a description of the trigger mechanism for posting special status.

NOTE: MSTR_CTRL_DWORD must be set to 1.

Figure 4-10. Segmentation Master Control Word

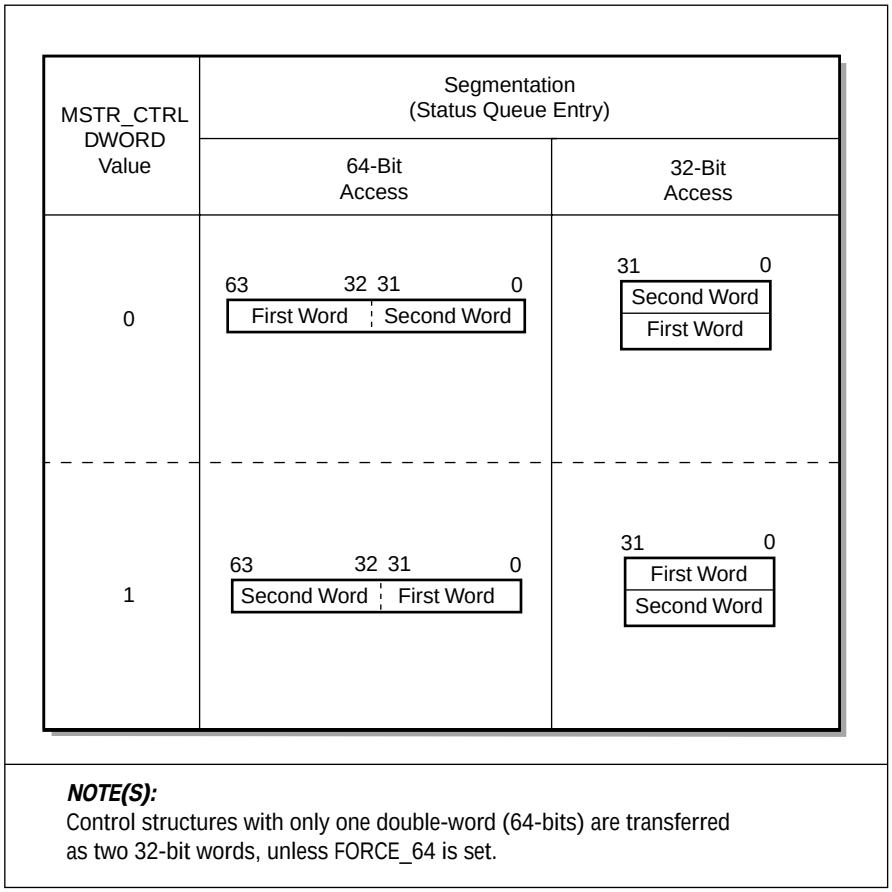


Table 4-18. Segmentation Status Queue Entry (64-bit)

Word	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	USER_PNTR																															
	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
	VLD	NCR=0	STOP	DONE	SINGLE	OVFL	I_EXP					I_MAN		Reserved			SEG_VCC_INDEX															

Table 4-19. Segmentation Status Queue Entry (32-bit)

Word	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	USER_PNTR																															
1	VLD	NCR=0	STOP	DONE	SINGLE	OVFL	I_EXP					I_MAN		Reserved			SEG_VCC_INDEX															

Table 4-20. Segmentation Status Queue Entry Field Descriptions

Field Name	Description
USER_PNTR	Copy of the USER_PNTR from the segmentation buffer descriptor. The SAR circulates this field from the SBD without using it internally. In Message Mode, the SAR returns the USER_PNTR of the BOM buffer. In Streaming Mode, the SAR returns the USER_PNTR of all buffers.
VLD	0 = Entry invalid. Indicates that the SAR has not written the entry and the host should halt status processing. 1 = Entry valid. Indicates that the host may process the entry. Written to 1 by the SAR. Written to 0 by the host.
NCR	When set to a 0, indicates a normal status queue entry.
NCR_DIR	Indicates direction of ACR/ER trigger. Logic high for HI and logic low for LO.
STOP	VCC has stopped because no more segmentation data is available.
DONE	Set when buffer segmentation is complete and buffer is released to host.
SINGLE	Set if SINGLE option is set in the segmentation buffer descriptor.
OVFL	Overflow: status entry is last entry available. (See Status Queue Overflow, below.)
I_EXP	Current I_EXP rate parameter of the VCC. This field is written only when VCC_INDEX(SCH_MODE) = ABR. (See Chapter 6.0.)
I_MAN	The two MSBs of the current I_MAN rate parameter for the VCC. This field is written only when VCC_INDEX(SCH_MODE) = ABR. (See Chapter 6.0.)
SEG_VCC_INDEX	Segmentation VCC index on which the SAR transmitted the buffer or PDU.
Reserved	Always set to 0.

Table 4-21. Segmentation Status Queue Format for ACR/ER (64-bit)

Word	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	ER																ACR															
	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
	VLD	NCR=1	NCR_DIR	SCR_NOT_DEST	Reserved	OVFL	Reserved										SEG_VCC_INDEX															

Table 4-22. Segmentation Status Queue Format for ACR/ER (32-bit)

Word	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	ER																ACR															
1	VLD	NCR=1	NCR_DIR	SCR_NOT_DEST	Reserved	OVFL	Reserved										SEG_VCC_INDEX															

Table 4-23. Status Queue Entry Field Descriptions for ACR/ER

Field Name	Description
ER	For SRC_NOT_DEST = 0, reflects TA_ER, current ER field in BCK RM cell. Not valid for SRC_NOT_DEST = 1.
ACR	For SRC_NOT_DEST = 0, reflects TA_CCR current CCR field in BCK RM cell. For SRC_NOT_DEST = 1, reflects current CCR field in FWD RM cell.
VLD	Entry valid. Written to 1 by the SRC. Written to 0 by the host.
NCR	When set to a 1, indicates that status entry is an ACR Notification status entry.
SRC_NOT_DEST	A value of 1 indicates a source ACR change notification. A value of 0 indicates a destination ACR change notification.
OVFL	Overflow: status entry is last entry available.
SEG_VCC_INDEX	Segmentation VCC index for buffer.
Reserved	Always set to 0.

4.3.6.2 Status Queue Management

At initialization, the host assigns the location and size of up to 32 queues by initializing internal registers, the segmentation status queue base table entries. The location and size of each queue is independently programmable via these base tables.

The SAR tracks its current position and the most recent known host position in the queues with fields in the base table entries. The host manages the queues as write-only status queues. The status queue base table entry contains all of the SAR's write-only control variables.

Tables 4-24, 4-25, and 4-26 describe the format of these entries.

Table 4-24. Segmentation Status Queue Base Table Entry (64-bit)

Word	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	BASE_PNTR																														Reserved	
	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
	SIZE		Rsvd		WRITE												Reserved				READ_UD											

Table 4-25. Segmentation Status Queue Base Table Entry (32-bit)

Word	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	BASE_PNTR																														Reserved	
1	SIZE		Rsvd		WRITE												Reserved				READ_UD											

Table 4-26. Segmentation Status Queue Base Table Entry Field Descriptions

Field Name	Description
BASE_PNTR	Points (Bits 31:3) to base of status queue. Bits 2:0 are always 0 (word-aligned).
SIZE	Number of entries in this status queue: 00 = 64 01 = 256 10 = 1024 11 = 4096
WRITE	SAR write pointer. Represents the SAR's current position in the queue.
READ_UD	Last update of the host processor read pointer. This field is written by the host processor.
Reserved	Always set to 0.

4.3.6.3 Status Queue Overflow

Since status queues contain a finite number of entries, it is possible that the SAR will exhaust the available entries. Although the SAR handles this condition, the host should attempt to prevent overflows.

The CN8237 detects when it writes the last available entry in a status queue (WRITE=READ_UD-1), and alerts the host to this condition by setting the OVFL bit in the status entry. Until the host services the queue and increments the READ_UD pointer in the base table register, the CN8237 inhibits segmentation on all channels that report on the overflowed status queue. All other channels are unaffected.

4.3.7 Segmentation Internal SRAM Memory Map

As indicated in Table 4-27, the segmentation internal SRAM is in the address range 0x1400–0x17FF.

The segmentation status queue base table registers (SEG_ST_QUn) are in the address range 0x1400–0x14FF.

The internal transmit queue base table registers (SEG_TQ_QUn) are in the address range 0x1500–0x15FF.

Other internal segmentation and scheduler registers are in the address range 0x1640–0x17FF.

Table 4-27. Segmentation Internal SRAM Memory Map

Address	Name	Description
Segmentation Status Queue Base Table Registers (SEG_SQ_QUn):		
0x1400–0x1407	SEG_SQ_QU0	Status Queue 0 Base Table
0x1408–0x140F	SEG_SQ_QU1	Status Queue 1 Base Table
⋮	⋮	⋮
0x14F8–0x14FF	SEG_SQ_QU31	Status Queue 31 Base Table
Internal Transmit Queue Base Table Registers (SEG_TQ_QUn):		
0x1500–0x1507	SEG_TQ_QU0	Transmit Queue 0 Base Table
0x1508–0x150F	SEG_TQ_QU1	Transmit Queue 1Base Table
⋮	⋮	⋮
0x15F8–0x15FF	SEG_TQ_QU31	Transmit Queue 31Base Table
0x1600–0x163F	Reserved	Not implemented
Internal Segmentation and Scheduler Registers:		
0x1640–0x174F	Reserved	Initialize to 0xFFFFFFFF
0x1750–0x17FF	Reserved	Not implemented

5.0 Reassembly Coprocessor

5.1 Overview

The reassembly (RSM) coprocessor processes cells received from the ATM PHY Interface block. The coprocessor extracts the AAL SDU payload from the received cell stream and reassembles this information into buffers supplied by the host system. The CN8237 supports AAL5 and AAL0 reassembly, as well as cell mode (1-cell PDUs through a Virtual FIFO buffer, for CBR voice traffic).

The CN8237 reassembles up to 64 K VCCs simultaneously. Individual connections are identified through separate VPI and VCI Index table structures. The VPI/VCI Index table mechanism provides very fast consistent channel identification over the full range of VPI/VCI addresses.

CPCS-PDU payload data, the CPCS-SDU, fills the host-supplied data buffers assigned to each VCC. The host assigns each VCC to one or two of 32 independent buffer pools, from which the RSM coprocessor draws buffers as needed.

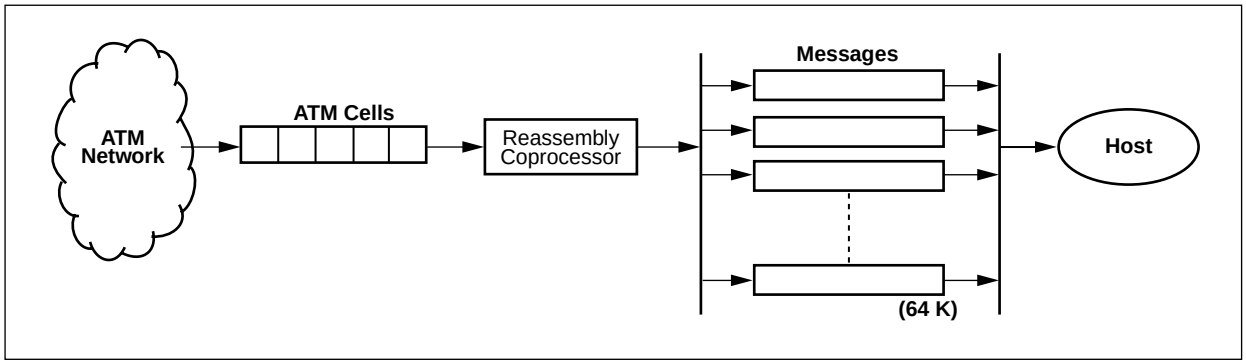
The CN8237 extracts the CPCS-SDU from the CPCS-PDU, writes the SDU to host-supplied buffers, and performs all CPCS-PDU checks. The results of these checks, as well as AAL information, are passed to the host on one of 32 independent status queues.

This chapter provides information on the functions and data structures of the reassembly coprocessor. For detailed information on how the CN8237 handles PM cells, deals with OAM functions and interacts with the segmentation coprocessor in handling traffic management and scheduling, refer to Chapter 6.0 and 7.0.

5.2 Reassembly Functional Description

Each cell received from the ATM PHY Interface block belongs to any one of a possible 64 K virtual channels, or simultaneous messages. Due to the asynchronous nature of ATM, the cell contained in any incoming cell slot can belong to any VCC. Thus, the reassembly coprocessor must assign each arriving cell to the proper VCC, thereby demultiplexing the incoming messages. Figure 5-1 illustrates the basic reassembly process flow.

Figure 5-1. Reassembly—Basic Process Flow



5.2.1 Reassembly VCCs

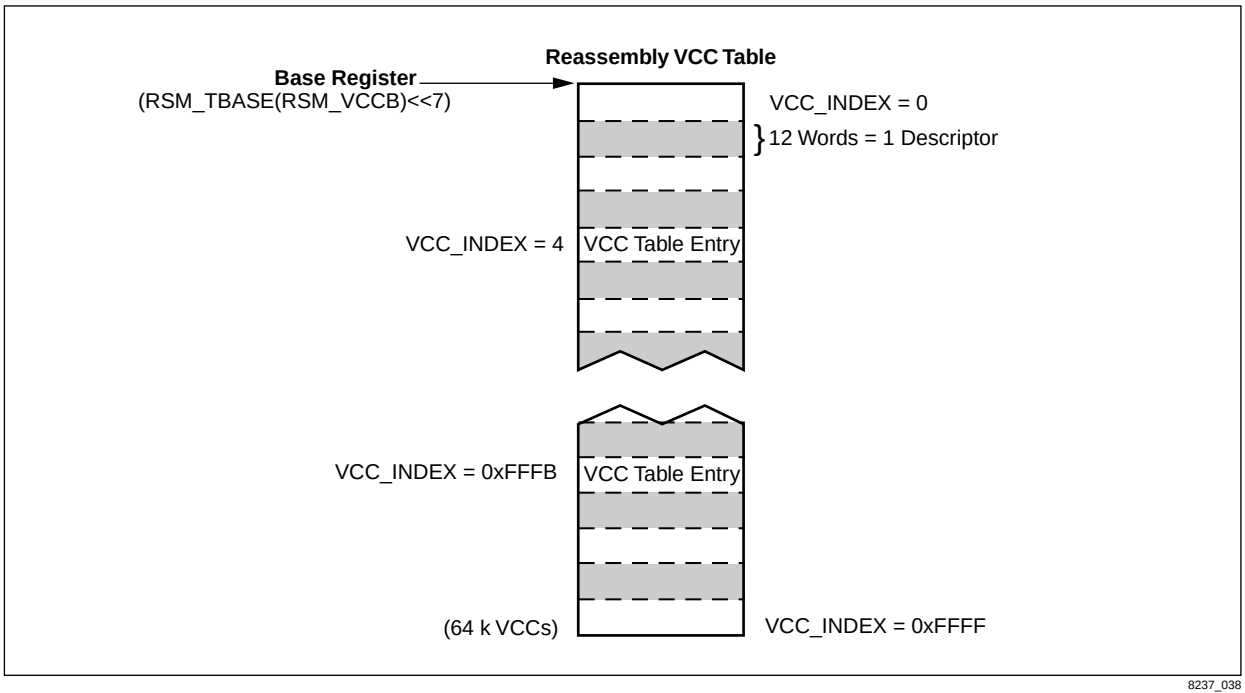
As with segmentation VCCs, the CN8237 supports up to 64 K reassembly VCCs, referenced by VCC_INDEX, which identifies a location in the reassembly VCC table.

Each entry in the reassembly VCC table consists of 12 words and describes a single VC. Each VC may be processed as either AAL5 or AAL0. AAL0 VCs can optionally be treated as Virtual FIFO buffers.

While the CN8237 will accept any reassembly VCC index within the range of 64 K VCC indexes, the actual number of reassembly VCCs allowed by the SAR is limited by the amount of SAR RSM-shared memory available in which to allocate and create RSM VCC tables, free buffer queues, etc.

Figure 5-2 illustrates how entries in the RSM VCC table are indexed by VCC_INDEX.

Figure 5-2. Reassembly VCC Table



5.2.1.1 Relation to Segmentation VCCs

The reassembly VCC index assignment is independent from the assignment of segmentation VCC indexes. A full duplex connection can have a segmentation VCC_INDEX of 0x100, while its receive channel has a reassembly VCC_INDEX of 0x800. This is especially important when a VP is represented by a single segmentation VCC_INDEX, but each of its VCs is represented by its own distinct reassembly VCC table entry.

For several operations, most notably ABR, the CN8237 provides a method to associate a reassembly VCC with a segmentation channel. The SEG_VCC_INDEX field in the reassembly VCC table allows one or many reassembly channels to correlate to one specific segmentation VCC.

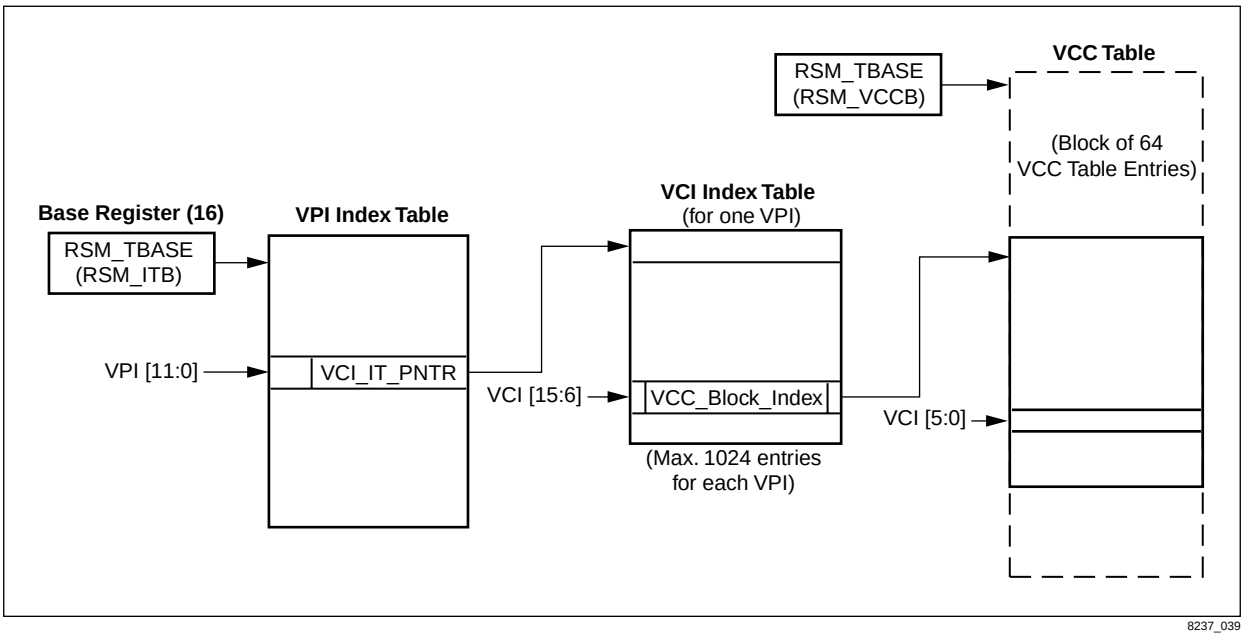
5.2.2 Channel Lookup

The CN8237's reassembly coprocessor implements a VPI/VCI Index table mechanism using direct index lookup in order to assign each cell to a virtual channel, based on its VPI/VCI value. Each channel is thus identified by its internally generated index value, the VCC_INDEX.

This VPI/VCI table Index mechanism dynamically maps VPI/VCIs to concatenated index values. It simplifies user channel assignment, provides flexible provisioning for received traffic, and provides fast, consistent lookup times regardless of the VPI/VCI values. In addition, using VPI/VCI table indexes minimizes the memory impact in preallocating large numbers of channels by requiring that only the VCI Index table entries be preallocated instead of the VCC table entries.

Figure 5-3 illustrates the direct index channel lookup mechanism.

Figure 5-3. Direct Index Method for VPI/VCI Channel Lookup



5.2.2.1 Programmable Block Size for VCC Table/ VCI Index Table

Some users might have a requirement or desire to limit the amount of memory allocated to VPI/VCI channel lookup. To enable this, the CN8237 provides the user with the choice of enabling an alternative scheme for the memory allocation and table handling involved in the Direct Index Lookup mechanism. In this scheme, the user programs the size of the memory block of RSM VCC table entries for all VCI Index table entries, to fit a range of 1 to 64 entries, instead of the default 64 entries.

Each RSM VCC entry requires 12 words of memory. By thus limiting the amount of memory space set aside for RSM VCCs, the total memory space required for VCC allocation can be substantially lessened.

To enable this scheme, set EN_PROG_BLK_SZ in the RSM_CTRL1 register to a logic high. The SAR will thus allocate block memory for VCC entries per VCI, based on the value entered in VCI_IT_BLK_SZ (RSM_CTRL1).

The data structures that facilitate this scheme are illustrated in Section 5.7.1. Figure 5-4 illustrates the alternate lookup mechanism.

Figure 5-4. Programmable Block Size Alternate Direct Index Method

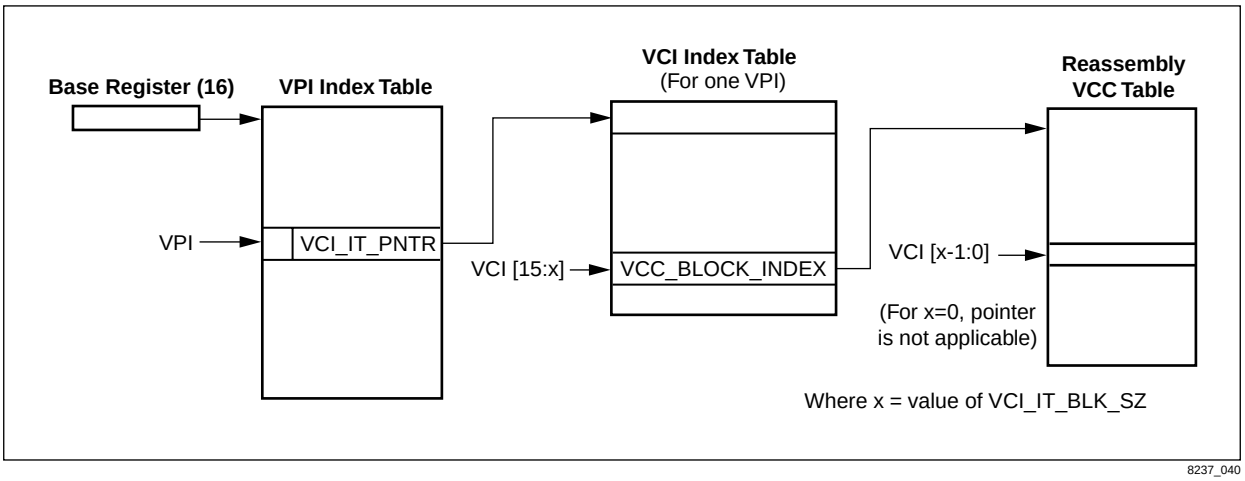


Table 5-1 shows the relationships of the values in VCI_IT_BLK_SZ, and the values for the number of VCI Index table and VCC table entries per VCI.

Table 5-1. Programmable Block Size Values for Direct Index Lookup

Value of VCI_IT_BLK_SZ	Value of x	VCI Index Table Portion of Cell's VCI	Number of Possible VCI Index Table Entries per VPI	VCC Table Portion of Cell's VCI	Number of Possible VCC Entries per VCI Index Table Entry
000	0	VCI[15:0]	65536	(No pointer)	1
001	1	VCI[15:1]	32768	VCI[0]	2
010	2	VCI[15:2]	16384	VCI[1:0]	4
011	3	VCI[15:3]	8192	VCI[2:0]	8
100	4	VCI[15:4]	4096	VCI[3:0]	16
101	5	VCI[15:5]	2048	VCI[4:0]	32
110	6	VCI[15:6]	1024	VCI[5:0]	64

5.2.2.2 Setup

At system initialization, the user configures the CN8237 to comply with either the 8-bit UNI VPI field or the 12-bit NNI VPI field, by setting the RSM_CTRL0 (VPI_MASK) bit to a logic high for UNI operation or a logic low for NNI operation. This configuration determines whether the CN8237 will treat the upper nibble of the first header octet of each received cell as the GFC field (in the UNI VPI definition), or as an extension of the VPI address. This gives an address range for VPIs of either 256 entries (for UNI) or 4096 entries (for NNI). This sets the VPI Index table size and dictates the number of VCI Index tables to be allocated.

The user can also enable the programmable block size for VCC table entries as described in the Section 5.2.2.2, by setting EN_PROG_BLK_SZ(RSM_CTRL1) to a logic high.

At system initialization, the user can also limit the valid range of both VPI and VCI addresses to be processed, in order to reduce the memory size of the lookup structures being accessed. VPIs are limited by VP_EN; and VCIs are limited by VCI_RANGE in the VPI Index table entry, and BLK_EN in the VCI Index table entry when EN_PROG_BLK_SZ is enabled.

VPI/VCI address pairs can now be preallocated in groups by mapping VCI Index table entries to blocks in the reassembly VCC table.

Once the reassembly process has been initiated, additional channels, Switched Virtual Circuits (SVCs), can be dynamically allocated with simple on-the-fly index updates.

5.2.2.3 Operation

Upon reception of a cell, the reassembly coprocessor uses the VPI field as an index into the VPI Index table, the base address of which is located at RSM_TBASE(RSM_ITB) x 0x80. The maximum allowed VPI value for UNI header operation is 255, and the maximum allowed VPI value for NNI operation is 4095, controlled by the RSM_CTRL0(VPI_MASK) field. If the VPI_MASK bit is a logic high (indicating UNI header operation), the four most significant bits of the ATM header are ANDed with 0000. The RSM coprocessor uses the VPI value to read the VPI Index table entry.

The VCI_RANGE field in the VPI Index table entry is used to set the maximum allowed value of VCI[15:x] values for that VPI, and thus sets the useable size of the VCI Index table for that VPI. If the value of VCI[15:x] of the received VCI field in the ATM header is greater than the VCI_RANGE field in the VPI Index table entry, or VP_EN is a logic low, the reassembly coprocessor discards the cell and increments the CELL_DSC_CNT counter.

The VCI_IT_PNTR indicates the base address of the VPI's VCI Index table. The CN8237 then reads the appropriate entry in the VCI Index table. The address of the VCI Index table entry is derived as follows:

$$VCI\_IT\_PNTR \times 4 + VCI[15:x] \times 4$$

The VCC_BLOCK_INDEX in the VCI Index table entry selects a contiguous block of 64 reassembly VCC State table entries (or from one to 64 VCC State table entries if EN_PROG_BLK_SZ is enabled), offset from the base address of the reassembly VCC table. The VCC_INDEX value is derived by concatenating the VCC_BLOCK_INDEX value with the VCI[x-1:0] bits from the received cell header. Thus, VCI[x-1:0] from the received header points to the reassembly VCC State table entry for that VCC.

$$VCC\_INDEX = VCC\_BLOCK\_INDEX + VCI[x\text{-}1:0]$$

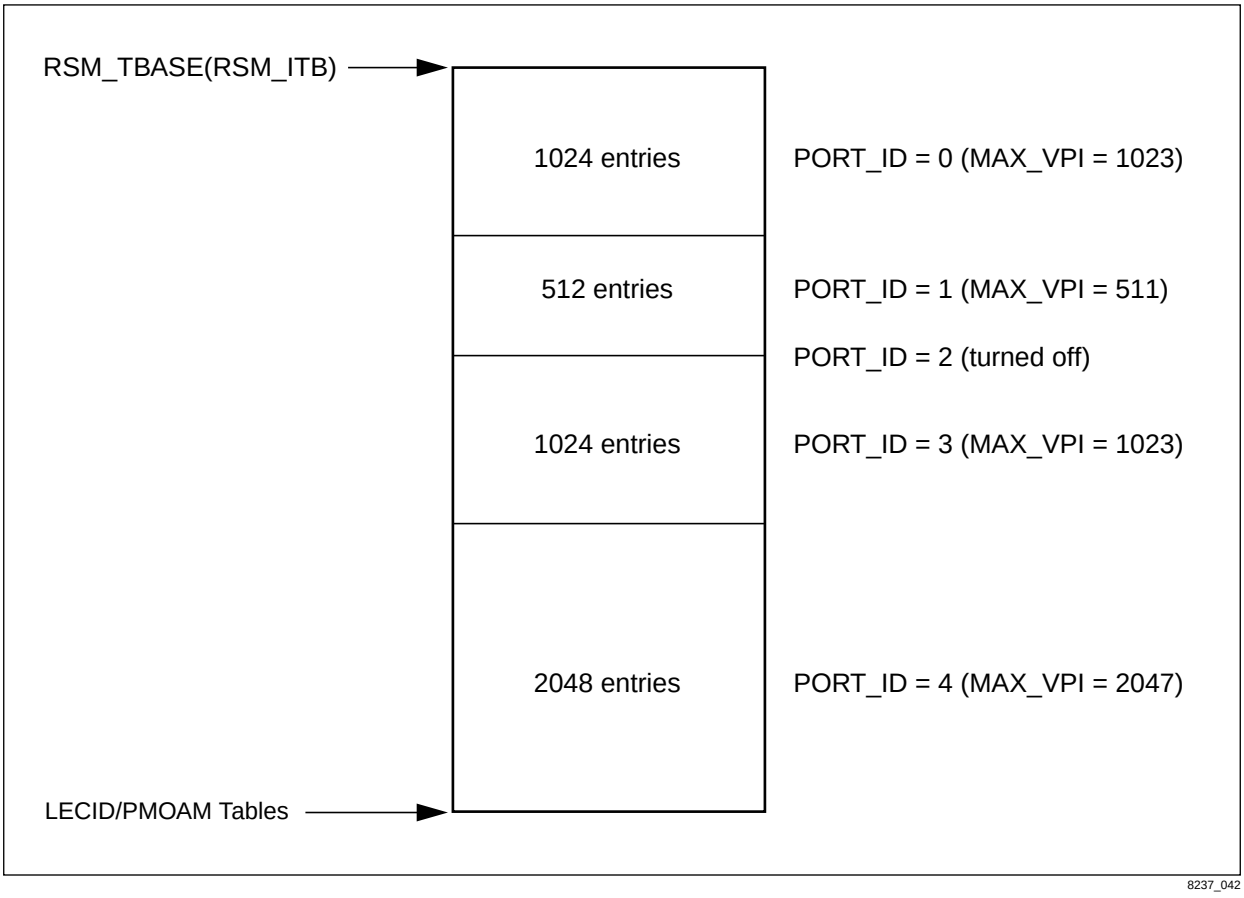
The reassembly coprocessor reads the first word of the VCC table entry. If VC_EN is a logic low, the cell is discarded and the CELL_DSC_CNT counter is incremented. Optionally, the counter is not incremented if the AAL_TYPE field has a value of 11. VC_EN allows idle cells to be filtered if the PHY layer has not already done so.

If the channel is active, the CN8237 increments the CELL_RCVD_CNT counter.

5.2.2.4 Variable VPI/PORT_ID Lookup (Multiphy Support)

In order to support muliphy operation and/or reduce memory requirements of the VPI Index table, a new Variable VPI/PORT_ID lookup mechanism has been implemented. This mechanism is enabled by setting RSM_CTRL1(EN_VPI_SIZE) = 1. When enabled, a new register called VPI_SIZE determines the maximum allowable VPI on each port in binary increments up to 4095. In addition, the VPI can be turned off. Figure 5-5 shows the VPI INDEX table with CONFIG1(NUM_PORTS) = 4 and VPI_SIZE = 0x000b_a09a. For all ports ≥CONFIG1(NUM_PORTS), their VPI_SIZE entry should be set to 0x0 in order to turn it off.

Figure 5-5. VPI Index Table with Multiple Ports



5.3 CPCS-PDU Processing

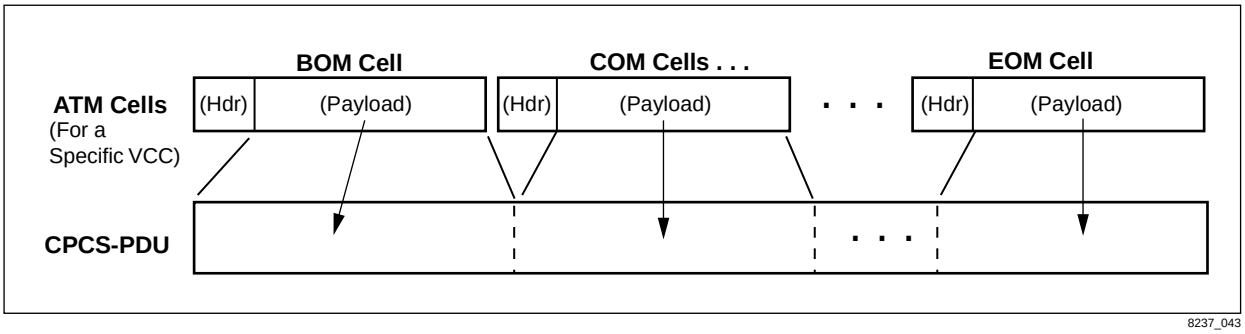
After the VCC has been identified via channel lookup, the reassembly coprocessor performs the appropriate CPCS-PDU processing according to the AAL_TYPE field in the reassembly VCC table.

The reassembly process is essentially the extraction and concatenation of consecutive ATM cell payloads on a specific VCC to form a CPCS-PDU. This processing is either reassembly into an AAL5 PDU, according to the specification

in ANSI T1.635 or reassembly into a transparent AAL0 PDU. The exact process is governed by the AAL type described in detail in the subsections below.

Figure 5-6 illustrates the basic process function.

Figure 5-6. CPCS-PDU Reassembly



SETUP

Each active reassembly VCC must have a corresponding entry in the reassembly VCC table to describe its state. At channel setup time—either during system initialization for Permanent Virtual Connections (PVCs), or dynamically for Switched Virtual Connections (SVCs)—the host allocates a reassembly VCC table entry and configures the VCC according to its provisioned or negotiated characteristics. This includes the AAL in use, its assigned buffer pools and allocation priority, as well as the associated segmentation VCC index (SEG_VCC_INDEX) for full duplex connections.

OPERATION

Once the reassembly process is activated, this RSM VCC table entry will be used to track the current state of the connection and direct the CN8237 to perform specific functions as described throughout this chapter.

5.3.1 AAL5 Processing

Except for the EOM cell, all of the data within AAL5 cell payloads is user data. The reassembly coprocessor writes all user data to memory as described in Section 5.4. The EOM cell contains both user data and CPCS-PDU overhead, and delineates the end of an AAL5 PDU.

5.3.1.1 AAL5 COM Processing

During reassembly of the PDU, the reassembly coprocessor calculates a CRC-32 value on the received AAL5 PDU, and counts the length of the PDU. The CRC-32 value is collected in an accumulator, and the LENGTH value is collected in a Length Counter. After each received cell is processed, the reassembly coprocessor writes the CRC-32 and LENGTH values to the reassembly VCC state table entry for that channel.

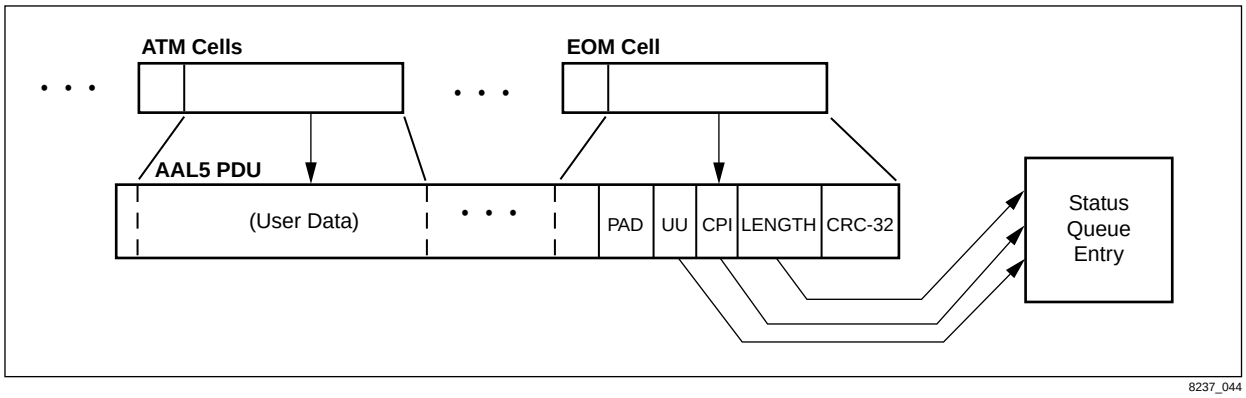
5.3.1.2 AAL5 EOM Processing

During reassembly of the AAL5 PDU, certain bytes of the PDU other than user data are written to a status queue entry for that PDU. The RSM coprocessor writes these specific fields to the status queue entry:

- UU information
- CPI field
- LENGTH field

Figure 5-7 illustrates these process functions.

Figure 5-7. AAL5 EOM Cell Processing—Fields to Status Queue

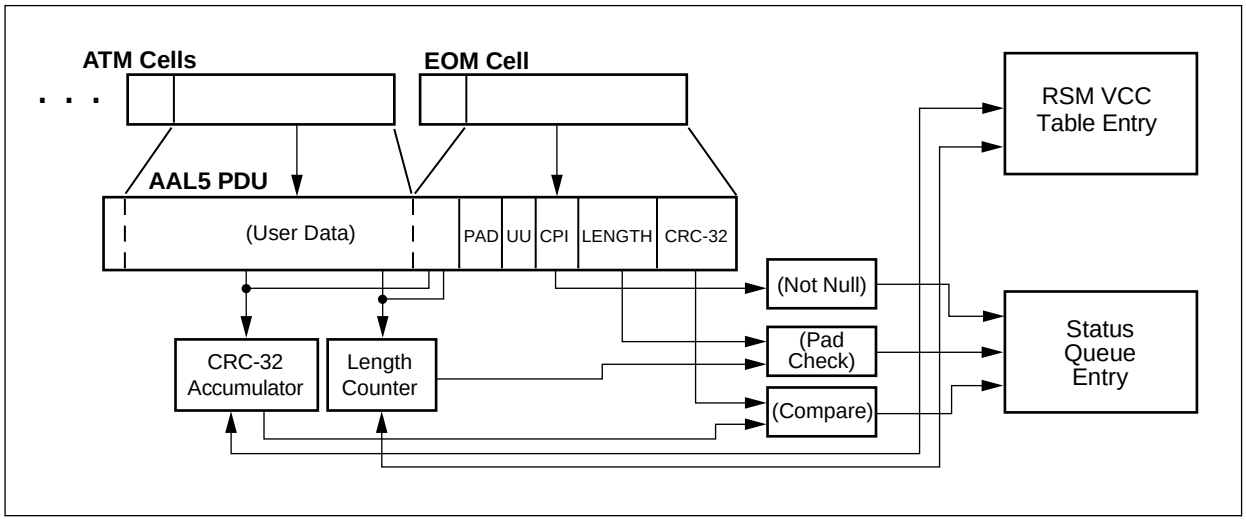


When the EOM cell is processed, the reassembly coprocessor performs the following checks:

- If the LENGTH field in the trailer of the AAL5 PDU is 0, the reassembly coprocessor sets the ABORT bit in the status queue entry to a logic high.
- Compares the calculated CRCREM value to the CRC-32 value in the trailer of the AAL5 PDU. If different, the reassembly coprocessor sets the CRC_ERROR bit in the status queue entry to a logic high.
- Compares the value collected in the Length Counter to the value in the LENGTH field in the trailer of the AAL5 PDU. If the number of PAD bytes is less than 0 or greater than the 47 the reassembly coprocessor sets, the PAD_ERROR bit in the status queue entry to a logic high.
- If the CPI field in the AAL5 trailer is not 0, the CPI_ERROR bit in the status queue entry is set to a logic high.
- All of the AAL5 trailer information is also written into the end of the PDU buffer(s) in memory.

Figure 5-8 illustrates these process functions.

Figure 5-8. AAL5 Processing—CRC and PDU Length Checks



The CN8237 reports all PDU termination events, with or without errors, in a status queue entry for that channel. See Section 5.6, for full details.

5.3.1.3 AAL5 Error Conditions

The user can set a global variable for the reassembly coprocessor, RSM_CTRL0 (MAX_LEN), dictating maximum SDU delivery length. The maximum allowable length, in bytes, of any AAL5 CPCS-PDU, including trailer and pad, is

$$\min[\text{RSM\_CTRL0(MAX\_LEN)} \times 1024, 65568]$$

During reassembly, this MAX_LEN value is checked to ensure that the PDU under reassembly does not exceed the maximum SDU delivery length.

If the CN8237 receives a non-EOM cell, where

$$\text{TOT\_PDU\_LEN} + 48 > \text{MAX\_LEN} \times 1024 \text{ (or } 65568)$$

Early Packet Discard is performed.
The CN8237 reports this condition via a status queue entry, with the LEN_ERROR and EPD status bits set. The AAL5_DSC_CNT counter is also incremented. Refer to Section 5.4.7, for details on how this process is handled.

For each EOM cell where

$$\text{TOT\_PDU\_LEN} + 48 > \text{MAX\_LEN} \times 1024 \text{ (or } 65568)$$

the PDU is completed, with BA_ERROR status bit set.

5.3.2 AAL0 Processing

AAL0 is a transparent adaptation layer, allowing for pass-through of raw data cells during CPCS-PDU processing. AAL0 channels are intended to be used for AAL proprietary adaptation layers.

5.3.2.1 Termination Methods

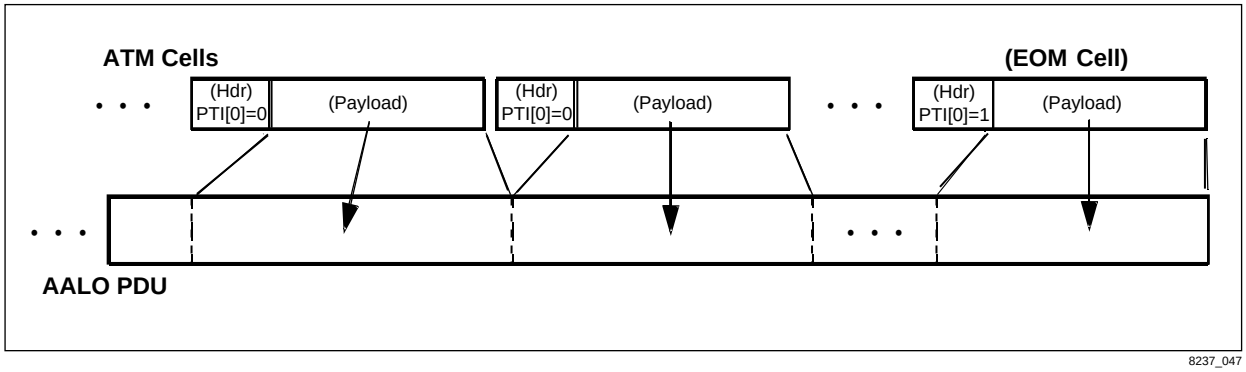
The CN8237 provides two methods of terminating an AAL0 PDU: Cell Count EOM and PTI termination.

The TCOUNT field in the RSM VCC state table entry determines the method for each VCC.

- If TCOUNT = non-0, Cell Count EOM PDU termination is enabled. PDUs will terminate when a fixed number of cells (TCOUNT) have been received. CCOUNT must be initialized to a value of one in this mode.
- If TCOUNT = 0, then PTI termination is enabled. In this case, a received cell with PTI[0] set to one indicates the end of the AAL0 message. The total maximum allowable length of an AAL0 PDU in this mode is (CCOUNT × 2) bytes.

Figure 5-9 provides an illustration of this.

Figure 5-9. AAL0 PTI PDU Termination



The CN8237 reports all PDU termination events, with or without errors, in a Status Queue entry for that channel. See Section 5.6.

5.3.2.2 AAL0 Error Conditions

If the CN8237 receives a non-EOM cell in PTI termination mode, where

$$TOT\_PDU\_LEN + 48 > CCOUNT \times 2,$$

EPD is performed. The CN8237 reports this condition via a status queue entry, with the LEN_ERROR and EPD status bits set. Refer to Section 5.4.7, for details on how this process is handled.

For each EOM cell where

$$TOT\_PDU\_LEN + 48 > CCOUNT \times 2,$$

the PDU is completed, with BA_ERROR status bit set.

The CN8237 processes error conditions for AAL0 (such as free buffer queue underflow, status queue overflow, and per-channel buffer firewall), in the same way as AAL5 CPCS-PDUs are processed.

5.3.3 ATM Header Processing

ATM level CI and CLP are mapped to the CPCS-PDU status queue entry in the following manner:

- LP: value of the ATM Header CLP bit ORed across all cells in a CPCS-PDU.
- CI_LAST: value of ATM Header PTI[1] bit in last cell of CPCS-PDU.
- CI: value of ATM Header PTI[1] bit ORed across all cells in a CPCS-PDU.

5.3.4 BOM Synchronization Signal

The STAT[1:0] output pins can be programmed to provide an indication that a BOM cell is being written across the PCI bus. Additional external circuitry could snoop the BOM cell for a service level protocol header, and perform appropriate lookup as the CPCS-PDU is being reassembled. To configure the STAT pins, set the STATMODE field in the CONFIG0 register to 0x00. The STAT output truth table illustrated in Table 5-2.

Table 5-2. STAT Output Pin Values for BOM Synchronization

	STAT[1]	STAT[0]
NOT BOM	0	0
AAL5 BOM	0	1
AAL0 BOM	1	0
NOT USED	1	1

The STAT output pins are valid during a SAR PCI master write address cycle. External circuitry would detect a BOM cell transfer by detecting a logic high on either STAT pin during a SAR PCI master write address cycle. External circuitry can then snoop the subsequent data cycles of the BOM cell transfer to extract the appropriate protocol overhead.

5.3.4.1 Prepend Index

In order to allow protocol processing to start upon reception of the BOM cell, the VCC_INDEX can be optionally prepended to the beginning of the BOM cell. This, in conjuction with the BOM SYNC signals via the STAT pins, can be used to eliminate the need for a host lookup process before starting protocol processing. When RSM_CTRL0(PREPEND_INDEX) is a logic high, the VCC_INDEX is appended to the BOM cell as follows:

Table 5-3. Prepend Index Table Format

Word	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	Reserved																VCC_INDEX															

5.4 Buffer Management

Once CPCS-PDU processing has been implemented, the cell payloads are written to data buffers. Each channel retrieves the location of its buffers from one of 32 free buffer queues. The reassembly coprocessor tracks the location of the buffers from the VCC table entry for that channel.

NOTE: The process cycle time of a read transaction across the PCI bus is much longer than a write transaction, due to the PCI bus being held in a busy state while the remote processor accesses and processes the read request. Therefore, to speed up processing flow during reassembly, the CN8237 uses only control and status writes across the PCI bus between host and local systems.

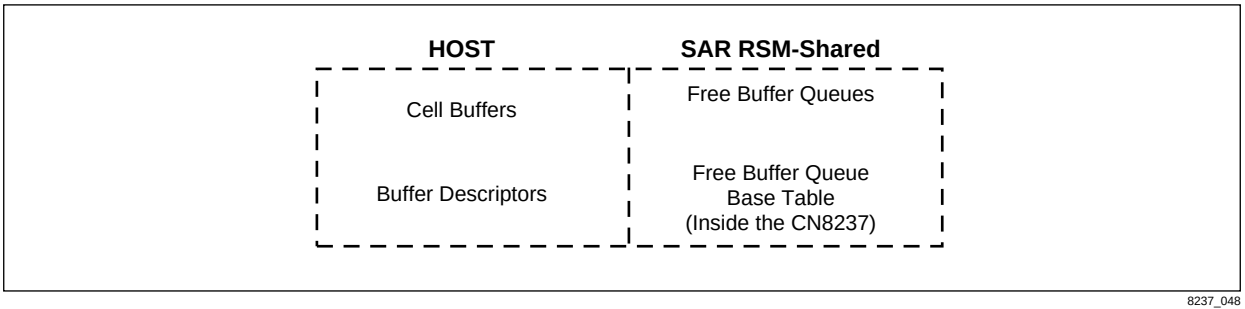
Data buffers are supplied according to the mechanisms detailed below.

5.4.1 Scatter Method

The CN8237 uses an intelligent scatter method to write cell payload data to host memory. During reassembly to host memory, the reassembly coprocessor uses the DMA coprocessor to control the scatter function. The reassembly coprocessor controls the incoming DMA block during scatter DMA to host memory.

Four data structures are maintained, as illustrated in Figure 5-10: two in the host memory, one in SAR RSM-shared memory, and one in internal memory. The linked cell buffers (HCELL_BUFF) and reassembly buffer descriptors reside in host memory, and the free buffer queues (HFR_BUFF_QU) reside in SAR RSM-shared memory. The free buffer queues also have an associated free buffer queue base table. This table is in internal memory. The CN8237 allows for up to 32 independent free buffer queues.

Figure 5-10. Host and SAR RSM-Shared Memory Data Structures for Scatter Method



5.4.2 Free Buffer Queues

The free buffer queue structure consists of a free buffer queue base table, two base address registers, RSM_FQBASE(FBQ0_BASE and FBQ1_BASE), and the corresponding free buffer queues.

The reassembly VCC table entry for any channel contains two 5-bit fields: BFR0 and BFR1. These fields identify the free buffer queues that have been assigned to this channel by the host during initialization of the VCC table entry. BFR0 contains the BOM free buffer queue number, and BFR1 contains the COM free buffer queue number. Typically, the BFR0 number is for free buffer queues 0–15, and the BFR1 number is for free buffer queues 16–31.

The free buffer queue is configured in two banks. Bank 0 contains free buffer queues 0–15, and Bank 1 contains free buffer queues 16–31.

The user can set BFR0 = BFR1 to disable this two-tier buffer structure.

Depending on the type of arriving cell (whether BOM or COM), the corresponding BFRx buffer number is used as an index to the appropriate free buffer queue base table entry.

The base addresses for these banks are in RSM_FQBASE(FBQ0_BASE and FBQ1_BASE). The reassembly coprocessor calculates the address of the first entry for any of the 32 free buffer queues as follows:

FBQx_BASE + [(size of each free buffer queue) x BFRx MOD 16]

Each of the 32 free buffer queues is a circular queue whose entries are sequentially read by the SAR. The reassembly coprocessor calculates the index for each sequential read as follows:

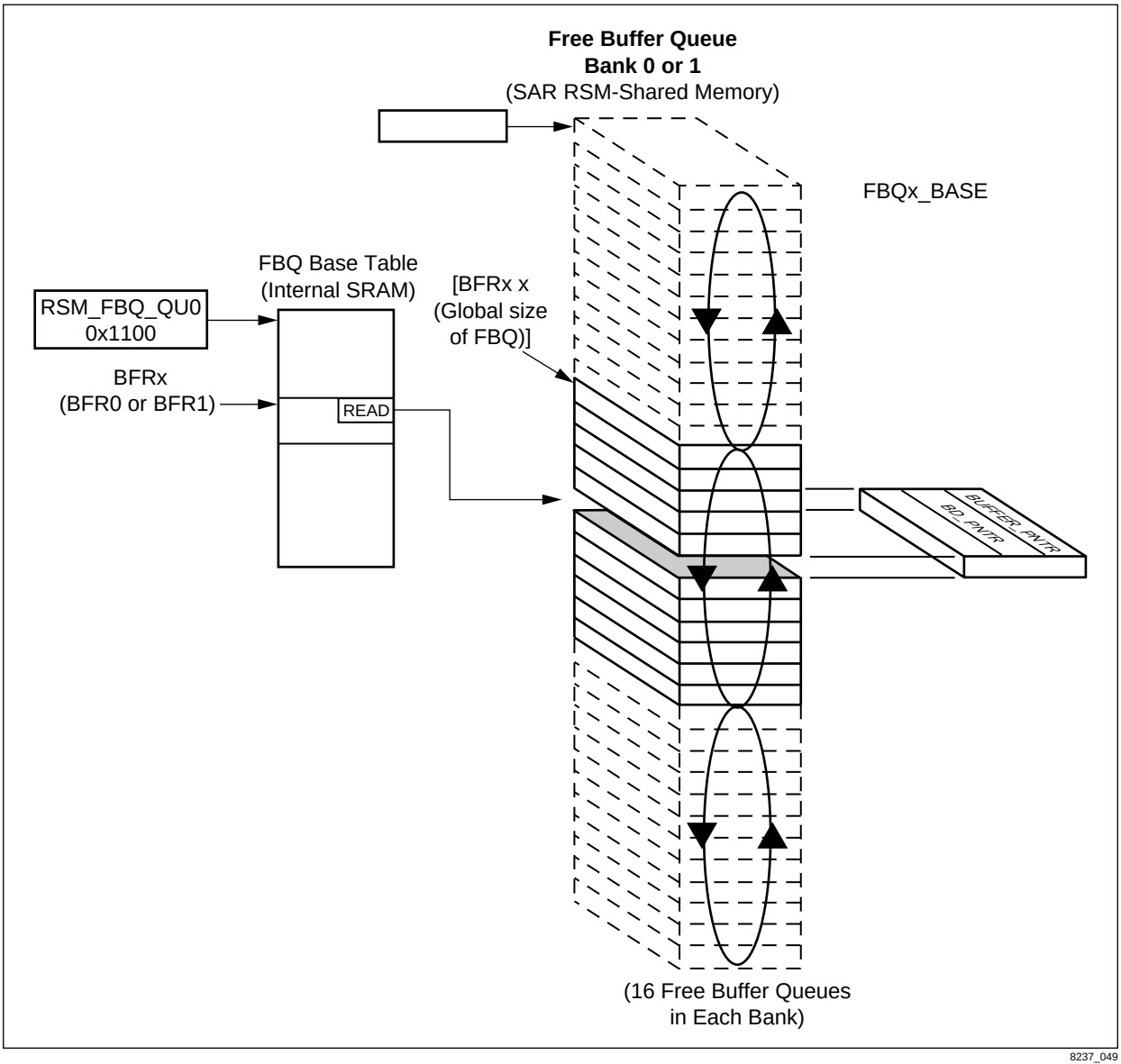
(index of first entry for the queue) + [(READ index pointer) x (size of each free buffer queue entry)]

The READ field in the base table entry for any free buffer queue is the current READ index pointer, and is continually updated with each read of that queue.

Each free buffer queue entry contains a pointer to a buffer descriptor (BD_PNTR) and a pointer to a data buffer (BUFFER_PNTR); when the free buffer queue entry is read it returns those pointers.

Figure 5-11 illustrates this structure. Refer to Chapter 3.0, for more information on the operation of the free buffer queue.

Figure 5-11. Free Buffer Queue Structure



5.4.3 Linked Data Buffers

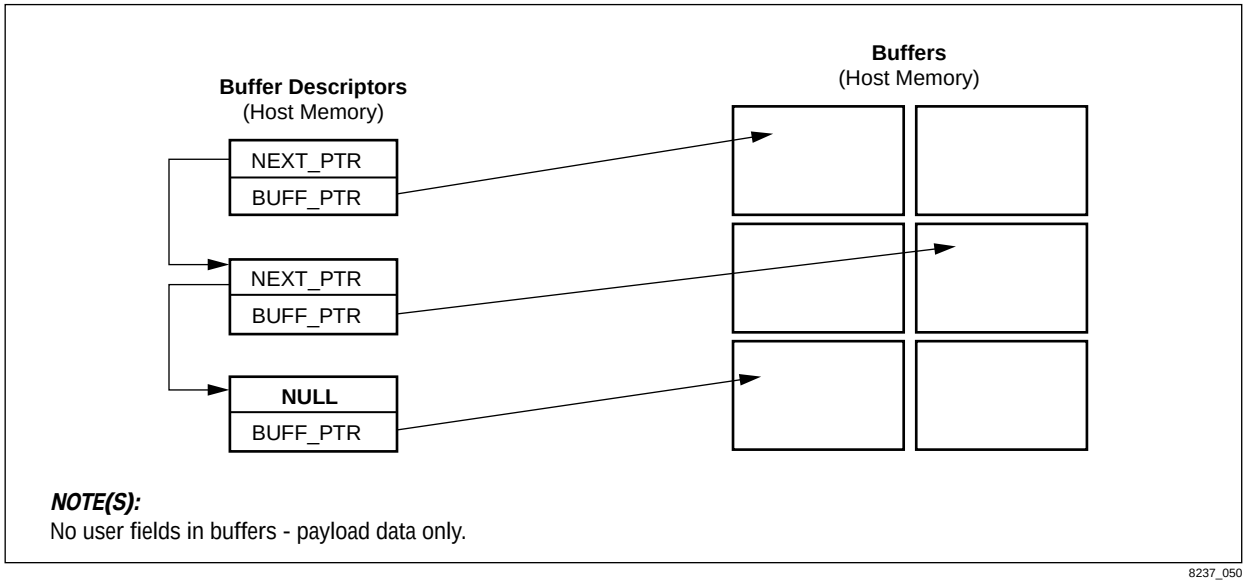
After the free buffer queue has returned pointers to a buffer descriptor and a cell buffer, the reassembly coprocessor writes payload data to the buffer.

The linked cell buffers contain the payload portions of the ATM cells. The buffers do not contain any control information. A pointer in a separate buffer descriptor structure links the buffers.

Thus, the pointer in the free buffer queue (BD_PNTR) points to a buffer descriptor, which has a pointer (BUFF_PTR) pointing to a buffer. The use of this buffer locating mechanism offers a layer of indirection in buffer assignment that maximizes system architecture flexibility.

Figure 5-12 illustrates this structure.

Figure 5-12. Data Buffer Structures



The data buffers are linked by a pointer (NEXT_PNTR) in the first word of the buffer descriptor, which is written by the reassembly coprocessor when the current buffer is completed. The host writes the second word of the buffer descriptor to point to the next associated data buffer. The link pointer of the last buffer descriptor in a chain is written to NULL.

The link pointer (NEXT_PNTR) is not written if the LNK_EN bit in the RSM VCC table entry for that channel is a logic low.

NOTE: For more information about big and little endian processing, refer to Section 8.4.

5.4.4 Initialization of Buffer Structures

Before operation of the reassembly coprocessor is enabled, the host must initialize these buffer structures. The initialization in this section assumes that the firewall function is disabled (that is, RSM_FQCTRL(FBQ0_RTN) = 0), and therefore, all free buffer queue entries are two words.

5.4.4.1 Buffer Descriptors

In every buffer descriptor entry, write the pointer to an available data buffer in the BUFF_PTR field. This assigns every data buffer to its own buffer descriptor.

5.4.4.2 Free Buffer Queue Base Table

Allocate the size (in number of entries) of each of the 32 free buffer queues in the RSM_FQCTRL register, (FBQ_SIZE) field, based on these values:

00 = 64
01 = 256
10 = 1024
11 = 4096

Initialize the free buffer queue update INTERVAL, that is, how many buffers are taken off the free buffer queue before the CN8237 writes the current READ index pointer to host memory. This is written to RSM_FQCTRL(FBQ_UD_INT).

Initialize the FORWARD, READ, UPDATE, and EMPT fields in each free buffer queue base table entry to 0s.

Initialize the READ_UD_PTR field in each base table entry with the appropriate address.

Write the appropriate length (in bytes) for the data buffers in that queue in the LENGTH field of each free buffer queue base table entry. Only MOD 8-byte buffer lengths are allowed.

5.4.4.3 Free Buffer Queue Entries

Write the base addresses of free buffer queues Banks 0 and 1 in RSM_FQBASE (FBQ0_BASE and FBQ1_BASE).

For each allocated free buffer queue entry, write the BD_PNTR and BUFFER_PTR fields corresponding to the buffer and buffer descriptor pair. Also write the VLD bit to a logic high.

For each unallocated free buffer queue entry, write the VLD bit to a logic low.

5.4.4.4 Other Initialization

The user can globally disable free buffer underflow protection by setting RSM_CTRL(RSM_FBQ_DIS) to a logic high.

5.4.5 Buffer Allocation

The reassembly coprocessor performs buffer allocation when a new channel is being reassembled, or when a buffer on an existing channel in process of being reassembled, is full.

The reassembly coprocessor reads the appropriate free buffer queue base table entry and free buffer queue entry. If the VLD bit is a logic low, a queue empty condition has occurred. (The processing of this condition is described in Section 5.4.6.) If the VLD bit is a logic high, the reassembly coprocessor uses the assigned buffer to store payload data.

The VLD bit is then written to a logic low without corrupting the BD_PNTR value. The READ index pointer and UPDATE counter are incremented. If the UPDATE counter equals RSM_FQCTL(FBQ_UD_INT), then the READ index pointer is written to the location pointed to by READ_UD_PNTR, and the UPDATE counter is reset to 0.

When the host wants to return a buffer to a free buffer queue, the host WRITE index pointer is compared to the CN8237 READ index pointer located at READ_UD_PNTR. If the WRITE index pointer plus one is equal to the READ index pointer, an overflow condition has been detected and the host does not return the free buffer yet. Otherwise, the host writes and updates the free buffer queue entry with a new buffer pointer, buffer descriptor pointer, and VLD bit set to a logic high. The host then increments its WRITE index pointer.

5.4.6 Error Conditions

An empty condition occurs when a buffer is needed and there are no available buffers in the free buffer queue. If the BFR1 queue is empty and BFR1 does not equal BFR0, the RSM coprocessor checks the BFR0 queue before declaring an empty condition.

If an empty condition occurs after the first buffer of a CPCS-PDU is written, the reassembly coprocessor will perform early packet discard on the channel and write a status queue entry with the EPD and free buffer queue Underflow (UNDF) bits set to a logic high. EPD functions are described in Section 5.4.7. Also, if a buffer queue empty condition initially occurs at the beginning of a BOM cell, a status queue entry is written with UNDF set to a logic high and BD_PNTR null. In both cases, the RSM_HF_EMPT bit is set in the HOST_ISTAT1 register.

All cells of a PDU up to and including the next EOM cell are discarded. Upon receiving a BOM or SSM cell, the reassembly coprocessor checks the queue indicated by BFR0 for a valid free buffer. If a free buffer exists, the RSM coprocessor stores the cell in the assigned buffer.

For AAL5 channels, the AAL5_DSC_CNT counter is incremented for each CPCS_PDU discarded during this error condition.

Channels that have outstanding buffers from an empty queue are not affected until they need a new buffer. Once the host has written more free buffers on the queue with VLD bit set to a logic high, the reassembly coprocessor automatically recovers from the empty condition.

5.4.7 Early Packet Discard

The packet discard feature provides a mechanism to discard complete or partial CPCS-PDUs, based upon service discard attributes or error conditions.

5.4.7.1 General Description

The EPD feature performs these basic functions:

- Halts reassembly of the CPCS-PDU marked for discard until the next BOM cell and the error condition has cleared.
- Writes a status queue entry with the EPD bit set and other appropriate STATUS and PDU_CHECKS bits set, based on the reason for the discard.

5.4.7.2 Frame Relay Packet Discard

The frame relay discard attribute is contained in the BOM cell of a CPCS-PDU. If the FRD_EN bit in the RSM VCC table is a logic high, the frame relay packet discard function for that VCC is enabled, and the functions below are performed.

When the reassembly coprocessor receives a BOM cell on a VCC with this feature enabled, it checks the 1-bit DE field in the frame relay header. If this bit is a logic high, and the channel priority (the DPRI in the RSM VCC table) is less than or equal to the global priority, RSM_CTRL (GDIS_PRI), the RSM coprocessor discards the cell, marks the rest of the packet for discard, and increments the SERV_DIS counter in the VCC table. All cells on that channel up to the next BOM are discarded.

If the SERV_DIS counter rolls over, the CNT_ROVR bit in the next status entry for this channel will be set to a logic high. The CNT_ROVR bit in the VCC table holds this flag information until a status is sent.

5.4.7.3 CLP Packet Discard

If the CLPD_EN bit in the RSM VCC table is a logic high, the Cell Loss Priority packet discard function for that VCC is enabled, and the following functions below are performed:

- When the RSM coprocessor receives a cell and this function is enabled, it checks the 1-bit CLP field in the ATM header. If this bit is a logic high, and the channel priority is less than or equal to the global priority, RSM_CTRL (GDIS_PRI), the RSM coprocessor discards the cell, marks the rest of the packet for discard, and increments the SERV_DIS counter in the VCC table. All cells on that channel up to the next BOM are discarded.
- If the SERV_DIS counter rolls over, the CNT_ROVR bit in the next status entry for this channel will be set to a logic high. The CNT_ROVR bit in the VCC table holds this flag information until a status is sent.

5.4.7.4 LANE-LECID Packet Discard—Echo Suppression on Multicast Data Frames

The system designer can use this feature to discard superfluous traffic on the ATM network caused by LAN Emulation Clients (LECs) transmitting multicast frames, that is, point-to-multipoint Emulated LAN traffic.

If the LECID_EN bit in the RSM VCC table is a logic high, the LANE-LECID discard function for that VCC is enabled, and the functions below are performed.

The DPRI field is used as an index into the LECID table. This allows support for up to 32 LECIDs, each a unique identifier for a single LAN Emulation Client.

When the RSM coprocessor receives a BOM cell with this function enabled, it checks the 16-bit LECID field in the LANE header against the value in the LECID table. If a match occurs, the RSM coprocessor discards the cell, marks the rest of the packet for discard, and increments the SERV_DIS counter in the VCC table.

If the SERV_DIS counter rolls over, the CNT_ROVR bit in the next status entry for this channel will be set to a logic high. The CNT_ROVR bit in the VCC table holds this flag information until a status is sent.

5.4.7.5 DMA FIFO Buffer Full

The purpose of this function is to allow a graceful recovery from an incoming DMA FIFO buffer full condition. Without this function, the reassembly coprocessor is stalled when the FIFO buffer is full until recovery from the full condition. This causes the cells to be dropped indiscriminately on the upstream side of the reassembly block without any record of which VCCs the cells belonged to. Upon recovery from the full condition, cells belonging to corrupted PDUs continue to be processed, which wastes PCI bandwidth during the recovery phase. This function provides for a more efficient use of host and SAR resources by allowing the reassembly block to process and drop cells during the full condition.

The reassembly block marks all channels that receive a cell during the full condition for subsequent early packet discard. Upon recovery from the full condition, the reassembly block performs early packet discard on the appropriate channels as cells are received on those channels. In addition, cells continue to be dropped on each channel until after an EOM cell is received for that channel. Early packet discard processing is delayed until recovery from the full condition, since the status entry also requires the use of the incoming DMA FIFO buffer.

This function is enabled by setting the FF_DSC bit in each VCC entry to a logic high.

Similarly, if RSM_CTRL1(OAM_QU_EN) is a logic high, RSM_CTRL1 (OAM_FF_DSC) should be set to a logic high.

Early packet discard due to a FIFO buffer full condition is indicated by the FFPD bit in the RSM status queue entry being a logic high.

NOTE: Avoid having the Status Queue Overflow (or Full Condition) and DMA FIFO Buffer Full conditions at the same time.

5.4.7.6 Error Conditions

Partially reassembled CPCS-PDUs will be recovered for the following error conditions:

- Non-EOM Max PDU Length exceeded
- Free buffer queue underflow
- Status queue overflow

5.4.8 Hardware PDU Time-Out

The CN8237 automatically detects active CPCS-PDU time-out for reassembly channels. A PDU time-out occurs when a partially received PDU does not complete within a set time period. When it detects this time-out condition, the CN8237 provides a status queue indication to the host. This indication allows the host to recover the buffers held by the partially completed PDU. The CN8237 supports up to eight reassembly time-out periods.

5.4.8.1 Reassembly Time-Out Process

A background hardware process performs the reassembly time-out function. The process is activated at a user-selected interval. The process is globally enabled by setting the GTO_EN bit in the RSM_CTRL0 register. The RSM_TO register controls the process activity once enabled. The process is activated every RSM_TO_PER rising edges of SYSCLK on cell boundaries.

NOTE: GTO_EN set to 0 resets the internal time-out interrupt counter.

Each time the process is activated, it examines a single VCC, identified by TO_VCC_INDEX. This is a 16-bit variable located at address 0x1350, in internal SRAM. The host should initialize this register to 0 at system initialization.

To enable hardware time-out on an individual VCC, the host must set TO_EN in the VCC table entry. The host also assigns one of eight time-out periods to each VCC by initializing the TO_INDEX field in the VCC table entry.

The CN8237 checks the TO_EN bit and the active PDU indicator bit, ACT_PDU, to see if time-out processing is enabled and necessary, respectively, for the current connection. If either bit is 0, TO_VCC_INDEX is incremented by one and compared to RSM_TO_CNT in the RSM_TO register. If TO_VCC_INDEX = RSM_TO_CNT, then TO_VCC_INDEX is reset to 0 and the time-out search is restarted at the beginning of the VCC table.

If both bits are set, the CN8237 increments CUR_TOCNT in the RSM VCC table entry. It then compares CUR_TOCNT to the time-out value selected, TERM_TOCNTx, where x = TO_INDEX. TERM_TOCNT0 through TERM_TOCNT7 are located at address 0x1340 through 0x134c in internal SRAM. They must be initialized to appropriate values during system initialization.

If CUR_TOCNT = TERM_TOCNTx, a time-out condition has occurred on the current VCC. The CN8237 follows the procedure described in Section 5.4.8.4.

5.4.8.2 Halting Time-Out Processing

To halt time-out processing, the host "must" set the TO_LAST bit to one in the RSM VCC table entry for the last VCC_INDEX that the host wishes to have enabled for time-out processing. When the CN8237 detects this bit set to one, it halts time-out processing.

When time-out processing is halted, the time-out process is still activated, but the VCC will not be checked for a time-out condition. The CN8237 simply increments TO_VCC_INDEX and compares it to RSM_TO_CNT. If they are equal, TO_VCC_INDEX is reset to 0 and the full time-out processing is re-enabled.

5.4.8.3 Timer Reset

The CN8237 reassembly time-out process increments the CUR_TOCNT value. If it reaches a threshold value, a time-out condition has occurred. In AAL5 and AAL0, PTI termination modes, the reception of a non-EOM cell resets the counter.

5.4.8.4 Reassembly Time-Out Condition

The CN8237 reports reassembly time-out conditions via the VCC's reassembly status queue. The TO bit in the STATUS field of the status queue entry is set to one. In Message Mode, the BD_PNTR points to the beginning of the partial buffer descriptor chain. In Streaming Mode, the BD_PNTR points to the last buffer descriptor in the chain. The only other valid fields in the status queue entry are VCC_INDEX and VLD.

Once status has been reported, the CN8237 re-initializes the VCC table entry to begin accepting a CPCS-PDU.

5.4.8.5 Time-Out Period Calculation

The following equation determines the time-out period of a VCC:

Period = SYSCLK period x RSM_TO_PER x RSM_TO_CNT x TERM_TOCNTx

RSM_TO_CNT must be greater than or equal to the maximum number of VCCs that require time-out processing.

5.4.9 Virtual FIFO Buffer Mode

This mode provides a logical FIFO buffer port for cell data to host memory. Its principal use is for AAL0 CBR voice traffic.

5.4.9.1 Setup

To enable this mode on any channel, set FIFO_EN in the RSM VCC table to a logic high. The user initializes the CBUFF_PNTR field in the RSM VCC table to the address of the FIFO buffer port. The channel should also be configured for AAL0 fixed length termination mode, with a termination length of one cell.

5.4.9.2 Operation

Whenever a buffer is required during reassembly in this mode, the CBUFF_PNTR address is used without accessing the free buffer queue.

No status entries are written in this mode because there is no way to maintain synchronization between status entries and cells in the FIFO buffer under FIFO buffer overflow conditions.

5.4.9.3 Errors

When the FIFO buffer port is on the PCI bus, the CBUFF_PNTR address must be on a 64 byte boundary, and a decode of any address in the 64 byte block accesses the FIFO buffer. External circuitry must also ensure that only complete cells are written into the host FIFO buffer.

The beginning of a cell transfer can be detected by the PCI address being 64-byte aligned.

5.4.10 Firewall Functions

Implementation of multiple free buffer queues and EPD performs a firewalling functionality on a group basis.

The user can also set up per-VCC firewalling on a channel-by-channel basis. The firewall mechanism allows the user to allocate buffer credits on a per-channel basis.

NOTE: When firewalling is enabled in the RSM coprocessor and an FBQ empty (underflow) condition is encountered, the RX_COUNTER field in the VCC table(s) still decrements each time the VCC receives a BOM cell. The RX_COUNTER should not be decremented when the FBQ is empty. There is no workaround for this problem. The user "must" avoid FBQ empty conditions when firewalling is enabled.

5.4.10.1 Setup

Set RSM_FQCTRL(FBQ0_RTN) to a logic high. This sets free buffer queue block 0 to contain queues with 4-word entries. This is used to support per-VCC firewall credit update.

Set the global firewall control bit to a logic high in register RSM_CTRL0, field (FWALL_EN), to globally enable firewall processing on a per-channel basis.

Set the following fields of the VCC table entry for the channel being set up for firewall processing:

- The FW_EN bit set to a logic high enables firewall processing on that channel.
- Set RX_COUNTER[15:0] to assign the initial buffer credit for the channel.

Initialize the FORWARD fields in the free buffer queue base tables to point to the entry where credit is initially returned. Typically, this is the first entry after the initial buffers placed on the queue. Write the FWD_VLD bit in all free buffer queue entries to a logic low.

5.4.10.2 Operation

Whenever a buffer is taken off free buffer queues 0–15 during reassembly on a channel enabled for firewall processing, the RSM coprocessor decrements the RX_COUNTER[15:0] in the RSM VCC table entry for that channel. This allows COM buffers to be placed on queues 16–31 without being firewalled.

If the RX_COUNTER[15:0] for a channel is 0 when a buffer is required, the RSM coprocessor declares a firewall condition. If the firewall condition occurs on a BOM or SSM, the CN8237 writes a status queue entry with the FW bit set and a NULL in the BD_PNTR field.

If the firewall condition occurs on a COM or EOM, the RSM coprocessor initiates EPD and writes a status queue entry with the FW and EPD bits set. It then discards cells on that channel until the channel has recovered from the firewall condition.

All AAL5 PDUs discarded under the firewall condition cause the AAL5_DSC_CNT counter to be incremented. Recovery occurs only on a BOM or SSM cell when the credit is rechecked.

5.4.10.3 Credit Return

The user returns credit, at the same time the buffer is recovered to the free buffer queue, by writing the third word of the free buffer queue. The VCC_INDEX is written to the channel to which credit is returned. The FWD_VLD bit is set to a logic high, and the QFC bit is set to a logic low. The RSM coprocessor increments the RX_COUNTER[15:0] of the applicable channel. For proper operation of the update interval function, buffers must be returned at the same time as credits are returned.

Credits are returned to VCCs through Bank 0 free buffer queues. In order to return buffer credits independently from buffer usage, the CN8237 maintains a separate read pointer into free buffer queues that return credits. This pointer name is FORWARD, located in the free buffer queue base table entry. The host determines the number of Bank 0 free buffer queues that return credits by setting FWD_EN in the RSM_FQCTRL register.

The CN8237 snoops writes to free buffer queues that return firewall credits. When a write completes, the CN8237 will begin processing firewall return credits on that queue. The third word of each entry is read, and if FWD_VLD is set, a credit is added to the VCC_INDEX indicated. The CN8237 continues to process credit return entries until FWD_VLD is 0. Multiple free buffer queues might have credit return entries outstanding at one time. The CN8237 processes the entries according to the priority set in FWD_RND in the RSM_FQCTRL register. If FWD_RND is a logic low, the CN8237 exhausts the credit returns on the highest number active queue before proceeding to other queues. Otherwise, it services the queues in round-robin order.

Before the reassembly coprocessor is enabled, the host must initialize the FORWARD read pointer to the first entry where credit is returned. Typically, this is the first entry after the initial buffers placed on the queue.

5.5 Global Statistics

To meet the requirements of ILMI (ATM Forum) and AToM (RFC1695) documents, three register-based counters are implemented:

- CELL_RCVD_CNT—Number of cells received that map to active channels.
- CELL_DSC_CNT—Number of cells received that map to inactive channels. This includes idle cells, since those channels will be turned off.
- AAL5_DSC_CNT—Number of AAL5 CPCS-PDUs discarded due to per channel firewall, buffer queue underflow, FIFO buffer full packet discard, status queue overflow, or maximum CPCS-PDU length exceeded on non-EOM cells.

The first two counters are implemented as 32-bit counters, and the third is a 16-bit counter. All three are set to 0 upon a reset and are not reset to 0 upon a read of the counter by the host. The counters roll over and optionally cause an interrupt upon rollover.

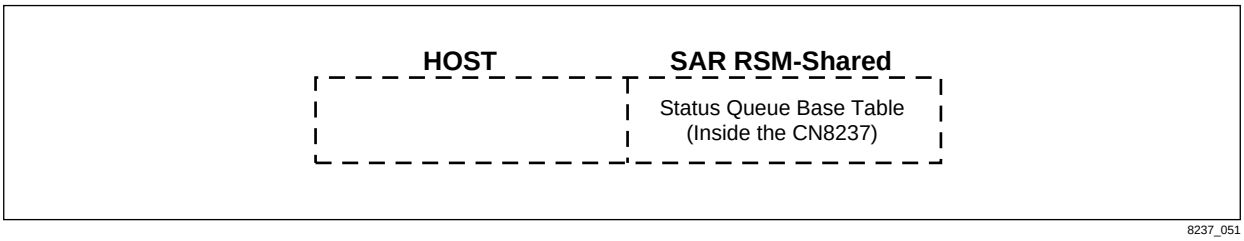
5.6 Status Queue Operation

The CN8237 reports reassembly status to the host using the reassembly status queue. The reassembly coprocessor normally writes a status queue entry when a complete CPCS-PDU has been reassembled. One field of the status queue entry (BD_PNTR) points to the first buffer descriptor in the linked list of buffer descriptors for that reassembled PDU, and the rest of the fields of that status queue entry provides data on the status of the reassembled PDU. These fields are then used by the host in directing further processing.

5.6.1 Structure

Two data structures are maintained as illustrated in Figure 5-13: one in host memory and one in SAR RSM-shared memory. The status queues (HSTAT_QU) reside in host memory, and the status queue base table resides in SAR RSM-shared memory, and allows for up to 32 independent circular status queues.

Figure 5-13. Data Structure Locations for Status Queues

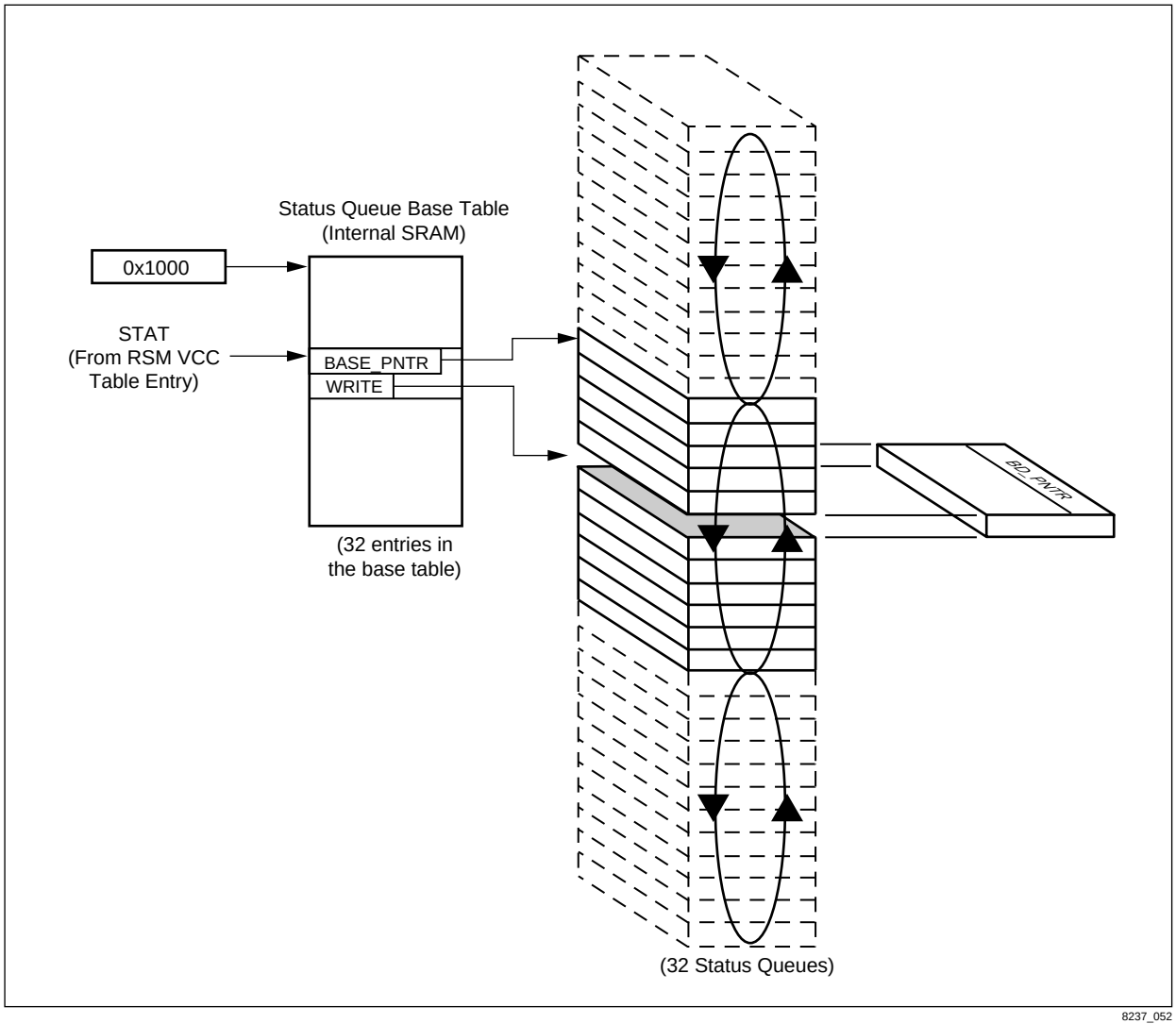


The status queue base table is located at 0x001000 in internal SRAM. The status queue base table contains status queue base information for up to 32 queues. The queues are accessed via a status pool number, the STAT field in the RSM VCC table. This field is used as an index to the correct status queue base table entry.

The BASE_PNTR field in the status queue base table entry points to the base address of the status queue associated with that status queue base table entry. The WRITE field is the index pointer maintained by the CN8237, incremented each time a status queue entry is written, to point to the next status queue entry for that status queue.

Figure 5-14 illustrates this structure.

Figure 5-14. Status Queue Structure Format



5.6.1.1 Setup

At system initialization, set up the following fields in each of the status queue base table entries:

- Write the base address of the corresponding status queue in the BASE_PNTR field.
- Initialize the WRITE and READ_UD fields to 0s.
- Set the SIZE field for the size of the corresponding status queue.

In addition, initialize each status queue entry in all 32 status queues by setting the VLD bit to a logic low.

Initialize the READ pointers to 0 for each status queue.

5.6.1.2 Operation

The reassembly coprocessor normally writes a status queue entry when a complete CPCS-PDU has been reassembled. It also writes a status queue entry for each received OAM cell.

Each time the RSM coprocessor writes a status queue entry, it sets the VLD bit in the entry to a logic high and increments the WRITE pointer in the status queue base table entry for that status queue.

When the host processes the status queue, it reads entries based on the host READ pointer for that status queue. It reads only the VLD bit at first before reading any other word to maintain data coherency.

When the host finds the VLD bit set to a logic high, it processes the status queue entry, increments the host READ counter, and resets the VLD bit to a logic low. The host also periodically writes the READ counter value to the READ_UD field in the status queue base table entry for that queue.

When in Message Mode, the RSM coprocessor writes a status entry at the completion of a CPCS-PDU. Optionally, a status entry can be written at both the beginning and end of a message, to allow the host to initiate protocol header processing in advance of receiving the complete message. The host can then traverse the linked cell buffers to collect the complete CPCS-PDU.

If the BINTR bit in the RSM VCC table entry is a logic high, the RSM coprocessor writes an additional status queue entry at the completion of the first buffer of a CPCS-PDU. This status queue entry is delineated by the BOM bit set and the EOM bit cleared. This allows the host to begin packet processing before reception of the complete CPCS-PDU. In this case only the BD_PNTR and VCC_INDEX fields are valid in that status queue entry.

The STM_MODE bit in the RSM VCC table, being set to a logic high, activates Streaming Mode for that channel. In this mode, the RSM coprocessor writes a status entry for each completed buffer. The BD_PNTR field in the status entry points to the corresponding buffer descriptor for that single buffer. Only the last status entry for that CPCS-PDU, with EOM bit a logic high, contains valid status data for that PDU.

Refer to Chapter 2.0, for more detailed information on the operation of status queues.

5.6.1.3 Errors

The RSM coprocessor also writes a status entry for several error conditions:

- Reassembly time-out
- Early packet discard
- Per-channel firewall
- CPCS abort

To ensure that an error indication occurs even if no CPCS-PDUs are being reassembled on channels having free buffer queues in the empty state, a BOM cell causes a status queue entry to be written. If a BOM cell is received and no early packet discards have occurred on channels mapped to the empty free buffer queue, status queue entry is written with the BOM and UNDF bits set to a logic high. In addition, the RSM_HF_EMPT bit in the HOST_ISTAT1 register is set to a logic high. This status does not point to a linked list of buffer descriptors. It is written a maximum of once per free buffer queue empty condition.

5.6.1.4 Host Detection of Status Queue Entries

The host can use either a polling operation or an interrupt routine to detect new status queue entries.

To poll each status queue, the host continuously reads the VLD bit at the current READ position until it returns a logic high. The host then processes the status entry, writes the VLD bit to a logic low and increments its current READ pointer. Periodically, the host writes the current READ index value into the READ_UD field of the status queue base table entry.

The host can also use an interrupt routine to process status queues. When the reassembly coprocessor writes a status queue entry into host memory, the HOST_ISTAT0 (RSM_HS_WRITE) bit is set to a logic high to prompt an interrupt. Upon receiving an interrupt, the host reads HOST_ST_WR (RSM_HS_WRITE[15:0]) to determine which host memory status queue(s) caused the interrupt. See Section 3.3.2.4 for alternative methods.

NOTE: Only status queues 0 through 15 are reported in this register.

A typical operation for the interrupt manager would be to only read HOST_ISTAT1 upon receiving an interrupt, and periodically read HOST_ISTAT0 to ensure that no error conditions have occurred. Once the interrupt manager has determined which status queue(s) caused the interrupt, the host starts reading the appropriate status queues at their current read location. The host processes status entries until reading an entry with the VLD bit set to logic low. Again, the host periodically writes the current READ index value into the READ_UD field of the status queue base table entry.

5.6.2 Status Queue Overflow or Full Condition

A status queue overflow or full condition is entered when the last available status queue entry is written. The reassembly coprocessor detects the condition by comparing the WRITE and READ_UD index pointers in the corresponding status queue base table. Upon detecting a status overflow condition, the RSM coprocessor sets the internal OVFL bit in the last status queue entry written to a logic high to indicate the condition. To prompt an interrupt, the RSM coprocessor also sets to 1 the RSM_HS_FULL bit in the HOST_ISTAT1 register.

While the reassembly coprocessor is in status full condition, it discards all cells. If a COM or EOM cell is received while the status queue is full, the channel is marked for status full packet discard. Once an SSM, EOM, or OAM cell is received during a status full condition, the cell is discarded and the status queue checked. If there is now room in the status queue, the status full condition is exited.

For multiple peer configurations, an interrupt manager can be configured by the user to detect the full condition and advise the peers to check if their queues have overflowed. Each peer would then check the OVFL bit in the last status queue entry written (pointed to by READ_UD – 1) to determine if that peer's status queue has filled. If the OVFL bit is not set to a logic high, the host should also check the entry pointed to by (READ_UD – 2) to determine if an overflow condition occurred during a host update of the READ_UD index pointer. Because the reassembly coprocessor recovers from the overflow condition automatically, the host does not have to determine which queue overflowed.

After a status group has exited a full condition, the RSM coprocessor performs EPD on channels marked for packet discard due to the status overflow condition, when a cell is received on any of those channels. Cells up to and including the next EOM are discarded. Status queue overflow protection can be globally disabled by setting RSM_CTRL0(RSM_STAT_DIS) to a logic high.

NOTE: Avoid having the Status Queue Overflow (or Full Condition) and DMA FIFO Buffer Full conditions at the same time.

5.7 Reassembly Control and Data Structures

5.7.1 Channel Lookup Structures

The reassembly coprocessor utilizes a VPI/VCI table index mechanism employing direct index lookup in order to assign each arriving cell to a virtual channel, based on its VPI/VCI value. Figure 5-15 illustrates this lookup mechanism.

Figure 5-15. VPI/VCI Channel Lookup Structure

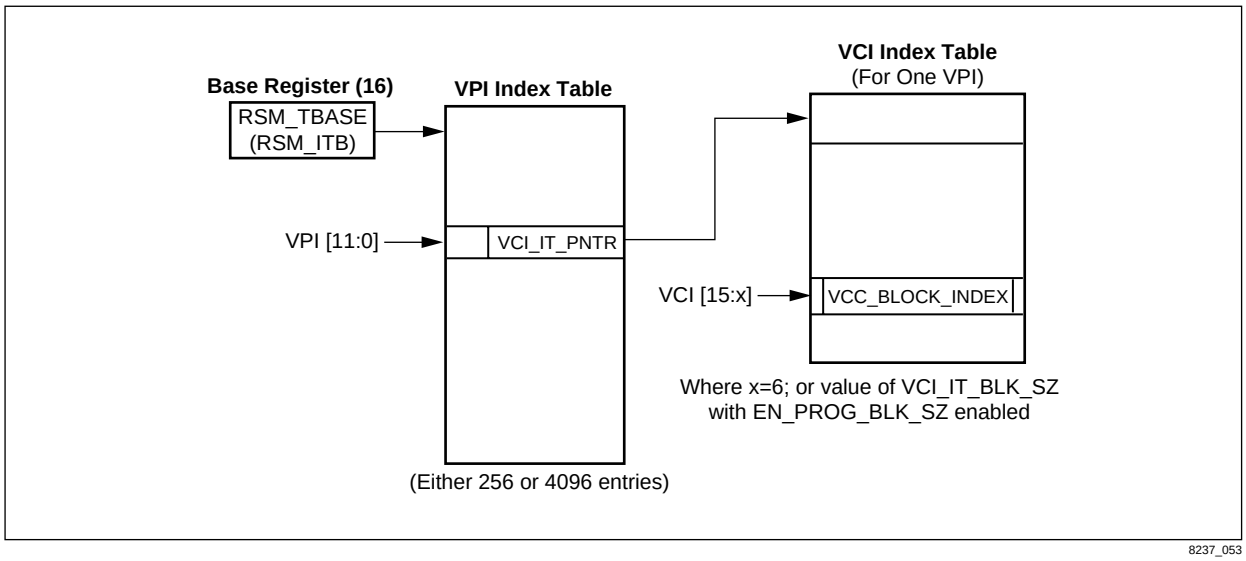


Table 5-4 describes the normal VPI Index table format (one word per entry) without EN_PROG_BLK_SZ (RSM_CTRL1) enabled.

Table 5-5 describes the VPI Index table format (two words per entry) with programmable VCC table and VCI Index table block size enabled.

Table 5-6 describes the field definitions for the VPI Index table fields.

Table 5-4. Normal VPI Index Table Entry Format

Word	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	VP_EN	VCI_RANGE										VCI_IT_PNTR																				

Table 5-5. VPI Index Table Entry Format with EN_PROG_BLK_SZ(RSM_CTRL1) Enabled

Word	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	VP_EN	Reserved										VCI_IT_PNTR																				
1	Reserved																								VCI_RANGE							

Table 5-6. VPI Index Table Entry Descriptions *(1 of 2)*

Field Name	Description/Function
VP_EN	Enables the VPI for lookup processing. If not enabled, cell is discarded, and the CELL_DSC_CNT counter is incremented.
VCI_RANGE	Determines the maximum VCI value allowed. If cell VCI exceeds maximum, cell is discarded, and counted as CELL_DSC_CNT. In normal operation, the 10 most significant bits can be set by the user, with the six least significant bit set at 1. When EN_PROG_BLK_SZ(RSM_CTRL1) is enabled, all 16 bits of the field can be set by the user.

Table 5-6. VPI Index Table Entry Descriptions *(2 of 2)*

Field Name	Description/Function
VCI_IT_PNTR	VCI Index Table Base Pointer. Points to the base of the VCI Index table for the VPI by appending two least significant 0 bits to form a byte address.

Table 5-7 describes the VCI Index table format without the programmable VCC table and VCI Index table block size enabled.

Table 5-8 describes the VCI Index table format with EN_PROG_BLK_SZ (RSM_CTRL1) enabled.

Table 5-9 describes the VCI Index table Descriptions.

Table 5-7. Normal VCI Index Table Format

Word	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	Reserved																VCC_BLOCK_INDEX										Reserved						

Table 5-8. VCI Index Table Format with EN_PROG_BLK_SZ (RSM_CTRL1) Enabled

Word	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	BLK_EN	Reserved															VCC_BLOCK_INDEX																

Table 5-9. VCI Index Table Descriptions

Field Name	Description/Function
BLK_EN	Block Enable. If logic high, enables entry. If a logic low, the cell is discarded.
VCC_BLOCK_INDEX	VCC block index. When EN_PROG_BLK_SZ is not enabled, this is the index to the block of 64 RSM VCC table entries allocated to VCI[15:6]. In this case, VCC_BLOCK_INDEX is concatenated with the six least significant bits of the VCI to form the index into the RSM VCC table to access the VCC table entry for that channel. When EN_PROG_BLK_SZ is enabled, the [16 – (value of VCI_IT_BLK_SZ)] most significant bits of VCC_BLK_INDEX are concatenated with the [(value of VCI_IT_BLK_SZ) – 1] least significant bits of the VCI to form the index into the VCC table to access the VCC table entry. For example, if the value of VCI_IT_BLK_SZ = 4, the resultant RSM VCC_INDEX pointer = {VCC_BLK_INDEX[15:4], CELL_VCI[3:0]}.

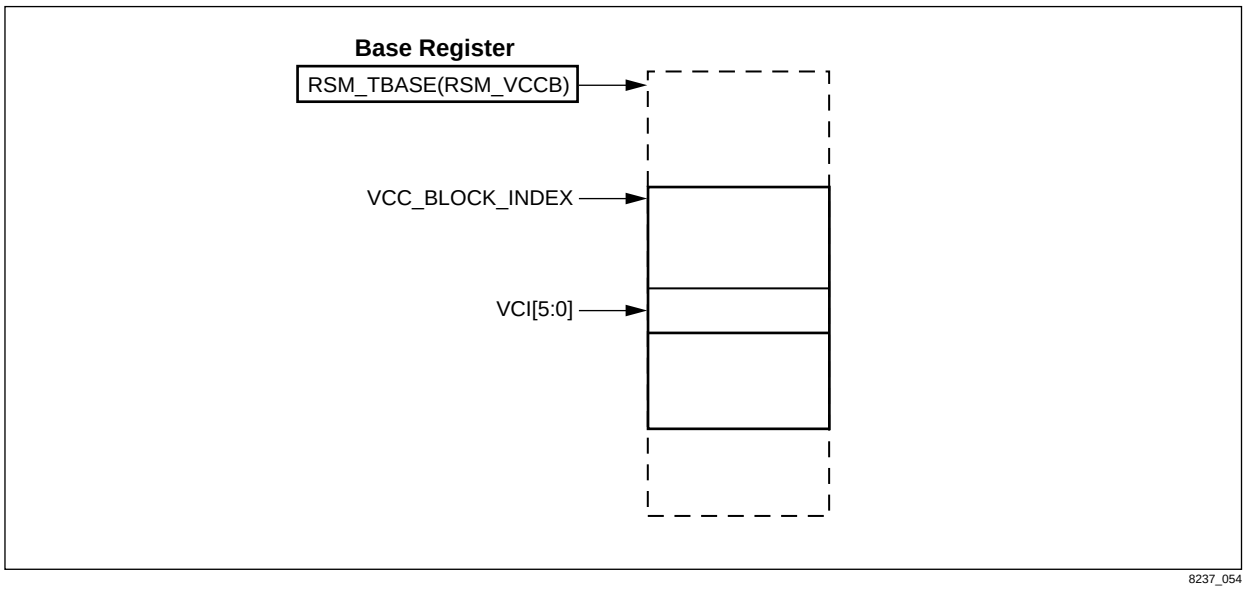
5.7.2 Reassembly VCC Table

Each reassembly VCC table entry occupies one 12-word descriptor of the reassembly VCC table.

There are two basic formats for the reassembly VCC table entry—AAL5 and AAL0. Each format completely describes the state of the reassembly process for individual VCCs.

Figure 5-16 illustrates the VCC table entry lookup mechanism as a continuation from Figure 5-15.

Figure 5-16. Reassembly VCC Table Entry Lookup Mechanism



5.7.2.1 AAL5, AAL0 and VCC Table Entries

Table 5-10 describes the format of AAL5 reassembly VCC table entries. Table 5-11 describes the format of AAL0 RSM VCC table entries.

Table 5-10. Reassembly VCC Table Entry Format—AAL5

Word	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	VC_EN	AAL_TYPE		DPRI					Reserved	PASS_OAM	Reserved	FF_DSC	TO_INDEX			PM_INDEX							AAL_EN									
1	TO_LAST	TO_EN		CUR_TOCNT													ER_EFCI	Reserved										ABR_CTRL				
2	PDU_FLAGS										Reserved						TOT_PDU_LEN															
3	CRCREM																															
4	CBUFF_LEN																FW	STAT					BFR1					BFR0				
5	CBUFF_PNTR																															
6	BOM_BD_PNTR																														Rsvd	
7	CURR_BD_PNTR																														Rsvd	CBFR1
8	SEG_VCC_INDEX																SERV_DIS															
9	Reserved								ERS_INDEX								RX_COUNTER/VPC_INDEX															
10	D_EN_NCR	ACR_NOT_ER	D_NCR_TRIG	D_NCR_DIR	Reserved				CONG_ID								EN_EXP_CNG	EN_IMP_CNG	EXP_TA_CI	EXP_TA_NI	EXP_TA_ER											
11	Rsvd			D_NCR_HI_EXP					D_NCR_HI_MANT								Rsvd		D_NCR_LO_EXP				D_NCR_LO_MANT									

KEY:

= Written by host at VCC setup.

= May be dynamically modified during active reassembly.

Table 5-11. Reassembly VCC Table Entry Format—AAL0

Word	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	VC_EN	AAL_TYPE		DPRI					Reserved	PASS_OAM	Reserved	FF_DSC	TO_INDEX			PM_INDEX							AAL_EN									
1	TO_LAST	TO_EN		CUR_TOCNT													ER_EFCI	Reserved													ABR_CTRL	
2	PDU_FLAGS										Reserved						TOT_PDU_LEN															
3	CCOUNT																TCOUNT															
4	CBUFF_LEN																FW	STAT					BFR1					BFR0				
5	CBUFF_PNTR																															
6	BOM_BD_PNTR																														Rsvd	
7	CURR_BD_PNTR																														Rsvd	CBFR1
8	SEG_VCC_INDEX																SERV_DIS															
9	Reserved								ERS_INDEX								RX_COUNTER/VPC_INDEX															
10	D_EN_NCR	ACR_NOT_ER	D_NCR_TRIG	D_NCR_DIR	Reserved				CONG_ID								EN_EXP_CNG	EN_IMP_CNG	EXP_TA_CI	EXP_TA_NI	EXP_TA_ER											
11	Rsvd		D_NCR_HI_EXP						D_NCR_HI_MANT								Rsvd	D_NCR_LO_EXP					D_NCR_LO_MANT									

Table 5-12 details the descriptions of the reassembly VCC table fields.

Table 5-12. Reassembly VCC Table Descriptions *(1 of 3)*

Field Name	Description/Function
VC_EN	Enables the VCC table entry. If disabled, the cell is discarded.
AAL_TYPE	When VC_EN is a logic high, configure channel to process specific AA as listed below; otherwise, when VC_EN is a logic low, a value of 11 does not increment the CELL_DSC_CNT counter. 00 = AAL5 01 = AAL0 10 = Reserved 11 = Reserved
DPRI	Discard Priority value. Compared against global priority in CLP and Frame Relay discard modes to determine discard eligibility. In LANE-LECID echo suppression mode, this field is the index into the LECID table that holds channel LECID values.
PASS_OAM	When bit is set OAM cells are processed and data cells are discarded without incrementing any discard counters.
FF_DSC	FIFO buffer Full Discard. When a logic high, cells are discarded when the incoming DMA FIFO buffer is almost full. This includes OAM cells when RSM_CTRL1(OAM_QU_EN) is a logic low.
TO_INDEX	Selects one of eight TERM_TOCNTx values in internal SRAM.
PM_INDEX	Pointer to a PM OAM processing word. Index with reference to top of VPI Index table.
AAL_EN	Enable various cell processes. The AAL_EN field contains the following control bits: 9 8 7 6 5 4 3 2 1 0 PM_EN | FIFO_EN | LNK_EN | FW_EN | M52_EN | BINTR | STM_MODE | LECID_EN | FRD_EN | CLPD_EN PM_EN = Enable PM OAM processing on this channel. FIFO_EN = Enable Logical FIFO Buffer mode. LNK_EN = Enable writing of buffer descriptor NEXT field. FW_EN = Enable firewall processing. Used in conjunction with FWALL_EN bit in RSM_CTR0. M52_EN = If set high, all 52 octets of the cell are written to a cell buffer. BINTR = Enable interrupt after BOM buffer filled in Message Mode. STM_MODE = Enable Streaming Mode. LECID_EN = Enable LANE-LECID echo suppression. FRD_EN = Enable frame relay DE (Discard Eligibility) mode. CLPD_EN = Enable CLP discard mode.
TO_LAST	Indicates the last VCC table entry to process for time-out.
TO_EN	Enable time-out processing on the channel.
CUR_TOCNT	Current time-out counter for the channel.
ER_EFCI	ER EFCI bit of the previous data cell.

Table 5-12. Reassembly VCC Table Descriptions *(2 of 3)*

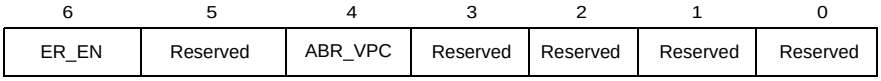
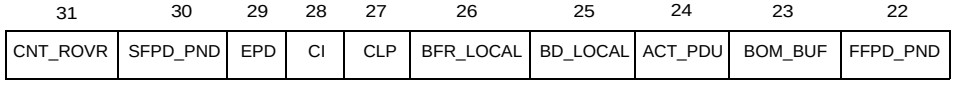
Field Name	Description/Function
ABR_CTRL	Set various control bits related to QFC. The QFC_CTRL field contains the following control bits:  ER_EN = Enable ER operation. ABR_VPC = Indicates ER connection is VPC (0 indicates VCC). For VPC operation, all VCC entries in the VPC group except VCI = 6, must be initialized with VPC_INDEX pointing to the VCC entry corresponding with VCI = 6. The VCI = 6 entry contains the integrated EFCI bit over the VPC group. Also, all entries in the VPC group including VCI = 6 must set the ABR_VPC bit to a logic high.
PDU_FLAGS	Set various flags related to PDUs. The PDU_FLAGS field contains the following control bits:  CNT_ROVR = Indication that SERV_DIS counters have rolled over. The next status entry indicates this condition. SFPD_PND = Status Full Packet Discard Pending. Set when status queue is full, and CPCS must be discarded when status entries available. EPD = Early Packet Discard Flag. Set when EPD occurs on a channel. Cleared when new packet starts and error condition cleared. CI = Congestion Indication. PTI[1] header bit ORed across the CPCS-PDU. CLP = Cell Loss Priority. CLP header bit ORed across the CPCS-PDU. BFR_LOCAL = Buffer Local. If high, the current cell buffer is located in local memory; otherwise, host memory. SAR maintains this bit. BD_LOCAL = Buffer Descriptor Local. If high, the buffer descriptors reside in local memory; otherwise, host memory. SAR maintains this bit. ACT_PDU = Active PDU. Indication that at least one buffer has been taken off of the free buffer queue for the current PDU being received. BOM_BUF = BOM buffer flag. Set high when filling the first buffer of a PDU. FFPD_PND = DMA FIFO buffer Full Packet Discard Pending.
TOT_PDU_LEN	Total PDU length in bytes.
CRCREM	Cycle Redundancy Check Remainder. CRC32 remainder used in AAL5 only.
CCOUNT	Termination Cell count. Used in AAL0 to terminate packet. When PTI termination mode is enabled, the maximum total length of a CPCS-PDU is CCOUNT × 2 bytes. In fixed length mode, initialize to 1.
TCOUNT	Termination Count. Used in AAL0 to determine the number of cells in a packet. If this field is 0 in AAL0 mode, PTI termination mode is enabled.
CBUFF_LEN	Current buffer length. Unused space of the current buffer in bytes.
FW	Firewall condition. Indicates that a firewall condition has occurred.
STAT	Status queue pool number.
BFR1	COM free buffer queue pool number.
BFR0	BOM free buffer queue pool number.
CBUFF_PNTR	Current buffer pointer. Pointer to the current unused position of the cell buffer where cell payload data may be written. In Logical FIFO Mode, the address of the FIFO.
BOM_BD_PNTR	BOM buffer descriptor pointer. Pointer to the BOM buffer descriptor.

Table 5-12. Reassembly VCC Table Descriptions *(3 of 3)*

Field Name	Description/Function
CURR_BD_PNTR	Current buffer descriptor pointer. Pointer to the buffer descriptor corresponding to the current buffer in use.
CBFR1	Current BFR1 indication. A logic high indicates that current buffer is from BFR1 pool and logic low indicates from BFR0 pool.
SEG_VCC_INDEX	Channel index of corresponding segmentation channel. Used by ER and PM-OAM processing.
SERV_DIS	Service Discard counter. Counts the number of CPCS-PDUs discarded due to either LANE_LECID echo suppression, CLP discard, or frame relay DE discard.
ERS_INDEX	Index into ER_SHIFT table for implicit congestion ER reduction. Base table address is given by ER_SHIFT_B register field.
RX_COUNTER/ VPC_INDEX	When ABR_VPC is a logic low, RX_COUNTER is the firewall mode credit counter. When ABR_VPC is a logic high, VPC_INDEX is used to control a VPC group. See ABR_VPC for description of this field.
D_EN_NCR	Enable destination ACR/ER change notification processing.
ACR_NOT_ER	Logic high results in CR change notification, logic low results in ER change notification.
D_NCR_TRIG	Destination ACR/ER change has been triggered. Initialize to logic low.
D_NCR_DIR	Indicates direction of destination ACR/ER trigger, logic high for HI, logic low for LO.
CONG_ID	Congestion Identification number
EN_EXP_CNG	Enables explicit congestion ER reduction mechanism
EN_IMP_CNG	Enables implicit congestion ER reduction mechanism—note: not valid when EN_EXP_CNG is logic high.
EXP_TA_CI	Set CI to logic high in turned-around BCK RM cells.
EXP_TA_NI	Set NI to logic high in turned-around BCK RM cells.
EXP_TA_ER	Set ER to minimum of this value and in coming FWD RM ER value in turned-around BCK RM cells when EN_EXP_CNG is logic high.
D_NCR_HI_EXP	Destination ACR/ER upper threshold, exponent portion.
D_NCR_HI_MANT	Destination ACR/ER upper threshold, mantissa portion.
D_NCR_LO_EXP	Destination ACR/ER lower threshold, exponent portion.
D_NCR_LO_MANT	Destination ACR/ER lower threshold, mantissa portion.

5.7.3 Reassembly Buffer Descriptor Structure

Reassembly buffer descriptors reside on word-aligned addresses in host memory. The host controls the allocation and management of reassembly buffer descriptors.

During initialization in each buffer descriptor entry, the host writes a pointer to an associated reassembly data buffer, in the BUFF_PTR field.

Tables 5-13 and 5-14 describe the format of the reassembly buffer descriptors.

NOTE: MSTR_CTRL_DWORD must be set to 1.

Figure 5-17. Reassembly Master Control Word

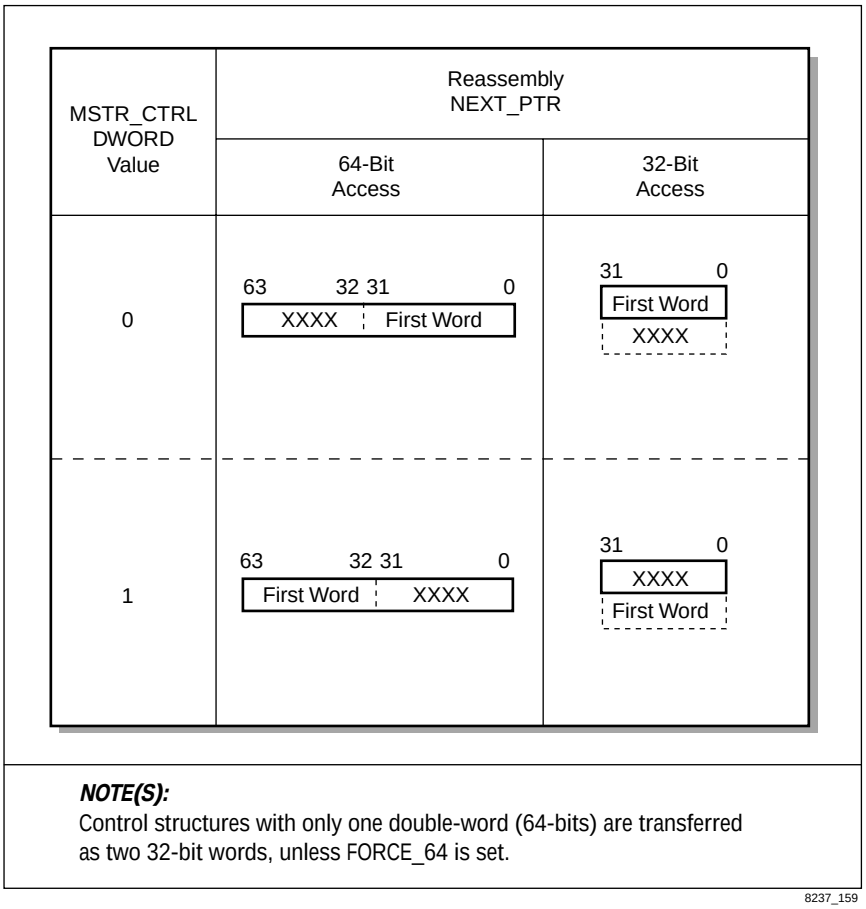


Table 5-13. Reassembly Buffer Descriptor Structure

Word	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	BUFF_PTR																															
1	NEXT_PTR																														000	

Table 5-14. Reassembly Buffer Descriptor Structure Definitions

Field Name	Description/Function
BUFF_PTR	Pointer to the address of the data cell buffer. ***NOTE(S):*** The SAR does not access this word; the user may place it anywhere in the buffer descriptor.
NEXT_PTR	Pointer to the address of the next buffer descriptor.

5.7.4 Free Buffer Queues

The host initializes the free buffer queues in SAR RSM-shared memory, and during reassembly processing, submits data buffers for reassembly to the free buffer queues.

The free buffer queue base table is located in CN8237 internal memory. Tables 5-15 and 5-16 describe the format of the free buffer queue base table entries.

Figure 5-18. Reassembly Master Control Word

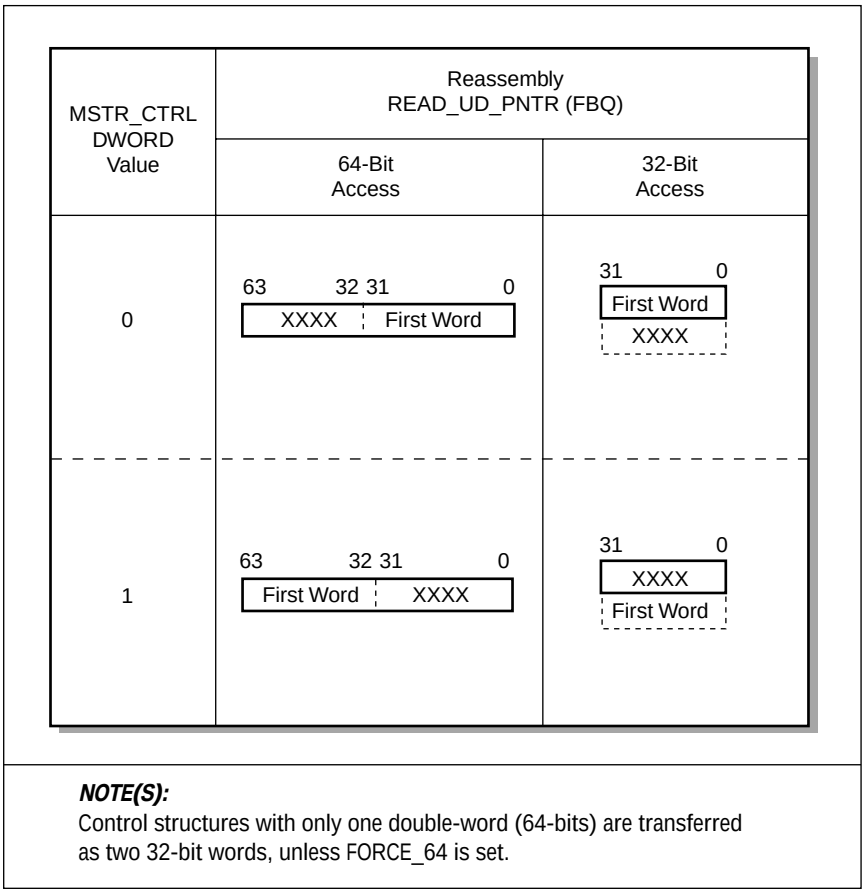


Table 5-15. Free Buffer Queue Base Table Entry Format

Word	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	Reserved								UPDATE								EMPT	Reserved			READ											
1	LENGTH																Reserved				FORWARD											
2	READ_UD_PNTR																														Reserved	
3	Reserved																															

Table 5-16. Free Buffer Queue Base Table Entry Descriptions

Field Name	Description/Function
UPDATE	Read Index Pointer Update Interval counter.
EMPT	Free buffer queue Empty flag. Used to indicate that an appropriate status entry has been written to indicate the empty condition.
READ	Current READ index pointer.
LENGTH	Length in bytes of buffers in queue. Even though LENGTH is in bytes, the user must set the length to a mod-64 bit boundary.
FORWARD	Current Forward processing Read index pointer (firewalling).
READ_UD_PNTR	Word address in user memory to write READ pointer every UPDATE interval. Read update will be written to host as 8-byte entity with lower four bytes disabled. ***NOTE(S):*** 1. MSTR_CTRL_DWORD must be set to 1. 2. Control structures (which SAR writes) with only double-word (64 bits) will be transferred as two 32-bit words, unless FORCE64 is set.
Reserved	Always set to 0.

Tables 5-17 and 5-18 describe the format of the free buffer queue entries.

Table 5-17. Free Buffer Queue Entry Format

Word	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	BUFFER_PNTR																															
1	BD_PNTR																													Rsvd		VLD
2(1)	Reserved															FWD_VLD	VCC_INDEX															
3(1)	Not Used																															

NOTE(S):
(1) These words are used only in Bank 0 if RSM_FBQCTL(FBQ0_RTN) = 1.

Table 5-18. Free Buffer Queue Entry Descriptions

Field Name	Description/Function
BUFFER_PNTR	Pointer to beginning of cell buffer.
BD_PNTR	Pointer to corresponding cell buffer descriptor.
VLD	Free buffer Valid Bit. If high, location has a valid free buffer.
Reserved	Always set to 0.
FWD_VLD	Forward Valid. If logic high, word contains valid buffer return information.
VCC_INDEX	Channel of corresponding buffer return.

5.7.5 Reassembly Status Queues

The CN8237 reports reassembly status to the host on any one of 32 reassembly status queues.

At initialization, the host assigns the location and size of up to 32 RSM status queues by initializing the reassembly status queue base table entries in CN8237 internal memory. The location and size of each RSM status queue is independently programmable via these base table entries.

Tables 5-19 and 5-20 describe the format of the reassembly status queue base table entries.

Tables 5-21 through 5-28 describe the formats of the reassembly status queue entries.

NOTE: MSTR_CTRL_DWORD must be set to 1.

Figure 5-19. Reassembly Master Control Word

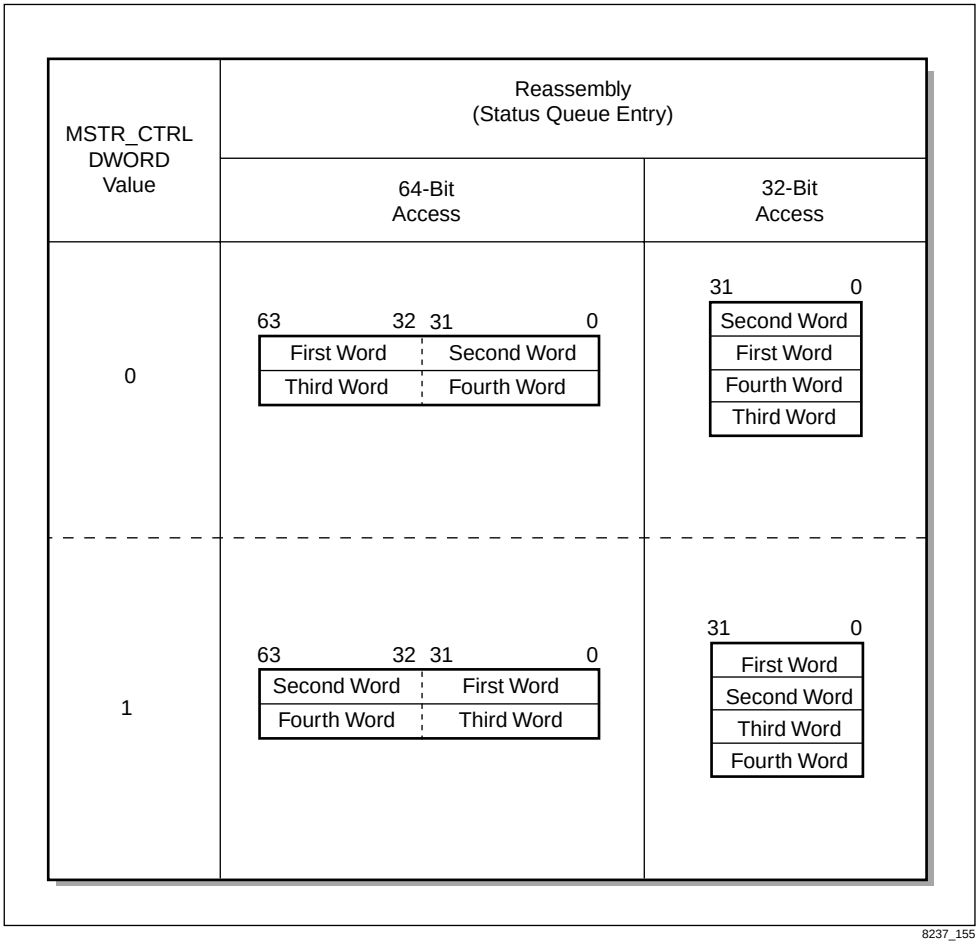


Table 5-19. Reassembly Status Queue Base Table Entry Format

Word	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	BASE_PNTR																													Reserved		
1	SIZE		Rsvd		WRITE												Reserved				READ_UD											

Table 5-20. Reassembly Status Queue Base Table Entry Descriptions

Field Name	Description/Function
BASE_PNTR	Base pointer. Pointer to the base word address of the status queue.
SIZE	Size of the status queue. 00 = 64 01 = 256 10 = 1024 11 = 4096
WRITE	Current Write index pointer maintained by the SAR.
READ_UD	Periodic Read update index pointer maintained by the user.
Reserved	Always set to 0.

Table 5-21. Reassembly Status Queue Entry Format with FWD_PM = 0

Word	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	BD_PNTR																													000		
1	UU								CPI								CPCS_LENGTH															
2	Rsvd			PDU_CHECKS													VCC_INDEX															
3	VLD	Reserved															STATUS								FWD_PM	OAM			STM			

Table 5-22. Reassembly Status Queue Entry Format with FWD_PM = 1

Word	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	BD_PNTR																													000		
1	TRCC0																TRCC0+1															
2	Rsvd		PDU_CHECKS														VCC_INDEX															
3	VLD	Rsvd							BIPV								Rsvd	FW	UNDF	OVFL	Rsvd				FWD_PM = 1	OAM			STM			

NOTE(S): OVFL and CNT_ROVR are defined in Table 5-26 under "Status."

Table 5-23. Reassembly Status Queue Entry Format with FWD_PM = 1 and NEW_PMOAM = 1

Word	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	BD_PNTR																													000		
1	TRCC0																TRCC0+1															
2	SECBC								Rsvd	CRC_ERROR	Reserved						VCC_INDEX															
3	VLD	Rsvd							BIPV								Rsvd	FW	UNDF	OVFL	Rsvd				FWD_PM = 1	OAM			STM			

Table 5-24. PDU_CHECKS Field Bits

Bit	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	BA_ERROR	Reserved	LEN_ERROR	PAD_ERROR	CPI_ERROR	Reserved	CRC_ERROR	Reserved	Reserved	Reserved	LP	CI_LAST	CI

Table 5-25. PDU_CHECKS Field Bits with CNT_ROVR = 1

Bit	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	BA_ERROR	Reserved	LEN_ERROR	PAD_ERROR	CPI_ERROR	Reserved	CRC_ERROR	Reserved	Reserved	Reserved	A3L2_ERR		

Table 5-26. STATUS Field Bits

Bit	16	15	14	13	12	11	10	9	8
	FFPD	EPD	FW	UNDF	OVFL	SFPD	TO	ABORT	CNT_ROVR

Table 5-27. STM Field Bits

Bit	3	2	1	0
	EOM	BOM	STM_MODE	BFR1

Table 5-28. Reassembly Status Queue Entry Descriptions *(1 of 2)*

Field Name	Description/Function
BD_PNTR	Buffer descriptor pointer. In Message Mode, pointer to the first buffer descriptor in the linked list. In Streaming Mode, pointer to an individual buffer descriptor.
UU	AAL5 CPCS-UU field.
CPI	AAL5 CPCS-CPI field. In AAL0 PTI terminated mode, the least significant bit contains the most significant bit of the total PDU length. The CPCS_LENGTH field contains the 16 least significant bits.
CPCS_LENGTH	AAL5 CPCS-LENGTH field. In AAL0 PTI terminated mode, it is the 16 least significant bits of the total PDU length. The CPI field contains the most significant bit.
TRCC0	PM total received cell count for CLP = 0.
TRCC0+1	PM total received cell count for CLP = 0+1.
PDU_CHECKS	Set various error flags related to PDUs. BA_ERROR = AAL5: Indicates that the total PDU length exceeds MAX_LENGTH when AUU = 1. LEN_ERROR = Total CPCS-PDU length exceeds maximum length and not at end of message. PAD_ERROR = PAD field length is not correct. CPI_ERROR = CPI field is not all 0. CRC_ERROR = AAL5 CPCS or OAM CRC error. LP = Value of CLP header bit ORed across the CPCS-PDU. CI_LAST = Value of the PTI[1] header bit in the last cell of the CPCS-PDU. CI = Value of the PTI[1] header bit ORed across the CPCS-PDU.
VCC_INDEX	VCC index of channel.
VLD	Valid. Indication that status entry is valid. The host must set to 0 after processing status.
STATUS	Sets various status bits. The STATUS field contains the following control bits: FFPD | EPD | FW | UNDF | OVFL | SFPD | TO | ABORT | CNT_ROVR FFPD = DMA FIFO buffer Full Packet Discard. EPD = Early Packet Discard occurred. A partially reassembled CPCS-PDU has been discarded due to firewall, buffer underflow, LI_EPD, SN_EPD, ST_EPD, CLP discard or Max PDU length exceeded. FW = Firewall error condition occurred. If early packet discard occurs, EPD is set. UNDF = Free Buffer Queue Underflow occurred. If early packet discard occurs, EPD is set. OVFL = Last available Status Queue entry. SFPD = Status Full Packet Discard occurred. EPD is not set. TO = Reassembly time-out condition occurred on this channel. EPD is not set. ABORT = Abort function detected. EPD not set. CNT_ROVR = Indication that the SERV_DIS counter has rolled over on the channel.
FWD_PM	PM-OAM Forward Monitoring cell detected.
OAM	A non-0 field indicates that an OAM or management cell has been received as follows: 001 = F4 OAM 010 = F4 OAM End to End 100 = F5 OAM 101 = F5 OAM End to End 110 = PTI = 6 111 = PTI = 7

Table 5-28. Reassembly Status Queue Entry Descriptions *(2 of 2)*

Field Name	Description/Function
STM	Sets various bits related to Streaming Mode. The STM field contains the following control bits: EOM | BOM | STM_MODE | BFR1 EOM = In Streaming Mode or BOM Interrupt mode, this bit identifies that the buffer contains an EOM. BOM = In Streaming Mode or BOM Interrupt mode, this bit identifies that the buffer contains a BOM. In Message Mode with BOM interrupt enabled, indicates that status entry points to only one buffer which contains a BOM. STM_MODE = Indication that Streaming Mode is enabled on the channel. BFR1 = In Streaming Mode, indicates which free buffer queue the buffer came from. Logic high indicates COM(BFR1), logic low indicates BOM(BFR0).
BIPV	BIP 16 violations. Same as BLER0+1 field in Backward Reporting PM-OAM cell.
TRCCO	PM total received cell count for CLP = 0.
TRCCO+1	PM total received cell count for CLP = 0 + 1.

5.7.6 LECID Table

The LECID table illustrated in Figure 5-20 includes up to 32 unique identifiers for LAN Emulation Clients (LECIDs). The DPRI field in the reassembly VCC table entry is used as the index into this table. Tables 5-29 and 5-30 display the LECID table entries and field definitions.

Figure 5-20. LECID Table, Illustrated

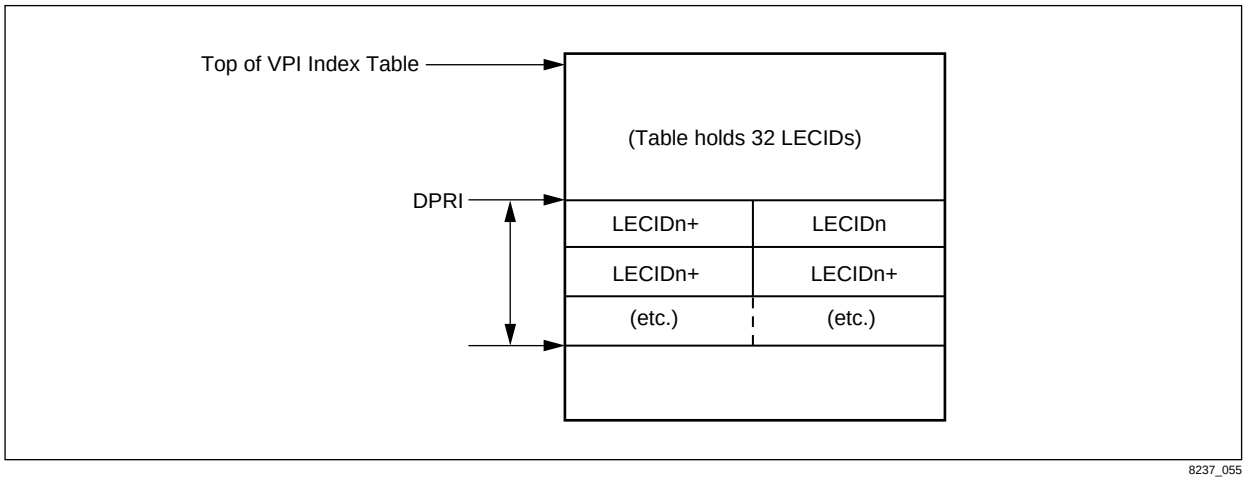


Table 5-29. LECID Table Entries

Word	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	LECID1																LECID0															
...	...																...															
1-15	LECID31																LECID30															

Table 5-30. LECID Table Field Definition

Field Name	Function/Description
LECIDn	LAN Emulation Client Identifier: a unique identifier for a LAN Emulation Client (LEC). The LECID table is capable of storing 32 LECIDs. The system designer can initialize this table to contain the LECIDs of LAN Emulation Clients that are transmitting multicast frames (point-to-multipoint Emulated LAN traffic) on an ATM network. Thus, with the LECID_EN bit set to a logic high on a channel, the RSM Coprocessor looks for a match in the LECID table with the LECID in each received LANE frame, and if a match, the frame is discarded. This implements echo suppression of superfluous multi-broadcast LANE traffic on the ATM network.

5.7.7 Global Time-Out Table

This table exists in internal SRAM starting at address 0x1340. The values in this table should be initialized during system initialization, before reassembly processing is started. These values set the selectable hardware PDU timeout values as described in Section 5.4.8. Tables 5-31 and 5-32 display the entries and field definitions for the Global Time-Out table.

Table 5-31. Global Time-Out Table Entry Format

Word	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	TERM_TOCNT1																TERM_TOCNT0															
1	TERM_TOCNT3																TERM_TOCNT2															
2	TERM_TOCNT5																TERM_TOCNT4															
3	TERM_TOCNT7																TERM_TOCNT6															
4	Reserved																TO_VCC_INDEX															

Table 5-32. Global Time-Out Table Entry Descriptions

Field Name	Description/Function
TERM_TOCNTx	Time-Out expiration count.
TO_VCC_INDEX	Time-Out VCC_INDEX tracking variable.

5.7.8 Reassembly Internal SRAM Memory Map

As indicated in Table 5-33, the Reassembly Internal SRAM is in the address range 0x1000-0x13FF.

Table 5-33. Reassembly Internal SRAM Memory Map

Address	Name	Description
Internal Reassembly Status Queue Base Table Registers:		
0x1000-0x1007	RSM_SQ_QU0	Status Queue 0 Base Table
0x1008-0x100F	RSM_SQ_QU1	Status Queue 1 Base Table
⋮	⋮	⋮
0x10F8-0x10FF	RSM_SQ_QU31	Status Queue 31 Base Table
Internal Reassembly Free Buffer Queue Base Table Registers:		
0x1100-0x110F	RSM_FBQ_QU0	Free Buffer Queue 0 Base Table
0x1110-0x111F	RSM_FBQ_QU1	Free Buffer Queue 1 Base Table
⋮	⋮	⋮
0x12F0-0x12FF	RSM_FBQ_QU31	Free Buffer Queue 31 Base Table
Other Internal Reassembly Registers:		
0x1300-0x133F	Reserved	
0x1340-0x1353	GBL_TO	Global Time-Out Table
0x1354-0x13FF	Reserved	

6.0 Traffic Management

6.1 CN8237 Overview

The traffic management capabilities of ATM differentiate it from other communication technologies. The CN8237 xBR Traffic Manager implements the complete set of ATM Service Categories as defined in the *ATM Forum's Traffic Management TM 4.1* specification. These categories include CBR, Real-Time and Non-Real-Time Variable Bit Rate (rt-VBR and nrt-VBR), UBR, GFR, and ABR. Table 6-1 provides a list of ATM attributes detailed in the *TM 4.1* specification (that is, traffic parameters, QoS parameters, and feedback characteristics), and identifies whether the CN8237 supports these for each service category. The shaded areas do not indicate that the service category and attribute are undefined for the SAR—they indicate that the *TM 4.1* specification does not detail them.

Table 6-1. ATM Service Category Parameters and Attributes

Attribute	ATM Layer Service Category					
	CBR	rt-VBR(1)	nrt-VBR(2)	UBR(3)	ABR	GFR
TRAFFIC PARAMETERS	—					
Peak Cell Rate (PCR)	Specified/Supported					
Cell Delay Variation Tolerance (CDVT), at PCR	Supported					
Sustainable Cell Rate (SCR)	—	Supported		—		
Maximum Burst Size (MBS)	—	Supported		—		
Cell Delay Variation Tolerance (CDVT), at SCR	—	Supported		—		
Minimum Cell Rate (MCR)	—				Supported	
QoS PARAMETERS	—					
Peak-to-peak Cell Delay Variation (CDV)	Supported		—			
Max Cell Transfer Delay (CTD)	Supported(4)		Not Supported(4)			
Cell Loss Ratio (CLR)	Supported(4)			Not Supported(4)	Supported(4)	Not Supported(4)
OTHER ATTRIBUTES	—					
Feedback(5)	—				Supported	—

NOTE(S):

(1) The rt-VBR service category is intended for real-time applications; that is, those requiring tightly constrained delay and delay variation, as would be appropriate for voice and video applications.

(2) The nrt-VBR service category is intended for non-real-time applications which have bursty traffic characteristics.

(3) The UBR service category is intended for non-real-time applications which do not require tightly constrained delay and delay variation. Examples of such applications are traditional computer communications applications, such as file transfer and e-mail.

(4) This is a network parameter. The CN8237 provides counters that monitor this parameter.

(5) Feedback refers to the several types of control cells called Resource Management Cells (RM cells), which are conveyed back to the source in order to control the source transmission rate in response to changing ATM layer transfer characteristics.

In order to supply these services, the xBR Traffic Manager directs the segmentation on each active VCC by controlling the segmentation coprocessor. By intelligently selecting when each VCC transmits a cell, the Traffic Manager guarantees that the output of the CN8237 conforms to the negotiated traffic contract. This selection process (called scheduling) executes dynamically, based upon per-VCC parameters.

The xBR Traffic Manager consists of two primary components. The first is the Dynamic Cell Scheduler, which provides for CBR, VBR, UBR, and GFR traffic. Second, for ABR service classes, is the ABR Flow Control Manager, an additional state machine, which works in conjunction with the xBR Cell Scheduler.

Due to the complexities of ABR, a dedicated state machine is required to achieve full rate performance—one which reacts to feedback from the network and adjusts cell transmission accordingly. This specification models ABR to align with the *TM 4.1* ABR specification. The CN8237 supports the rate based flow control and service models specified for ABR in *TM 4.1*.

The CN8237 also provides Generic Flow Control. This XON/XOFF protocol complements the GFC algorithm by allowing switches to significantly overallocate port bandwidth.

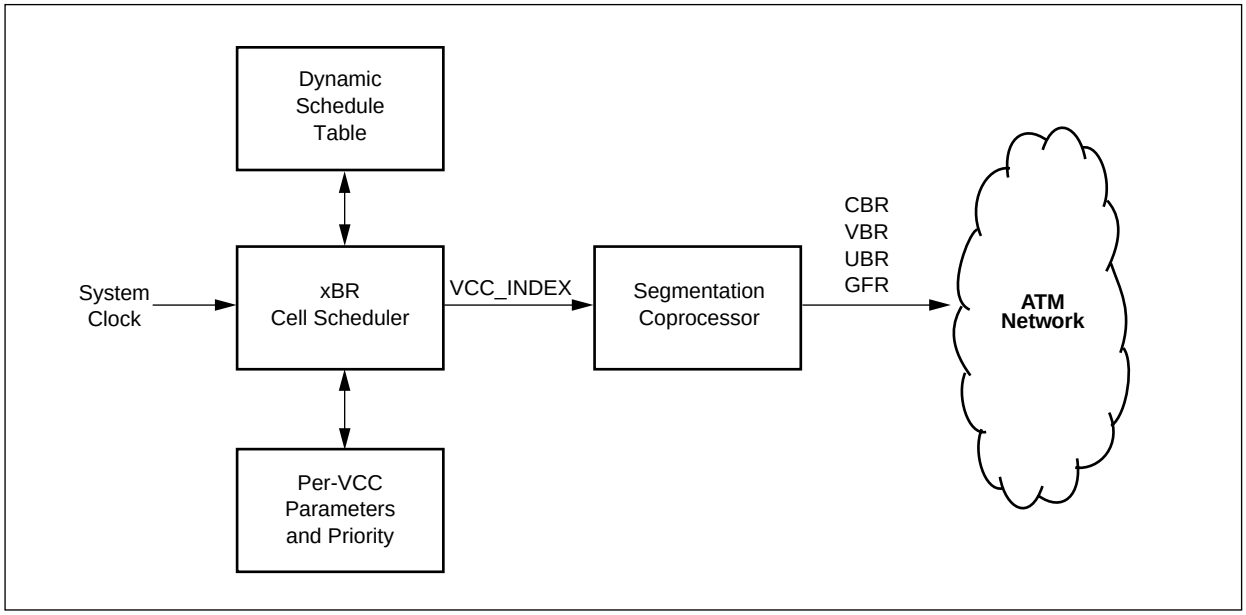
6.1.1 xBR Cell Scheduler

The Cell Scheduler maintains the QoS guarantees for each service category. It rate-shapes all segmentation traffic according to per-channel parameters. This provides each channel with appropriate transmission opportunities and guarantees conformance with network traffic contract policing algorithms applied to the outputs.

The Cell Scheduler selects individual channels based upon the dynamic schedule table. Schedule Slots in this table can be pre-reserved for CBR services. The VPs and VCs with CBR service reserve cell slots for transmission, and are intended for highly regular data sources, such as voice circuits. The CN8237 schedules all other traffic dynamically. The proprietary dynamic scheduling algorithm uses the remaining bandwidth to statistically multiplex all other service classes onto the line. The CN8237 can manage VCC traffic on either a VC or a VP level. In addition, it can schedule traffic as a CBR tunnel (or pipe), that is, several VCCs assigned to a single CBR scheduling priority, with individual VCCs within that tunnel scheduled based on their traffic parameters. Traffic into this CBR tunnel can be of types UBR, VBR and ABR. To enhance flexibility, the CN8237 supports 16 priorities of non-CBR traffic.

Figure 6-1 shows a high-level block diagram of the Cell Scheduler control flow for CBR, VBR, GFR, and UBR traffic. The Cell Scheduler tracks cell slots using the system clock and decides which VCC should send during each slot. This decision is based upon per-VCC parameters and the current condition of the Dynamic Schedule table. Unlike ABR, these service categories are open loop, and need no feedback from the network for run-time cell scheduling.

Figure 6-1. Non-ABR Cell Scheduling



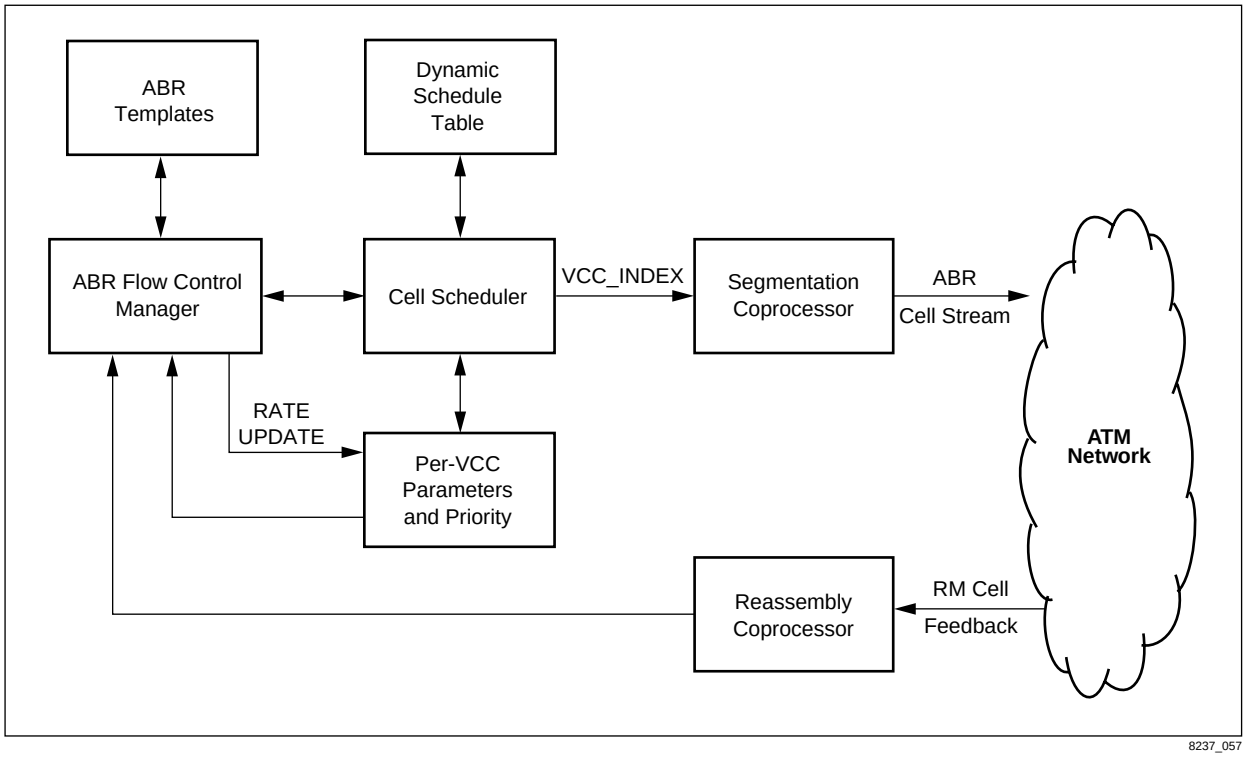
6.1.2 ABR Flow Control Manager

The CN8237 implements the ATM Forum ABR flow control algorithms, referred to in the aggregate as ABR by this document. The ABR service category effectively allows zero cell loss transmission through an ATM network, by regulating transmission based upon network feedback. The ABR algorithms regulate the rate of each VCC independently. Figure 6-2 shows a high level block diagram of the CN8237's ABR feedback control loop.

Network feedback is received by the reassembly coprocessor and routed to the ABR Flow Control Manager. This state machine processes the feedback information according to per-VCC parameters and user-programmable ABR templates, both located in SAR SEG-shared memory. These templates define ABR service parameters and policies for groups of VCCs.

Once the feedback is processed, the Flow Control Manager provides the Cell Scheduler with an updated rate. Under the policy imposed by the template, the rate complies with the *TM 4.1* Source Behavior specifications. Until the next update, the Cell Scheduler uses this rate to dynamically schedule the connection.

Figure 6-2. ABR Flow Control



For optimal performance, the ABR Flow Control Manager implements the ABR algorithm in a hardware state machine. However, to provide flexibility against minor changes in the immature *TM 4.1* specification, the state machine is programmable through Mindspeed-supplied ABR templates. These templates, resident in SAR SEG-shared memory, also provide a policy tuning mechanism for interoperability and performance. Separate groups of VCCs can be assigned to application-optimized templates according to their path through the network.

APPLICATION EXAMPLE: Application-Specific Templates

Each template can have different flow control parameters. For instance, a system designer may wish to configure a VCC traversing a network with all ER switches with a Rate Increase Factor (RIF) and Rate Decrease Factor (RDF) = 1. However, a VCC traversing a network with switches doing CI/NI (binary) marking will desire an RIF and RDF ≠ 1. Thus, separate templates would be desired for these VCCs.

6.2 xBR Cell Scheduler Functional Description

6.2.1 Scheduling Priority

6.2.1.1 16 Priority Levels + CBR

The CN8237 supports 16 scheduling priorities (the PRI field in the SEG VCC table entry), in addition to the optional CBR service category, with 15 being the highest priority, down to 0 being the lowest priority. CBR channels are assigned a priority which is in effect higher than the 16 scheduling priorities discussed here. From one to 16 of these scheduling priorities can be assigned to VBR and ABR service classes. The others are shared UBR priorities. The VBR/ABR priorities must be contiguous within the 16 scheduling priorities, and thus accessible by an offset pointer. The host sets the offset of the VBR/ABR priorities in the VBR_OFFSET field of SEG_CTRL or SCH_CTRL.

6.2.1.2 VCC Priority Assignment

The host assigns the priority of all VCCs, except CBR VCCs, by setting the PRI field in the Segmentation VCC table entry. This priority should be set at connection setup and should not be changed dynamically.

6.2.2 Dynamic Schedule Table

6.2.2.1 Overview

The xBR Cell Scheduler schedules traffic according to the Dynamic Schedule table. The table contains a programmable number of slots, determined by SCH_SIZE(TBL_SIZE). The duration of a single slot is a programmable number of clocks cycles, set as the number of clock cycles per schedule slot, in SCH_SIZE(SLOT_PER). The Cell Scheduler sequences through this table in a circular fashion to schedule CBR, VBR, and ABR traffic. UBR traffic is handled as described in Section 6.2.5. By configuring the number of slots and the duration of each slot, the system designer chooses a range of available rates. This range of available rates is dictated by the rate at which a single slot is scheduled, the size of the table, and how many slots in the Dynamic Schedule table each channel is assigned.

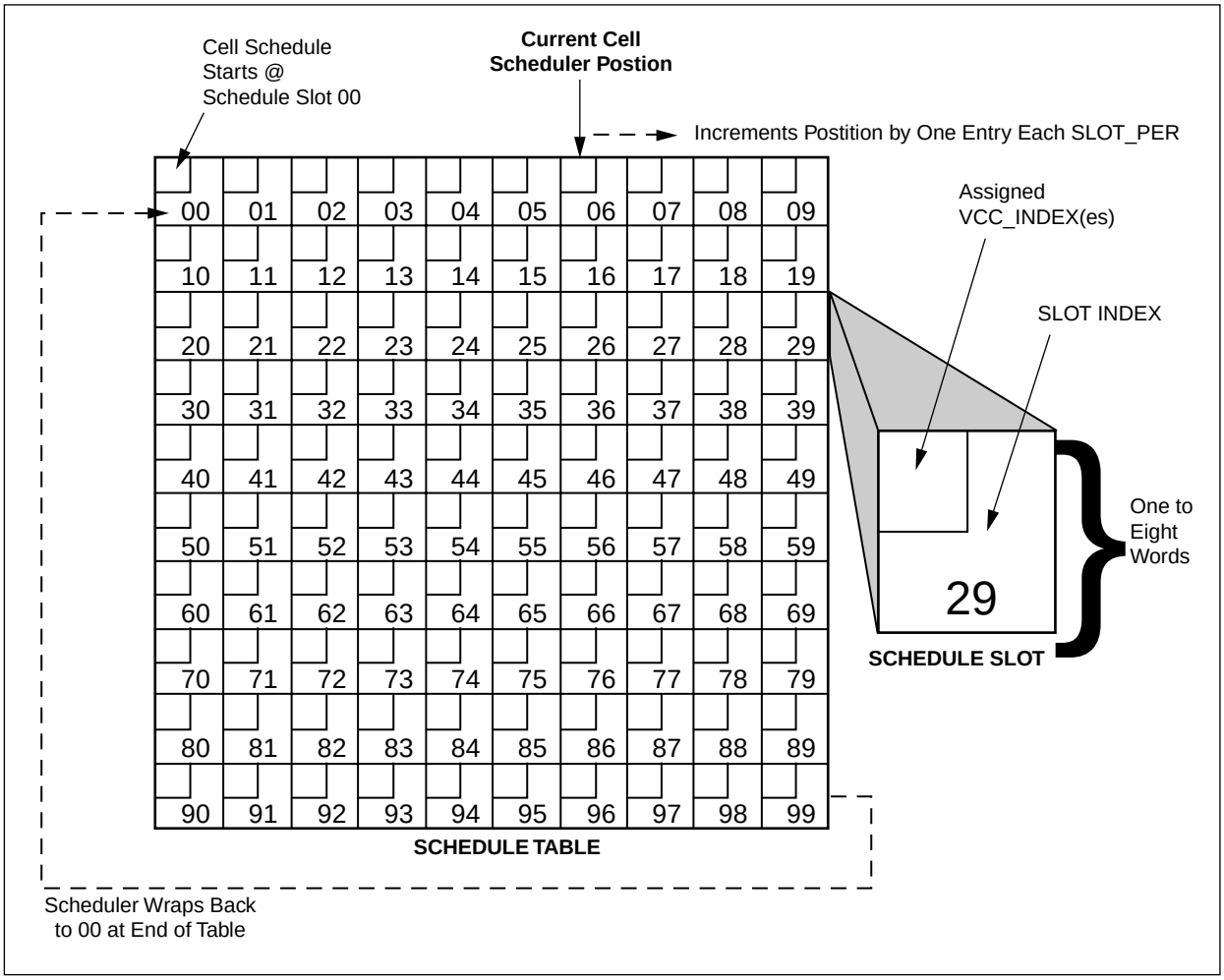
The scheduler clock is selected by bit 26 (USE_SCHREF) of the SCH_CTRL register, as shown in Table 6-2.

Table 6-2. Scheduler Clock Selection

USE_SCHREF	Scheduler Clock
0	SYSCLK
1	SCHREF

Figure 6-3 represents an example of a schedule table. In this example, the system designer has chosen 100 schedule slots. The duration of each slot is a programmable number of clock cycles. The system designer can choose to schedule cells close to the full payload data rate of STS-12C at 1.4128 M cells/second. At this cell rate, each cell slot takes 47 clocks with a clock frequency of 66 MHz. The scheduler begins at slot 00 and increments its position in the table every 47 clocks. After 4700 clocks, the scheduler position returns to 00.

Figure 6-3. Schedule Table with Size = 100



6.2.2.2 Schedule Table Slots

The user can select from a wide range of possible schedule slot formats. The user selects a format based on system requirements. If the system needs to generate CBR traffic, the first 16 bits of each schedule table slot are reserved for a CBR slot entry. The number of distinct VBR and ABR priorities required, and the enabling of CBR traffic, govern the size requirements for each slot.

The USE_SCH_CTRL bit in the SEG_CTRL register is used to turn on and turn off the mechanisms that create the 16 scheduling priorities. This maintains backward compatibility to earlier versions of the SAR, where only eight scheduling priorities were used. If the USE_SCH_CTRL bit is asserted, the user controls the size and format of all schedule slots through the SLOT_DEPTH, 4-bit VBR_OFFSET, and TUN_PRI0-OFFSET fields in the SCH_CTRL register, and the CBR_TUN bit in the SEG_CTRL register. If the USE_SCH_CTRL bit in the SEG_CTRL register is not asserted, the user controls the size and format of all schedule slots through the DBL_SLOT, 3-bit VBR_OFFSET, and CBR_TUN fields in the SEG_CTRL register. These factors, coupled with the size of the Schedule table, determine the memory requirements for the Schedule table.

Table 6-3. Selection of Schedule Table Slot Size by System Requirements

CBR Service	Number of VBR/ABR Priorities Required	Schedule Slot Size	CBR_TUN	SLOT_DEPTH	Available VBR/ABR Priority Levels
No	16	8 Words (256 bits)	0	111	0–15 (1)
Yes	15	8 Words (256 bits)	1	111	1–15 (1)
No	14	7 Words (224 bits)	0	110	VBR_OFFSET + 0–13
Yes	13	7 Words (224 bits)	1	110	VBR_OFFSET + 1–13
No	12	6 Words (192 bits)	0	101	VBR_OFFSET + 0–11
Yes	11	6 Words (192 bits)	1	101	VBR_OFFSET + 1–11
No	10	5 Words (160 bits)	0	100	VBR_OFFSET + 0–9
Yes	9	5 Words (160 bits)	1	100	VBR_OFFSET + 1–9
No	8	4 Words (128 bits)	0	011	VBR_OFFSET + 0–7
Yes	7	4 Words (128 bits)	1	011	VBR_OFFSET + 1–7
No	6	3 Words (96 bits)	0	010	VBR_OFFSET + 0–5
Yes	5	3 Words (96 bits)	1	010	VBR_OFFSET + 1–5
No	4	2 Words (64 bits)	0	001 (DBL_SLOT=1)(2)	VBR_OFFSET + 0–3
Yes	3	2 Words (64 bits)	1	001 (DBL_SLOT=1)(2)	VBR_OFFSET + 1–3
No	2	1 Word (32 bits)	0	000 (DBL_SLOT=0)(2)	VBR_OFFSET + 0 or 1
Yes	1	1 Word (32 bits)	1	000 (DBL_SLOT=0)(2)	VBR_OFFSET + 1

NOTE(S):
(1) VBR_OFFSET would be set to 0 for this mode of operation.
(2) The bottom four rows of this table describe the slot size formats when USE_SCH_CTRL is not asserted.

6.2.2.3 Schedule Slot Formats without USE_SCH_CTRL Asserted

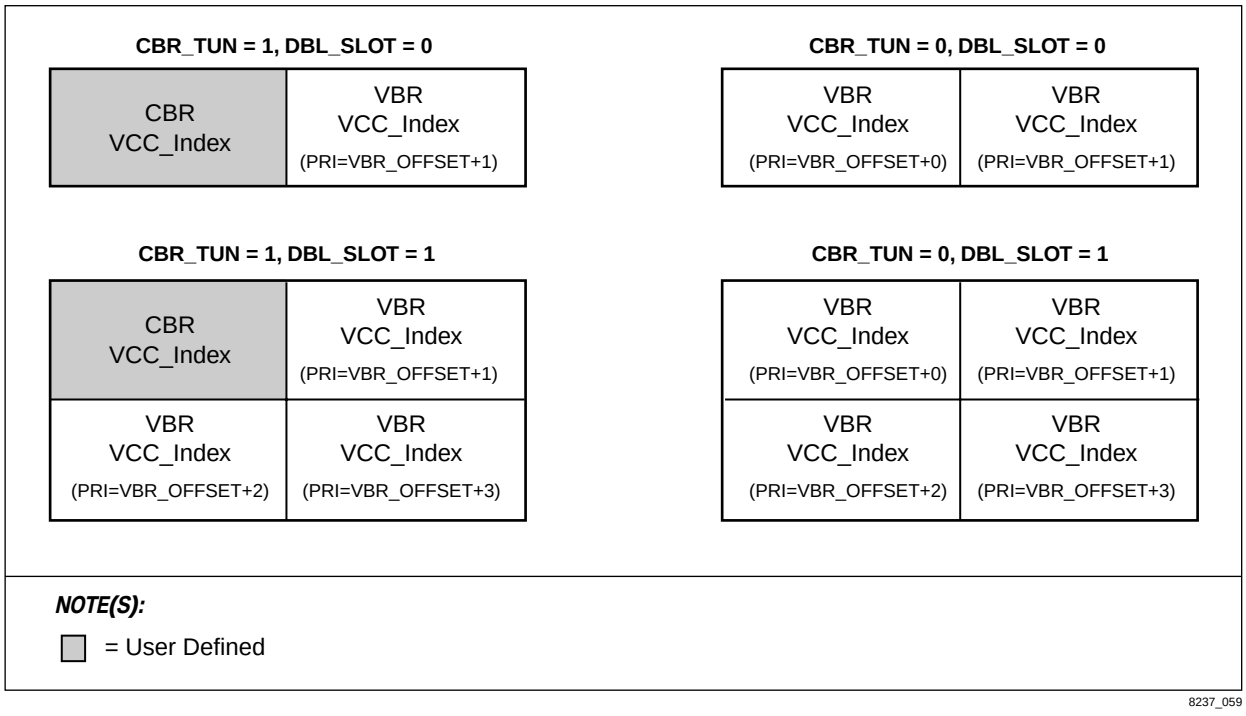
The VCC index contents of each Schedule table slot, based on the settings in the last four rows of the table, are illustrated in Figure 6-4. The SAR can assign VBR VCC index(es) to any or all of the VBR VCC_Index fields in a schedule table slot. Each of the VBR fields in a schedule table slot that are assigned a VBR VCC_Index have a different scheduling priority (PRI). Also, any of these VBR VCC_Indexes assigned by the SAR can be the first of a linked list of VBR VCCs.

VBR_OFFSET Described

VBR_OFFSET gives the offset difference in priority level between what the user or system designer wishes to assign VBR channels and what the SAR assigns via the VBR Schedule Slot format. Thus, the value of VBR_OFFSET is the difference between the highest VBR/ABR priority being actively scheduled and the largest priority of the VBR VCC_Index field in the VBR scheduling slot (either one or three). For example, if the system designer assigns VBR/ABR VCCs to three different scheduling priorities with the highest of those priorities being 5 (that is, PRI = 5), then VBR_OFFSET = 5 – 3 = 2.

Figure 6-4 illustrates how these priorities are assigned to the VBR fields.

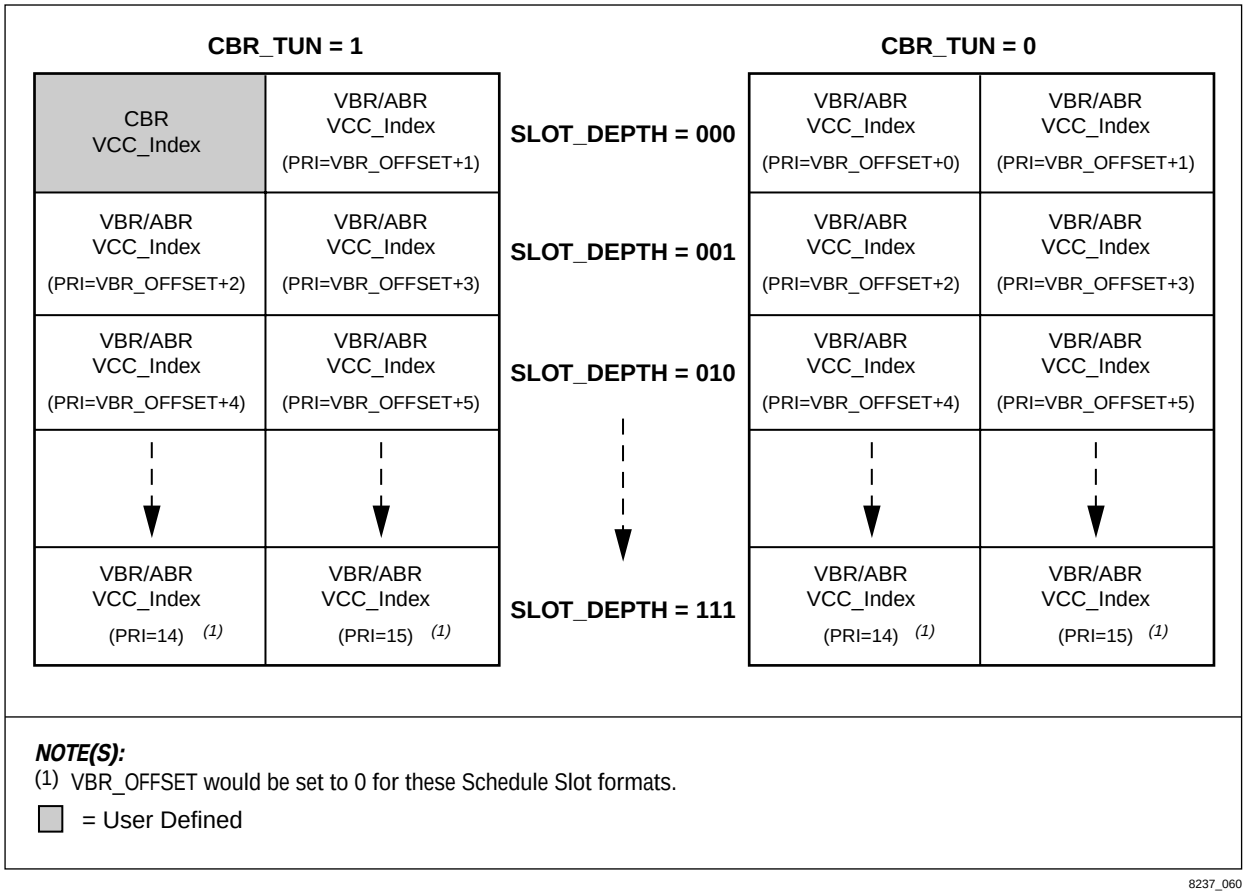
Figure 6-4. Schedule Slot Formats without USE_SCH_CTRL Asserted



6.2.2.4 Schedule Slot Formats with USE_SCH_CTRL Asserted

When USE_SCH_CTRL is asserted, the SLOT_DEPTH and VBR_OFFSET fields in the SCH_CTRL register and the CBR_TUN field in the SEG_CTRL register dictate the format of each schedule slot. This format is illustrated in Figure 6-5.

Figure 6-5. Schedule Slot Formats with USE_SCH_CTRL Asserted



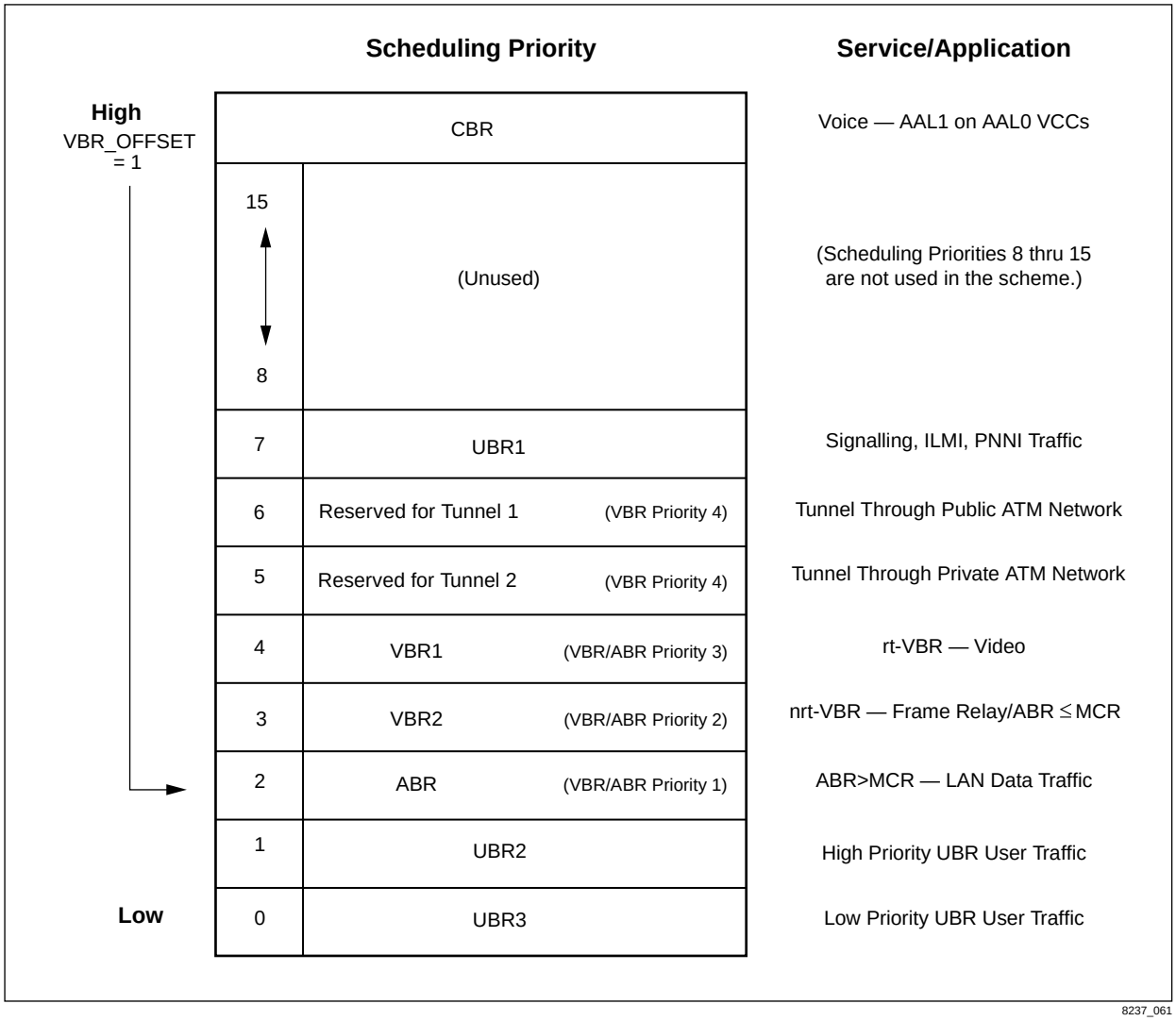
8237_060

6.2.2.5 Some Scheduling Scenarios

This section discusses a few examples of scheduling priority schemes that system designers may decide to implement. These examples demonstrate the range of flexibility that the CN8237 affords system designers and network administrators.

Figure 6-6 illustrates how scheduling priorities are assigned in scheduling slots. In this illustration, the number of VBR/ABR priorities = 4. In addition, the tunnel at PRI = 5 has VBR traffic (CBR_TUN = 1, SLOT_DEPTH = 010). The highest VBR/ABR scheduling priority (PRI) is 5. Thus, VBR_OFFSET = 1.

Figure 6-6. One Possible Scheduling Priority Scheme with the CN8237



The VBR/ABR priorities must be contiguous in order to be accessible by VBR_OFFSET. To ensure that these priorities remain contiguous when accessed by VBR_OFFSET, the following condition must be met:

$$\text{VBR\_OFFSET} + (\#\ \text{of VBR / ABR priorities}) \leq 15$$

6.2.3 CBR Traffic

The CBR service category guarantees end-to-end bandwidth through the network. Certain data sources, such as voice circuits and constrained Cell Delay Variation (CDV), require this guarantee. CDV is a measure of the burstiness of traffic. The ATM network supplies a minimum CDV for CBR channels by reserving cell transmission opportunities for the connections.

The CN8237 generates CBR traffic by assigning specific slots in the Schedule table to a CBR VCC. This connection always sends a single cell during its assigned slots. The system clock serves as a time reference for CBR cell generation. The system's designer programs the duration of schedule slots in clock cycles in the SLOT_PER (slot period) field of the SCH_SIZE register.

6.2.3.1 CBR Rate Selection

Maximum Rate

For each CBR assigned cell slot, the CN8237 generates one cell on the specified VCC. The maximum or base rate of CBR channels is determined by the duration of a cell slot according to the equation below, where R_{max} is the maximum rate in cells per second.

$$\text{SLOT\_PER} = \frac{\text{clock frequency (SYSCLK or SCHREF)}}{R_{max}}$$

SLOT_PER must be nominally set to the number of clock cycles needed to transmit a full cell, and its minimum bound should be no less than 40.

APPLICATION EXAMPLE: Determining Maximum Rate

frequency(SYSCLK) = 66 MHz
SLOT_PER = 47 clocks/slot

R_{max} = 1.4128 M cells/second

To achieve the maximum rate, the user assigns one VCC to every cell slot in the Schedule table. This prevents any other VCC from being scheduled since this channel uses all of the available slots.

6.2.3.2 Available Rates

Once a maximum rate (R_{max}) has been selected, the size of the Schedule table determines the rate granularity and minimum rate. The system designer specifies the number of table slots in SCH_SIZE(TBL_SIZE). The available CBR rates are as follows:

$$R_{max},\ R_{max}x(TBL\_SIZE - 1) / TBL\_SIZE,\ R_{max}x(TBL\_SIZE - 2)/ TBL\_SIZE,\ldots\ \ldots,\ R_{max} / TBL\_SIZE$$

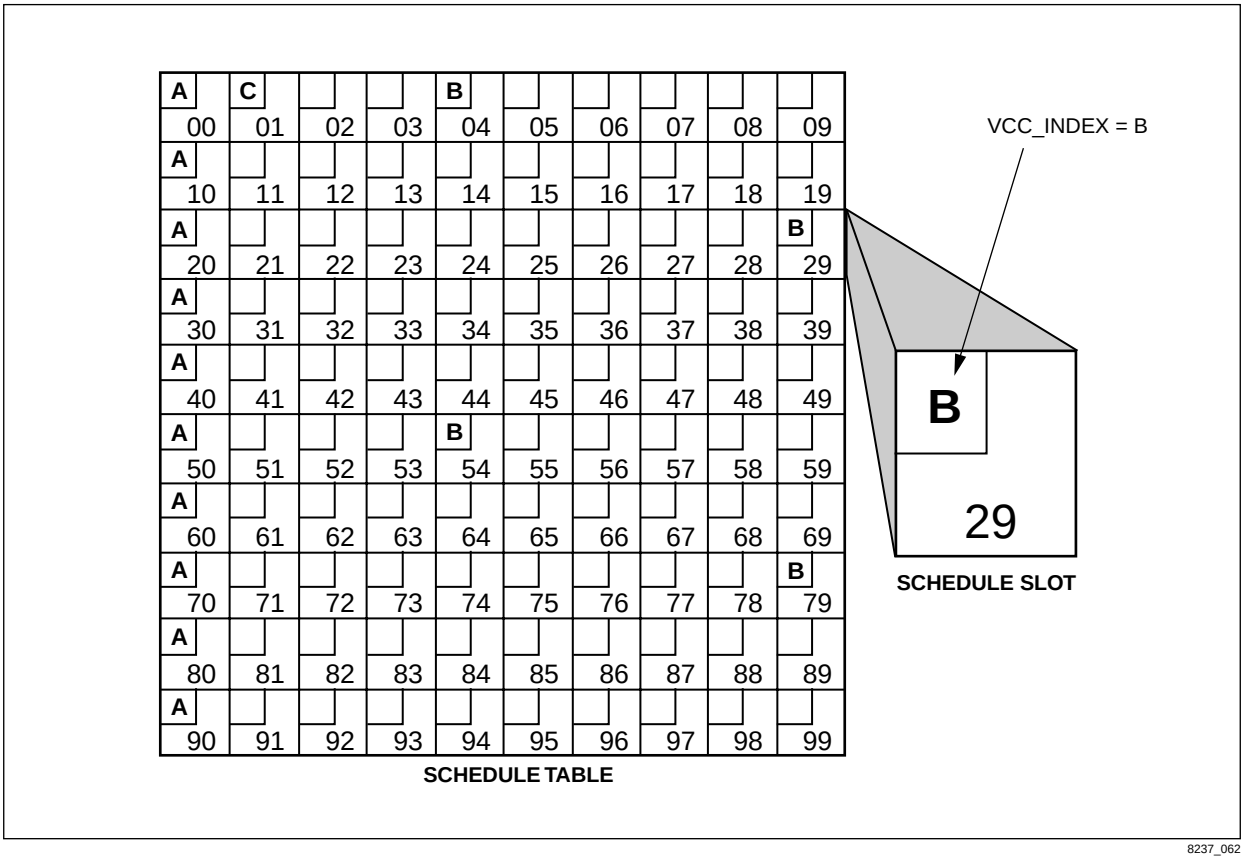
Minimum Rate

The minimum rate available is therefore R_{max} / TBL_SIZE.

Assigning CBR Cell Slots

Figure 6-7 displays an example of a Schedule table with slots assigned to various CBR channels, each with a different rate. In this example, TBL_SIZE = 100 and SLOT_PER = 47. VCC_INDEX A occupies every tenth schedule slot; therefore, it transmits at R/10, or 141.28 K cells/second. VCC_INDEX B occupies a slot every 25 cell slots and transmits at a rate of R/25, or 56.512 K cells/second. VCC_INDEX C occupies only one schedule slot and therefore transmits at the minimum rate, R/TBL_SIZE, or 14.128 K cells/second. Not all cell slots have been assigned to CBR channels. During these slots, the CN8237 dynamically schedules traffic from the other service classes. However, the total bandwidth of channels A, B, and C is reserved.

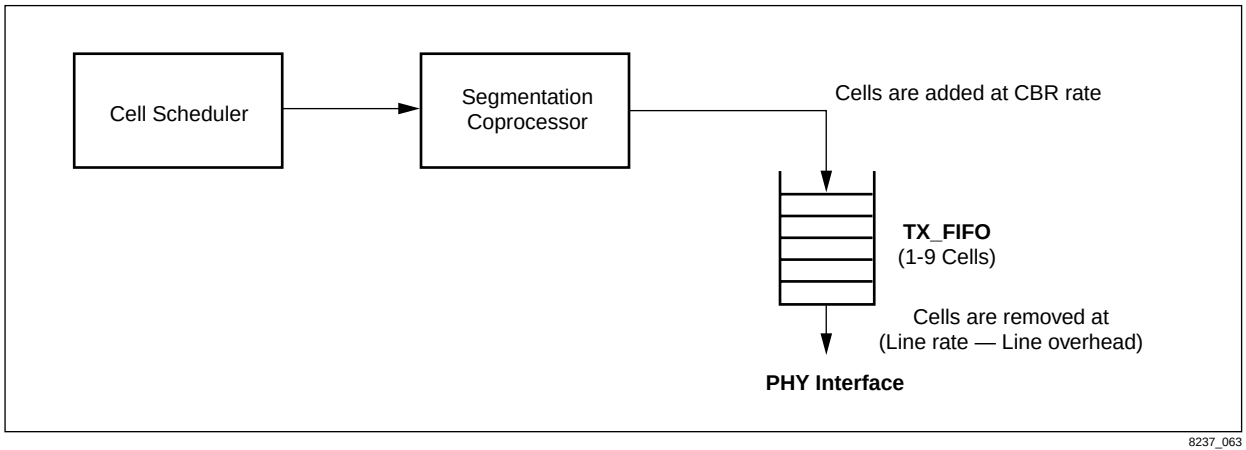
Figure 6-7. Assigning CBR Cell Slots



6.2.3.3 CBR Cell Delay Variation (CDV)

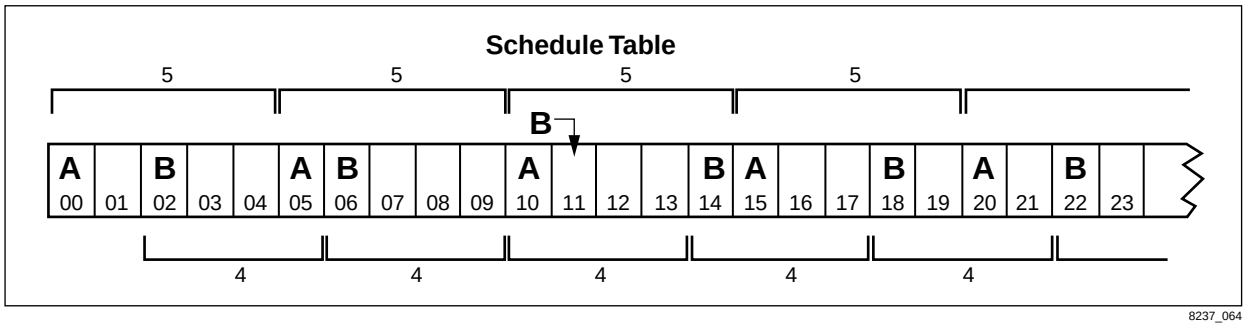
CBR connections are sensitive to CDV. (See *ATM Forum UNI 3.1* or *TM 4.1* for a complete definition of CDV.) The CN8237's Traffic Manager minimizes CDV by basing all traffic management on the Scheduler clock frequency and providing the user the ability to explicitly decide the transmit time for CBR traffic. However, no system is without some CDV. In the case of terminals using the CN8237, the dominant factor in CDV is the variation introduced between the segmentation coprocessor and the PHY layer device at the Transmit FIFO buffer (TX_FIFO). Line overhead created by the framer in the PHY layer device causes this variation. Figure 6-8 illustrates this interface.

Figure 6-8. Introduction of CDV at the ATM/PHY Layer Interface



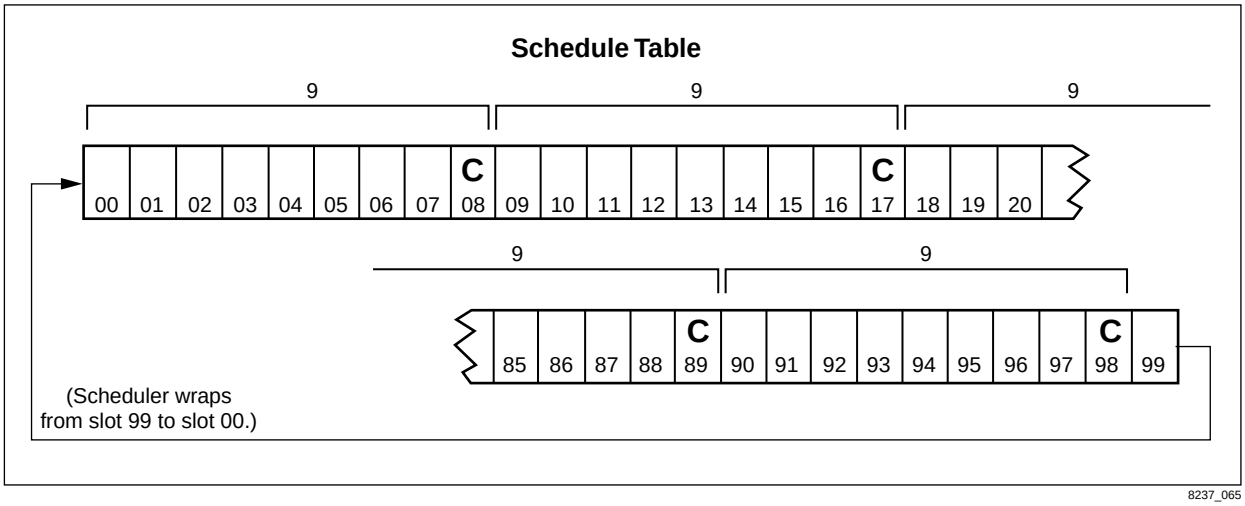
A possible source of Schedule table-dependent CDV is created when the host contracts for different CBR rates on more than one CBR channel, causing schedule slot conflicts in the Schedule table. Figure 6-9 illustrates an example of a linear representation of a Schedule table with 100 schedule slots. CBR channel A has reserved bandwidth for a rate of 282.56 K cells/second or every fifth cell slot. CBR channel B has reserved bandwidth for a rate of 353.2 K cells/second or every fourth cell slot. In this case there is a schedule slot reservation conflict between channels A and B every 20th cell slot, and one of the channel's slots has to be reserved one slot later.

Figure 6-9. Schedule Table with Slot Conflicts at Different CBR Rates



Another possible source of Schedule-table-dependent CDV occurs for certain CBR rates whose schedule slot spacings do not evenly divide the table slot size. An example of this is illustrated in Figure 6-10, where the beginning and end of a 100-slot Schedule table is shown. The system designer has reserved bandwidth for CBR channel C at a rate of 156.98 K cells/second or every ninth cell slot. When the Scheduler wraps from the end of the table to the beginning of the table, the number of schedule slots between the last C channel slot at the end of the table and the first C channel slot at the beginning of the table is 10 slots, not 9.

Figure 6-10. CDV Caused by Schedule Table Size at Certain CBR Rates



The worst-case CDV is determined by the following formula:

$$CDV_{max} = 1 / (\text{CBR rate in cells / sec}) + TX\_FIFO\_LEN / (\text{line rate in cells/sec})$$

The first term in the formula above is introduced by CBR Rate Matching (see Section 6.2.2.4).

The second term in the formula above is introduced by the TX_FIFO itself. The FIFO buffer introduces worst case CDV when one CBR cell is transmitted through an empty FIFO buffer, and the next CBR cell from that channel is segmented to a full FIFO buffer.

The system designer programs the TX_FIFO_LEN in the SEG_CTRL register. By reducing the TX_FIFO_LEN, the system designer reduces CDV. However, the TX_FIFO also absorbs PCI bus latency. To prevent PCI latency from impacting line utilization, set the TX_FIFO length according to the following formula:

$$TX\_FIFO\_LEN > 1 + (\text{worst case PCI latency}) / (\text{line rate in cells / sec})$$

6.2.3.4 CBR Channel Management

The host initializes the segmentation VCC table entry of a CBR VCC. The SCH_MODE of the VCC must be set to 001 (CBR). Thereafter, the CN8237 ignores further processing based on the SCH_MODE field in the VCC table entry for CBR VCCs.

CBR PVC Provisioning

Once a Schedule table has been created, the system assigns specific cell slots to any CBR-provisioned virtual connections (PVCs). This process takes place before the segmentation process is initiated. Once segmentation has begun, the CN8237 dynamically allocates these cell slots to other channels until the data is supplied for the CBR VCC.

CBR SVC Setup and Teardown

It is also possible to set up and tear down CBR-switched virtual connections (SVCs) without disrupting ongoing segmentation. The user simply configures a new VCC table entry. Once the VCC is initialized, the host assigns schedule slots to the VCC according to its rate. The host then submits data as with any segmentation VCC.

CBR Rate Matching

In reality, the CBR schedule rate of a channel does not exactly match the rate of its data source. To compensate for this asynchronous data source behavior, the CN8237 provides a rate-matching mechanism for use when CBR traffic is mapped to a Virtual FIFO buffer. (This method is needed only for Virtual FIFO buffers. For VCCs that are not Virtual FIFO buffers, the cell transmission is skipped automatically if there is no data available.)

For an individual CBR channel mapped to a Virtual FIFO buffer, the host directs the CN8237 to skip one cell transmission opportunity whenever data is unavailable. The host requests this rate matching adjustment by setting the SCH_OPT bit of the CBR channel. The next time the CN8237 encounters a CBR slot for this VCC, it does not transmit data on that VCC. Then, the CN8237 indicates to the host that a slot has been skipped by clearing the SCH_OPT bit.

With this method, the host effectively synchronizes the CN8237's scheduled rate to the external data source rate. The host must configure the CBR VCC rate slightly higher than the actual rate of the data source. Skipping cell transmission slots then compensates for the rate differential.

6.2.4 VBR Traffic

The CN8237 Cell Scheduler also supports multiple priority levels for VBR traffic. The VBR service class takes advantage of the asynchronous nature of ATM by reserving bandwidth for VBR channels at average cell transmission rates without hardcoding time slots, as with CBR traffic. This dynamic scheduling allows VBR traffic to be statistically multiplexed onto the ATM line, resulting in better use of the shared bandwidth resources.

6.2.4.1 Mapping CN8237 VBR Service Categories to *TM 4.1* VBR Service Categories

The *ATM Forum TM 4.1 Specification* describes the different categories of VBR service in a different manner than is employed in the CN8237 device. These relationships are described in Figure 6-4.

Table 6-4. CN8237 VBR to TM 4.1 VBR Mapping

CN8237 VBR	*TM 4.1* VBR	Comments
VBR1	(None)	*TM 4.1* does not employ single leaky bucket.
VBR2	VBR.1	Double leaky bucket.
VBR3 (or VBRC)	VBR.2 /VBR.3	*TM 4.1* defines two conformance standards for CLP(0+1).

6.2.4.2 Rate-Shaping vs. Policing

The Cell Scheduler rate-shapes the segmentation traffic for up to 64 K connections. The outgoing cell stream for each VCC is scheduled according to the GCRA algorithm. This guarantees compliance to policing algorithms applied at the network ingress point. Channels can be rate-shaped as VCs or VPs, according to one of three leaky bucket paradigms, set by the SCH_MODE bit in the channel's segmentation VCC table entry.

6.2.4.3 Single Leaky Bucket

The first and simplest bucket scheme is single leaky bucket. The user defines a single set of GCRA parameters — *I* (Interval) and *L* (Limit). *I* is used to control the per-VCC PCR, and *L* is used to control the CDVT of the outgoing cell stream. The user enables this scheme by setting the SCH_MODE bits to 100 (VBR1).

6.2.4.4 Dual Leaky Bucket

The user can also choose, on a per-VCC basis, to apply two leaky buckets to a single connection. The user enables this scheme by setting the SCH_MODE bits to 101 (VBR2).

When using VBR2 SCH_MODE, the user is limited to 256 values for the *I*2 and *L*2 parameters. These parameters are stored as bucket table entries. (See Table 6-14, for the definition of a bucket table entry.) There is complete flexibility with regard to using these 256 values to specify SCR or PCR. In VBR2, *I*1 and *L*1 can specify either PCR and CVDT, or SCR and Burst Tolerance (BT), with *I*2 and *L*2 used to specify the parameters not assigned to *I*1 and *L*1.

For example, the user can configure:

*I*1 = PCR, *L*1 = CDVT, *I*2 = SCR, *L*2 = BT

or

*I*1 = SCR, *L*1 = BT, *I*2 = PCR, *L*2 = CDVT

6.2.4.5 CLP-Based Buckets

The third option allows both buckets to apply to CLP = 0 cells. CLP=0 means high priority cells; CLP = 1 means cells are subject to discard. CLP = 1 cells are scheduled from the second bucket only. Therefore, the second bucket corresponds to PCR. This controls the PCR of the total cell stream, but only controls the SCR of CLP = 0 cells. The user enables this scheme by setting the SCH_MODE bits to 110 (VBRC—also called VBR3).

6.2.4.6 Rate Selection

The *I* and *L* parameters of the GCRA algorithm represent cell slots in the schedule table. Therefore, the VBR channels have the same maximum rate available as the CBR channels. Since VBR channels are internally scheduled by the Cell Scheduler based on that channel's *I* parameter, a floating point value, they are not constrained to repeat at some *n* integer value of R_{max} / (TBL_SIZE – *n*), Thus, VBR channels have a much finer rate granularity.

The minimum rate for VBR channels is R_{max} / (TBL_SIZE –1).

6.2.4.7 Real-Time VBR and CDV

Real-time VBR traffic should be assigned the highest scheduling priority to minimize cell delay variation. The worst-case CDV for rt-VBR traffic is calculated as

((# VBR VCCs at same-or-higher priority)+TX_FIFO_LEN) /
(line rate in cells/sec).

6.2.5 UBR Traffic

The remaining priority levels not already assigned to ABR, VBR, or CBR tunnel traffic are scheduled as UBR traffic. All UBR channels within a priority are scheduled on a round-robin basis.

To limit the bandwidth that a UBR priority consumes, use a CBR tunnel in that priority level. Another method of limiting the bandwidth that a UBR priority consumes is described in Section 6.2.8.

6.2.6 xBR Tunnels (Pipes)

A CBR tunnel occupies schedule table slots in the same manner as a CBR VCC. However, instead of serving one VCC, from one to four priority levels are served. The VCCs within a CBR tunnel can be UBR, VBR (VBR1 or VBR2), and/or ABR traffic. In the case of a UBR priority in a tunnel, the VCCs of the served priority level are serviced in round-robin order. For any VBR/ABR priorities in a tunnel, each VCC is shaped to its GCRA parameters within the CBR tunnel.

NOTE: The format of a CBR tunnel schedule table slot is not backward-compatible with the CBR tunnel format used in the Bt8233 SAR.

When the user establishes a CBR tunnel, the user-defined transmit bandwidth for that tunnel is reserved in the schedule table as schedule slots. The one-to-four priority levels assigned to that tunnel are named at this time, and are written to the PRI3, PRI2, PRI1, and PRI0 fields in the CBR_TUN_ID field for the schedule slots reserved. PRI3 should specify the highest priority level for that tunnel, down to PRI0 as the lowest assigned priority level for that tunnel. The scheduler services the highest priority level that has data available on that priority queue at each point a schedule slot for that tunnel becomes active. This serves as a secondary shaping of the traffic assigned to these tunnel priorities. Any one priority level can be assigned to only one CBR tunnel, and must not be assigned as a priority to more than one tunnel. Additionally, all priority levels utilized within a CBR tunnel need to be activated as a tunnel by setting the TUN_ENA(×) bit(s) in the SCH_PRI and SCH_PRI2 registers.

Sixteen tunnels can be active at once, if each of these tunnels has only one scheduling priority assigned. All can be VBR/ABR tunnels. On the other hand, the system designer can establish four tunnels, each of which has four scheduling priorities assigned to it, or any combination of tunnels, which includes up to a total of 16 scheduling priorities. Individual VCCs are assigned to the tunnel by the host, by setting the PRI field in the VCC table to the priority of the tunnel.

The 3-bit PRI0 field allows the entry of only priority levels 0–7. If the system designer wishes to enter a priority level higher than priority 7 in PRI0, assign the difference in values as an offset, in the TUN_PRI0_OFFSET field in the SCH_CTRL register.

If both non-tunnel and tunnel scheduling priorities exist, the host must assign the highest priority level(s) to CBR tunnel(s).

APPLICATION EXAMPLES: Tunnels

Figure 6-6, previously, shows priorities five and six used as CBR tunnels for UBR and VBR traffic. The host assigns a fixed number of Schedule table slots to the tunnel to reserve a fixed rate. Each time an assigned slot is encountered by the Cell Scheduler, it selects a VCC from a round-robin queue of active VCCs assigned to that priority.

For example, 100 UBR VCCs with PRI = 6 might currently be segmenting data. Each will get 1/100th of the CBR bandwidth assigned to the tunnel. Tunneling enables system-level end users to purchase CBR services from a WAN service provider. The purchaser can then dynamically manage the traffic within this leased CBR tunnel as CBR and/or a combination of other service categories.

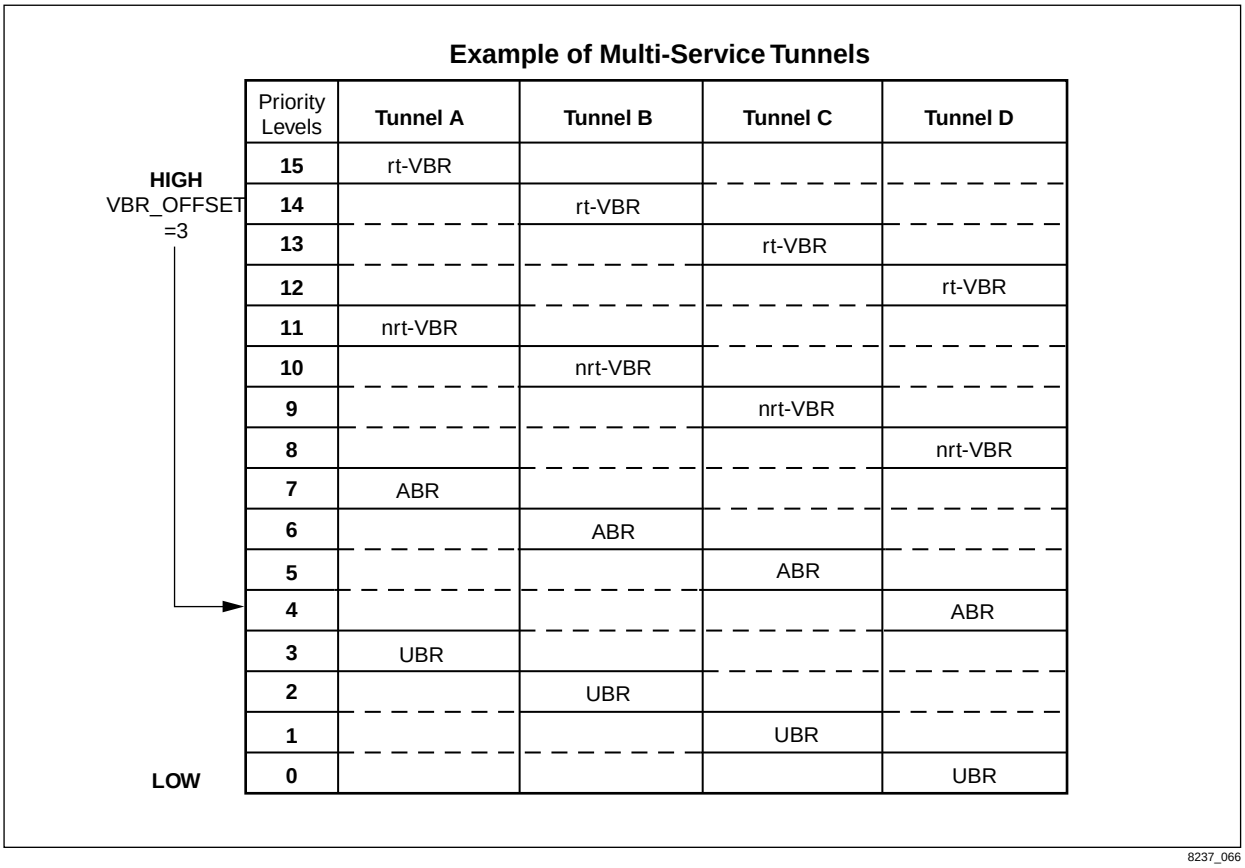
In this example, the user has configured the CN8237 to manage two independent tunnels. The first tunnel priority five, is through a private ATM network, perhaps a corporate ATM campus backbone. The other tunnel, priority six, carries traffic through a public network. This topology allows the end user to lease reserved CBR bandwidth from an administrative domain, but manage the usage of the tunnel in an arbitrary fashion.

Figure 6-11 illustrates a different use of CBR tunnels. In this example, the user has established 4 separate tunnels or pipes. Each can have an equal 1/4 share of the bandwidth available to the port by provisioning every fourth schedule table slot to one tunnel for each of the four tunnels.

In each of the tunnels, 4 priority levels are assigned. The highest priority in each tunnel is assigned to real-time VBR, the next highest priority to non-real-time VBR, the third highest priority to ABR, and the lowest priority to UBR. This in effect establishes four multi-service pipes, each on equal priority to the others (since the scheduler services each of the four pipes equally). Thus, the 4 rt-VBR priorities have effectively the same priority, and so on through the four levels of priority serviced in each pipe. number of VBR/ABR priorities is 12. CBR_TUN = 1, SLOT_DEPTH = 110.

Highest VBR/ABR Scheduling priority is 15 (that is, PRI=15). Thus, VBR_OFFSET = 3.

Figure 6-11. Another Possible Scheduling Priority Scheme with the CN8237



6.2.7 Guaranteed Frame Rate

GFR is a new service category defined by the ATM Forum to provide an MCR QoS guarantee for AAL5 CPCS-PDUs (or frames) not exceeding a specified frame length. This class of service is designed specifically to use the UBR service category. A GFR service connection is thus treated as UBR with a guaranteed MCR.

The CN8237 implements GFR by scheduling or shaping the connections using both the VBR1 scheduling procedure (for the MCR rate value) and a UBR priority queue, thereby providing fair sharing for all GFR connections to excess bandwidth. The VBR1 queue priority (the PRI field in the SEG VCC table entry) must be set to a higher priority than the UBR queue (the GFR_PRI field in the SEG VCC table entry). The priority level entered in the GFR_PRI field can only be one of the lower eight scheduling priorities. This establishes the condition where those GFR connections sharing the same GFR_PRI would fair-share any excess bandwidth above the MCR limits. Call control must not oversubscribe MCR on GFR channels and other rate-guaranteed services.

The CN8237 helps guarantee MCR on a GFR channel by providing a mechanism to trigger an increase in the scheduling priority by one priority level, if the transmit rate on that channel falls below its specified MCR. This is done using the MCRLIM_IDX field in that VCC's SCH_STATE entry in the SEG VCC table entry. MCRLIM_IDX is an 8-bit index into a table containing 256 entries (each formatted as described in Table 6-18), each containing MCR limit values, and each indicating a trigger point for an increase in the VCC's priority for that scheduled slot.

To disable this priority bumping, set MCR_LIMIT to its maximum value. The user can accomplish this by setting each of the values in the GFR MCR Limit bucket table entries as follows:

1. Set NONZERO bits to 1.
2. Set MCR_EXP fields to decimal value of 31 (that is, all-1s).
3. Set MCR_MAN fields to decimal value of 511 (that is, all-1s).

An additional level of traffic shaping can be established on GFR-UBR priority queues by shaping to a specified PCR. (See Section 6.2.8.)

6.2.8 PCR Control for Priority Queues

The CN8237 provides an optional method of creating tunnels (that is, limiting the bandwidth of a group of channels), by allowing the user to assign a PCR to a scheduling priority queue so that the aggregate of the rates of the channels assigned to that priority queue are not greater than the PCR assigned. This can be done for UBR, VBR, ABR, and UBR-GFR traffic classes.

A maximum of 4 of the 16 priority queues can be shaped to a PCR less than line rate. The queues to be shaped are identified using the QPCR_ENAx bits in the SCH_PRI and SCH_PRI_2 registers. The associated PCR for each queue thus enabled is specified in one of the two PCR_QUE_INTxx registers. These PCR values are stored as schedule table intervals. QPCR_INT3 maps to the highest priority queue that is enabled for PCR shaping, QPCR_INT2 to the next highest priority PCR shaped queue, and so on. If only one priority queue is enabled for PCR shaping, it is shaped using the QPCR_INT3 value.

If two priority queues are enabled for PCR shaping, QPCR_INT3 and QPCR_INT2 are used, and so on.

6.3 ABR Flow Control Manager

6.3.1 A Brief Overview of *TM 4.1*

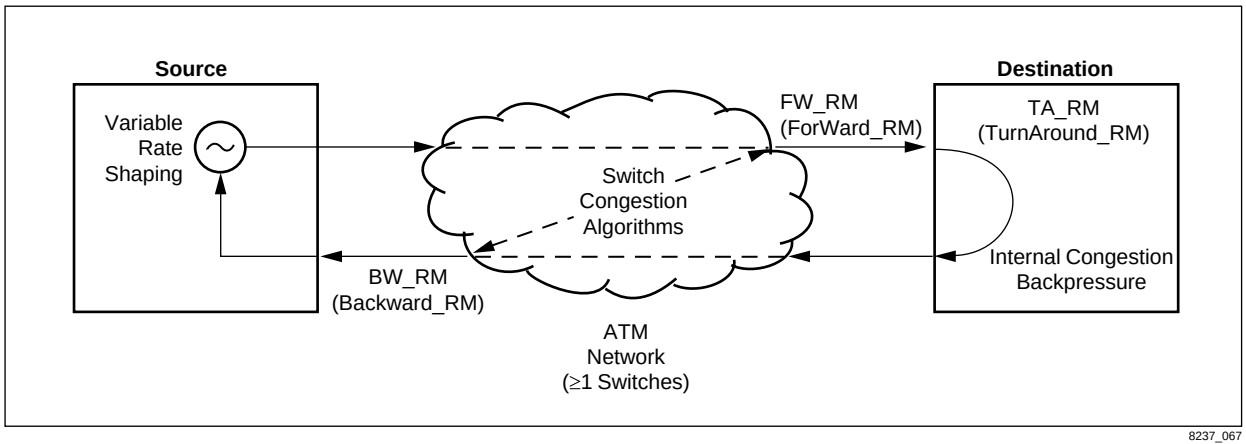
This section briefly describes the *TM 4.1* ABR Flow Control algorithms. However, it is strongly recommended that the reader be familiar with the *ATM Forum's TM 4.1 Specification* before attempting to understand the CN8237's ABR implementation.

6.3.2 Internal ABR Feedback Control Loop

As a complete implementation of the UNI ATM layer, the CN8237 acts as an ABR Source and Destination, complying with all required *TM 4.1* ABR behaviors. The CN8237 utilizes the dynamic rate adjustment capability of the xBR Cell Scheduler as the Source's variable rate-shaper. An internal feedback mechanism supplies feedback from the received cell stream to the ABR Flow Control Manager, a special purpose state machine. This state machine translates the feedback extracted from received Backward RM cells, to instructions for the xBR Cell Scheduler and segmentation coprocessor. It supports both binary and ER flow control methods.

Figure 6-12 illustrates this basic concept.

Figure 6-12. ABR Service Category Feedback Control

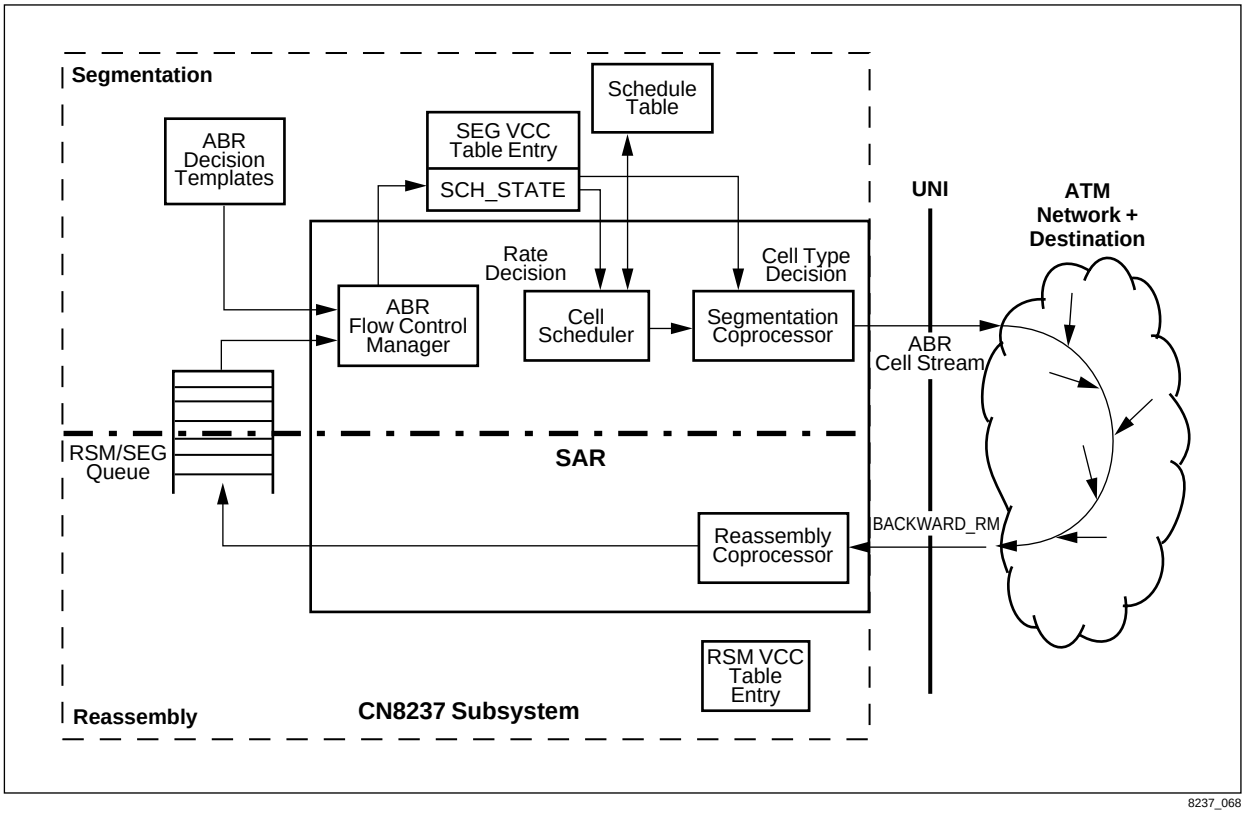


6.3.2.1 Source Flow Control Feedback

Figure 6-13 illustrates the feedback loop which controls the Source cell stream. The CN8237 injects an in-rate cell stream (CLP = 0) into the ATM network for each ABR VCC. Network elements modify the flow control fields in the cell stream's Forward RM cells. After a round trip through the network and destination node, these cells return to the CN8237 receive port as Backward RM cells. The reassembly coprocessor processes incoming Backward RM cells, and communicates with the segmentation state machines via the RSM/SEG Queue in SAR SEG-shared memory. Upon receiving this feedback from the queue, the ABR Flow Control Manager updates fields within the SCH_STATE portion of the segmentation VCC table entry. The xBR Cell Scheduler and segmentation coprocessor use these fields to generate the ABR in-rate cell stream, closing the Source Behavior feedback loop.

The SAR also processes the Explicit Forward Congestion Indication (EFCI) bit in the data cell header(s) per the rules in *TM 4.1*.

Figure 6-13. CN8237 ABR-ER Feedback Loop (Source Behavior)



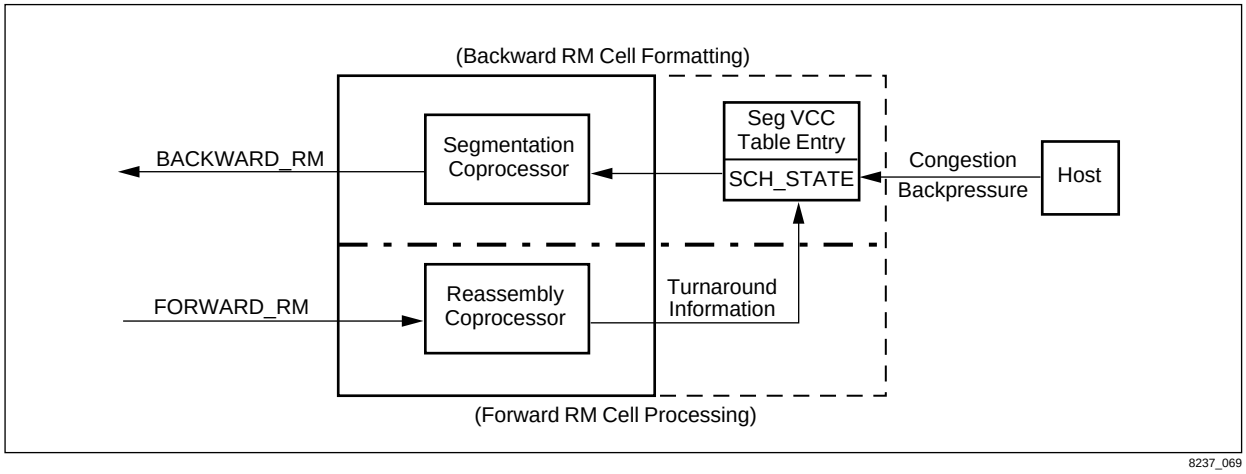
6.3.2.2 Destination Behavior

The CN8237 also responds to an incoming ABR cell stream as an ABR Destination. The reassembly coprocessor processes received Forward RM cells. It turns around this incoming information to the segmentation coprocessor via the SCH_STATE part of the segmentation VCC table entry. The segmentation coprocessor then formats Backward RM cells containing this information and inserts these turnaround RM cells into the transmit cell stream.

The CN8237 also provides a mechanism for the Destination host to apply backpressure to the Source. The host notifies the CN8237 of internal system congestion. The CN8237 passes this information to the Source using Backward RM cells, to flow control the transmitting Source. The SAR also processes the EFCI bit in the data cell header(s) per the rules in *TM 4.1*.

This process is illustrated in Figure 6-14.

Figure 6-14. CN8237 ABR-ER Feedback Generation (Destination Behavior)



6.3.2.3 Out-of-Rate Cells

In addition to in-rate cell streams, the CN8237 can also generate out-of-rate (CLP = 1) cell streams. These CLP = 1 streams compliment and enhance the information flow contained in the in-rate cell streams. An example of the use of this mechanism is to send out-of-rate Forward RM cells on a channel if the transmit rate on that channel has dropped below the schedule table minimum rate. This provides a mechanism to restart scheduling of an ABR VCC whose rate has dropped to 0 or below the schedule table minimum rate.

6.3.3 Source and Destination Behaviors

ABR's Source and Destination Behavior Definitions are listed in the *ATM Forum's TM 4.1 Specification*. Refer to this specification for these definitions.

6.3.4 ABR VCC Parameters

The state of each VCC is stored individually in its SEG VCC table entry. The ABR parameters are stored in the SCH_STATE portion of the segmentation VCC table entry. Due to the large number of ABR parameters, the SCH_STATE of ABR VCCs requires an additional location in the SEG VCC table.

6.3.5 ABR Templates

The ABR Templates reside in SAR SEG-shared memory in a region referred to as the ABR Instruction table. The ABR Instruction table contains one or more templates. Individual VCCs are assigned to a single template. Each template supports a group of VCCs. The key parameters contained in and controlled by the ABR Templates are PCR, Initial Cell Rate (ICR), MCR, RIF, RDF, and Nrm.

The user can define MCR and ICR from the ABR Templates or from fields in the SCH_STATE portion of the SEG VCC table entry for each connection, thus giving either per-template or per-connection control of MCR and ICR. This choice is globally set using the ADV_ABR_TEMPL bit in the SEG_CTRL register.

These ABR Templates are furnished by Mindspeed, and are downloaded to the CN8237 as a complete microcoded state machine.

Three components constitute a complete template: a Cell Decision table, a Rate Decision table, and an Exponent table.

The CN8237 uses the Cell Decision table to select a cell type for insertion into the transmitted in-rate cell stream.

The Rate Decision table maps ABR cell stream rates to logical states. Each state includes several vectors to other states, corresponding to different rates. When an event forces a rate decision, the CN8237 decides which vector to follow based on network stimulus (CI/NI), ER decisions, or internal timer/counter expirations (ADTF and CRM).

The Exponent table contains parameters for a piece-wise linear mapping function. This function maps a *TM 4.1* floating point rate representation (in particular, the RM cell Explicit Rate field) to a Rate Decision table index. The Exponent table is indexed by the Explicit Rate exponent. This mapping function normalizes the ER field rate to the CI/NI state-based rate decision. Once normalized, a rate can be chosen based on the minimum of the two rates.

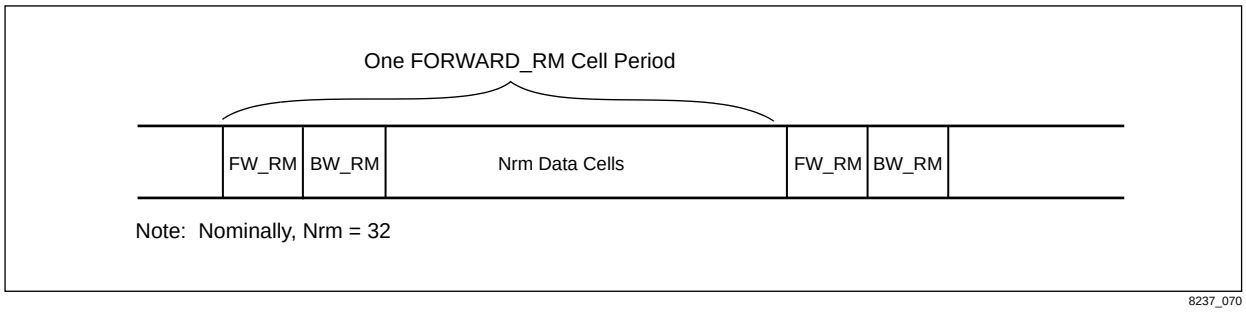
6.3.6 Cell Type Decisions

6.3.6.1 In-rate Cell Streams

Since every ABR connection in a network is full duplex, the CN8237 is a Source and a Destination for each VCC. The CN8237 multiplexes user data cells, Forward RM cells, and Backward RM cells into the in-rate ABR cell stream. *TM 4.1 Source Behaviors* specify rules for the in-rate cell stream.

Figure 6-15 shows the desired result of the specification. First, the Source transmits one Forward RM cell. A Backward RM cell follows. Finally, the Source sends Nrm user data cells. At steady state, this sequence would be repeated with a Forward RM cell marking the beginning of each sequence.

Figure 6-15. Steady State ABR-ER Cell Stream

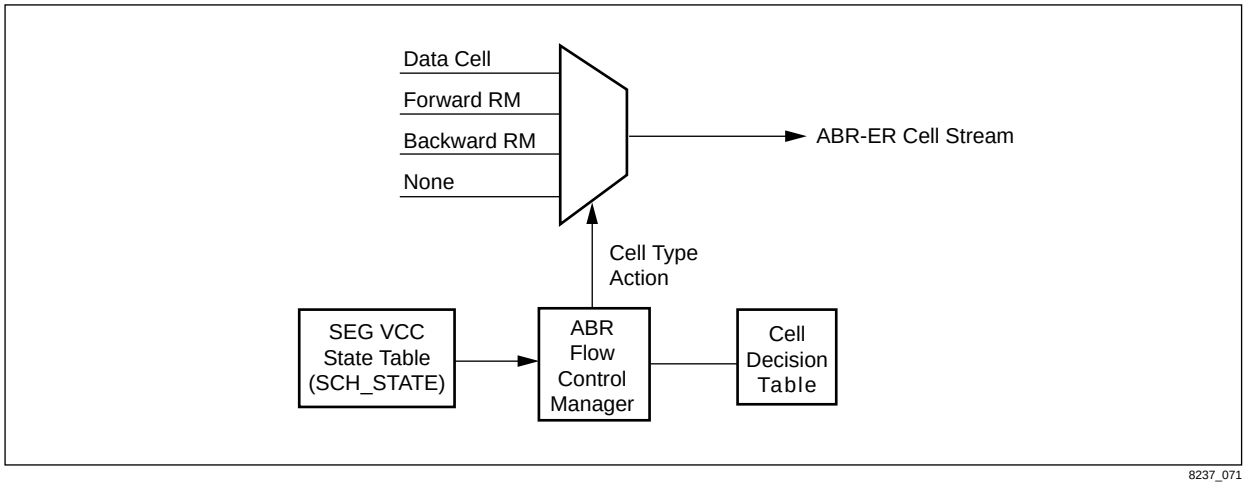


Unfortunately, due to the asynchronous, asymmetrical, and fluctuating characteristics of ABR connections, all connections do not maintain a steady state sequence. Furthermore, since it is an immature specification, the rules for cell interleaving of in-rate cell streams may be modified slightly.

The CN8237 provides a programmable mechanism to comply with *TM 4.1*'s specification on cell stream interleaving. This Cell Type Decision algorithm makes on-the-fly decisions concerning which type of cell to send, based on the current state of the connection. Figure 6-16 shows a block diagram of the algorithm. The ABR Flow Control Manager chooses a cell type based on a Cell Decision table and the VCC state. The segmentation coprocessor then formats the appropriate cell type and sends it on the ABR in-rate cell stream.

NOTE: The ABR Flow Control Manager may choose to send no cell. This occurs when the VCC has no user data to segment.

Figure 6-16. Cell Type Interleaving on ABR-ER Cell Stream



Since the Cell Decision table template resides in SAR SEG-shared memory, the user can modify it to optimize performance or react to changes in the ABR specification. Furthermore, multiple tables allow groups of VCCs to be tuned for different network policies.

6.3.6.2 ABR Cell Decisions

The Cell Decision Tables reside in SAR SEG-shared memory as a segment of the Flow Control Manager's ABR Instruction table. Each entry of a table is an ABR Cell Decision Block (ACDB). Each ACDB contains sixteen Cell Type Actions. Cell Type Actions result in one of four decisions: send in-rate Forward RM cell, send in-rate Backward RM cell, send user data cell, or no action. A Cell Type Action is chosen by a 4-bit ABR Cell Type Decision Vector (ACDV). The four bits of the ACDV correspond to four current ABR VCC conditions.

Initialization Instructions

At system initialization, the host loads one or more Cell Decision Tables into SAR SEG-shared memory. Mindspeed provides standard Cell Decision Tables. The system designer uses either these or customized templates.

To assign an ABR VCC to a table, the host initializes two SCH_STATE fields at connection setup time. The first field, CELL_INDEX, tracks the current ACDB position of the VCC. Source Behavior #2 specifies that the first in-rate cell on a VCC is a Forward RM cell. Therefore, CELL_INDEX should be initialized to an ACDB position which always decides to send a Forward RM cell. The second field, FWD_INDEX, provides a reset point for the in-rate cell stream sequence. Whenever the CN8237 transmits an in-rate Forward RM cell, it copies FWD_INDEX to CELL_INDEX.

Run-time Operation

Each time an ABR cell-transmit opportunity arises, the CN8237 makes a cell decision. The SAR retrieves CELL_INDEX from SCH_STATE and chooses a Cell Type Action location within the current ACDB. Table 6-5 shows the four conditions which are used to form this vector. The presence of a condition sets the vector bit to a 1.

Table 6-5. ABR Cell Type Decision Vector (ACDV)

Bit	Name	Description
3	TRM_EXP	Time since last forward RM transmitted ≥ TRM
2	TA_PND	TA_PND bit set in VCC table entry
1	TA_XMIT	TA_XMIT bit set in VCC table entry
0	RUN	RUN bit set in VCC table entry

The SAR updates the CELL_INDEX field after each cell decision is made. If an in-rate Forward RM cell is transmitted, FWD_INDEX is copied to CELL_INDEX; otherwise, CELL_INDEX is incremented by one (CELL_INDEX++). Therefore, the number of blocks in a Decision table is bounded by the maximum number of cell decisions made between transmitted Forward RM cells, typically Nrm.

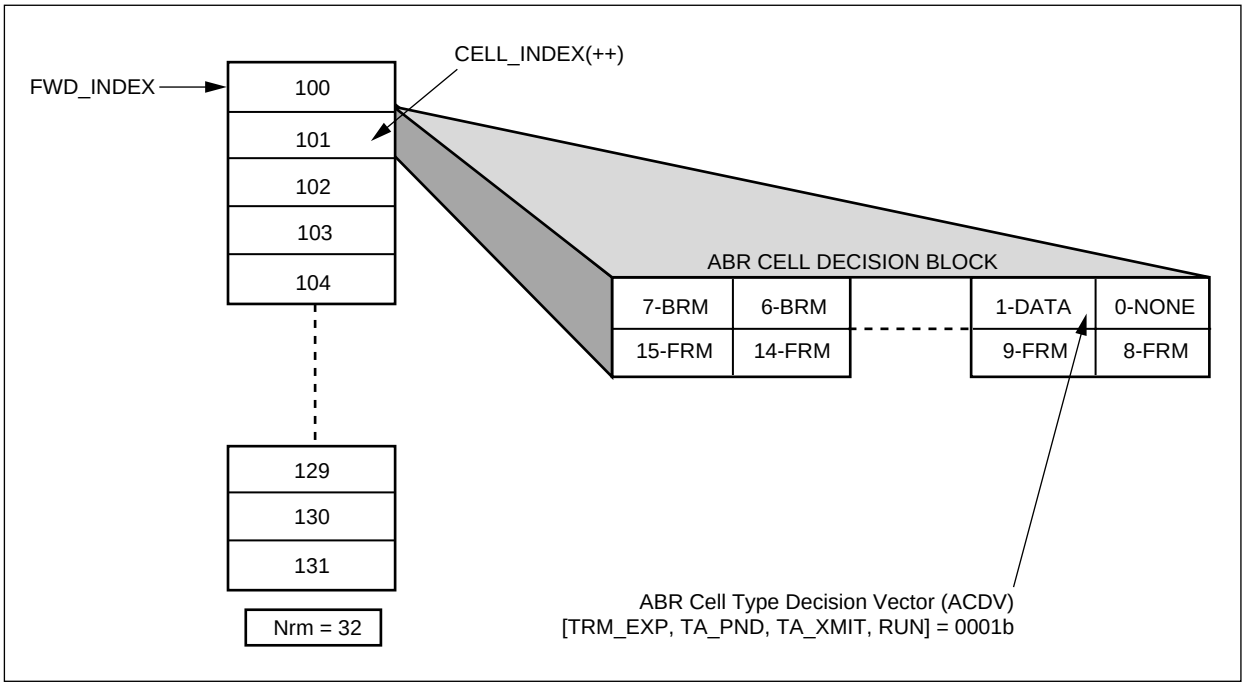
APPLICATION EXAMPLE: Cell Decision Table for Nrm=32

The figure below, shows a typical Cell Decision table. The anchor of the table, the FWD_INDEX ACDB, is located at index 100 of the ABR Decision table. Whenever a Forward RM cell is sent, this ARCB serves as the reset point for the decision state machine. When CELL_INDEX = FWD_INDEX, a steady state ABR channel transmits a Backward RM cell and advances the CELL_INDEX. The diagram shows the next ACDB (101) in more detail. It contains the 16 Cell Type Actions (for example, BRM = Send Backward RM). At the next cell transmission opportunity, the ABR Flow Control Manager state machine selects a cell type by indexing into the ACDB according to the ACDV. In this case, the only condition present is RUN=1, so ACDV = 0001b. Cell Type Action Index 1 is selected. For this ACDB, Cell Type Action 1 is DATA or Send Data Cell. Therefore, the CN8237 transmits a user data cell.

In steady state, CELL_INDEX is incremented until it reaches ACDB 131. This ACDB transmits a Forward RM Cell under all circumstances. Therefore, the CN8237 resets the CELL_INDEX for this VCC to FWD_INDEX = 100, when it reaches this ACDB.

In this example, the host should initialize CELL_INDEX to 131. The CN8237 would transmit a Forward RM cell, and the sequence would continue from the FWD_INDEX.

Figure 6-17. Cell Decision Table for Nrm = 32



6.3.7 Rate Decisions and Updates

6.3.7.1 ABR Traffic Shaping

The CN8237 schedules ABR VCCs as a single leaky bucket GCRA channel, but the rate is subject to continual adjustment. At initialization, the host assigns flow controlled VCCs *I* and *L* parameters corresponding to the negotiated ICR. The Cell Scheduler rate-shapes the initial traffic according to these GCRA parameters. This shaped cell stream includes in-rate Forward and Backward RM cells.

When feedback from the network results in flow control rate adjustments, the ABR Flow Control Manager overwrites the *I* and *L* parameters in the SCH_STATE of the VCC. The updated rates fall within the range of MCR and PCR, which are specified on a per-connection basis.

6.3.7.2 Rate Adjustment Overview

The CN8237 dynamically adjusts the rate of transmission for ABR VCCs based upon network feedback, individual VCC parameters, and the current transmission rate. The ABR Flow Control Manager uses all of these inputs to calculate the ACR of the connection. This ACR corresponds to *I* and *L* GCRA parameters. The CN8237 updates the *I* and *L* parameters of the VCC with the new ACR *I* and *L* values.

The Flow Control Manager updates the transmission rate in response to two types of events—the reception of a Backward RM cell or the transmission of a Forward RM cell. The decision process is unique for each event type.

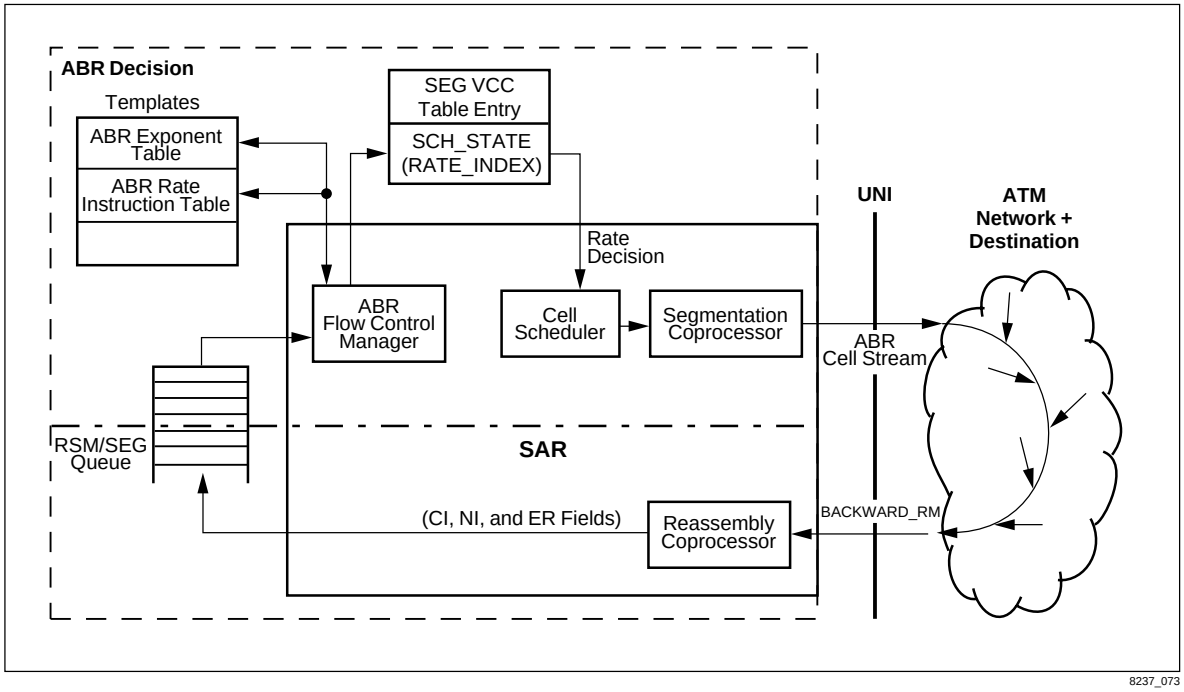
NOTE: Both of these event types may occur in one cell slot. In this case, the CN8237 makes both decisions before updating ACR.

6.3.7.3 Backward RM Cell Flow Control

The ABR Rate Decision table consists of many indexed entries. Each of these entries is called an ABR Rate Decision Block or ARDB. Each ARDB represents a possible transmission rate for an ABR VCC. The rate of an ARDB is a monotonically increasing function of the index of the block.

The CN8237 tracks the state of each ABR connection's transmission rate by storing its current ARDB index in the RATE_INDEX field of the SCH_STATE VCC structure. This RATE_INDEX uniquely identifies the current transmission rate of the VCC. Figure 6-18 illustrates a block diagram of Backward RM flow control.

Figure 6-18. Backward_RM Flow Control, Block Diagram



Backward RM cells deliver stimulus for two flow control algorithms. The first algorithm adjusts the state of a connection by a relative rate (RR) algorithm, in response to the CI and NI bits. The second algorithm allows the network to explicitly request a transmission rate in the ER RM cell field. Each of these algorithms produces a candidate rate. The Source must adjust its rate to be less than or equal to the lesser of these two candidates.

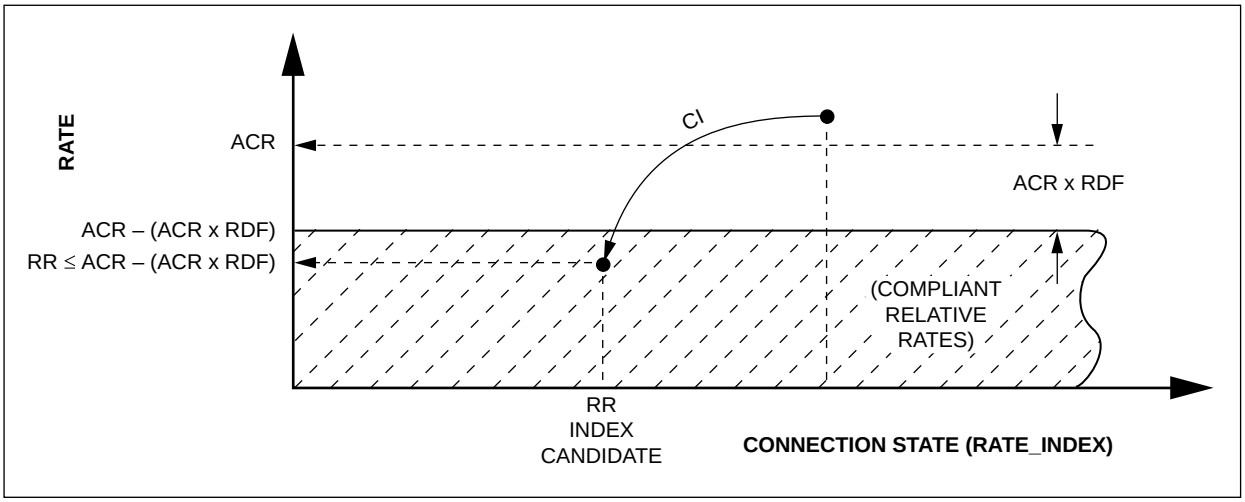
The CN8237 computes both rate candidates by first mapping all rate parameters into rate index space. Since rate indexes are a monotonically increasing function of rate, the CN8237 selects the candidate with the lowest rate index.

For the RR algorithm, the conversion is embedded in the ARDBs themselves. This algorithm can adjust the rate relative to the current rate. The adjusted rate relates to the current rate by a constant multiplicative decrease or additive increase. Specifically, the adjusted rate increases by (RIF × PCR) or decreases by (RDF × ACR). These values are pre-calculated and used to formulate a CN8237 ABR Rate Instruction table, avoiding real time math.

Each ARDB contains vectors to eight other ARDBs. Four of these vectors are responses to the relative rate elements in backward RM cells. The CN8237 uses the CI/NI bits to choose an ARDB candidate from one of these four vectors. For instance, if CI is set, the VCC must reduce its ACR by (ACR x RDF). The vectors in the ARDB that are chosen when CI is set point to ARDBs whose corresponding rates are less than [ACR – (ACR x RDF)]. The CN8237 recognizes the ARDB indicated by the chosen vector, as the RR rate index candidate.

Figure 6-19 illustrates the RR Rate_INDEX candidate selection.

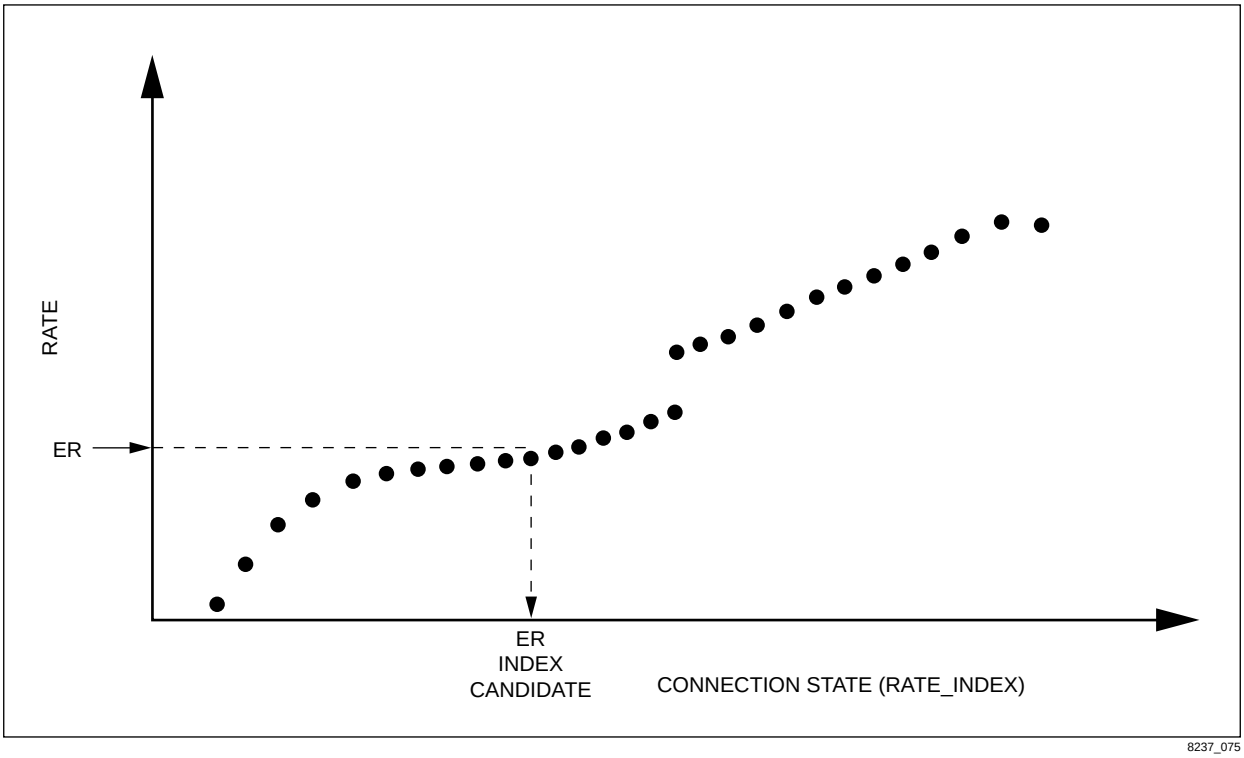
Figure 6-19. RR RATE_INDEX Candidate Selection



The CN8237 calculates the ER candidate by mapping the ER field from the Backward RM cell into rate index space. This mapping converts the floating point ER field rate representation to an absolute rate index. This mapping does not depend on the current rate of the connection. The CN8237 maps ER field to rate indexes with a programmable piece-wise linear function. Each piece-wise linear segment is described in the ABR Exponent table. Again, the system designer pre-calculates this mapping function to eliminate real-time floating point math.

Figure 6-20 illustrates the ER RATE_INDEX candidate selection.

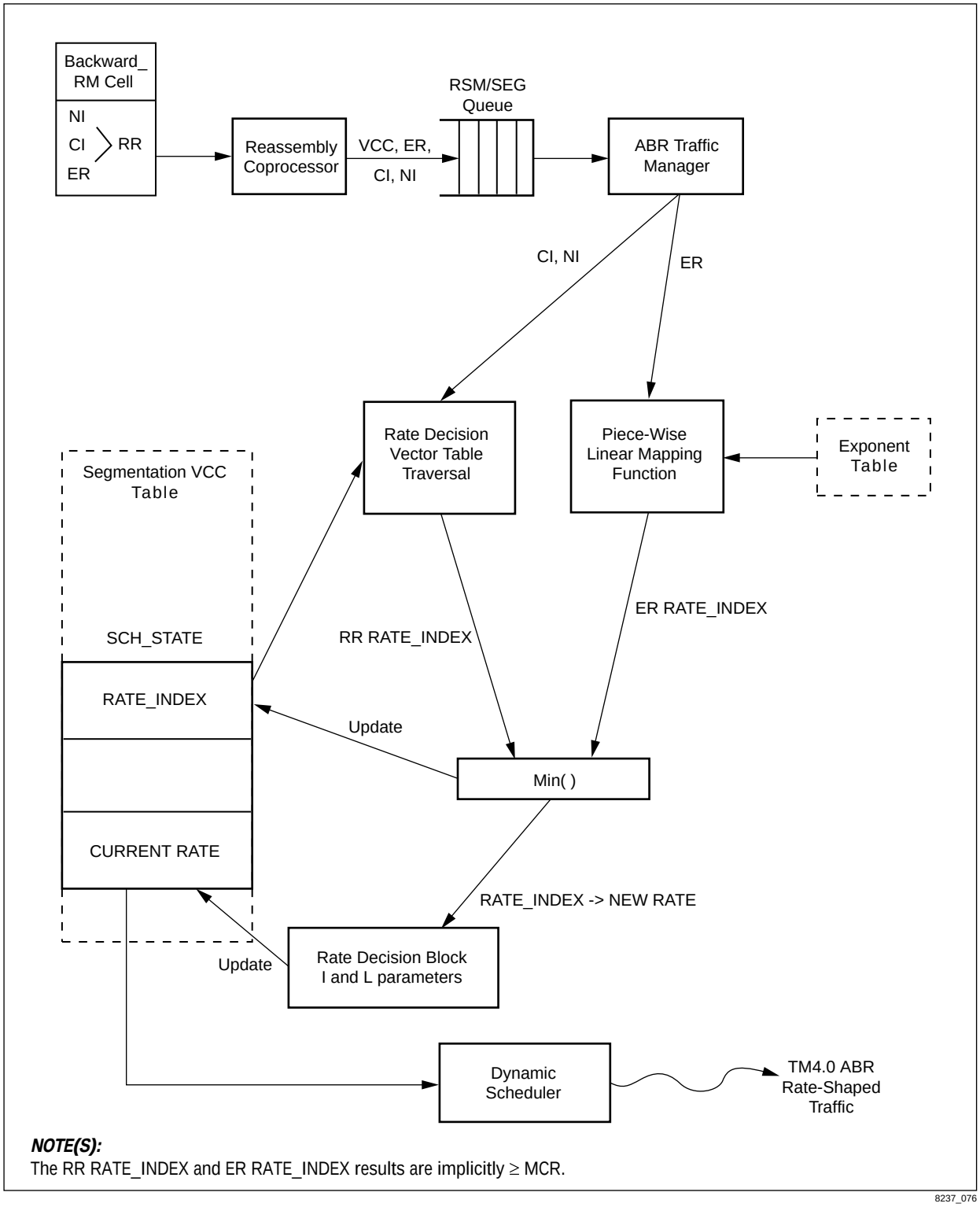
Figure 6-20. ER RATE_INDEX Candidate Selection



Once both candidates are identified, the CN8237 selects the smaller of the two, each of which are ≥ MCR. The winning candidate then replaces the current RATE_INDEX in the VCC's SCH_STATE entry. Then, the CN8237 uses this RATE_INDEX to point to the corresponding appropriate Rate Decision Block (RDB) and, as specified in *TM 4.1*, updates the VCC's rate parameter with the RDB's *I* and *L* parameters.

Figure 6-21 provides a block diagram of this process. The reassembly coprocessor passes the CI, NI, and ER fields from the RM cell to the ABR Flow Control Manager via the RSM/SEG Queue. The Flow Control Manager uses the ABR Rate Decision table and the Exponent table, both programmable components of an ER template, to calculate the two RATE_INDEX candidates, each of which are ≥ MCR. The min() function is applied to select a winning candidate. Then, the CN8237 updates the VCC's rate parameters with the newly selected rate index. This rate index points to a Rate Decision Block whose *I* and *L* parameters establish the new rate. The Cell Scheduler uses the updated rate to schedule all future traffic until the next rate update.

Figure 6-21. Dynamic Traffic Shaping from RM Cell Feedback



NOTE(S):
The RR RATE_INDEX and ER RATE_INDEX results are implicitly ≥ MCR.

8237_076

6.3.7.4 Forward RM Cell Transmission Decisions

The CN8237 also adjusts the transmission rates of an ABR in-rate cell stream before sending a Forward RM cell. This adjustment utilizes a Rate Decision Vector selection process. As stated previously, four of the eight ARDB vectors are used for CI/NI rate adjustment. The other four are used for Forward RM cell rate adjustments.

Instead of the CI/NI bits, the Forward RM cell vector selection is based upon the ADTF timer and the CRM counter. These internal measures may force the connection to decrease its rate when sending an RM cell.

The ER for Forward RM cells on any channel is set in word 18 of the VCC table entry (FWD_ER). The normal value for FWD_ER is PCR. When this field is initialized in the VCC table entry, it must be set to the rate specified by the exponent table entries so that the resulting selected rate is ≥ PCR. This happens when a Forward RM cell is eventually received as a Backward RM cell, the CN8237 maps the available cell rate (ACR) to a rate specified by the exponent table. If PCR falls between two exponent table rates and FWD_ER is set to PCR, the ACR of the connection is limited to the lower of the two exponent table rates, thereby lowering the rate below PCR.

6.3.7.5 ACR Change Notification

An ACR change notification mechanism per ATM Forum AF-SAA-0108* Appendix D is implemented for both source and destination. Five fields in the SEG VCC table entry (S_EN_NCR, S_NCR_LO, S_NCR_HI, S_NCR_TRIG, and S_NCR_DIR) are used for source ACR change notification. Six fields in the RSM VCC table entry (D_EN_NCR, D_NCR_LO_, D_NCR_HI, ND_CR_TRIG, D_NCR_DIR, and ACR_NOT_ER) are used for destination ACR/ER change notification. x_EN_NCR enables the ACR change notification mechanism. x_NCR_LO is the low threshold value, and x_NCR_HI is the high threshold value. x_NCR_TRIG indicates that a notification has been triggered, x_NCR_DIR indicates which threshold was crossed last, logic high for HI, logic low for LO. Finally, ACR_NOT_ER allows the user to choose between destination ACR or ER change notification. A special status queue entry is written when the following conditions occur:

- x_NCR_LO triggers a notification when
  - The new value of ACR$^{(1)}$ is less than or equal to NCR_LO, AND
  - Either this is the first notification or the last notification was triggered from NCR_HI.
- x_NCR_HI triggers a notification when
  - The new value of ACR$^{(1)}$ is greater than or equal to NCR_HI, AND
  - Either this is the first notification or the last notification was triggered from NCR_LO.

NOTE: $^{(1)}$For the destination change notification, this can be either ACR or ER depending on ACR_NOT_ER value.

The special segmentation status queue (see Section 4.3.6) containing the current ACR value and is indicated by the NCR bit set to a logic 1. The SRC_NOT_DEST bit indicates whether the notification is source- or destination-generated. In the case of a destination change notification, the current ER value (TA_ER) is also provided.

6.3.7.6 Rate Adjustment in Turnaround RM Cells

The system designer has various options of effecting a lowered explicit rate in Turnaround RM cells.

Implicit ER Modification

Added to word 10 of the RSM VCC table entry are EN_IMP_CHG and CONG_ID. Added to word 9 is ERS_INDEX.

When EN_IMP_CNG is a logic high and a forward RM cell is received, SCH_CNG(CONG_ID) is checked. If a logic high, the ER field is modified per the mapping mechanism described below and written into TA_ER field of the SEG VCC table entry. This mechanism provides an efficient per-connection hardware reduction of ER values when turning around RM cells.

The implicit ER reduction mechanism uses a ER_SHIFT table and the ER_HC_RATE table to provide programmable scaled reduction of the ER value based on the value's range. In addition, multiple ER_SHIFT tables and ER_HC_RATE tables may be used to allow different VCCs to utilize different mappings to ER_HC_RATE table entries. For example, different mappings may be associated with various ABR templates. A 6-bit index, ERS_INDEX, located in the RSM VCC table entry indicates which ER_SHIFT table to use. The exponent portion of the ER value is used as an index into the ER_SHIFT table. Thus, there are 32 entries in each ER_SHIFT table (see Figure 6-22). Each entry contains two values, the first, HC_SHIFT, specifies the mantissa shift mask, and the second, HC_INDEX, specifies the address of an entry into the ER_HC_RATE table. For HC_SHIFT values larger than nine, no ER_HC_RATE table lookup is performed, and the ER value from the incoming RM cell is written into the TA_ER field in the SEG VCC table.

The number of rates in the ER_HC_RATE table for each exponent value is determined by:

$$2^{\wedge\wedge}(9 >> HC\_SHIFT)$$

The address of an ER_HC_RATE table entry is calculated as:

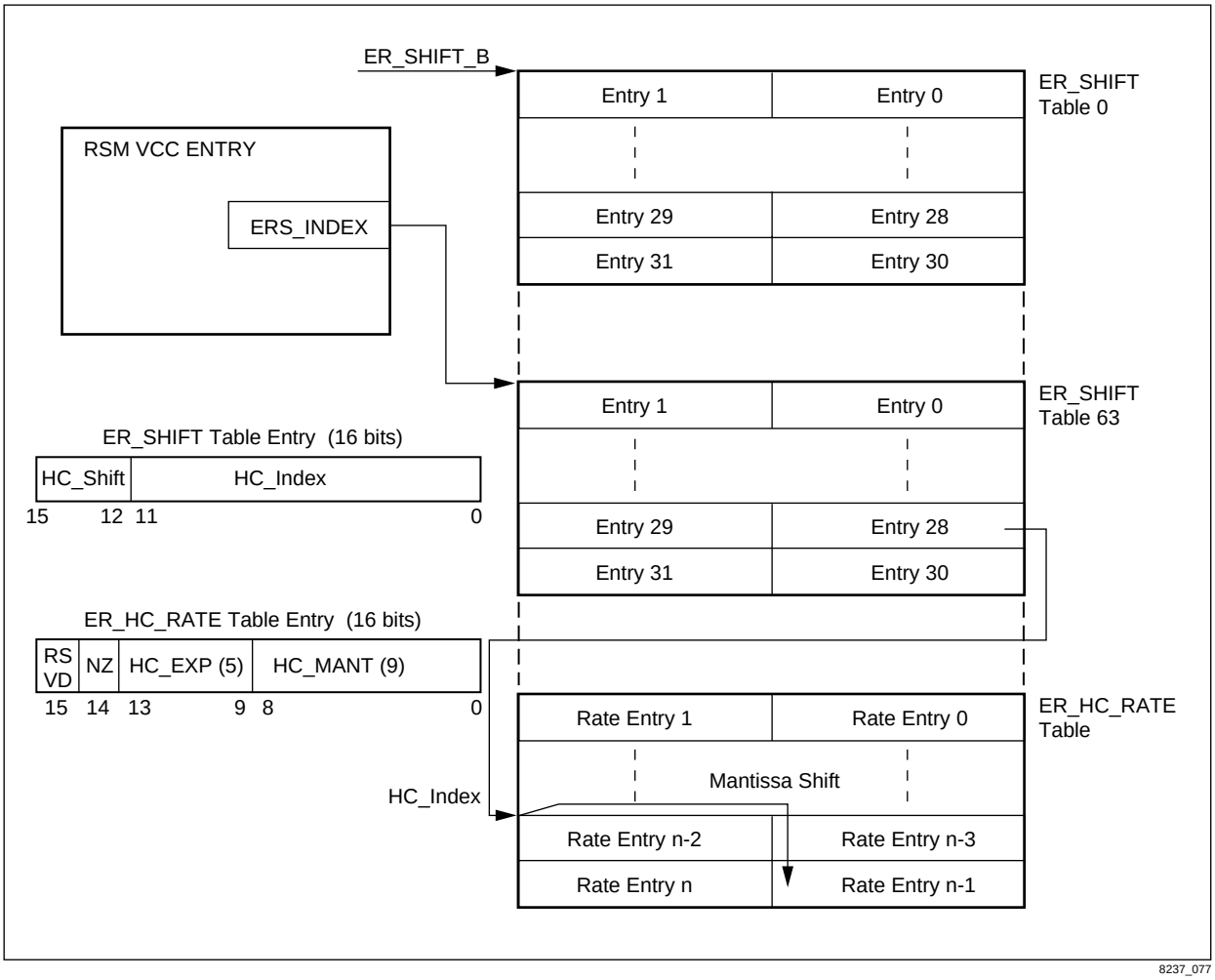
$$(ER\_SHIFT\_B << 5 + HC\_INDEX) + RM\_CELL\ mantissa\ [8:1] >> HC\_SHIFT$$

The right entry for this address is chosen, if the bit value of the RM_CELLS mantissa at its bit position HC_SHIFT is zero, and the left otherwise. The new reduced rate is then:

$$2\ ^{\wedge\wedge}\ HC\_EXP \times (1 + HC\_MANT/512)$$

The maximum index into the rate table has an offset of 4096 words to the ER_SHIFT_B base address. Multiple ER_HC_RATE tables might be used.

Figure 6-22. ER Reduction Mapping



Explicit ER Modification

Added to word 10 of the RSM VCC table entry are EN_EXP_CNG and EXP_TA_ER.

When EX_EXP_CNG is a logic high and a forward RM cell is received, the ABR block compares the ER value in the RM cell with EXP_TA_ER and writes the lower value to TA_ER field of the SEG VCC table entry.

Explicit CI and NI Modification

Added to word 10 of the RSM VCC table entry are EXP_TA_CI and EXP_TA_NI.

When EXP_TA_CI is a logic high, the value of CI in the turned-around RM cell is asserted. When EXP_TA_CI is a logic low, the value of CI in the turned-around RM cell remains the same; that is, it reflects the EFCI field of the last data cell ORed with the CI value from the FWD RM cell.

When EXP_TA_NI is a logic high, the value of NI in the turned-around RM cell is logic high; otherwise, the FWD RM cell value of NI is reflected in the turned-around BCK RM cell.

6.3.7.7 Optional Rate Adjustment Due to Use-It-Or-Lose-It Behavior

The use-it-or-lose-it behavior defined in *TM 4.1* describes a rate adjustment for a VCC that has a high ACR but is not actually using that bandwidth in transmission. In this case, the ACR is lowered to the level of ICR for that channel.

The CN8237's implementation of the *TM 4.1* use-it-or-lose-it behavior allows the system designer to exactly tailor its operation. It uses a simple timeout mechanism, dictatable for each explicit rate.

The designer selectively enables this function for the chosen rate(s) by setting the ACR_ENA bit to a logic high in each of the corresponding ABR Rate Decision Blocks. As a general guideline, this bit would only be set in those ARDBs dictating rates above ICR.

In those ARDBs which have this function enabled, the designer also specifies two other values: ACR_TO (the trigger time for the timeout value when the rate adjustment function activates), and ACR_INDEX (the new lower explicit rate index to be established when the use-it-or-lose-it behavior is triggered).

As a general guideline, ACR_TO should be patterned on the ACR Decrease Time Factor (ADTF), and ACR_INDEX should be patterned on the rate index for ICR.

During run time on those ARDBs with the ACR_ENA bit set, the CN8237 compares RM_TIME (the schedule slot count at the time the last Forward RM cell was generated) with ACR_TO. If the trigger point for the timeout has been reached, the CN8237 generates a new Forward RM cell with the ACR_INDEX value as the assigned new explicit rate.

6.3.8 Boundary Conditions and Out-of-Rate RM Cells

6.3.8.1 Calculated Rate Boundaries

When the system designer pre-calculates the rates for the ARDB tables, those rates must fall within the two obvious upper and lower rate boundaries—the lower rate boundary should not fall below MCR, and the upper rate boundary should not be calculated above PCR.

6.3.8.2 Out-of-Rate Forward RM Cell Generation

A mechanism must exist to restart scheduling of a VCC once the rate on that channel has dropped to 0, or is below the schedule table minimum rate (that is, the rate is less than one schedule slot on the schedule table). Out-of-rate Forward RM cell generation provides this mechanism.

To globally enable out-of-rate Forward RM cell generation, set SCH_ABRBASE(OOR_ENA) to a logic high.

The system designer can set the SET_OOR bit to a logic high in any ABR Rate Decision Block (ARDB). This enables out-of-rate Forward RM cell generation for that rate.

When the actual cell rate on a channel has lowered to the point where scheduling of the VCC has halted, and SET_OOR at that rate = 1, the CN8237 sets the SCH_OOR bit in the VCC's SCH_STATE to a logic high. The CN8237 then generates an out-of-rate Forward RM cell on that channel.

The CN8237 calculates the rate for generation of out-of-rate Forward RM cells as follows:

R = (maximum scheduled rate) = sysclk / SLOT_PER

generated rate = R / (OOR_INT + 1) / (VCC_MAX / 2 + 1)

6.3.8.3 Out-of-Rate Backward RM Cells

The system designer can set the TA_OOR bit to a logic high in any ABR Rate Decision Block. This enables the CN8237 sending a Backward RM cell out-of-rate when there is a pending Turnaround RM cell scheduled at that rate but not yet sent, and another Forward RM cell is received.

6.4 GFC Flow Control Manager

6.4.1 A Brief Overview of GFC

Generic Flow Control (GFC) is a one-way control mechanism which allows the network equipment to control the input from an end station, for the class(es) of traffic defined as controlled. This mechanism does not allow the end station to exert any control on traffic from the network.

GFC is useful because it allows overbooking of the bandwidth on the input side of the network switch buffers. This allows a much higher degree of multiplexing than is otherwise possible, and allows the network-side costs of connections to be significantly reduced. By overbooking the input bandwidth, a high degree of sharing is possible, and the buffer system can be used by more end nodes than full bandwidth input would allow. GFC is used to coordinate access to that bandwidth when temporary conflicts occur.

GFC provides a link-level, short-term, XON/XOFF-type flow control mechanism that only works on the link from the end station to the first piece of network equipment. The GFC protocols are defined and described in *ITU Recommendation I.361*.

6.4.2 The CN8237's Implementation of GFC

The CN8237 implements the GFC one-queue mode. The reassembly coprocessor provides Auto Configure and Command Detection. The segmentation coprocessor provides Halt Processing and Per-transmit Queue SET_A control. It does not implement the optional Queue B.

Once the link has been configured for GFC operation (as described in Section 6.4.2.1), a received HALT indication causes the segmentation coprocessor to halt processing of all channels, both controlled and uncontrolled. This halt condition continues until a cell is received without the HALT indication.

A received SET_A indication increments the GFC credit counter by one. A GFC-controlled cell can be sent only when the GFC credit counter is equal to 1. Transmission of a GFC-controlled cell decrements the credit counter by 1. Each of the eight transmit priority queues can be configured for GFC control by setting the appropriate GFC*n* bit(s) in the Scheduling Priority (SCH_PRI) register. In this way, the SAR can segment both GFC-controlled and GFC-uncontrolled traffic simultaneously. GFC-controlled queues are active only when the GFC credit counter is equal to 1.

CBR traffic is not affected by the SET_A command because it is not mapped into a transmit priority queue. In addition, the segmentation coprocessor implements a credit borrow algorithm that provides better utilization of the line when receive and transmit cell streams are not synchronized. Up to one credit can be borrowed.

The user must control the transmitted GFC field via the HEADER_MOD and GFC_DATA fields in the buffer descriptor entries. For GFC-controlled channels, GFC_DATA = 0101; and for non-GFC-controlled-channels, GFC_DATA = 0001.

6.4.2.1 Configuring the Link for GFC Operation

The following example describes a sequence of how to auto-configure a link for GFC operation after the link has been initialized:

1. Host sets a software GFC initialization timer = 0.
2. Disable reassembly coprocessor by setting RSM_CTRL0(RSM_EN) = 0.
3. Set the framer chip to pass unassigned cells.
4. Enable the GFC link interrupt (GFC_LINK), by setting HOST_IMASK0 (EN_GFC_LINK) = 1.
5. Read HOST_ISTAT0 register twice to clear it.
6. Enable the reassembly coprocessor and set the GFC initialization timer to some user-assigned value.
7. Upon occurrence of an interrupt and before the GFC initialization timer expires, read HOST_ISTAT0. If GFC_LINK is a logic high, continue. If the timer expires before GFC_LINK is detected, do not enable the link for GFC processing.
8. Set SEG_CTRL(SEG_GFC) to a logic high.
9. Set the GFCn bit(s) in the SCH_PRI register to enable the appropriate priority queue(s) for GFC-controlled operation.
10. Set the framer chip to generate unassigned cells with GFC field in cell headers set to the value of 0001.

6.5 Traffic Management Control and Status Structures

6.5.1 Schedule Table

At initialization, all words in the entire Schedule table space should be written to 0xFFFFFFFF. Individual schedule slot entries can then be initialized as either CBR slots or Tunnel slots as needed. The two formats are displayed in Table 6-6.

Table 6-6. Schedule Slot Entry—CBR/Tunnel Traffic

Word	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	CBR	CBR_TUN_ID															(Reserved for 1 VBR/ABR pointer)															
1-7[(1)(2)]	(Reserved for 2 16-bit VBR/ABR pointers)																															

NOTE(S):

(1) Word 1 is present only when the DBL_SLOT field in SEG_CTRL is set and USE_SCH_CTRL is not asserted, meaning two words per schedule slot.

(2) When USE_SCH_CTRL is asserted, the value of SLOT_DEPTH determines how many additional words are used in each schedule slot; from 0 to 7 additional words.

6.5.2 CBR-Specific Structures

6.5.2.1 CBR Traffic

A schedule slot is dedicated to a CBR connection by formatting the CBR bit to a logic high and the CBR_TUN_ID field in the slot entry as illustrated in Table 6-7.

Table 6-7. CBR_TUN_ID Field, Bit Definitions—CBR Slot

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Def.	1	CBR_VCC_INDEX (15 bits)														

6.5.2.2 Tunnel Traffic

A schedule slot is dedicated to a CBR tunnel by formatting the CBR bit to a logic low and the CBR_TUN_ID field of the slot entry as illustrated in Table 6-8. The Schedule Slot field descriptions are detailed in Table 6-9.

Table 6-8. CBR_TUN_ID Field, Bit Definitions—Tunnel Slot

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Def.	0	PRI3				PRI2				PRI1				PRI0		

Table 6-9. Schedule Slot Field Descriptions—CBR Traffic

Field Name	Description
CBR	Set high to indicate a CBR slot; set low to indicate a tunnel slot.
PRI3	Specifies the highest priority of four possible scheduling priority queues to service when a scheduling slot for this CBR tunnel becomes active.
PRI2	Specifies the second highest priority of four possible scheduling priority queues to service when a scheduling slot for this CBR tunnel becomes active.
PRI1	Specifies the third highest priority of four possible scheduling priority queues to service when a scheduling slot for this CBR tunnel becomes active.
PRI0	Specifies the lowest priority of four possible scheduling priority queues to service when a scheduling slot for this CBR tunnel becomes active.
CBR_VCC_INDEX	Segmentation VCC index for dedicated CBR schedule slot. CBR VCC indexes range from 0x0000 to 0x7FFE.
NOTE(S): If the user wants only one priority of traffic scheduled in the CBR tunnel, assign the priority level to PRI3, and make PRI2 the same priority. PRI1 and PRI0 values are Don't Cares in this case. If two priorities are to be assigned to the tunnel, assign the higher priority to PRI3 and the lower priority to PRI2, making PRI1 the same as PRI2. PRI0 is a Don't Care in this case. If three priorities are to be assigned to the tunnel, assigns the priorities by level to PRI3 (highest), PRI2 and PRI1 (lowest), making PRI0 the same as PRI1. TUN_PRI0_OFFSET can be used to set PRI0 = PRI1, as needed.	

6.5.2.3 SCH_STATE Fields For CBR

This section specifies the SCH_STATE portion (words 7–9) of the Segmentation VCC table entry, when SCH_MODE = CBR.

The CBR SCH_STATE is shown in Table 6-10, and the field descriptions are detailed in Table 6-11.

Table 6-10. SCH_STATE for SCH_MODE = CBR

Word	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
7	Reserved																	TUN_PRI_3				TUN_PRI_2				TUN_PRI_1				TUN_PRI_0		
8	Reserved																															
9	Reserved																															

Table 6-11. CBR SCH_STATE Field Descriptions

Field Name	Description
TUN_PRI_3	When SCH_MODE is CBR and CBR_W_TUN bit is set to 1, this field specifies the highest priority of four possible scheduling priority queues to service in place of the unused CBR slot. (This field is not active when CBR_W_TUN = 0.)
TUN_PRI_2	When SCH_MODE is CBR and CBR_W_TUN bit is set to 1, this field specifies the second highest priority of four possible scheduling priority queues to service in place of the unused CBR slot. (This field is not active when CBR_W_TUN = 0.)
TUN_PRI_1	When SCH_MODE is CBR and CBR_W_TUN bit is set to 1, this field specifies the third highest priority of four possible scheduling priority queues to service in place of the unused CBR slot. (This field is not active when CBR_W_TUN = 0.)
TUN_PRI_0	When SCH_MODE is CBR and CBR_W_TUN bit is set to 1, this field specifies the lowest priority of four possible scheduling priority queues to service in place of the unused CBR slot. (This field is not active when CBR_W_TUN = 0.)

6.5.3 VBR-Specific Structure

6.5.3.1 VBR SCH_STATE

Table 6-12 and 6-13 define the SCH_STATE part of the Segmentation VCC table entry, which consists of words 7–9 for VBR.

See Section 6.2.4 for key data on *I* and *L* values.

6.5.3.2 VBR1 or VBR2 Schedule State Table

Table 6-12. SCH_STATE for SCH_MODE = VBR1 or VBR2

Word	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
7	BUCKET2 (MSB)				L1_EXP					L1_MAN									I1_EXP					I1_MAN							
8	BUCKET2 (LSB)				Reserved																											
9	Reserved																															

Table 6-13. VBR1 and VBR2 SCH_STATE Field Descriptions

Field Name	Description
BUCKET2	Index into Bucket table to give I2 and L2 for SCHED_MODE = VBR2 and VBRC. Bucket table base is given by SEG_BCKB register field. Entries in Bucket table have same format as L1_EXP, L1_MAN, I1_EXP, and I1_MAN.
L1_EXP	GCRA L parameter exponent for bucket 1. L1_EXP must satisfy L1_EXP ≤ min(I1_EXP + 9 or 29). L1_EXP that does not satisfy L1_EXP ≥ I1_EXP – 9 will have an effective L1 value of 0.
L1_MAN	GCRA L parameter mantissa for bucket 1. Bucket 1 L value is $2^{(L1\_EXP - 10)}(1+L1\_MAN / 512)$.
I1_EXP	GCRA I parameter exponent for bucket 1. I1_EXP must satisfy 10 ≤ I1_EXP ≤ 25.
I1_MAN	GCRA I parameter mantissa for bucket 1. Bucket 1 I value is $2^{(I1\_EXP - 10)}(1+I1\_MAN / 512)$.
NOTE(S): The GCRA I value is expressed in cell slot intervals. Example (I is not expressed in cells/sec.)	$I = \frac{1}{PCR} \times R_{MAX}$

6.5.3.3 Bucket Table for VBR2 and VBRC

The Bucket table has only 256 entries. Table 6-14 and 6-15 display the entry format and field descriptions for the Bucket table.

Table 6-14. Bucket Table Entry

Word	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	Reserved				L2_EXP					L2_MAN									I2_EXP					I2_MAN									

Table 6-15. Bucket Table Entry Field Descriptions

Field Name	Description
L2_EXP	GCRA L parameter exponent for bucket 2. L2_EXP must satisfy L2_EXP ≤ min(I2_EXP + 9 or 29). L2_EXP that does not satisfy L2_EXP ≥ I2_EX – 9 will have an effective L2 value of 0.
L2_MAN	GCRA L parameter mantissa for bucket 2. Bucket 2 L value is $2^{(L2\_EXP - 10)}(1 + L2\_MAN / 512)$.
I2_EXP	GCRA I parameter exponent for bucket 2. I2_EXP must satisfy 10 ≤ I2_EXP ≤ 25.
I2_MAN	GCRA I parameter mantissa for bucket 2. Bucket 2 I value is $2^{(I2\_EXP - 10)}(1 + I2\_MAN / 512)$.

6.5.4 GFR-Specific Structures

6.5.4.1 GFR Schedule State Table

Tables 6-16 and 6-17 detail the SCH_STATE structure for GFR traffic.

Table 6-16. SCH_STATE for SCH_MODE = GFR

Word	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
7	MCRLIM_IDX (MSB)				L1_EXP					L1_MAN									I1_EXP					I1_MAN									
8	MCRLIM_IDX (LSB)				Reserved																											
9	Reserved				Reserved														Reserved					Reserved									

Table 6-17. SCH_STATE Field Descriptions for SCH_MODE = GFR

Field Name	Description
MCRLIM_IDX	Index into table of 256 MCR LIMIT values each specifying an increase in the VCC's priority by one if the VCC's rate falls below MCR. The table base is located on top of the BUCKET2 table whose base is given by SEG_BCKB register field. The BUCKET2 table is 1 KB in length. Table values are in the form of a non-0 bit, exponent, and mantissa such that MCR_LIMIT is: $MCR\_LIM\_NZ \times 2^{(MCR\_LIM\_EXP-10)}(1 + MCR\_LIM\_MAN / 512)$
L1_EXP	GCRA L parameter exponent for MCR bucket. L1_EXP must satisfy .1_EXP ≤ min(I1_EXP + 9 or 29) . LI_EXP that does not satisfy L1_EXP ≥ I1_EXP – 9 will have effective L1 value of 0.
L1_MAN	GCRA L parameter mantissa for MCR bucket Bucket 1 L value is $2^{(L1\_EXP-10)}(1 + L1\_MAN / 512)$
I1_EXP	GCRA I parameter exponent for MCR bucket. I1_EXP must satisfy 10 ≤ I2_EXP ≤ 25
I1_MAN	GCRA I parameter mantissa for MCR bucket. Bucket 1 I value is $2^{(I1\_EXP-10)}(1 + I1\_MAN / 512)$

6.5.4.2 GFR MCR Limit Bucket Table

The 128 words of this table contain 256 MCR Limit values, two bucket table entries per word. The table base is on top of the Bucket2 table. Table values are in the form of a non-0 bit, exponent and mantissa such that MCR_LIMIT is

$$MCR\_LIM\_NZ \times 2^{(MCR\_LIM\_EXP - 10)}(1 + MCR\_LIM\_MAN / 512)$$

Tables 6-18 and 6-19 describe the MCR Limit Bucket table.

Table 6-18. GFR MCR Limit Bucket Table Entry

Word	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	Reserved	NONZERO	MCR_EXP					MCR_MAN									Reserved	NONZERO	MCR_EXP					MCR_MAN								

Table 6-19. GFR MCR Bucket Table Entry Field Descriptions

Field Name	Description
MCR_EXP	GFR MCR limit exponent. MCR_EXP must satisfy MCR_EXP ≤ 29.
MCR_MAN	GFR MCR limit mantissa. MCR limit is: NONZERO × $2^{(MCR\_EXP-10)}$(1 + MCR_MAN / 512). VCC priority is increased by 1 when rate falls below 1/MCR limit.
NONZERO	GFR MCR limit is non-0.

6.5.5 ABR-Specific Structures

6.5.5.1 ABR Schedule State Table

Table 6-20 describes the SCH_STATE entries in the segmentation VCC table for the ABR service class. Table 6-21 details the field descriptions for the ABR SCH_STATE fields.

Table 6-20. SCH_STATE for SCH_MODE = ABR

Word	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
7	Reserved				L_EXP					L_MAN									I_EXP					I_MAN									
8	Reserved				PRESENT																											
9	Reserved	MCR_LIM_NZ	DELTA_SIGN	DELTA_NZ	MCR_LIM_EXP					MCR_LIM_MAN									DELTA_EXP					DELTA_MAN									
10	EN_SRC_NCR	S_NCR_TRIG	S_NCR_DIR	Rsvd		OOR_PRI				DCR_STAT	TM_EXP	BCK_OOR	FWD_OOR	SCH_OOR	TA_XMIT	TA_PND	CELL_INDEX															
11	FWD_INDEX																RM_TIME															
12	RATE_INDEX																EB_INDEX															
13	CRM																UNACK															
14	Rsvd		S_NRC_LO_EXP					S_NRC_LO_MANT									NEXT_OOR															
15	MCR_INDEX																ICR_INDEX															
16[1]	TA_ID								TA_DIR	TA_BN	TA_CI	TA_NI	TA_RA	D_NCR_DIR	Rsvd		TA_ER															
17[1]	TA_CCR																TA_MCR															
18	FWD_ID								FWD_DIR	FWD_BN	FWD_CI	FWD_NI	FWD_RA	Reserved			FWD_ER															
19	Rsvd		S_NCR_HI_EXP					S_NCR_HI_MANT									FWD_MCR															

NOTE(S):

(1) Words 16 and 17 are written directly by the RSM coprocessor (turnaround information).

KEY:

= Values are furnished by the ABR Templates.

Table 6-21. ABR SCH_STATE Field Descriptions *(1 of 2)*

Field Name	Description
L_EXP	GCRA L parameter exponent for ER rate.
L_MAN	GCRA L parameter mantissa for ER rate.
I_EXP	GCRA I parameter exponent for ER rate.
I_MAN	GCRA I parameter mantissa for ER rate.
PRESENT	Current Schedule table position.
MCR_LIM_NZ	MCR Limit is non-0. To disable priority bumping, MCR_LIMIT must be set to its maximum value. Thus, MCR_LIM_NZ must be set to 1.
DELTA_SIGN	Delta is negative for ER rate.
DELTA_NZ	Delta is non-0 for ER rate.
MCR_LIM_EXP	MCR Limit exponent. MCR_LIM_EXP must satisfy MCR_LIM_EXP ≤ 29. To disable priority bumping, MCR_LIMIT must be set to its maximum value. Thus, MCR_LIM_EXP must be set to a decimal value of 31 (that is, binary all-1s).
MCR_LIM_MAN	MCR Limit mantissa. MCR_LIMIT is $MCR\_LIM\_NZ \times 2^{(MCR\_LIM\_EXP-10)} (1 + MCR\_LIM\_MAN / 512)$ This format for calculating MCR_LIMIT is the same as used with the GCRA *I* parameter to calculate rates. VCC priority is increased by 1 when rate falls below 1 / MCR_LIM. To disable priority bumping, MCR_LIMIT must be set to its maximum value. Thus, MCR_LIM_MAN must be set to a decimal value of 511 (that is, binary all-1s).
DELTA_EXP	Delta exponent for ER rate.
DELTA_MAN	Delta mantissa for ER rate. Desired position for ER rate is $DELTA\_NZ \times 2^{(DELTA\_EXP-10)} (1+DELTA\_MAN / 512) + PRESENT$
EN_SRC_NCR	Enable source ACR change notification processing.
S_NCR_TRIG	Source ACR change has been triggered. Initialize to logic flow.
S_NCR_DIR	Indicate direction of source ACR trigger. 1 for HI, 1 for low.
OOR_PRI	Segmentation priority for out-of-rate RM cells.
DCR_STAT	Destination ACR change status to be sent. Initialize to logic low.
TM_EXP	RM_TIME value has expired and is no longer valid.
BCK_OOR	Backward RM cell has been scheduled out-of-rate.
FWD_OOR	Forward RM cell has been scheduled out-of-rate.
SCH_OOR	VCC has halted due to low ACR and has out-of-rate Forward RM cells enabled.
TA_XMIT	A Backward RM cell has been transmitted after the last Forward RM cell transmitted.
TA_PND	A Backward RM cell is waiting for transmission.
CELL_INDEX	ER cell type decision block index for cell type decisions. The cell type decision block is located at byte address SCH_ABRB × 128 + 8 × CELL_INDEX.
FWD_INDEX	ER cell type decision block index for cell type decisions after a Forward RM cell is transmitted.
RM_TIME	Global slot count [21:6] at time of last Forward RM transmission.

***Table 6-21. ABR SCH_STATE Field Descriptions** (2 of 2)*

Field Name	Description
RATE_INDEX	ER rate decision block index. The rate decision block is located at byte address SCH_ABRB × 128 + 32 × RATE_INDEX.
EB_INDEX	ER exponent table index for rate index block. The exponent table is located at byte address SCH_ABRB × 128 + EB_INDEX × 128.
CRM	ER parameter.
UNACK	Number of Forward RM cells transmitted since last Backward RM cell received. Initialize to 0.
S_NCR_LO_EXP	Source ACR lower threshold, exponent portion.
S_NCR_LO_MANT	Source ACR lower threshold, mantissa portion.
NEXT_OOR	Next VCC index for linking out-of-rate RM cells.
MCR_INDEX	Minimum Cell Rate decision block index. The rate decision block is located at byte address SCH_ERB × 128 + 32 × MCR_INDEX.
ICR_INDEX	Initial Cell Rate decision block index. The rate decision block is locate at byte address SCH_ERB × 128 + 32 × ICR_INDEX.
TA_ID	ID field from most recent received Forward RM cell. This field is written by the SAR.
TA_DIR	DIR field for turnaround (Backward) RM cell. This field is written by the SAR.
TA_BN	BN field for turnaround (Backward) RM cell. This field is written by the SAR.
TA_CI	CI field for turnaround (Backward) RM cell. This field is written by the SAR.
TA_NI	NI field for turnaround (Backward) RM cell. This field is written by the SAR.
TA_RA	RA field from most recent received Forward RM cell. This field is written by the SAR.
D_NCR_DIR	Indicates direction of destination ACR trigger. 1 for HI, 0 for low.
TA_ER	ER field for turnaround (Backward) RM cell. This field is written by the SAR.
TA_CCR	CCR field from most recent received Forward RM cell. This field is written by the SAR.
TA_MCR	MCR field from most recent received Forward RM cell. This field is written by the SAR.
FWD_ID	ID field for transmitted Forward RM cells. This field is supplied by the user.
FWD_DIR	DIR field for transmitted Forward RM cells. This field is supplied by the user.
FWD_BN	BN field for transmitted Forward RM cells. This field is supplied by the user.
FWD_CI	CI field for transmitted Forward RM cells. This field is supplied by the user.
FWD_NI	NI field for transmitted Forward RM cells. This field is supplied by the user.
FWD_RA	RA field for transmitted Forward RM cells. This field is supplied by the user.
FWD_ER	ER field for transmitted Forward RM cells. This field is supplied by the user.
S_NCR_HI_EXP	Source ACR upper threshold, exponent portion.
S_NCR_HI_MANT	Source ACR upper threshold, mantissa portion.
FWD_MCR	MCR field for transmitted Forward RM cells. This field is supplied by the user.

6.5.6 RS_QUEUE

Table 6-22. RS_QUEUE Entry—OAM-PM Reporting Information Ready for Transmission

Word	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	0000 OAM_PM				Reserved				BLER								SEG_VCC_INDEX															
1	BCK_TUC0																BCK_TUC01															
2	TRCC0																TRCC0+1															
3	Reserved																															

Table 6-23. RS_QUEUE Entry—Forward ER RM Cell Received

Word	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	0100 ER_FWD				Reserved												SEG_VCC_INDEX															
1	Reserved																															

Table 6-24. RS_QUEUE Entry—Backward ER RM Cell Received

Word	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	0101 ER_BCK				Reserved												SEG_VCC_INDEX															
1	Reserved									BN	CI	NI	Reserved				ER															

Table 6-25. RS_QUEUE Field Descriptions

Field Name	Description
OAM_PM	OAM PM backward reporting information ready for transmission.
BLER	Block Error Result.
SEG_VCC_INDEX	Segmentation VCC index for backward reporting OAM PM cell.
BCK_TUC0	TUC0 field in received forward monitoring cell. Written directly from Forward_RM cell.
BCK_TUC01	TUC01 field in received forward monitoring cell. Written directly from Forward_RM cell
TRCC0	Total Received Cell Count with CLP = 0.
TRCC0+1	Total Received Cell Count with CLP = 0 + 1.
ER_FWD	Forward ER RM cell received.
SEG_VCC_INDEX	Segmentation VCC index for transmitted Backward ER RM cell.
ER_BCK	Backward ER RM cell received.
BN	Backward Explicit Congestion Notification bit from received RM cell.
CI	Congestion Indication bit from received RM cell.
NI	No Increase bit from received RM cell.
ER	Explicit Cell Rate field from received RM cell.

6.5.7 Scheduler Internal SRAM Registers

NOTE: FOR SAR INTERNAL USE ONLY!

Application program should ignore these registers.

Scheduler SRAM registers are located in the address range 0x1640–0x17FF. Figure 6-23 shows the layout for the head and tail pointers. Table 6-26 describes the memory map of these registers.

Figure 6-23. Head and Tail Pointers

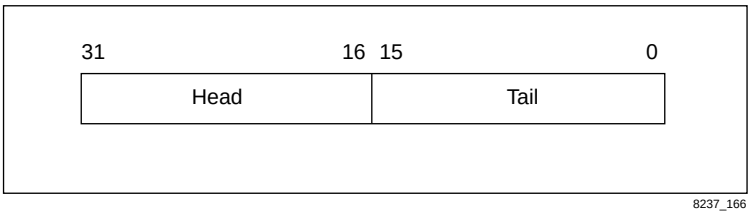


Table 6-26. Scheduler Internal SRAM Memory Map (Head/Tail Pointers)

Address	Name	Description
0x1640-0x1643	PRI_PNTR0	Global priority 0.
0x1644-0x1647	PRI_PNTR1	Global priority 1.
0x1648-0x164B	PRI_PNTR2	Global priority 2.
0x164C-0x164F	PRI_PNTR3	Global priority 3.
0x1650-0x1653	PRI_PNTR4	Global priority 4.
0x1654-0x1657	PRI_PNTR5	Global priority 5.
0x1658-0x165B	PRI_PNTR6	Global priority 6.
0x165C-0x165F	PRI_PNTR7	Global priority 7.
0x1660-0x1663	PRI_PNTR8	Global priority 8.
0x1664-0x1667	PRI_PNTR9	Global priority 9.
0x1668-0x166B	PRI_PNTR10	Global priority 10.
0x166C-0x166F	PRI_PNTR11	Global priority 11.
0x1670-0x1673	PRI_PNTR12	Global priority 12.
0x1674-0x1677	PRI_PNTR13	Global priority 13.
0x1678-0x167B	PRI_PNTR14	Global priority 14.
0x167C-0x167F	PRI_PNTR15	Global priority 15.
0x1680-0x17FF	Reserved	Not implemented.

7.0 OAM Functions

7.1 OAM Overview

OAM cells are ATM Layer management messages. These are generated by the host on the segmentation side of the CN8237, and are detected and either monitored or processed on the reassembly side of the CN8237.

ATM's OAM capabilities differentiate it from other less robustly managed communication technologies. The CN8237 provides internal support for the detection and generation of OAM traffic, including Performance Monitoring (PM) OAM.

The CN8237 supports the F4 and F5 OAM flows according to *ITU-T Recommendation I.610*. It also monitors the performance of up to 128 channels, generating PM-OAM cells according to the same specification.

7.1.1 OAM Functions Supported

Refer to *ITU-T Recommendation I.610* for complete information on the structures and functions of OAM cell generation, detection, and processing.

Table 7-1 lists the OAM functions of the F4 and F5 OAM flows, as defined in *ITU-T Recommendation I.610*.

Table 7-1. OAM Functions of the ATM Layer

OAM Function	Main Application
AIS (Alarm Indication Signal)	Reports defect indications in the forward direction.
RDI (Remote Defect Indication)	Reports remote defect indications in the backward direction.
CC (Continuity Check)	Continuously monitors the continuity of a connection. A continuity cell is used periodically to check whether a connection is idle or has failed.
Loopback	Used for • on-demand connectivity monitoring • fault localization • pre-service connectivity verification
PM (Performance Monitoring)	Estimates performance and reports performance estimations in the backward direction.
Activation/Deactivation	Activating/deactivating performance monitoring and continuity check.
System Management	Used by end-systems only.

The CN8237 internally supports the following functions as described in *ITU-T Recommendation I.610*:

- Full Performance Monitoring functions (designed for estimating transmission performance on any channel, and for reporting performance estimations in the backward direction)
- Detection of OAM cells, and routing them as directed by the host
- Generation of OAM cells, as directed by the host

Implementation of the full range of functions in processing OAM cells are done at the software level due to the low bandwidth of OAM traffic (less than 1% of the bandwidth of active connections).

7.1.2 OAM Flows Supported

7.1.2.1 F4 OAM Flow

The F4 OAM flow is the Virtual Path level, provided by OAM cells dedicated to ATM layer OAM functions for Virtual Path Connections (VPCs). The F4 flow is bidirectional.

There are two kinds of F4 OAM flows which can simultaneously exist in a VPC:

- End-to-end F4 flow (used for end-to-end VPC operations communications)
- Segment F4 flow (used for communicating operations information within the bounds of one VPC link, such segment having its source and sink defined by the user)

OAM cells for the F4 flow have the same VPI value as the user cells of the VPC.

F4 flow OAM cells are identified as F4 flow cells by a pre-assigned VCI value of three (segment flow cell) or four (end-to-end flow cell) as shown in Table 7-2. The same pre-assigned VCI value is used for both directions of the F4 flow.

Table 7-2. VCI Values for F4 OAM Flows

VCI Value	Interpretation
3	Segment OAM F4 flow cell
4	End-to-end OAM F4 flow cell

7.1.2.2 F5 OAM Flow

The F5 OAM flow is the Virtual Channel level, provided by OAM cells dedicated to ATM layer OAM functions for VCCs. The F5 flow is bidirectional.

There are two kinds of F5 OAM flows which can simultaneously exist in a VCC:

- End-to-end F5 flow (used for end-to-end VCC operations communications)
- Segment F5 flow (used for communicating operations information within the bounds of one VCC link, such segment having its source and sink defined by the user)

OAM cells for the F5 flow have the same VPI value and VCI value as the user cells of the VCC.

F5 flow OAM cells are identified as F5 flow cells by a pre-assigned PTI code value of 100 (segment flow cell) or 101 (end-to-end flow cell) and shown in Table 7-3. The same pre-assigned PTI value is used for both directions of the F5 flow.

Table 7-3. PTI Values for F5 OAM Flows

PTI Code	Interpretation
100	Segment OAM F5 flow cell
101	End-to-end OAM F5 flow cell

7.1.2.3 Performance Monitoring (PM)

Performance Monitoring includes a set of functions that monitor and process user information on a channel to produce maintenance information specific to that channel. This maintenance information is added to the in-rate data flow on that channel in the form of PM cells, added at the source of a connection or link, and extracted at the sink of a connection or link. With this maintenance information, the user can estimate and analyze the transport integrity of that channel.

The PM flow is bidirectional, and PM cells are of two basic function types: forward monitoring cells (which carry the forward error detection information), and backward reporting cells (which carry the results of the performance monitoring checks).

The CN8237 performs PM on up to 128 user-assigned channels. The SAR enables PM for a channel by setting the PM enable bits (PM_EN) in the Segmentation and Reassembly VCC State table entries for that channel.

The CN8237 performs PM processing on any channel by monitoring blocks of user cells on that channel. The size of this block of cells is set by the user, and can have the value (N) of 128, 256, 512, or 1,024 cells. The CN8237 inserts a PM cell after every N user cells on that channel. A block size of 0 is valid when the SAR is a destination point ONLY for PM processing. In this case, the segmentation coprocessor will only generate Backward reporting cells in response to reassembly performance monitoring calculations.

PM as performed by the CN8237 calculates the following:

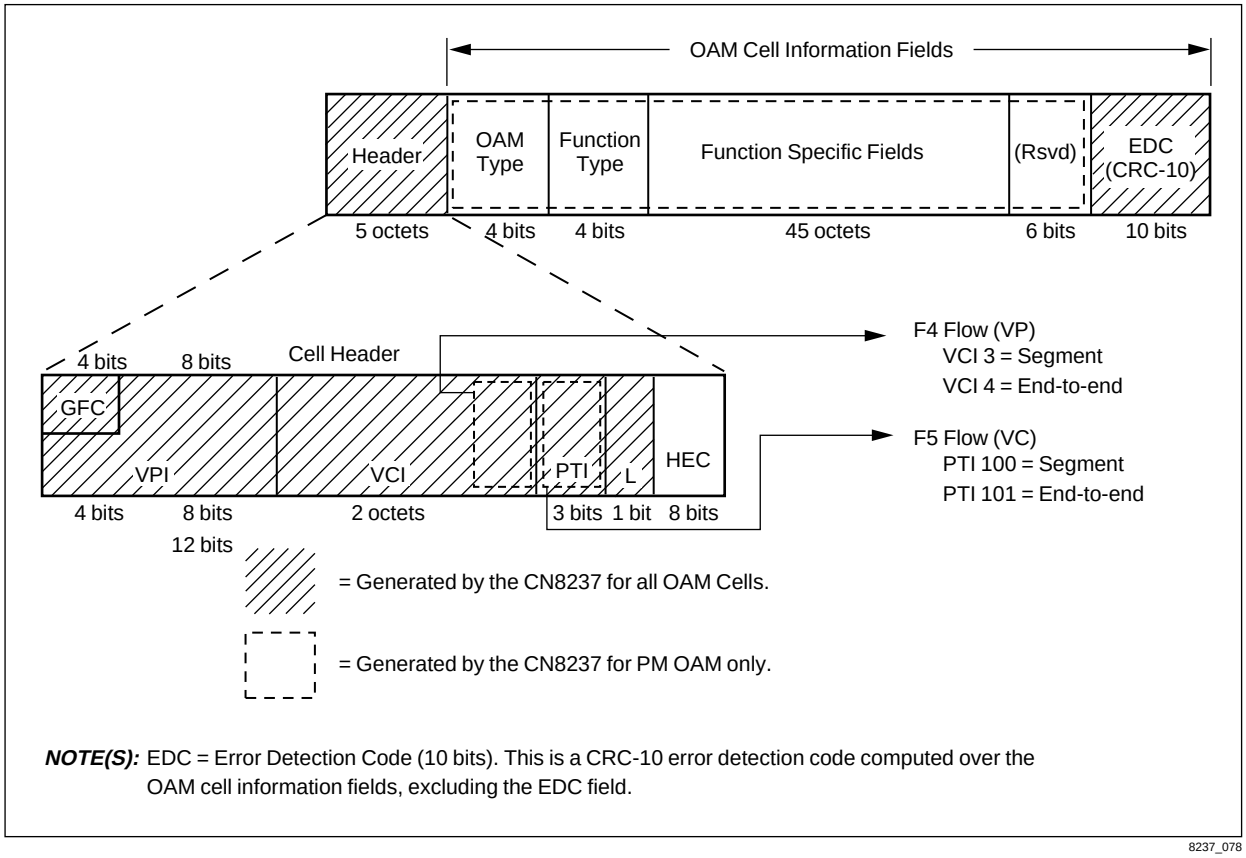
- Errored blocks (by means of a BIP-16 error detection code generated over the payloads of the user cells in the PM block)
- A count of misinserted PM forward monitoring cells

The user can activate PM on any channel either during connection establishment, or at any time after the connection has been established.

7.1.3 OAM Cell Format

Figure 7-1 illustrates the common OAM cell format and identifies the fields specific to OAM.

Figure 7-1. OAM Cell Format



NOTE(S): EDC = Error Detection Code (10 bits). This is a CRC-10 error detection code computed over the OAM cell information fields, excluding the EDC field.

8237_078

The OAM Type and Function Type identifiers as specified by *ITU-T Recommendation I.610* are listed in Table 7-4.

Table 7-4. OAM Type and Function Type Identifiers

OAM Type(1)	Coding	Function Type(2)	Coding
Fault Management	0001	AIS RDI Continuity Check Loopback	0000 0001 0100 1000
Performance Management	0010	Forward monitoring Backward reporting	0000 0001
Activation/Deactivation	1000	Performance Monitoring Continuity Check	0000 0001

NOTE(S):
(1) OAM Type indicates the type of management function performed by this cell; for example, fault management, performance management, etc.
(2) Function Type indicates the actual function performed by this cell within the management type indicated by the OAM Type field.

7.1.4 Global Queue Processing of OAM

ATM carries with it significant network management overhead. The CN8237 supports those robust OAM features described previously. Due to the breadth of ATM applications and the continuing evolution of ATM management standards, it is essential that a range of flexibility be provided in the processing of network management overhead.

To accomplish this, the CN8237 provides global OAM buffer and status queues (OAM_BFR_QU and OAM_STAT_QU, addresses for both assigned in the RSM_CTRL1 register). If the OAM_QU_EN bit in the RSM_CTRL1 register is set to a logic high, then the CN8237 routes OAM traffic to these global queues. In addition, the global OAM segmentation status queue, SEG_CTRL(OAM_STAT_ID), is used if the OAM_STAT bit in the segmentation buffer descriptor is a logic high.

7.2 Segmentation of OAM Cells

The host places OAM cells in a single buffer, and thus allocates a single segmentation buffer descriptor (SBD) for the OAM cell data buffer, not a linked list of SBDs.

The host then writes a pointer to that SBD in the next available transmit queue entry, and sets the VLD bit to 1.

When the segmentation coprocessor processes that transmit queue entry, it submits the OAM cell data buffer to the xBR Traffic Manager and the cell is thus scheduled for transmission.

7.2.1 Key OAM-Related Fields for OAM Segmentation

7.2.1.1 Segmentation Buffer Descriptors

Several fields in the SBD entry are used to facilitate segmentation of OAM cells.

- Set the 2-bit AAL_OPT field to SINGLE (value = 01). This enables reading 48 octets from a single buffer to form a single ATM cell.
- Set the OAM_STAT bit to a logic high. The CN8237 now reports status to the OAM-dedicated OAM_STAT_ID identified in the SEG_CTRL register, instead of the STAT specified in the SEG VCC table entry.
- Set the single-bit HEADER_MOD field to a logic 1. This activates the WR_PTI and WR_VCI bits in the buffer descriptor, which signal the CN8237 to overwrite the ATM header PTI and VCI fields for that cell with the values from the PTI_DATA and VCI_DATA fields. In this way, F4 and F5 flow OAM cells can be generated by the CN8237.
- The VCI_DATA field set to a value of three (segment cell) or four (end-to-end cell) generates an F4 flow OAM cell.
- The PTI_DATA field set to a value of 100 (segment cell) or 101 (end-to-end cell) generates an F5 flow OAM cell.
- Set the AAL_MODE field to 01 (AAL0).
- Set both BOM and EOM bits to 0.
- Set the CRC10 bit to a logic high.

7.2.1.2 Low Latency Transmission

For low latency, the LINK_HEAD bit in the transmit queue entry should be set to a logic high. This tells the CN8237 to link the buffer chain at the head of the existing chain for the corresponding VCC. This bit is intended for use with the SEG buffer descriptor's SINGLE option, to send in-line OAM cells. Only a single SEG buffer descriptor may be linked to a transmit queue entry when this bit is set.

This bit must also be set if the OAM SBD is placed on the transmit queue after a partial PDU, to ensure correct segmentation.

7.2.1.3 Segmentation Status Queue

The SINGLE bit in the SEG status queue entry should be set to a logic high. This bit is set if the SINGLE option in the AAL_OPT field of the SEG buffer descriptor is set. This bit indicates a special buffer is in use, rather than the normal system-assigned buffers for normal CPCS-PDUs.

7.2.1.4 F4 Flow

For F4 flow operation, a separate VCC table entry must be configured.

7.2.2 Error Condition During OAM Segmentation

Each OAM cell has a 10-bit Error Detection Code (EDC) field, for storing and transporting the calculated CRC-10 error detection code results (computed over the OAM cell information fields, excluding the EDC field). To enable this CRC-10 function, set the CRC10 bit in the SEG buffer descriptor entry to a logic high.

7.3 Reassembly of OAM Cells

To enable the reassembly coprocessor to detect and therefore further process OAM cells, set the OAM_EN bit in RSM_CTRL1 to a logic high.

The CN8237 detects the following OAM cell flows:

- Segment F4 Flow
- End-to-end F4 Flow
- Segment F5 Flow
- End-to-end F5 Flow
- PTI = 6
- PTI = 7

The CN8237 provides global OAM buffer and status queues (OAM_BFR_QU and OAM_STAT_QU, addresses for both assigned in the RSM_CTRL1 register). To activate these queues, set the OAM_QU_EN bit in the RSM_CTRL1 register to a logic high. The CN8237 then routes OAM traffic to these global buffer and status queues.

NOTE: The cell buffer size must be large enough to hold a complete cell when OAM detection is enabled.

7.3.1 Key OAM-Related Fields for OAM Reassembly

7.3.1.1 Reassembly VCC State Table

The reassembly VCC State table field, SEG_VCC_INDEX, should be written to point to the channel index of the corresponding segmentation channel. This is necessary for PM-OAM, as well as ABR channels.

7.3.1.2 Reassembly Status Queue

The 3-bit OAM field in the RSM status queue entry must be set to the value indicated in the RSM status queue structure description for that field. A non-0 value in the OAM field indicates that the cell is an OAM cell.

If the CRC-10 error detection code computation on the OAM cell shows an error, the CN8237 sets the CRC_ERROR bit in the status queue entry to a logic high.

7.3.1.3 F4 Flow

For F4 flow operation, a separate VCC table entry must be configured.

7.3.2 OAM Reassembly Operation

Received OAM traffic should be detected and routed to the global OAM buffer and status queues (OAM_BFR_QU and OAM_STAT_QU).

The reassembly coprocessor treats OAM cells as one-cell PDUs. The RSM coprocessor transfers the 48-octet OAM payload to the next available global data buffer, and writes a RSM status queue entry.

Once an OAM cell is detected, the RSM coprocessor checks the cell to determine whether it is a PM cell, by seeing if the OAM_TYPE field in the cell is set to value 0010. That PM detection is performed only on F4 and F5 OAM cells.

If the RSM coprocessor finds the cell is a PM cell, it is processed following the guidelines described in Section 7.4.

NOTE: OAM cells that get buffers from the global OAM free buffer queue do not effect the per-VCC firewall calculation.

7.3.3 Error Conditions During OAM Reassembly

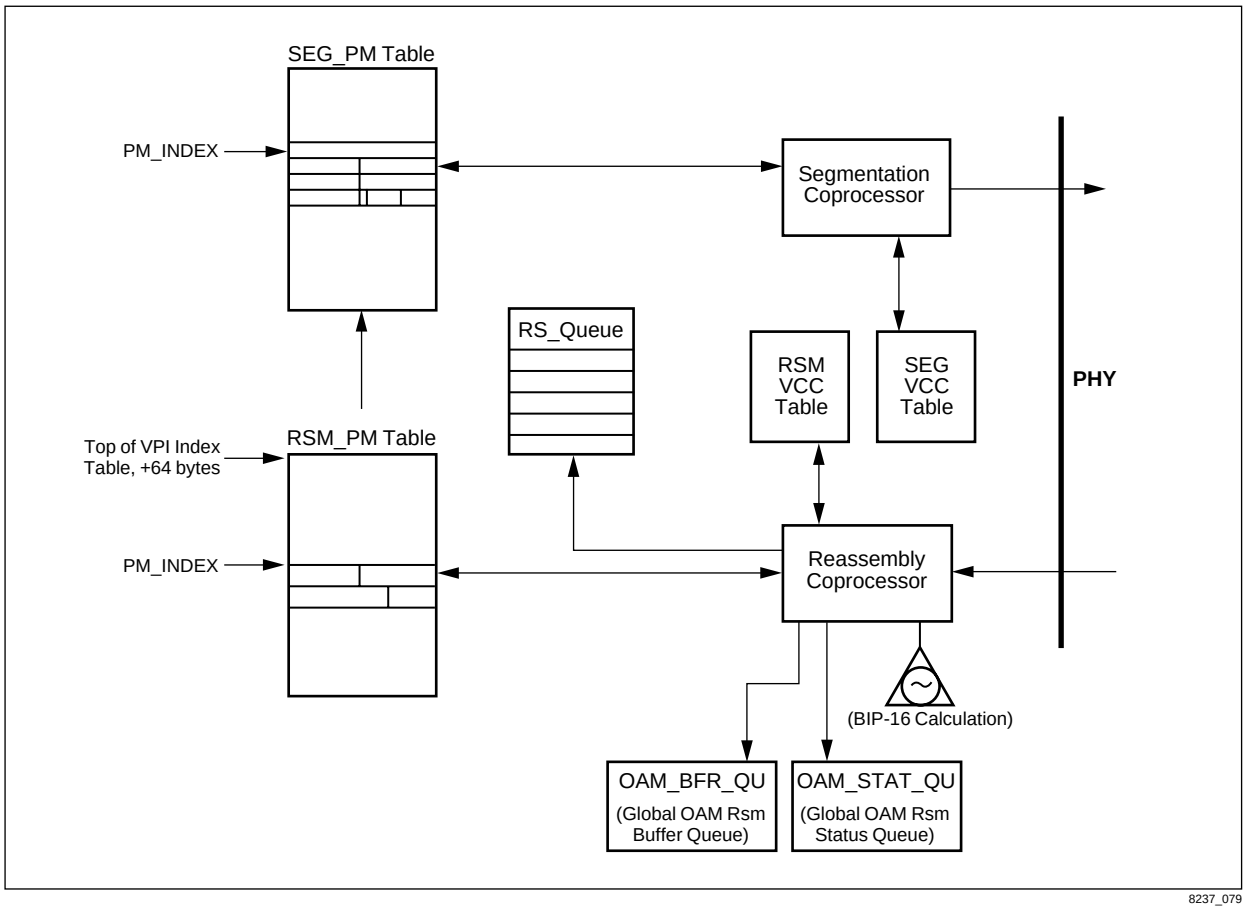
The RSM coprocessor computes the CRC-10 error detection code for each received OAM cell's information fields. If an error is detected (for example, the computed CRC value does not match the CRC-10 value written in the cell), the SAR sets the CRC_ERROR bit in the status queue entry to a logic high.

7.4 PM Processing

OAM Performance Monitoring may be enabled for up to 128 VCCs by setting the PM_EN bits in the RSM VCC table entry and the SEG VCC table entry for that channel.

Figure 7-2 illustrates the functional blocks for PM processing.

Figure 7-2. Functional Blocks for PM Segmentation and Reassembly



For any channel on which PM processing is enabled, the CN8237 performs these functions:

- The RSM coprocessor reassembles received PM-OAM cells in the global reassembly OAM buffer queue (OAM_BFR_QU).
- A reassembly status record is written to the global reassembly OAM status queue, for each received PM-OAM cell. The status entry for a forward monitoring PM-OAM cell includes the Block Error Result (BIPV) calculation, and both Total Received Cell Counts (TRCC0 and TRCC0+1). The two Total User Cell number fields (TUC0 and TUC0+1) can be extracted directly from the cell payload.
- The RSM coprocessor performs the BIP-16 calculation on each received data cell in the defined PM block, and writes this data to the RSM_PM table entry set aside for that PM_INDEX.
- For each forward monitoring PM cell received, the RSM coprocessor writes an entry for a backward reporting PM cell in the RS_Queue, causing the CN8237 to generate and transmit a backward reporting PM cell.
- The segmentation coprocessor automatically generates a forward monitoring PM cell at the end of each PM block. It gets the data for these cells from the SEG_PM table entry for that PM_INDEX. This can be optionally disabled by setting the FWD_MON field equal to 0.

7.4.1 Initializing PM Operation

The user must initialize the fields described in Table 7-5 before starting PM processing:

Table 7-5. PM-OAM Field Initialization For Any PM_INDEX

Register / Table	Field	Initialized Value	Notes
RSM_CTRL1 (Reassembly Control Register 1)	OAM_EN	0 – 1	—
	OAM_QU_EN	0 – 1	—
	OAM_BFR_QU	(User assigned)	—
	OAM_STAT_QU	(User assigned)	—
SEG_PMBASE (SEG PM Base Register)	SEG_PMB[15:0]	(User assigned)	Base address for SEG_PM table.

7.4.2 Setting Up Channels for PM Operation

When the CN8237 receives a PM activation cell (OAM Type = 1000, Function Type = 0000), or when the user decides to activate PM processing on a channel, the host (or local) processor must enable PM on the applicable channel by setting the PM_EN bits in the RSM and SEG VCC table entries to logic high, and selecting an unused PM_INDEX (0–127).

The host then initializes the corresponding SEG_PM and RSM_PM table entries. The format of each entry of the SEG_PM table is illustrated in Table 7-6, while the format of each entry of the RSM_PM table is illustrated in Table 7-8.

The assigned PM_INDEX value for that channel has to be written to both the RSM VCC table entry and the SEG VCC table entry for that channel.

For F4 flows, each VCI channel in the VPI group must have the PM_EN bits in both the RSM VCC table entry and the SEG VCC table entry set high, and the PM_INDEX pointing to the same location. In addition, a RSM and SEG VCC table entry must be configured corresponding to VCI=3 or VCI=4.

To initialize backward reporting without forward monitoring on any channel, set FWD_MON = 0.

Once the SEG_PM and RSM_PM table entries have been initialized, the processor sends an activation confirmed OAM cell to the originator.

7.4.3 PM Operation

PM processing operates automatically on any channel until stopped by clearing the PM_EN fields in the SEG VCC table entry and the RSM VCC table entry.

PM-OAM cells are not included in the NRM cell count, as part of ER processing. See Chapter 6.0, for details.

7.4.3.1 Generation of Forward Monitoring PM Cells

The segmentation coprocessor generates a forward monitoring PM cell at the end of each PM block, as defined by the BLOCK_SIZE field in the SEG_PM table for any PM_INDEX. It determines the point to generate a forward monitoring cell by following these processes:

- At the point of initialization of PM processing or when a forward monitoring cell is sent, the SEG coprocessor sets the BLOCK_COUNT field to 0. It also increments Monitoring Cell Sequence Number (MSN), and re-initializes the BIP field to 0.
- As each data cell is segmented, the SEG coprocessor increments the BLOCK_COUNT number and updates the BIP field.
- When the BLOCK_COUNT number reaches the block size specified by the BLOCK_SIZE field, signifying the end of the PM block, the SEG coprocessor generates a new forward monitoring PM cell and starts these processes again.

7.4.3.2 Reassembly of Forward Monitoring PM Cells

When the CN8237 receives a forward monitoring PM cell, the RSM coprocessor reads the RSM_PM table word pointed to by the PM_INDEX field in the RSM VCC table entry. The location of the RSM_PM table is above the LECID table. The BIPV, TRCC0, and TRCC0+1 fields are written to a special RSM-PM forward monitoring status queue entry. The TUC0 and TUC0+1 fields can be extracted directly from the RSM_PM cell payload.

When a new buffer is needed, the reassembly coprocessor uses the global OAM buffer queue if RSM_CTRL1(OAM_QU_EN) is a logic high. Otherwise, it uses the BFR0 pool identification number in the RSM VCC table to point to the appropriate free buffer queue.

A RSM status entry is written for each OAM cell reassembled.

The reassembly coprocessor uses the global OAM status queue if the RSM_CTRL1(OAM_QU_EN) bit is a logic high. Otherwise, it uses the STAT field in the RSM VCC table to determine which status queue to use for that channel.

7.4.3.3 Reassembly of Backward Reporting PM Cells

Backward reporting cells are reassembled in the same manner as non-PM OAM cells.

7.4.3.4 Turnaround and Segmentation of Backward Reporting PM Cells

For each forward monitoring cell received, the CN8237 also writes the BIPV, TRCC0, TRCC0+1, TUC0, and TUC0+1 fields to the RS_Queue, for further processing by the segmentation coprocessor. The segmentation coprocessor generates a backward reporting cell.

7.4.3.5 Turnaround of Backward Reporting PM Cells ONLY

To enable turnaround of backward reporting PM cells without generation of forward monitoring PM cells, set the segmentation PM_EN bit in the SEG VCC table entry to a logic high, set the PM_INDEX, and set the FWD_MON field to 0.

7.4.4 Error Conditions During PM Processing

If OAM cells are not using the global reassembly OAM buffer pool, the cells are treated as Single Segment Messages (SSMs) for purposes of the firewall protection. OAM cells using the global OAM buffer pool do not have per channel protection.

If the RS_Queue fills, the PM-OAM information is dropped, and the RS_QUEUE_FULL status indication is set.

7.4.5 PASS_OAM Function

A PASS_OAM only function is activated by setting bit 22 of word 0 in the RSM VCC table. When active, OAM cells are processed, and data cells are discarded without incrementing any discard counters.

7.5 OAM Control and Status Structures

Refer to Sections 4.3 and 5.7, for information on VCC Tables, buffer descriptors, the transmit queue, and status queues, as they apply to generating and processing OAM cells.

The base address of the SEG_PM table is given by the SEG_PMB field in the SEG_PMBASE register. The address of each entry is located at byte address,

SEG_PMB × 128 + (PM_INDEX) × 32.

The RSM_PM table is located above the LECID table, which is located above the VPI table. The address of each entry is located at byte address,

if

RSM_CTRL0(VPI_MASK)
RSM_ITB × 128 + 1024 + 64 + (PM_INDEX) × 16

else

RSM_ITB × 128 + 16384 + 64 + (PM_INDEX) × 16

7.5.1 SEG_PM Structure

Tables 7-6 and 7-7 describe the structure and field definitions of the SEG_PM table.

Table 7-6. SEG_PM Structure

Word	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	ATM_HEADER																															
1	FWD_TUC0																FWD_TUC01															
2	FWD_PM	Reserved		FWD_MON	Reserved								BLOCK_SIZE				BIP															
3	Reserved								BLER								BCK_MSN								FWD_MSN							
4	BCK_TUC0																BCK_TUC01															
5	TRCC0																TRCC01															
6	BLOCK_COUNT																FWD_MCSN								SECBC							
7	Reserved																															

Table 7-7. SEG_PM Field Descriptions

Field Name	Description
ATM_HEADER	ATM header to use for both backward-reporting and forward-monitoring cells.
FWD_TUC0	Total User Cell number with CLP = 0 for forward-monitoring.
FWD_TUC01	Total User Cell number with CLP = 0,1 for forward-monitoring.
FWD_PM	A forward PM cell is due to be sent.
BLOCK_SIZE	Size in cells of forward-monitoring block: 1000: block size = 128 0100: block size = 256 0010: block size = 512 0001: block size = 1024 1001: block size = 2048 0101: block size = 4096 0011: block size = 8192 1011: block size = 16384 0111: block size = 32768
FWD_MON	Set to enable both forward-monitoring and backward-reporting. Clear to enable only backward-reporting.
BLOCK_COUNT	Number of cells in current monitoring block.
BIP	BIP-16 for forward-monitoring.
BLER	Block Error Result for backward-reporting cells.
BCK_MSN	Monitoring Cell Sequence Number for backward-reporting cells.
FWD_MSN	Monitoring Cell Sequence Number for forward-monitoring cells.
BCK_TUC0	TUC0 field for backward-reporting cells.
BCK_TUC01	TUC01 field for backward-reporting cells.
TRCC0	Total Received Cell Count with CLP = 0 for backward-reporting.
TRCC01	Total Received Cell Count with CLP = 0,1 for backward-reporting.
FWD_MCSN	Copied value from received forward monitoring cell to be placed in backward reporting cell.
SECBC	Severely errored cell block as generated by the RSM block.

7.5.2 RSM_PM Table

Tables 7-8 and 7-9 describe the structure and definitions of the RSM_PM table.

Table 7-8. RSM_PM Table Entry

Word	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	Reserved																								MSN							
1	BIP16																BCNT															
2	TRCC0																TRCC0+1															
3	Reserved																															

Table 7-9. RSM_PM Table Field Descriptions

Field Name	Description
MSN	Monitoring Cell Sequence Number for forward-monitoring PM cells. Backward-reporting PM cells are not included in this sequence. This field allows for the detection of lost or mis-inserted PM cells containing forward-monitoring information.
BIP16	The BIP-16 error detection code generated over the payloads of the user information cells in the PM block.
BCNT	Block Count. This field contains the calculated number of cells in the current PM block.
TRCC0	Total received cell count with CLP = 0.
TRCC0+1	Total received cell count.

8.0 DMA Coprocessor

8.1 Overview

The DMA coprocessor performs high-speed sustained data transfers to and from the host memory space. It is controlled by the segmentation and reassembly coprocessors.

The major functions of the DMA coprocessor include transferring data from the host memory (through the PCI bus) to the segmentation coprocessor and transferring data from the reassembly coprocessor to the host memory space (through the PCI bus).

In all modes of operation, the DMA coprocessor maintains a high level of performance. It uses burst transfers when possible to maximize host bus bandwidth utilization, perform byte switching to accommodate misaligned transfers, and carry out concurrent input and output transfers (alternating burst reads and burst writes) to support simultaneous input and output data streams.

8.2 DMA Read

For outgoing messages, DMA read cycles move data from host memory to the segmentation coprocessor using a gather DMA method. The maximum burst size is thirteen 32-bit words, which correspond to one cell.

8.3 DMA Write

For incoming messages, DMA write cycles move data from the reassembly coprocessor to host memory using a scatter DMA method. The maximum burst size is fourteen 32-bit words, which correspond to one ATM cell and a status word appended to PM cells.

8.4 Master Transactions

The SAR supports both big endian and little endian processors in the PCI address space. Bit 15 (ENDIAN) in SRC Configuration Register 0 controls all data byte swapping. Bit 30 (MSTR_CTRL_SWAP) in the Special Status Register (PCI Config Address 0x40) controls all control byte swapping. Bit 27 (MSTR_DATA_DWORD) in the Special Status Register (PCI Config Address 0x40) controls Data Dword Conversion. Bit 26 (MSTR_CTRL_DWORD) in the Special Status Register (PCI Config Address 0x40) controls the Control Dword Conversion.

8.4.1 Data Transfers

The SAR can perform a full matrix of byte lane swapping and Dword conversion. Dword Conversion is used to determine which 32-bit word is placed on the PCI Bus first during a 32-bit transaction. Data transfers do not have to be byte aligned.

8.4.2 Control Word Transfers

The SAR internally distinguishes between data and control transactions. The SAR performs byte swapping based on bit 30 (MSTR_CTRL_SWAP) in the Special Status Register (PCI Config Address 0x40). The SAR performs Dword Conversion based on bit 26 (MSTR_CTRL_DWORD) in the Special Status Register (PCI Config Address 0x40). This allows the order of the 32-bit words to be swapped. Control word transfers are byte aligned.

Control transfers include the following data structures: reassembly and segmentation status queue writes, read update pointer writes, and reassembly buffer descriptor next pointers.

8.5 Slave Transactions

All slave transactions consist of control words. There are two types of control word manipulation that can take place on slave transactions: Byte swapping and Quadword (64-bit) to Dword (32-bit) Conversion.

8.5.1 Control Word Transfers

The SAR accepts 64-bit or 32-bit data as a slave. Since the Local Memory is 32 bits, a 64-bit access is converted to two 32 accesses. Under programmable control by bit 28 (SLAVE_DWORD) in the PCI Special Status Register (PCI Config Address 0x40), either bits [31:0] or bits [63:32] are mapped to address 0x00. Dword Conversion is performed on 64-bit transfers only.

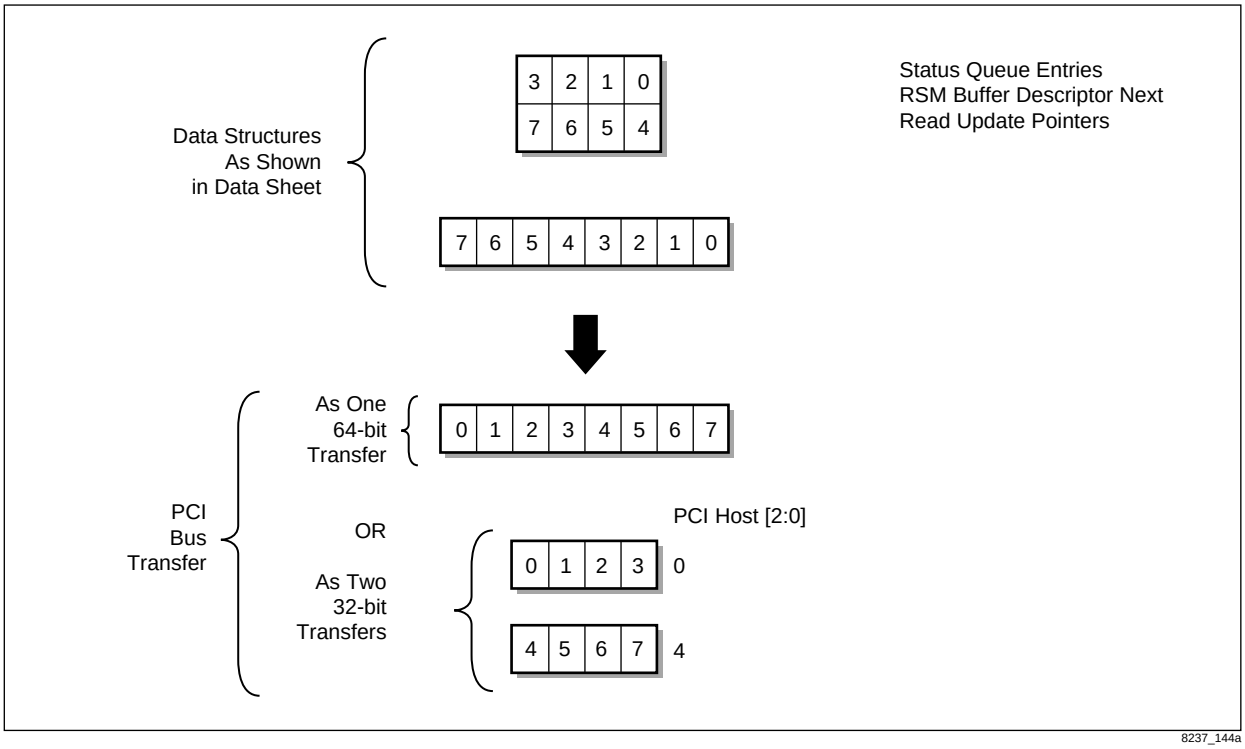
Bit 29 (SLAVE_SWAP) in the PCI Special Status Register (PCI Config Address 0x40) control byte swapping. Bytes are mapped from PCI address to local memory address. Byte swapping is performed on both 32-bit and 64-bit transactions.

8.6 Control Bit by Bit Endian Transfers

8.6.1 Master Control Swap

Special Status bit 30 control swaps the byte lanes of control words written by SAR to host memory. Includes SEG/RSM Status Queues, Free Buffer and Transmit queue read update values, and RSM buffer decriptor next pointer. See Figure 8-1.

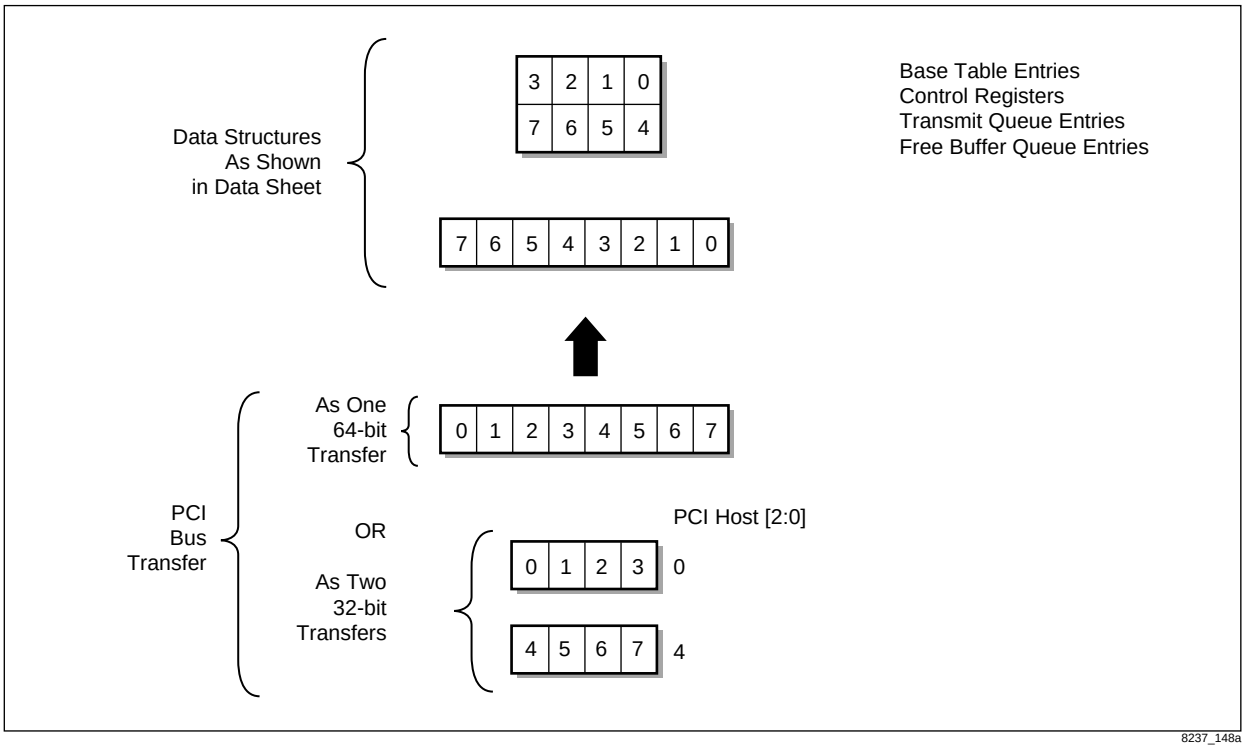
Figure 8-1. Master Control Swap



8.6.2 Slave Swap

Special Status bit 29 control swaps bytes within 32-bit words when host accesses SAR control Regs and local SRAM. See Figure 8-2.

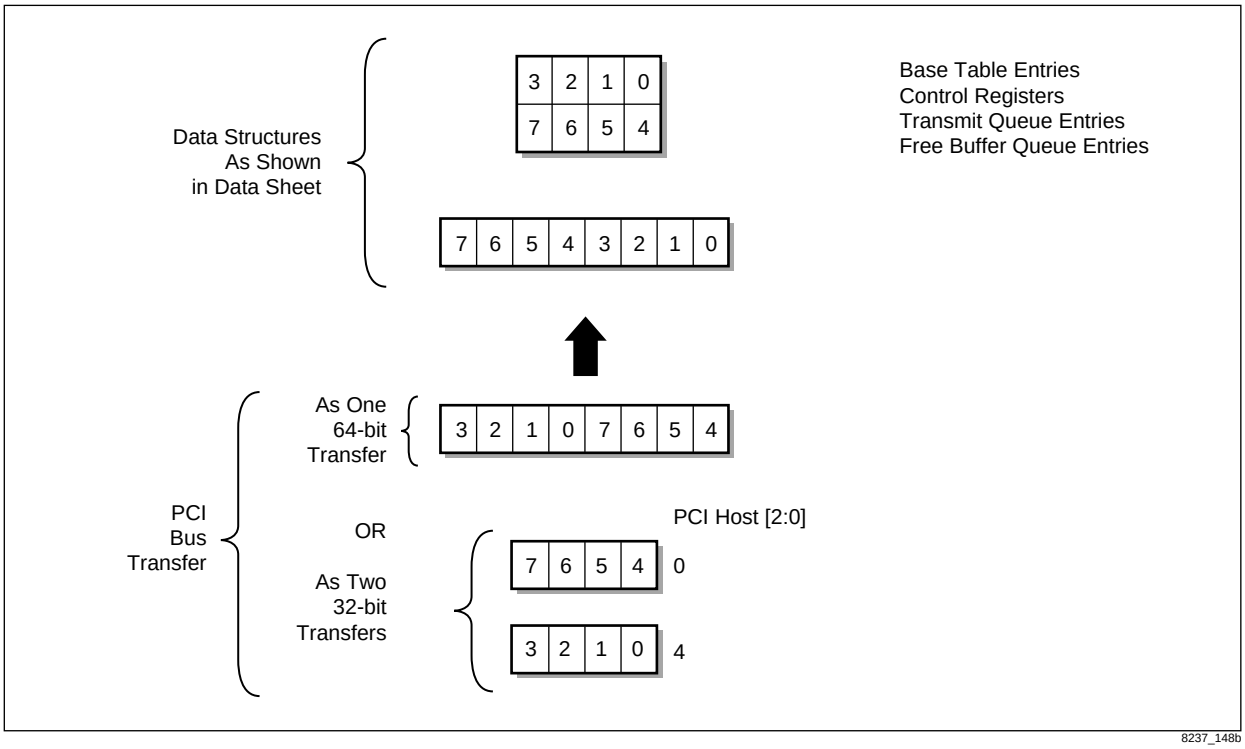
Figure 8-2. Slave Swap



8.6.3 Slave Dword Swap

Special Status bit 28 control swaps 32-bit word in 64-bit word when host accesses SAR control Regs and local SRAM. See Figure 8-3.

Figure 8-3. Slave Dword Swap



8.6.4 Master Data Dword Swap

Special Status bit 27 control swaps 32-bit words in 64-bit transaction of host writing data to host buffers and reading host buffers from host buffers. Bit 15 (ENDIAN) in SRC Configuration Register 0 is 0. See Figure 8-4 and Figure 8-5.

Figure 8-4. Master Data Dword Swap—64-bit Transfer

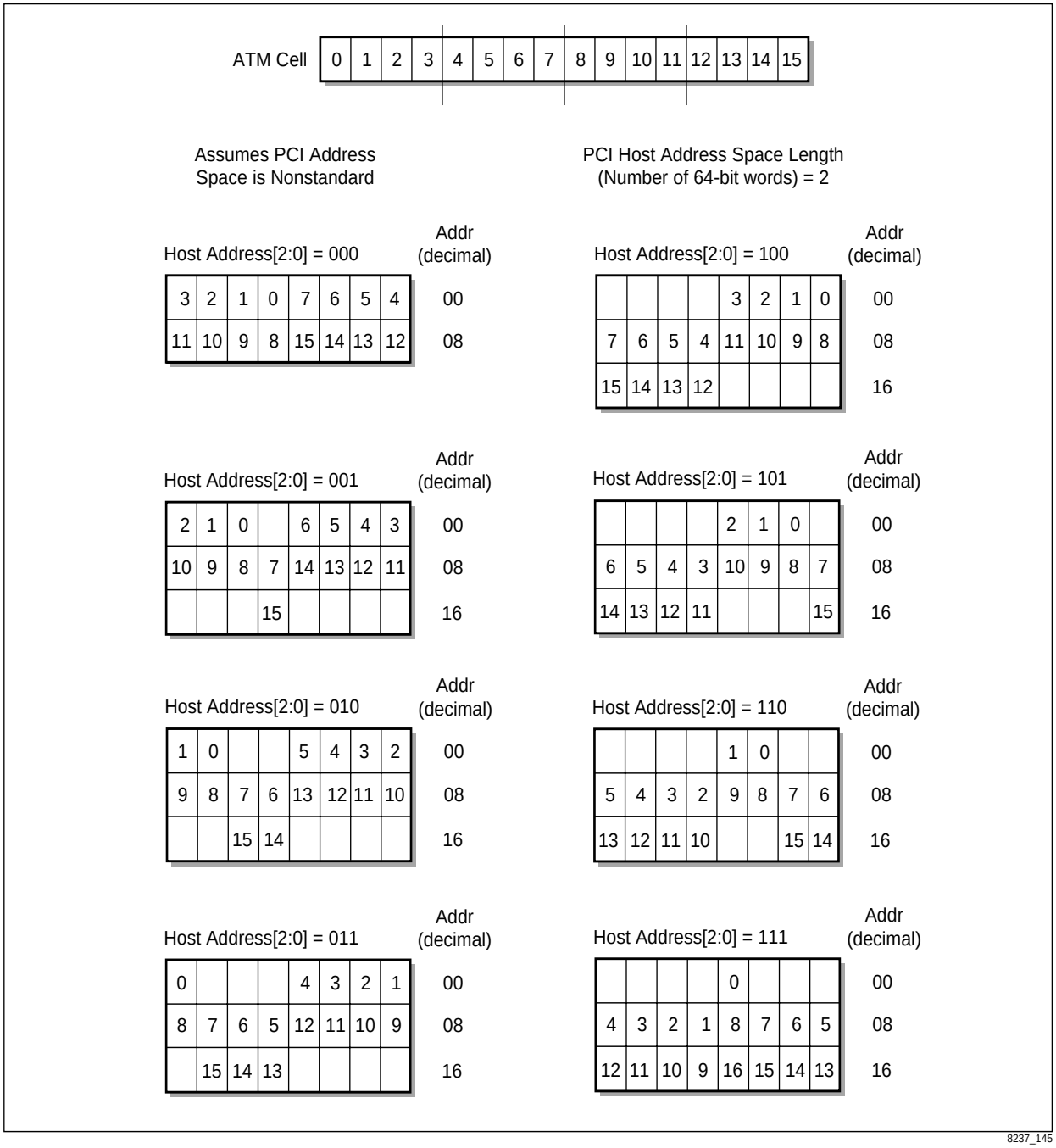
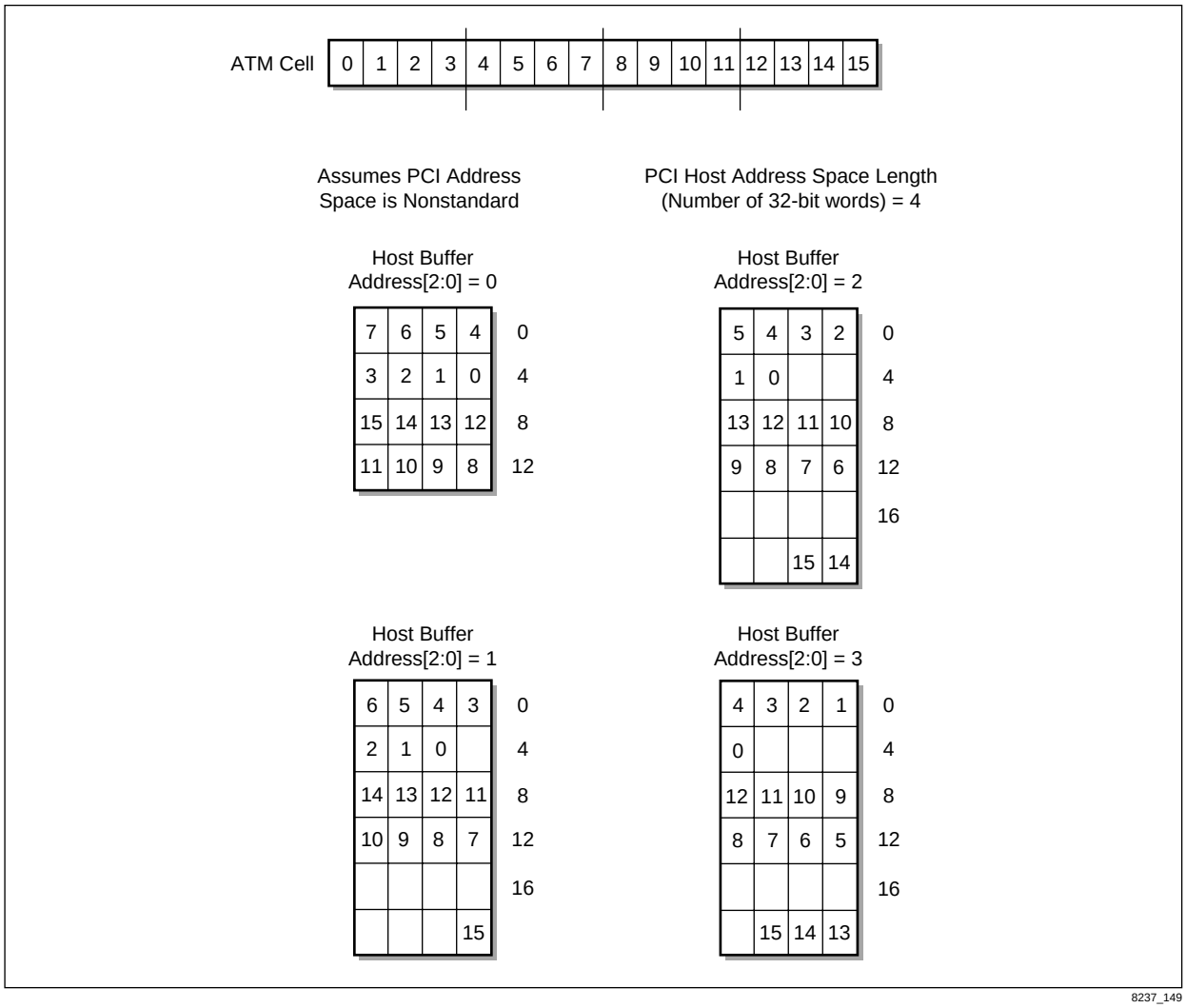


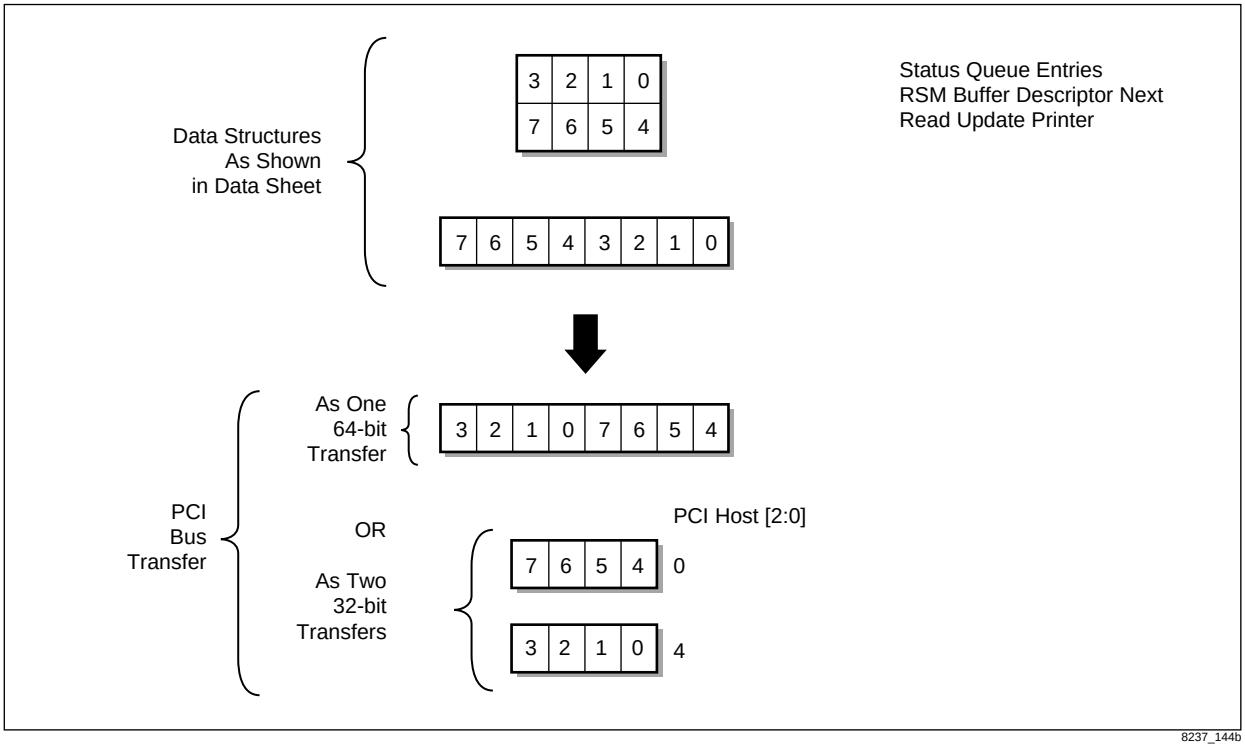
Figure 8-5. Master Data Dword Swap—32-bit Transfer



8.6.5 Master Control Dword Swap

Special Status bit 26 controls swapping of control writes by the SAR to host memory. It swaps the two 32-bit words within a 64-bit transaction. See Figure 8-6.

Figure 8-6. Master Control Dword Swap



8.6.6 Master Data Byte Swap

CONFIG0 bit 12 controls byte swapping of master reads and writes of data within 64-bit words. See Figure 8-7 and Figure 8-8.

Figure 8-7. Master Data Byte Swap—64-bit Transfer

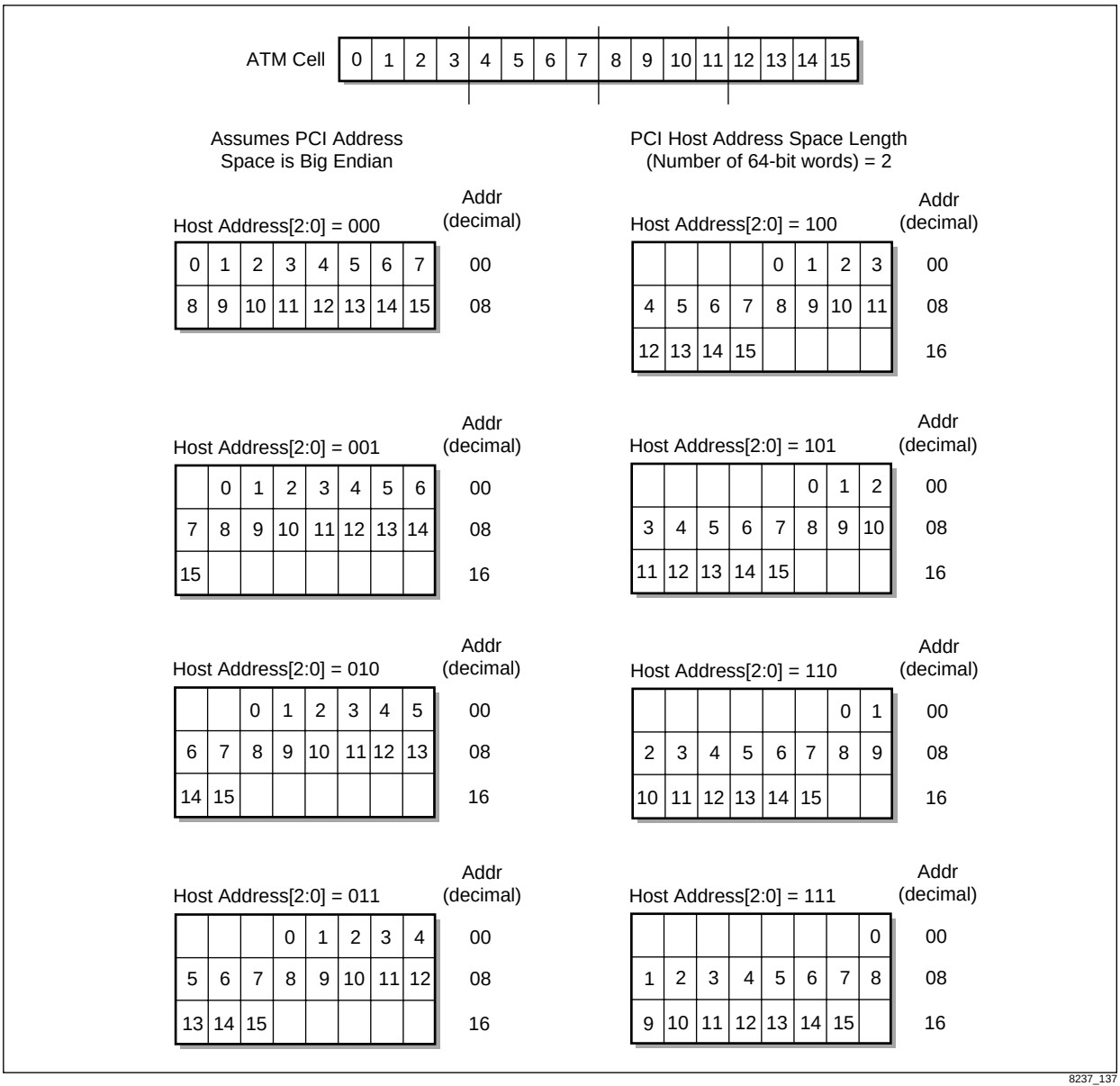
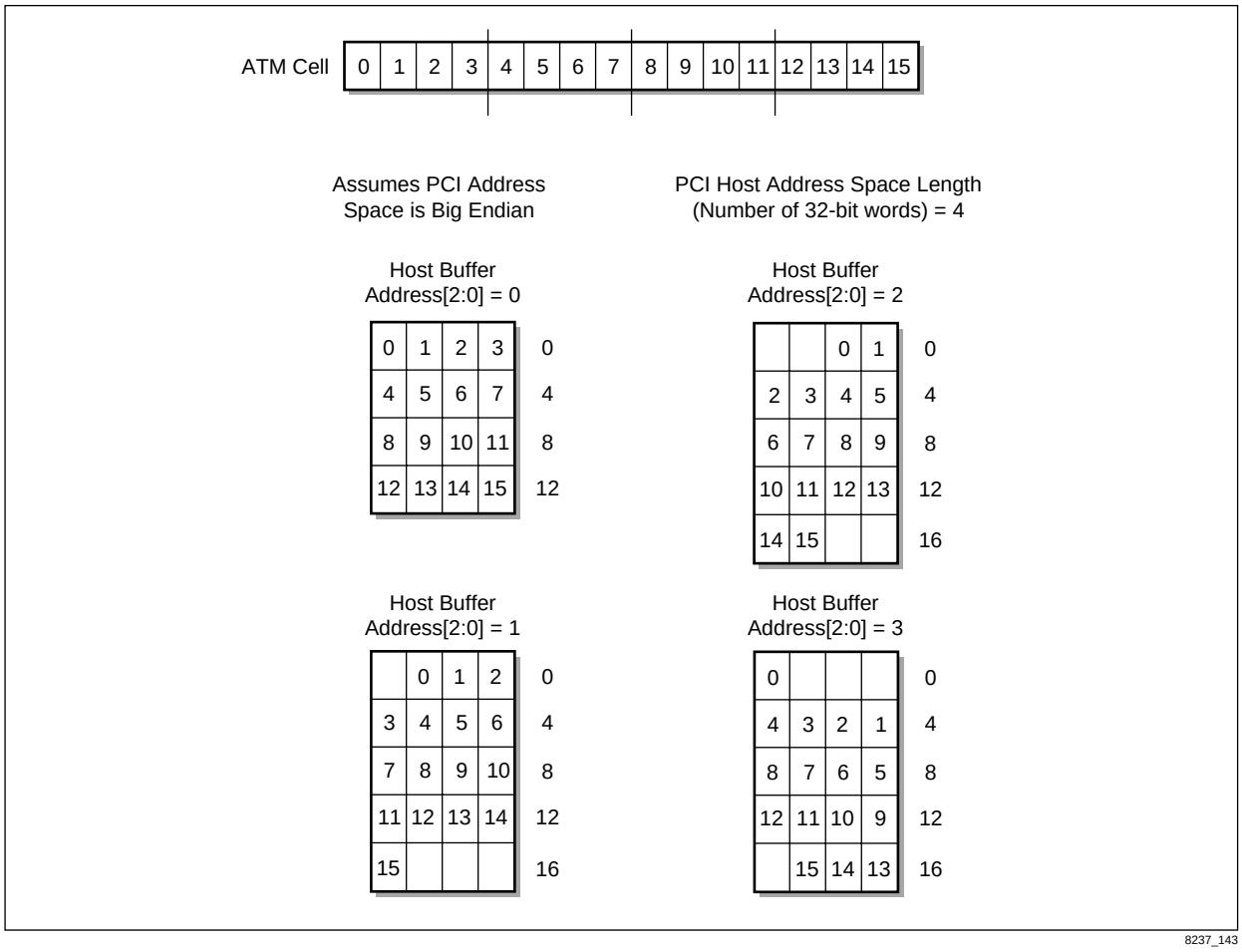


Figure 8-8. Master Data Byte Swap—32-bit Transfer



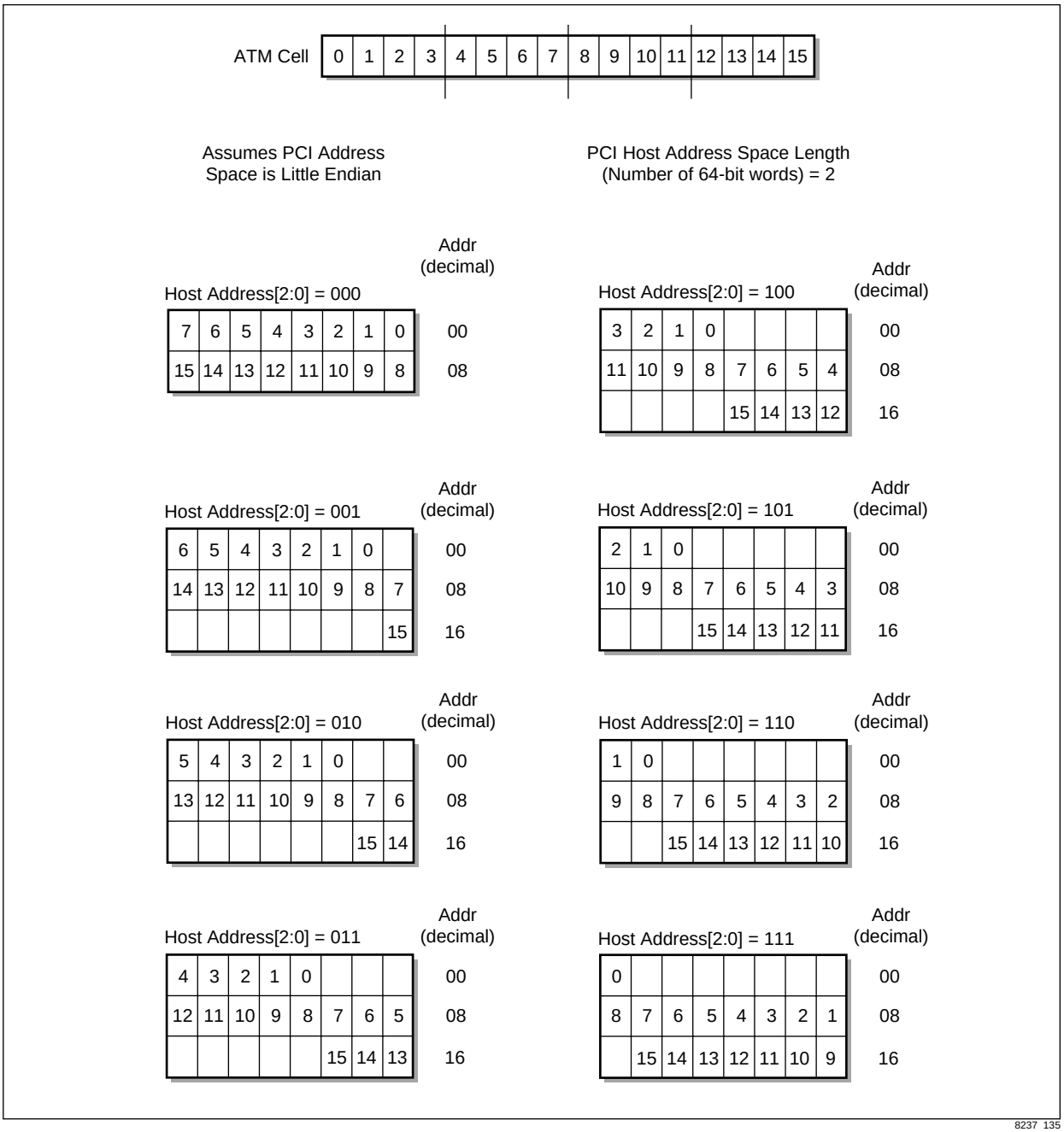
8.7 Little Endian Mode

When no Endian bits are set, the SAR comes up in little Endian mode. The following section describes transfers as seen in the PCI bus when the SAR is set in default mode.

NOTE: MSTR_DWORD_SWAP must be set to 1.

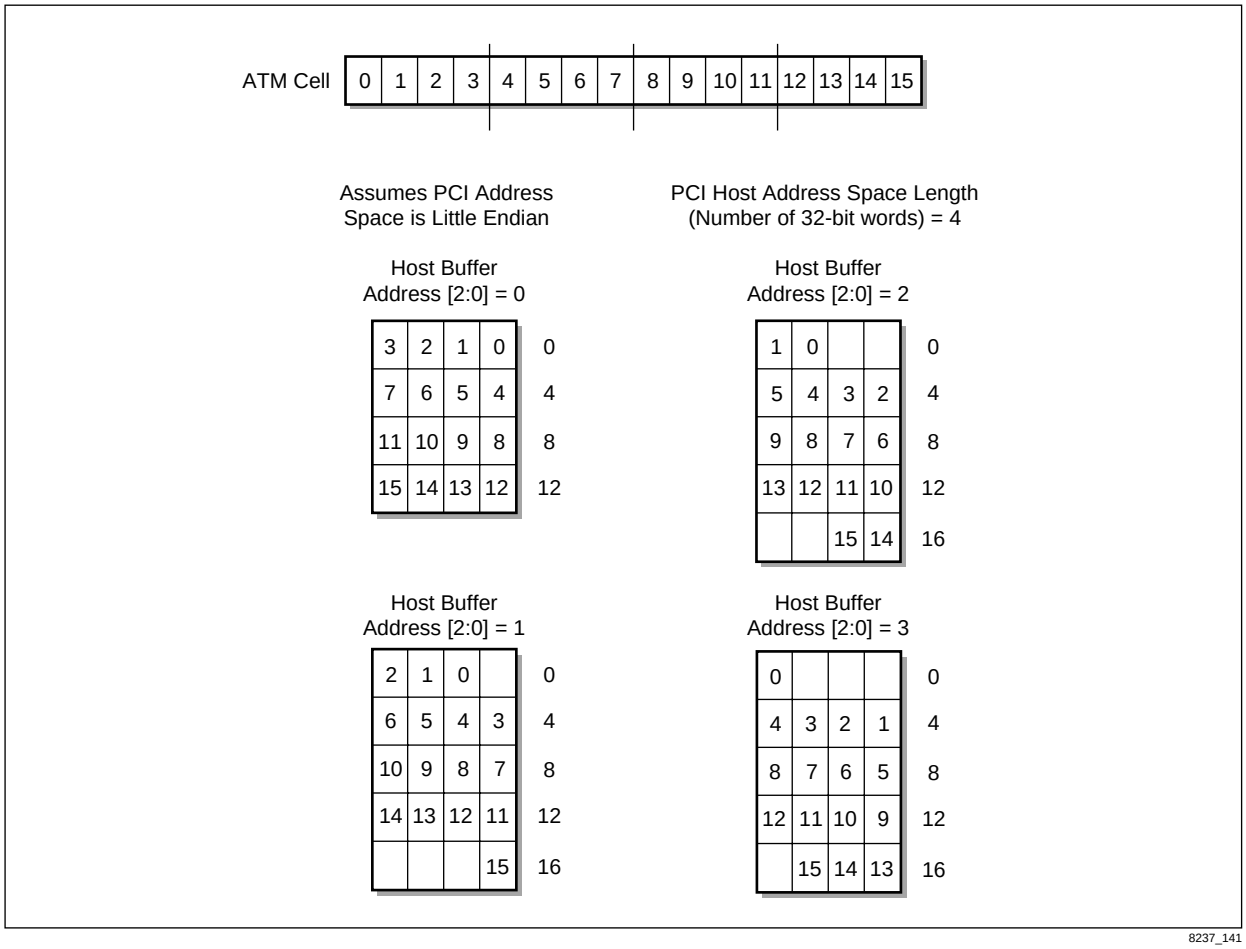
8.7.1 Little Endian 64-bit Master Data Transfers

Figure 8-9. Little Endian 64-bit Master Data Transfers



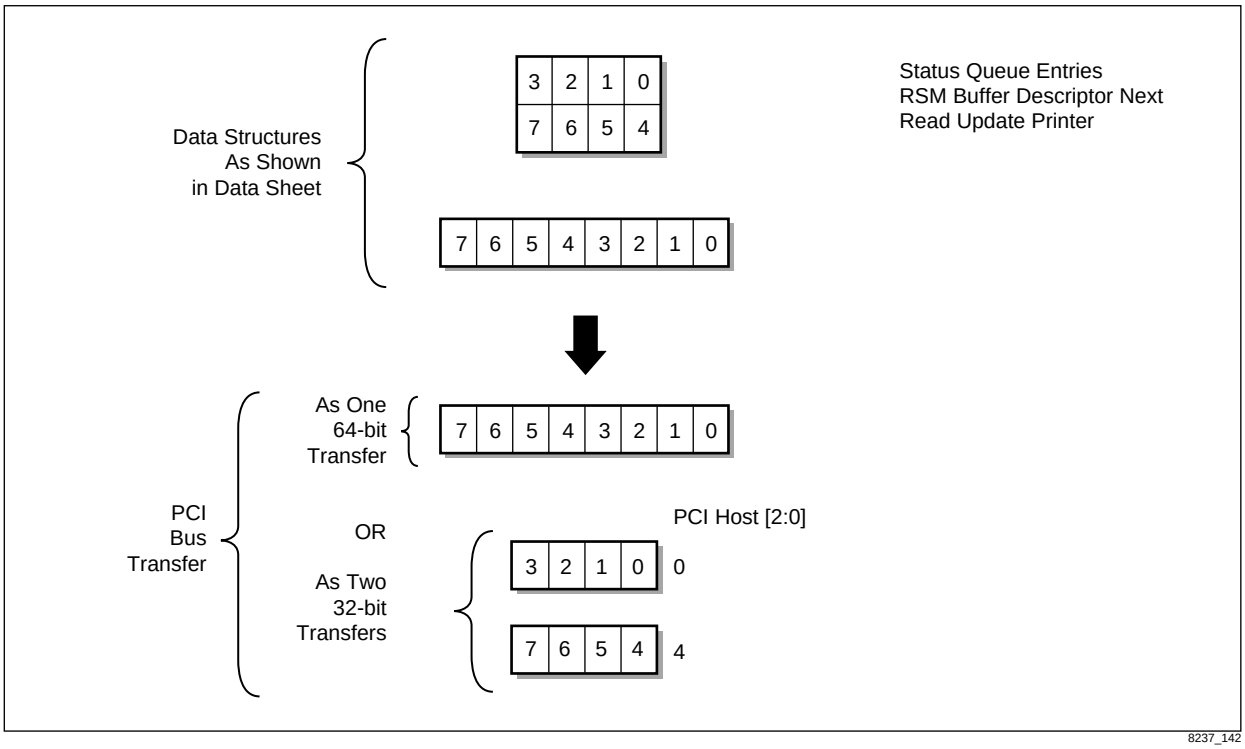
8.7.2 Little Endian 32-bit Master Data Transfers

Figure 8-10. Little Endian 32-bit Master Data Transfers



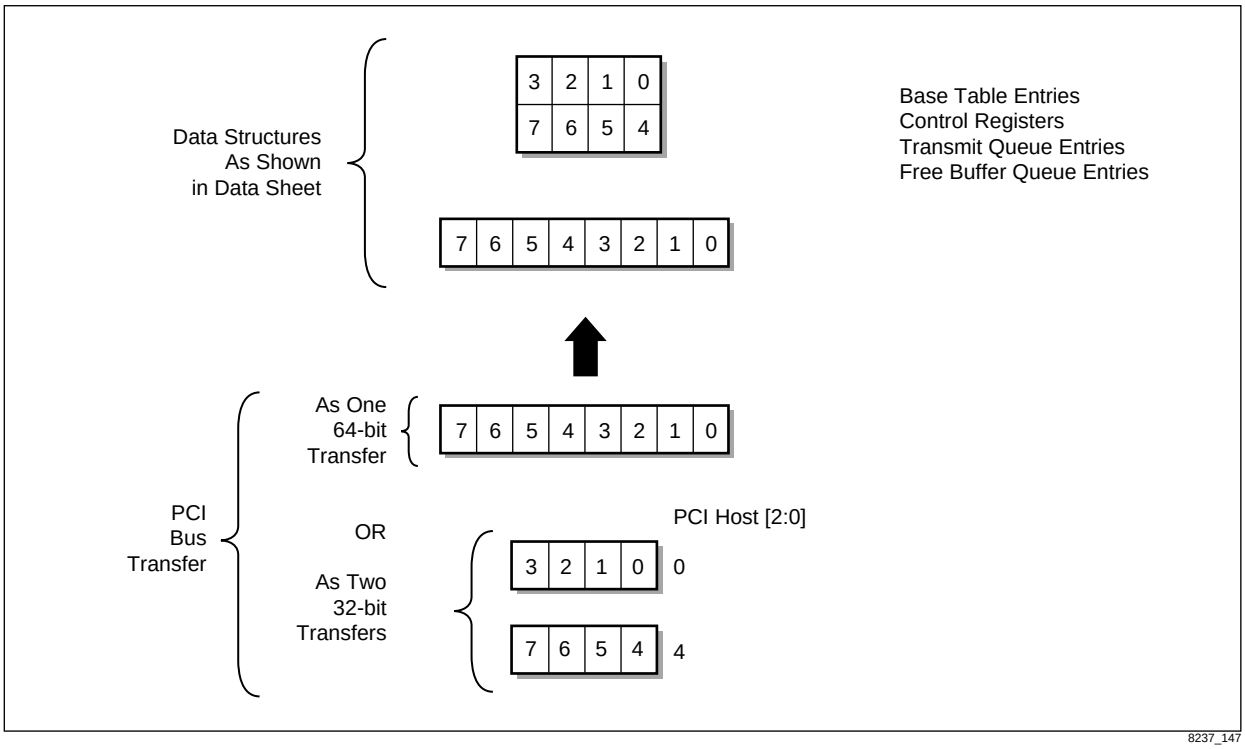
8.7.3 Little Endian SAR as Master Control Transfers

Figure 8-11. Little Endian SAR as Master Control Transfers



8.7.4 Little Endian Host Slave Interface Control Transfers

Figure 8-12. Little Endian Host Slave Interface Control Transfers



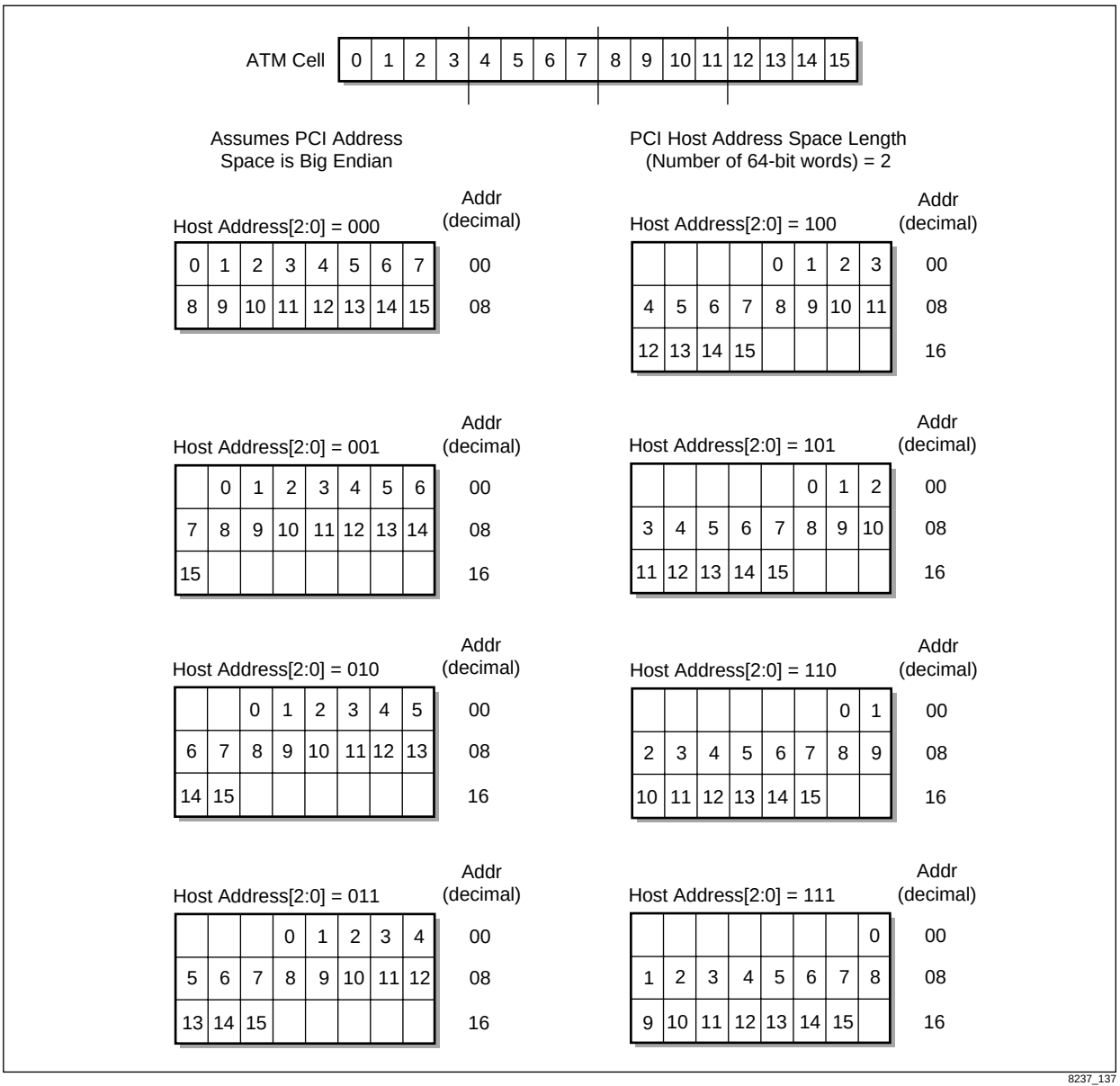
8.8 Big Endian Mode

For pure big Endian systems, set the Endian bits CONFIG0 bit 12, PCI CONFIG Special Status bits 28 and 30.

NOTE: Do not mix 64-bit and 32-bit transfers.

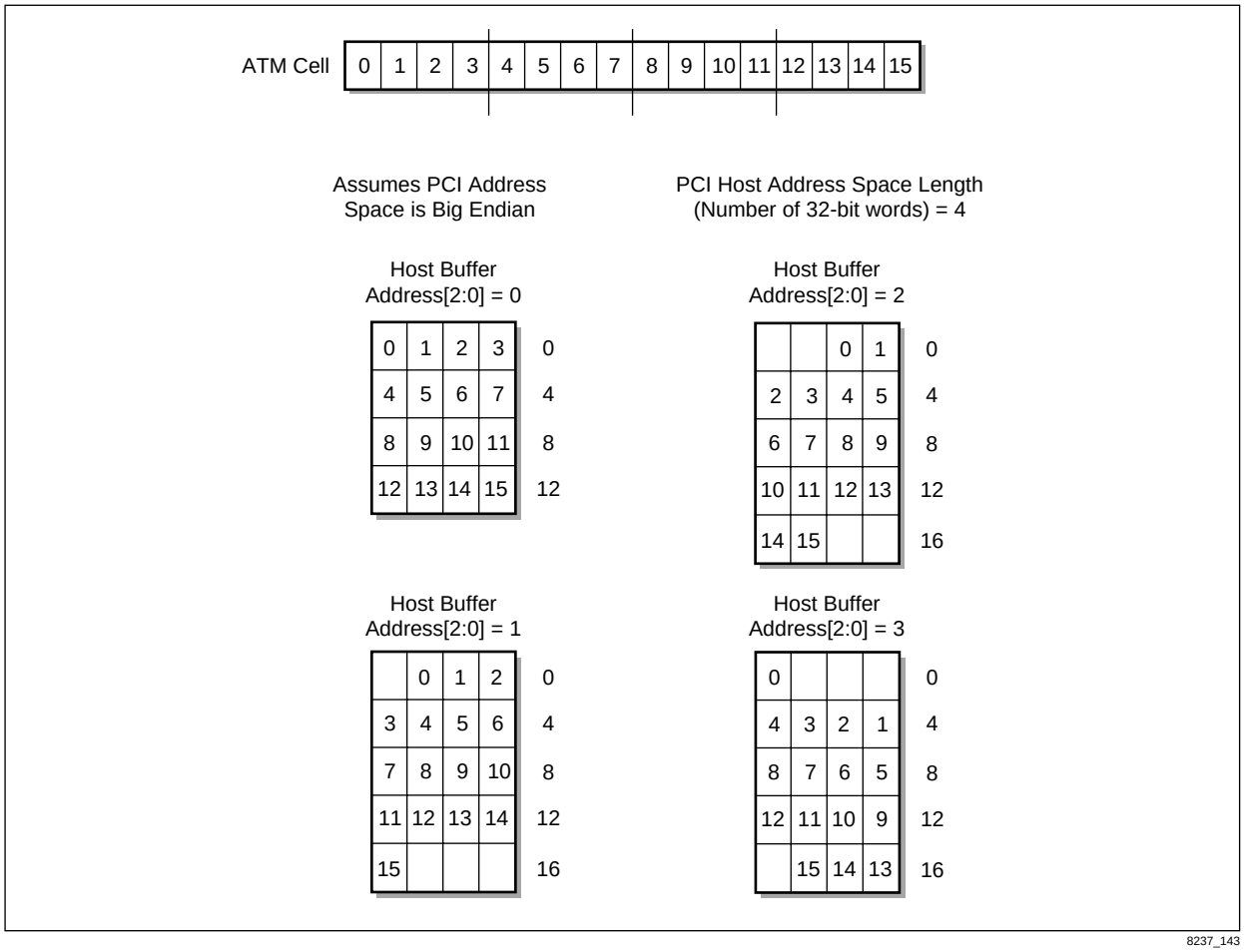
8.8.1 Big Endian 64-bit Master Data Transfers

Figure 8-13. Big Endian 64-bit Master Data Transfers



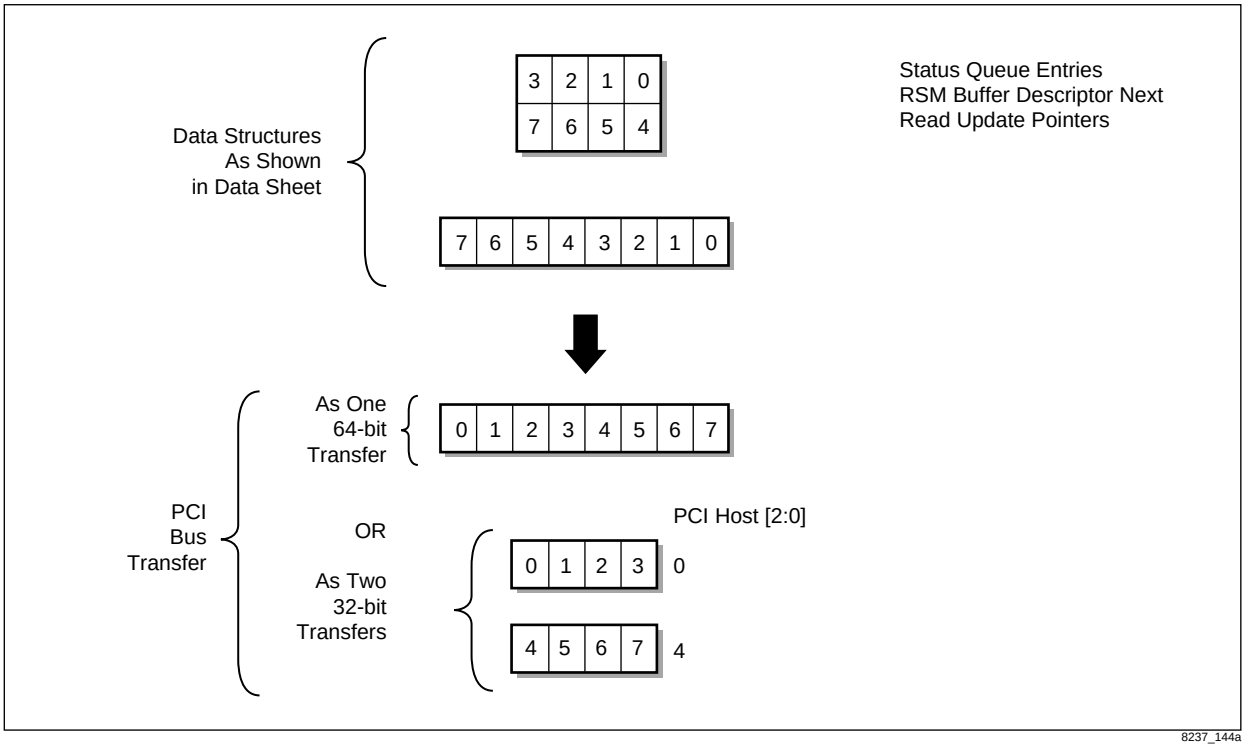
8.8.2 Big Endian 32-bit Master Data Transfers

Figure 8-14. Big Endian 32-bit Master Data Transfers



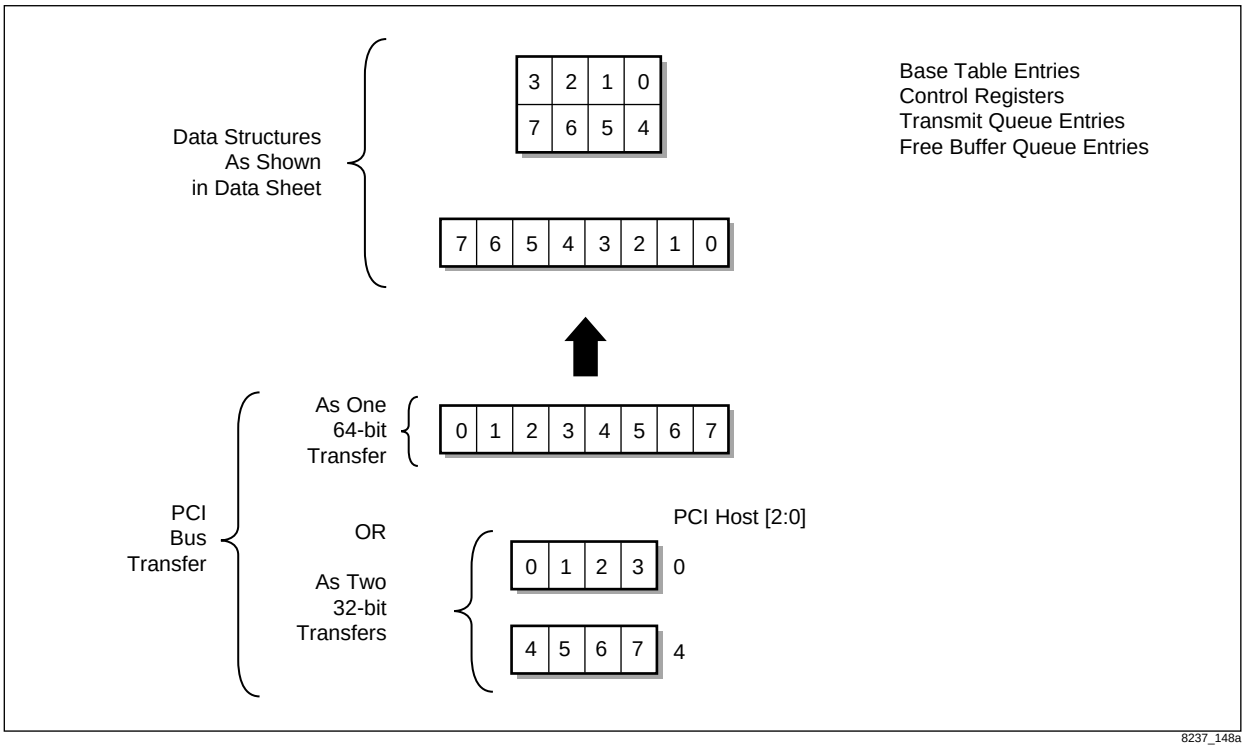
8.8.3 Big Endian SAR as Master Control Transfers

Figure 8-15. Big Endian SAR as Master Control Transfers



8.8.4 Big Endian Host Slave Interface Control Transfers

Figure 8-16. Big Endian Host Slave Interface Control Transfers



9.0 System Interface

9.1 System Clocking

PCI CLK is used as a reference to a PLL that generates several system clocks, including SYSCLK, CLKD3, and PCLK. The PLL multiplier is programmable from value of 4 to 10 register in PC configuration. The VCO frequency range is 200 MHz to 264 MHz. A system with a 66 MHz HCLK, will set the multiplier to 4 in order to generate a 264 MHz VCO frequency which in turn generates a 66 MHz SYSCLK, 33 MHz PCLK, and 44 MHz CLKD3 signals. The multiplier value resides at address 0x40 of the PCI configuration space. The M66EN input is used to determine the default value of the multiplier upon application of HRST*. If logic high, the multiplier is set to 4 for operation with a 66 MHz PCI clock. If logic low, it is set to 8 for operation with a 33 MHz PCI clock. After system reset, the user may change the frequency by writing the PLL_MUL field in the PCI configuration space. The resultant value should not exceed 66 MHz for SYSCLK, 33 MHz for PCLK and 50 MHz for CLKD3 (if used to drive UTOPIA clocks).

9.2 Real-Time Clock Alarm

A real-time clock counter and alarm registers are built into the CN8237. This real-time clock consists simply of a 7-bit prescaler (configured via the DIVIDER field in the CONFIG0 register) that accepts the SYSCLK input and outputs a constant (nominally 1 MHz) pulse train, and a 32-bit read/write counter (the Real-Time Clock register [CLOCK;0x00]), that counts the number of pulses output by the prescaler since the system was initialized. When the prescaler is set to generate a 1 MHz pulse train, the CLOCK counter counts in 1 µs intervals. An interrupt is generated when the CLOCK counter overflows, that is, more than 2^{32} pulses have occurred since it was cleared to zero. If this happens, the CLOCK counter simply wraps around to zero and starts counting over. The control processor or host software is responsible for noting the overflow.

One simple real-time alarm is implemented in the CN8237. This is Alarm register 1 [ALARM1;0x04], which is continuously compared to the Clock register. When a match is detected, an interrupt is generated to the host processor. The processor can then respond to this interrupt and reload a new value into the ALARM1 register.

9.3 CN8237 Reset

The CN8237 must be reset by the host processor prior to system initialization for proper operation. This can be done in one of two ways: by asserting the external HRST* pin (which is normally connected to the system power-up reset circuitry), or by setting the GLOBAL_RESET bit in the Configuration register 0 [CONFIG0;0x28]. The HRST* pin must be deasserted and GLOBAL_RESET must be cleared before beginning the CN8237 initialization process.

10.0 Local Memory Interface

10.1 Overview

To simplify system implementations, the CN8237 integrates a complete memory controller, designed for direct interface to local memory Zero Bus Turnaround (ZBT) or syncburst synchronous SRAMs (each containing up to 4 MB of external memory, 8 MB future expansion).

When CONFIG0 (MEMTYPE) is logic low, ZBT flowthrough synchronous memory is selected. When MEMTYPE is a logic high, non-ZBT flowthrough synchronous memory is selected. ZBT memory does not require a dead cycle between a read cycle and a write cycle whereas non-ZBT synchronous SRAM does.

There are two coprocessors located inside the CN8237. Each coprocessor has its own local memory. One coprocessor is designated for segmentation (transmitting), while the other coprocessor is designated for reassembly (receiving).

Each local memory interface is 32 bits wide and operates up to 66 MHz.

When the multiple consumers try to access the segmentation local memory, the segmentation coprocessor has priority, followed by the reassembly coprocessor, then the host. When multiple consumers try to access the reassembly local memory, the reassembly coprocessor has priority followed by host.

The amount of memory required is heavily dependent on the number of VCCs implemented, as well as the number of VCCs that are currently active. Memory requirements are discussed in Section 10.3.

Table 10-1. Memory Map

Byte Address Ranges	Block Selected	Source/Destination
0 x FF FFFFh >> (7 – (BANKSIZE + NUMBANKS)) 0 x 80 0000h >> (7 – (BANKSIZE + NUMBANKS))	SCS3* SCS2* SCS1* SCS0*	SEG Local Memory
0 x 7F FFFFh >> (7 – (BANKSIZE + NUMBANKS)) 0 x 00 1800h	RCS3* RCS2* RCS1* RCS0*	RSM Local Memory
0 x 00 1000h – 0 x 00 17FF	Internal SRAM	SAR
0 x 00 0800h – 0 x 00 0FFFh	PHY	PHY
0 x 00 0400h – 0 x 00 07FFh	Reserved	X
0 x 00 0000h – 0 x 00 03FFh	8237 Internal Registers	SAR

Table 10-2. Example of Memory Map

Byte Address Ranges	Block Selected	Source/Destination
0 x 3F FFFFh 0 x 20 0000h	SCS1* (for NUMBANKS = 2) SCS0* (for NUMBANKS = 2)	SEG Local Memory
0 x 1F FFFFh 0 x 00 1800h	RCS1* (for NUMBANKS = 2) RCS0* (for NUMBANKS = 2)	RSM Local Memory
0 x 00 17FFh 0 x 00 1000h	Internal SRAM	SAR
0 x 00 0FFFh 0 x 00 0800h	PHY	PHY
0 x 00 0FFFh 0 x 00 0700h	Reserved	X
0 x 00 03FFh 0 x 00 0000h	8237 Internal Registers	SAR
NOTE(S): If BANKSIZE = 100 (1 Mbytes per bank) and NUMBANKS = 01 (2 banks)		

10.2 Memory Bank Characteristics

The external memory is organized in one to four banks of up to 4 MB each (8 MB future). The system can use any number of banks to fulfill the memory requirements; the only stipulation is that the banks must be of the same size and organization. BANKSIZE[2:0] (bits 23–21) in the CONFIG0 register denotes the size of the memory banks and allows the CN8237 to incorporate the various bank sizes into contiguous memory. Table 10-3 gives the coding of the BANKSIZE[2:0] control bits.

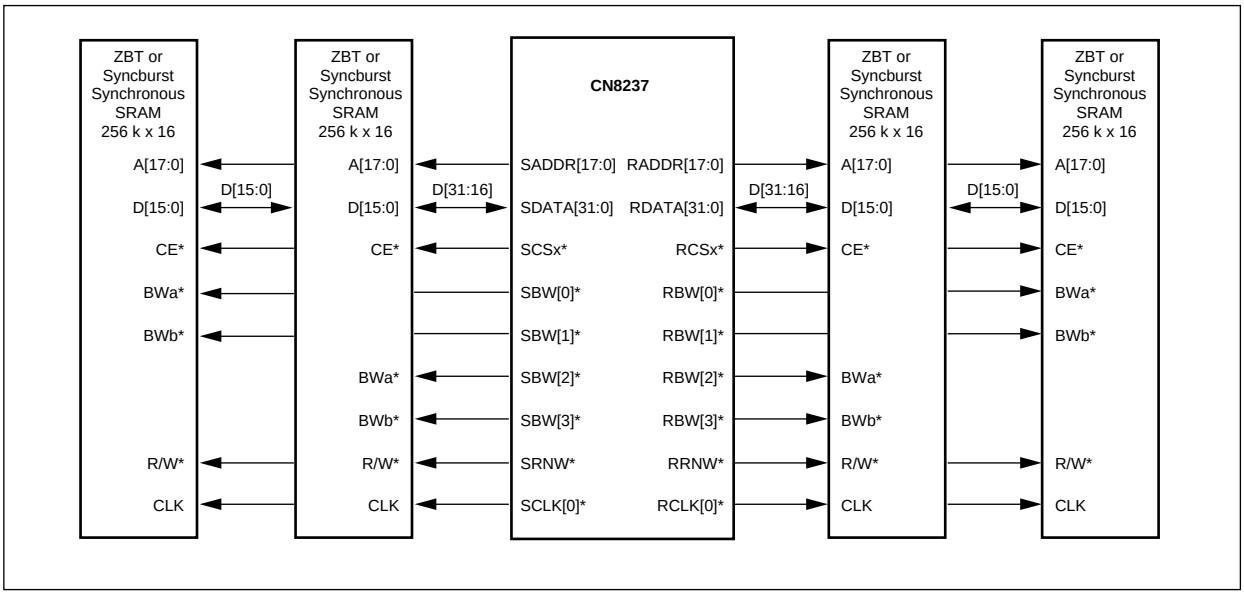
Table 10-3. BANKSIZE Selection Number

BANKSIZE (CONFIG0)	Bank Memory Organization	Total Bank Size (Bytes)	Typical Implementation
111	2M × 32	8 M	Future expansion
110	1M × 32	4 M	Two 1 M × 16, one 1 M × 32
101	512 k × 32	2 M	Two 512 k × 16, one 512 k x 32
100	256 k × 32	1 M	Two 256 k × 16, one 256 k x 32
011	128 k × 32	512 k	One 128 k × 32, two 128 k × 16
010	64 k × 32	256 k	One 64 k × 32, two 64 k × 16
001	32 k × 32	128 k	One 32 k × 32
000	Reserved	—	—

The memory controller is designed to work with either ZBT or syncburst synchronous SRAM. The local memory interface consists of one SEG and one RSM coprocessor. Each port is 32 bits wide and operates up to 66 MHz. Each coprocessor is able to address up to 4 MB (8 MB future expansion) of either ZBT or syncburst synchronous SRAM. For syncburst SRAM, the RBW[3:0]* and SBW[3:0]* outputs become byte enables for both reads and writes. When reading local memory, entire 32-bit words are always read. Figure 10-1 shows a typical bank using by_16 RAM. To connect different sized RAM banks, simply use more or less address bits; all other control remains the same.

NOTE: The number and type of ZBT or syncburst synchronous SRAM chips used affect the address and data bus capacitance and, therefore, the AC timing specifications and the required ZBT or syncburst synchronous SRAM speed. The use of by_16 devices causes more address bus loading than the use of by_32 devices. See Chapter 16.0 for detailed timing information.

Figure 10-1. ZBT or Syncburst Synchronous SRAM Bank Utilizing By_16 Devices



8237_083

The memory map contains space allocated to the RS825x (physical layer device).

The MEMCTRL bit in the CONFIG0 register selects the number of wait states that the memory controller uses to access the ZBT or syncburst synchronous SRAM. A logic 0 indicates 0 wait state or single-cycle memory, while a logic 1 indicates one wait state or two-cycle memory. The power-on default is MEMCTRL=1, selecting one wait state or two-cycle memory accesses.

The internal register and internal memory accesses from the PCI slave interface are always 0 wait state. When the CN8237 decodes a PCI slave read to its address space, the CN8237 performs a prefetch of four subsequent (contiguous) word locations.

ZBT or syncburst synchronous SRAM access time requirements are directly proportional to the system clock speed, as well as the amount and organization of the memory. The required system clock speed for a given application is dependent on the physical line rate, number of VCCs, and the percentage of idle cells versus assigned cells. The system designer can choose the appropriate ZBT or syncburst synchronous SRAM characteristics to suit the amount of memory and organization required for the application. See Chapter 16.0 for timing information.

10.3 Memory Size Analysis

Table 10-4 lists the memory size requirements for 1,024 configured VCCs under the following assumptions:

SEGMENTATION

1. The Schedule table is 8,448 schedule slots and each slot is a double word. This allows CBR and three-priority VBR schedule in 64 Kbps increments for an OC-12 connection.

2. 32 OAM PM channels active.
3. Transmit queues configured for 256 entries.
4. Each active channel has an average of one active segmentation buffer.
5. RS_Queue size is 256 entries.

REASSEMBLY

1. Eight VPIs per 1,024 channels.
2. 1,024 entries preallocation on VPI=0 only.
3. UNI VPI space.
4. 32 OAM PM channels active.
5. Free buffer queues configured for 256 entries.
6. FBQ pools have 2-word entries.
7. Variable block size disabled.

Table 10-4. Memory Size in Bytes

Data Structure	One Peer	16 Peers	32 Peers
SEGMENTATION			
Schedule Table	67,584	67,584	67,584
PM Tables	1,024	1,024	1,024
Buffer Descriptors	24,576	24,576	24,576
Transmit Queue	2,048	32,768	65,536
VCC Tables	40,960	40,960	40,960
RSM Q	4,096	4,096	4,096
Total	140,288	171,008	203,776
REASSEMBLY			
VPI Index Table	1,024	1,024	1,024
VCI Index Tables	32,768	32,768	32,768
VCC Tables	49,152	49,152	49,152
PM Tables	256	256	256
Free Buffer Queue	2,048	32,768	65,536
Total	85,248	115,968	148,736
Grand Total	225,536	286,976	352,512

11.0 PHY Interface

11.1 Overview

A physical microprocessor interface provides access to the microprocessor port of one physical device directly through the PCI interface. The interface consists of dedicated control signals and address/data signals. The PHY interface directly decodes accesses from the slave interface without using local memory bus bandwidth.

A PHY clock that runs up to 33 MHz (the PHY clock is a divide by 2 of SYSCLK) is provided. Also, each eight byte access across the PCI bus to the PHY memory mapped space accesses one byte from the PHY device. To allow for 256 bytes of PHY control and status space 0x800 bytes of PCI memory map space is reserved for PHY access.

Each 64-bit PCI access generates only one access to the PHY. The upper DWORD is ignored. In addition, burst reads into PHY space cause a disconnect after the first quad word is transferred. If a burst read is performed, the bridge chip performs a retry with the address incremented. In PCI lite mode, dummy words are inserted after the first valid transfer of a burst.

Multi-PHY or extended addressing for a PHY is provided through a PHY page mechanism. This allows address bit to be appended to PHY control access without increasing the size of the PHY memory map as seen by the PCI. The PHYBANK field in the CONFIG1 control register provides up to 5 bits of page addressing for a total of 13 bits of PHY addressing.

11.2 Microprocessor Interface to PHY Device(s)

Microprocessor interface pins and descriptions are given in Table 11-1. Figure 11-1 shows the signal interface between the CN8237 and the RS825x ATM receiver/transmitter device. The PCS*, PAS*, and PWNR pins are outputs providing chip select, address strobe, and write/read control to the RS825x. PINT* is an input connected to the interrupt sources of the RS825x. The PDS* output is active and indicates the cycles in which the data transaction occurs. The PWAIT* input is active and can be used to prolong the cycle as shown in Figure 11-2. Physical interface devices other than Mindspeed PHY devices, such

as the RS825x, can be connected by using PWAIT* to extend the read or write cycle to the CN8237.

Table 11-1. Microprocessor Interface to PHY Devices Interface Pins

Signal	Dir(1)	Description
PCS*	O	Chip select output for PHY device number 1. Synchronous to PCLK.
PAS*	O	PHY address strobe. Synchronous to PCLK.
PWNR	O	PHY write/read select. A logic 1 on this output indicates a write cycle, a logic 0 indicates a read cycle. Synchronous to PCLK.
PDS*	O	PHY interface ready signal. A logic low on this signal at rising edge of PCLK indicates that the data cycle has been completed.
PWAIT*	I	PHY wait input. Allows external logic to insert wait states to extend data cycles. Only active when PDS* is active.
PINT*	I	PHY interrupt input, active low, level sensitive(2).
PADDR[12:0]	O	Address bus to PHY device.
PCLK	O	Interface clock to PHY device.
PDATA [7:0]	I/O	PHY Device Data Bus.
PRST*	O	Asserted by the CN8237 whenever the HRST* input is asserted.
PCLK	O	Interface clock to PHY device.

NOTE(S):
(1) Direction given with respect to the CN8237.
(2) See the HOST_ISTAT0 register for details.

Figure 11-1. SAR (CN8237) and OC-3 PHY (RS825x) Interface

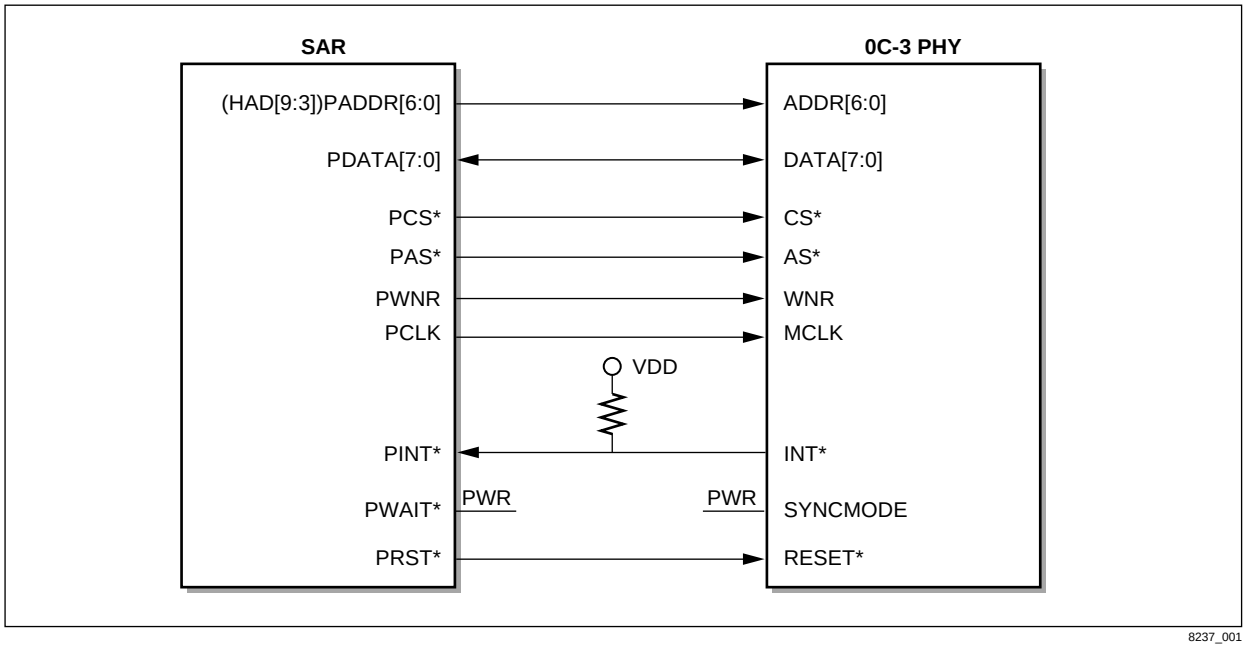


Figure 11-2. CN8237/PHY Functional Timing with Inserted Wait States (Normal Mode)

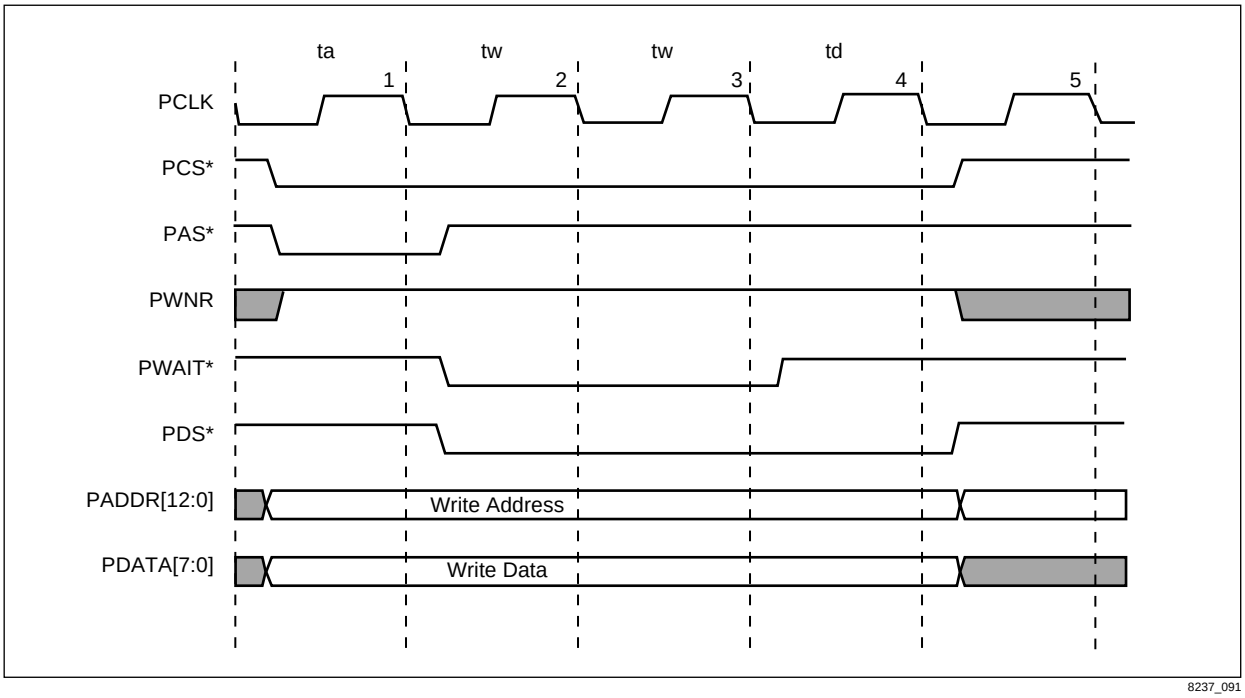
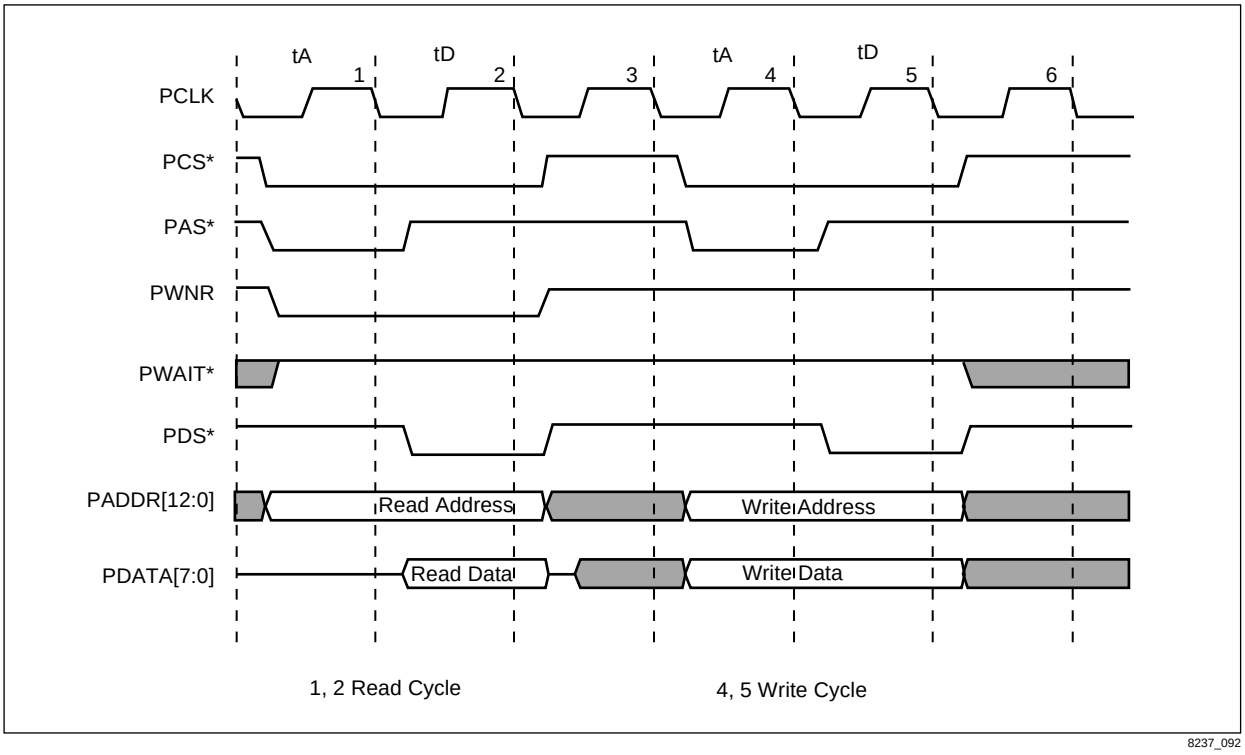


Figure 11-3 shows a read and write RS825x access. At cycle one (the rising edge of PCLK) the RS825x samples PCS*, PAS*, and PWNR low, indicating a read cycle. By the next rising edge of PCLK at cycle two, the data is output by the RS825x to be latched by the CN8237. The same procedure occurs for a write except that at cycle four, PWNR is sampled high. The RS825x then latches the data to the appropriate internal register on the next PCLK rising edge at cycle five.

Figure 11-3. CN8237/RS825x Read/Write Functional Timing (Normal Mode)



11.3 Microprocessor Interface for Multiple Physical Devices

Multi-PHY or extended addressing for a PHY is provided through a PHY page mechanism. This allows address bits to be appended to PHY control access without increasing the size of the PHY memory map as seen by the PCI. The PHY BANK field in the CONFIG1 control register provides up to five bits of page addressing for a total of 13 bits of PHY addressing. The contents of PHY BANK are placed on PADDR[12:8] during PCS* access.

The PDS* connection is only required where a data strobe handshake signal is required. Likewise, PWAIT* is only required where the returned read data is expected to be variable or slower than four PCLK clock cycles. In the event the PHY device does not require externally controlled wait states, the PWAIT* signal can be tied high. In a multi-device connection, this signal may be supported via a pull-up in typical wired-OR fashion. None of the devices shown in this document require externally provided PWAIT* control.

NOTE: In some applications, due to the specific timing of the interface signals, use of the PWAIT* signal can needlessly slow down the interface.

The PHY interrupt outputs shown here are open-drain outputs and require pullup resistors. This pull-up function is not provided at the PINT* input.

Figure 11-4 and Figure 11-5 show the signal interface between the PHY block package I/O and some typical PHY devices. CN8237 signals ending in '*' are active low.

NOTE: PHY parts in these diagrams use the '~' notation to denote the same.

Figure 11-4. Typical PHY Connection to RS825x

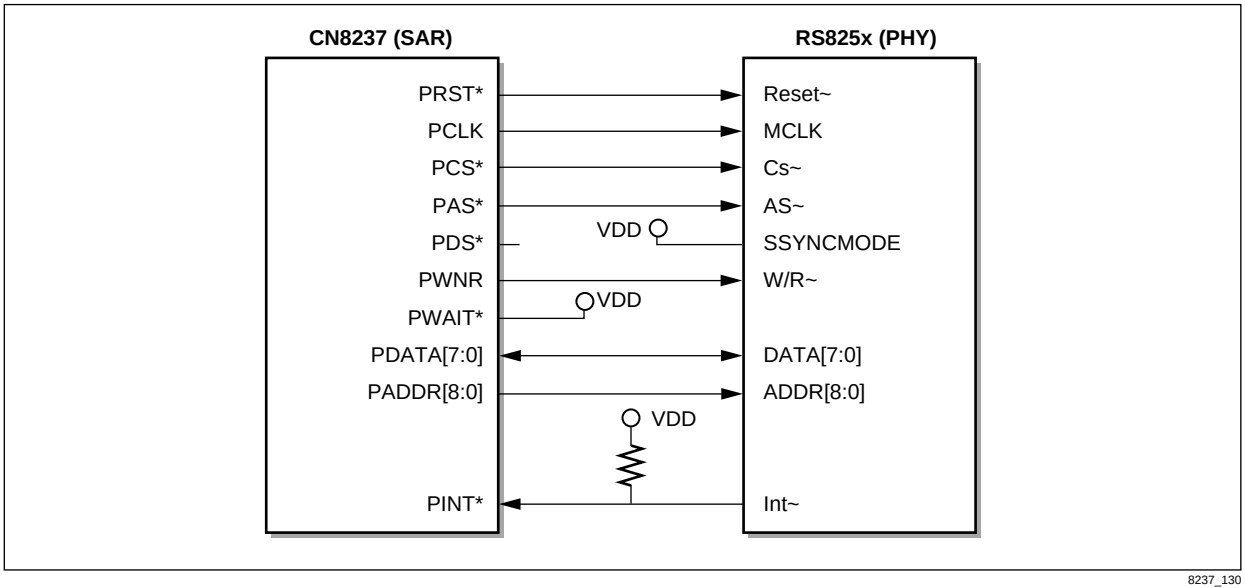
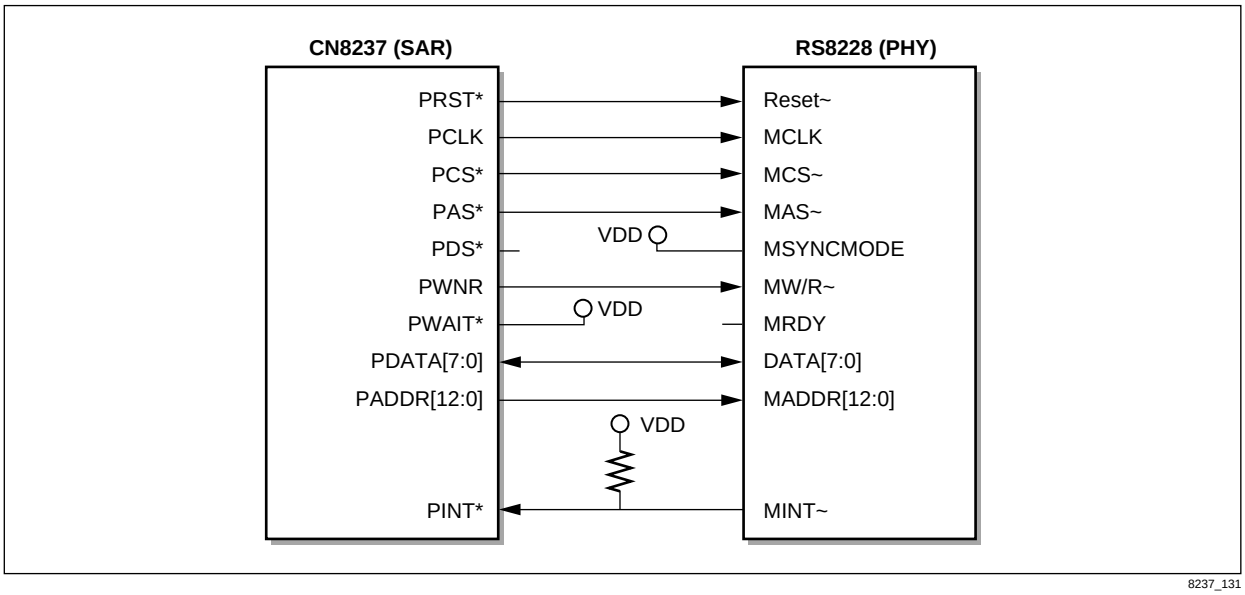


Figure 11-5. Typical PHY Connection to RS8228



11.4 Interface Control

The two modes of interface control are normal and strobe.

In normal mode, the external PHY device is clocked by PCLK which is generated by a dividing SYSCLK by two.

NOTE: PCLK is reset and held low while the HRST_ input is active.

Within the control register block, HRST_ active sets PHY_RST active high. Access to the control register by the PCI interface allows control of the PHY_RST control input which is inverted to drive the PRST* output to the PHY device. This allows the PRST* output to be controlled via software.

NOTE: HRST_ always sets this output high.

In normal mode, all other interface control signals are clocked internally by SYSCLK and transition at a PCLK falling edge. This provides one SYSCLK period of setup and one SYSCLK period of hold for all PHY interface signal transitions. It is expected that in normal mode, the external interface uses the rising edge of PCLK for synchronous processing of the interface and handshake signals. This allows setup and hold time on these signals around each access to be met.

In strobe mode, the same sequence applies but the PCLK output is typically not used.

The PHY_BANK control register bits allow setting of the upper five address bits of the outgoing PADDR bus via a software accessible control register. This provides a paged type interface to the PHY device which extends the fully addressable range while limiting the amount of the available PCI address space that this interface consumes.

NOTE: The three lower address bits of the PCI address bus are not used by the PHY block. Because a byte-only interface is supported and the PCI interface is a full 64-bit, quad-word interface, the upper bits of all accesses are ignored on writes and return zeros on reads. The addressing is shifted to accommodate this as is standard fashion in this type of interface.

11.4.1 Strobe Mode

To activate the strobe mode, set PHY_STROBE_IO (bit 29 of the CONFIG1 register) high. In strobe mode, PAS*, PDS*, and PWNR are redefined to be an active high address strobe, an active low read enable strobe, and an active low write enable strobe, respectively. This provides a glueless interface to most PHY devices using standard RAM-type interface signals. See the timing diagram in Section 16.1.4 for more information.

11.4.1.1 Wait Mode

Wait states can be internally generated by setting PHY_WAIT (bits 27 and 28 of the CONFIG1 register) to 1, 2, or 3, while a setting of zero disables internal wait state generation. When set to other than zero, the interface state machine automatically inserts the appropriate number of wait states during the data phase of the access or during the read/write strobe time if in strobe mode. The PWAIT* input remains enabled and can generate additional externally controlled wait states.

NOTE: The internally set wait period cannot be shortened by use of the PWAIT* input.

12.0 PCI Bus Interface

12.1 Overview

The PCI bus interface is compliant with *PCI Local Bus Specification*, Revision 2.1. With the exception of HRST* and HINT*, this interface is completely synchronous to the PCI bus clock (HCLK). All inputs are sampled at the rising edge of HCLK, and all outputs are driven by the CN8237 to be valid before the next rising edge of HCLK.

The maximum PCI bus clock rate supported by the CN8237 is 66 MHz. The PCI bus interface logic is clocked directly from the PCI bus clock, while the remainder of the CN8237 logic runs off separate clocks. Synchronizing registers and FIFOs are implemented in the PCI bus interface in order to transfer data between the PCI bus clock (HCLK) and the system (SYSCLK) clock domains.

The PCI bus drivers are shared between master and slave bus interface functional blocks. The PCI bus master logic (within the device) arbitrates via the PCI bus arbiter (external to the device) for access to the PCI bus; access to the PCI bus automatically implies access to the bus drivers, since no other master can be concurrently communicating with the slave logic. The bus master logic contends for the bus on a transaction by transaction basis.

The PCI bus interface responds to read and write requests by the host CPU, allowing access to chip resources by host software. The CN8237 can also act as DMA bus master on the PCI bus. As a result, the PCI bus interface implements the full set of address, data, and control signals required to drive the bus as master, and contains the logic required to support arbitration for the PCI bus. The DMA coprocessor and the PCI bus interface are closely linked and, hence, are shown as one unit.

The PCI bus interface functional blocks are as follows:

- I/O drivers and receivers that drive the pins connected to the PCI bus signals.
- PCI bus master logic that allows the bus interface to acquire mastership of the PCI bus and act as a transaction initiator. The bus master logic also contains a command decoder that interprets access commands generated by the DMA coprocessor, and a burst controller for controlling the duration of each read or write burst. In addition, the bus master logic contains address counters that allow it to restart and retry burst transfers if required by the transaction target.
- Burst FIFO buffers that store and transfer bursts of data words between the DMA coprocessor and the PCI bus master logic.
- PCI bus slave logic that responds to transactions initiated by other masters on the PCI bus with the CN8237 as a target. The bus slave logic also synchronizes data passed back and forth across the clock boundary between the PCI bus interface and the internal chip logic.
- Configuration registers holding initialization parameters and PCI bus status information.
- Logic that allows the host CPU to read/write the internal CN8237 registers via the PCI slave port.
- Logic to enable read/write access to the SAR-shared memory space from the host CPU, again via the PCI slave port.
- An Interface module that allows the PCI core to connect to a serial EEPROM.
- Implement Sections 3.1.8 and Section 7.2 of the *Compact PCI Hot Swap Specification*. Assume that a logic low on the HSWITCH* input is switch locked and a logic high is switch unlocked. The HSWITCH* input to the SAR is asynchronous and may cause the state machine to transition to the wrong state. Use the PCI clock to latch HSWITCH* and then provide the latched signal to the SAR. Put PCI Reset on the flip flop to clear the latch on power up.

NOTE: The arming/disarming of the INS/EXT bits provides a switch debounce function.

If HSWITCH* is not used, it must be tied to ground.

12.2 Unimplemented PCI Bus Interface Functions

The PCI bus interface on the CN8237 does not implement all the transaction types defined by the PCI bus specification; only those sections of the protocol that are necessary for slave and DMA memory accesses are implemented. In particular, the following transaction types are not implemented:

- The Dual Address Cycle command.
- Snooping and cache support. Memory Read Line, Memory Write, and Invalidate commands are internally aliased to the Memory Read and Memory Write commands as per the PCI specification.
- Locked and exclusive accesses: the PCI LOCK* line is not driven by the CN8237, and the PCI slave interface does not handle locked accesses by other bus masters in any special manner.
- I/O accesses (the I/O Read and I/O Write commands).
- Interrupt acknowledge cycles, including the Interrupt Acknowledge command.
- The Special Cycle command and Special Cycle transactions.
- Burst transfers that do not have simple, sequentially incrementing addresses for consecutive data phases. The PCI master logic always performs sequentially incrementing burst transfers. The two LSBs of the PCI address lines (AD[1,0]) must be 0 during the address phase of any transfer made to the PCI slave logic (indicating sequentially incrementing burst addresses). If AD[1,0] is not equal to 0, the slave logic signals a type A or B target disconnect after the first data phase, forcing the external master to perform a single word transfer as per the PCI specification.

12.3 PCI Configuration Space

In accordance with the *PCI Bus Specification*, Revision 2.1, the CN8237 PCI bus interface implements a 128-byte configuration register space. These configuration registers can be used by the host processor to initialize, control, and monitor the SAR bus interface logic. The complete definitions of these registers and the relevant fields within them is given in the PCI bus specification, and will not be repeated here. The descriptions and definitions of these register fields as implemented in the CN8237 are shown in Chapter 14.0.

The incoming DMA FIFO size is programmable and may be set to 2 KB or 8 KB depending on the value of the INCFIFO_SZ bit in the CONFIG1 register.

12.4 PCI Bus Master Logic

The PCI bus master logic block is responsible for accepting read and write commands from the DMA coprocessor (passed via the burst FIFO buffers), and in turn acquiring mastership of the PCI bus and generating transactions to perform the actual data transfers. The bus master logic contains the following:

- A command decoder that interprets commands issued from the DMA coprocessor.
- A burst controller that counts off read and write cycles in each burst on the PCI bus (and also latches and drives the address and command during the address phase of each transfer).
- Arbitration logic that acquires control of the PCI bus.
- Supported arbitration parking.
- A bus state machine that sequences and controls transfers.

It is possible for the addressed slave to request a disconnect or a retry during a read or a write transfer, using the defined PCI protocol sequence. In this case, the bus master logic terminates the current burst, maintain its bus request, and restarts the transfer at the point of termination. Disconnects and retries are not regarded as errors.

Five possible sources of error are present during any PCI bus master transaction. If any of the following five errors occur, the bus master logic permanently terminates the transaction, flags an error, and ceases to process any more commands.

- Target Abort—The PCI transaction terminates if the addressed target signals a target abort. In this case, the RTA and MERROR bits in the PCI Configuration register space are set and the PCI_BUS_STATUS[4] bit in the SYS_STAT register is set.
- Master Abort—If the addressed target does not respond with an HDEVSEL* assertion, then a master abort is flagged. In this case, the RMA and MERROR bits in the PCI Configuration register space are set, and the PCI_BUS_STATUS[3] bit in the SYS_STAT register is set.
- Parity Error—If the data parity checked during read transfers is inconsistent with the state of the HPAR signal, then a parity error is signaled. In this case, the DPR and MERROR bits in the PCI Configuration register space are set and the PCI_BUS_STATUS[2] bit in the SYS_STAT register is set.
- Interface Disabled—If the driver or application software on the PCI host CPU has disabled the CN8237 PCI bus master logic (using the M_EN bit in the Command field of the PCI bus configuration registers), then any attempt to perform a DMA transaction to the PCI bus results in an error. In this case, the MERROR and INTF_DIS bits in the PCI configuration space are set and the PCI_BUS_STATUS[1] bit in the SYS_STAT register is set.
- Internal Failure—Upon a synchronization error between the DMA coprocessor and the PCI master logic, an internal failure is flagged. In this case, the MERROR and INT_FAIL bits in the PCI configuration space are set, and the PCI_BUS_STATUS[0] bit in the SYS_STAT register is set.

NOTE: The above errors permanently affect system level operation. Because of this the system should be re-initialized, since full system level recovery is unlikely. The bus protocol errors can be cleared either by a software reset of the associated status flag or flags, that is, RTA, RMA, or DPR, or with a reset of the PCI bus master logic using the HRST* input pin. For example, a master abort error can be cleared by writing a logic 1 to the RMA status bit in the PCI Configuration register space, causing the status bit to be cleared. Internal failures (attempting to initiate a master transaction with the interface disabled, or loss of synchronization with the DMA controller) can only be reset by applying the global reset, CONFIG0 (GLOBAL_RESET), or by asserting the HRST* signal.

Next, the MERROR bit must be cleared. The MERROR bit in the PCI Configuration register drives the PCI_BUS_ERROR interrupt. To clear this interrupt, a logic high must be written to the MERROR bit location. The MERROR bit can also be cleared by a logic low on the HRST* input pin.

Several fields are provided in the PCI configuration space to aid in recovering from a PCI master error. The PCI host software can determine that an error occurred by checking the MERROR bit. It can also determine if the transaction was a read or write by inspecting the MRD bit, and then retrieve the read or write address at which an error occurred by reading the MASTER_READ_ADDR or MASTER_WRITE_ADDR fields.

The PCI Read and PCI Read MULTIPLE commands issued by the PCI block are under the control of PCI_READ_MULTI bit 22 in the CONFIG0 register.

NOTE: If all targets are 64 bit, then the SAR can be programmed to master 64-bit transactions only. This results in a modest performance increase. This is programmable with the FORCE64 (bit 14 in CONFIG0).

12.5 Burst FIFO Buffers

Two small FIFO buffers are implemented to support PCI slave burst-mode operation (Read = 8 × 32, Write = 64 × 32), to allow synchronization between the CN8237 internal logic and the PCI bus interface, and to carry commands from the DMA coprocessor to the PCI bus logic. The incoming master FIFO is 512 × 32 or 2 K × 32 bits, the outgoing Master FIFO is 16 × 36 bits.

12.6 PCI Bus Slave Logic

The PCI slave logic permits the host CPU on the PCI bus to access and modify CN8237 resources (the external SAR-shared memory, internal memory, and internal registers). Because the control processor also has access to these resources, the PCI slave logic must arbitrate for access prior to performing any read or write transaction. The slave logic also contains the PCI configuration registers. These registers control the PCI slave and master interfaces, and can be read or written at any time by the PCI host. The slave logic implements the synchronizers required for rate-matching between the PCI bus clock and the internal CN8237 system clock. Also, small FIFOs are used to speed up burst reads and writes performed by the host processor to local resources, by buffering prefetched read data and absorbing latency during consecutive writes.

In general, the PCI slave interface functions as a normal memory-mapped PCI target, responding to Memory Read, Memory Write, Configuration Read, and Configuration Write commands from any initiator on the PCI bus. The slave interface responds only to Memory Read and Memory Write commands if the MS_EN bit of the Command field in the PCI Configuration register has been set.

The PCI slave logic does not implement special cycle commands, or respond to special cycles on the PCI bus. If a master performs a special cycle on the PCI bus, the following occurs:

- The slave logic never asserts HDEVSEL*.
- Parity errors during the address phase of the special cycle command are reported to be asserting HSERR* in the normal fashion, if SE_EN and PE_EN in the command register are both set.
- Parity errors during the data phase are ignored.

12.7 Byte Swapping and Dword Conversion of Control Structures

Four control bits in the PCI configuration space are used to configure byte swapping, in order to align with various big and little endian host system requirements.

The SLAVE_SWAP control bit is bit 29 of address offset 0x40 in the PCI Configuration register. When SLAVE_SWAP is set to a logic high, the slave interface swaps the bytes of a slave write or read access. The default setting for this bit is logic low.

The MSTR_CTRL_SWAP control bit is bit 30 of address offset 0x40 in the PCI Configuration register. When MSTR_CTRL_SWAP is set to a logic high, the control structures that the SAR writes are written with bytes swapped. The default setting for this bit is logic low.

Under programmable control by bit 28 (Slave Dword Conversion) in the PCI Special Status Register (PCI Config Address 0x40), either bits [31:0] or bits [63:32] are mapped to address 0x00.

Bit 26 (Master Control Dword Conversion) in the PCI Special Status Register (PCI Config Address 0x40) controls the Control Dword Conversion.

The HRST* pin made active causes both of these bits to be logic low.

Refer to Section 8.4 and Section 8.5 for more information.

12.8 Power Management

Power Management, as a defined class of functions, consists of mechanisms in software and hardware to minimize system power consumption, manage system thermal limits, and maximize system battery life. The CN8237 supports Power Management on the PCI bus according to the *PCI Bus Power Management Interface Specification*, Revision 1.0.

Power management states are defined as varying, distinct levels of power savings. The CN8237 device supports the two mandatory power states, D0 and D3, of the PCI Bus Power Management Interface Specification. D0 is the maximum powered state (on) and D3 is the minimum powered state (off). When in the D3 state, SYSCLK is turned off.

Refer to Section 14.7 for the detailed information on the PCI Configuration space and PCI registers concerned with implementing the Power Management functions.

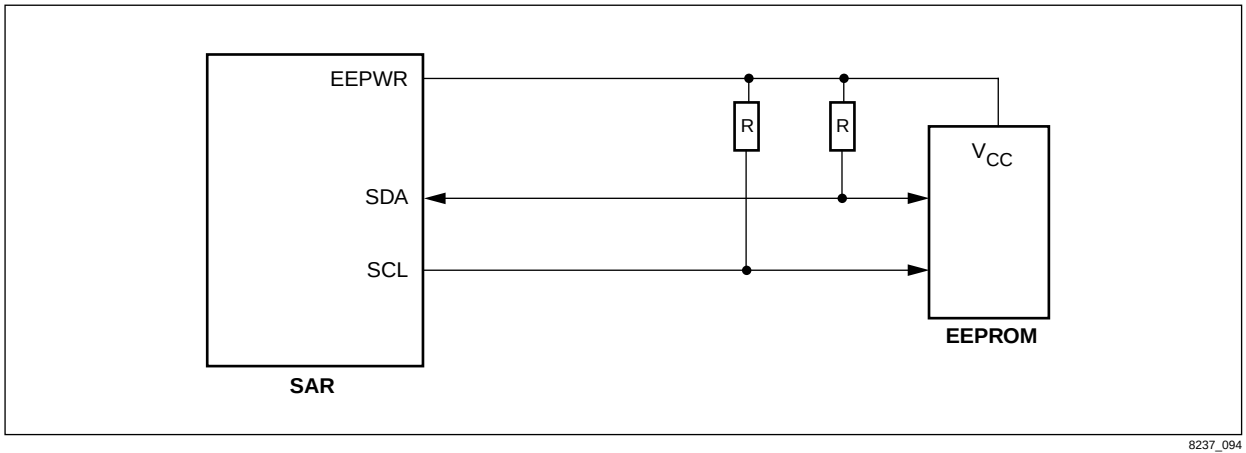
Power Management is enabled by default. This capability can be disabled using bit 2 of the EEPROM (Disable Capability register), which sets bit 20 of the PCI Status register to a logic low. See the *PCI Bus Power Management Interface Specification*, Revision 1.0, for specific information on the functions involved in Power Management.

12.9 Interface Module to Serial EEPROM

The Interface Module implements the protocol to allow the PCI core to connect to a serial EEPROM.

A condition can arise where the protocol is violated and unknown operation of the EEPROM can occur. If HRST* is applied while the EEPROM is being accessed, that is, right after HRST* goes inactive, the access is abnormally terminated with indeterminate effects on the EEPROM. In order to reset the EEPROM, its power must be cycled. A 12 mA pin, EEPWR is added to supply power to the EEPROM if the above condition cannot be guaranteed from happening. Whenever HRST* is active, EEPWR is a logic low. When HRST* goes to a logic high (inactive), EEPWR immediately goes to a logic high. Figure 12-1 illustrates the suggested connection of the EEPROM.

Figure 12-1. EEPROM Connection



12.9.1 EEPROM Format

The first 32 bytes of the 128-byte EEPROM are used to store PCI configuration information, loaded into the PCI Configuration space at reset. Unless otherwise specified, all unused bytes are reserved and should be programmed to 0x00. Bytes above address offset 0x20 can be used by application software or device drivers as needed. The EEPROM fields are described in Table 12-1.

Table 12-1. EEPROM Fields

Address Offset	Name	Description
0x00	FIELD_ENABLES	Bit 6: Load PLL_MUL from EEPROM Bit 5: Load Memory Size Mask from EEPROM Bit 4: Load Latency Timer from EEPROM Bit 3: Load General Enables from EEPROM Bit 2: Disable Capability registers (for Power Management) Bit 1: Load Subsystem ID (SID) from EEPROM Bit 0: Load Subsystem Vendor ID (SVID) from EEPROM
0x01-0x03	Reserved	Set to 0s.
0x04-0x05	SVID	Subsystem Vendor ID.
0x06-0x07	SID	Subsystem ID.
0x08	General Enables	Bit 7: Special Status register Bit 26 (MSTR_CTRL_DWORD, Master Control DWORD) Bit 6: Special Status register Bit 27 (MSTR_DATA_DWORD, Master Data DWORD) Bit 5: Special Status register Bit 28 (SLAVE_DWORD, Slave DWORD) Bit 4: Special Status register Bit 29 (SLAVE_SWAP, Slave Control Byte Swap) Bit 3: Special Status register Bit 30 (MSTR_CTRL_SWAP, Master Control Byte Swap) Bit 2: PCI Command register Bit 6 (PE_EN, Enable Detection of Parity Errors) Bit 1[(1)]: PCI Command register Bit 2 (M_EN, Master Enable) Bit 0[(1)]: PCI Command register Bit 1 (MS_EN, Memory Space Enable)
0x09	Latency Timer	Master Latency Timer
0x0a	Memory Size Mask	Valid Mask Values: Bit 7 6 5 4 3 2 1 0 = Size 0 0 0 0 0 0 0 0 = 16 M 1 0 0 0 0 0 0 0 = 8 M
0x0b	PLL_MUL	Bits 7–4: Reserved Bits 3–0: PLL_MUL

NOTE(S):
(1) Do not set until after PCI Base Address register has been set.

12.9.2 Loading the EEPROM Data at Reset

At reset, the PCI Configuration block first reads byte 0x00 of the EEPROM to determine which fields of the EEPROM should be read. It first looks at the FIELD_ENABLES bits. If bit 2 is set, it sets bit 20 in the PCI Status register to 0, to disable Power Management capabilities. It then looks at bits 1 and 0 to see if the corresponding SVID and/or SID fields are to be loaded into the corresponding PCI Configuration register fields. If either of these is set to 0, the SUBSYSTEM_ID and/or SUBSYSTEM_VENDOR_ID fields will default to all 0s.

12.9.3 Accessing the EEPROM

The EEPROM is accessed through the PCI Configuration space at offset 0x4C, the EEPROM register. See Section 14.7 for a description of the EEPROM register's contents.

Before starting an EEPROM operation, the BUSY bit must be a logic 0 (not busy). When in this state, the EEPROM can be either written to or read from. After initiating a read or write operation, the BUSY bit will be a logic 1 (busy) until the transfer completes. During this time, the application software must poll the BUSY bit to determine when the transfer has completed. Once completed, the NO_ACK bit indicates the status of the operation. A logic 1 (no acknowledge) indicates that no device responded to the request.

To initiate a write operation, the application software must write the BYTE_ADDR and DATA fields and set the READ_WRITE bit to 0. The module then transfers the data in bits 7:0 (the DATA field) to the device on the bus at the hardware address at the BYTE_ADDR specified. Application software then polls the register until the BUSY bit is read as 0 (not busy), which indicates that the transfer has completed. Software must then check the NO_ACK bit to ensure that the transaction completed normally. If not, the software should retry the transaction or signal the error to the user. Since the EEPROM might not respond until after a few milliseconds after a write transaction, it is recommended that all operations resulting in NO_ACK=1 be retried several times before issuing the failure.

For read operations the application software must also write to the EEPROM register, specifying the BYTE_ADDR to be read and setting the READ_WRITE bit to 1. The software must then poll the BUSY bit until the operation completes. At this point the data is returned in the DATA field of the EEPROM register. The software should check the NO_ACK bit to ensure proper completion of the transfer with no error.

12.9.4 Using the Subsystem ID Without an EEPROM

For applications that can use BIOS or boot code to initialize devices before loading high-level operating system software, the CN8237 allows for the programming of the PCI Configuration space fields, SUBSYSTEM_ID and SUBSYSTEM_VENDOR_ID. This feature allows a user to employ the CN8237 without an EEPROM, but still allowing for unique Subsystem IDs.

To program the SUBSYSTEM_ID and SUBSYSTEM_VENDOR_ID fields, BIOS must first write a logic 1 to bit 31 of the PCI Special Status register. This enables the writing to these two fields in the PCI Configuration space. BIOS can then update the IDs by writing the desired values to the PCI Configuration space at offset 0x2C. Once the values are written, BIOS should then disable the writing to these fields by setting bit 31 of the PCI Special Status register to logic 0. When bit 31 is set to 0, writes to these two fields are ignored.

12.10 PCI Host Address Map

The address map of the CN8237 resources seen by the PCI bus is the same as that seen by the host processor. The base address of the CN8237 resource mapping is defined in the BASE_ADDRESS_REGISTER_0 field, located in the PCI configuration space.

Burst reads of the Control and Status registers only return valid data for the first address. Subsequent data words read during a burst read are indeterminate.

13.0 ATM UTOPIA Interface

13.1 Overview of ATM UTOPIA Interface

The ATM UTOPIA interface contains receive and transmit interface logic and receive error detection logic. The ATM UTOPIA interface block also interfaces with the segmentation and reassembly coprocessors.

The ATM UTOPIA interface for the CN8237 accepts 52 octet cells from the segmentation coprocessor and transmits them to the PHY device while inserting a dummy HEC. The interface also receives 53 octet cells from the PHY device, removes the HEC, and saves them in a FIFO buffer to be used by the reassembly coprocessor.

The ATM UTOPIA interface is responsible for communicating with and controlling the ATM link interface device, which carries out all the transmission convergence and physical media dependent functions defined by the ATM protocol. The block performs the following functions:

- Receives and transmits ATM UTOPIA interface logic. The ATM UTOPIA interface accommodates the Mindspeed RS825x, RS8228, or Bt8223 UTOPIA layer devices, a UTOPIA-compatible framer or a Mindspeed-conceived slave UTOPIA interface, and is responsible for converting between these devices and the internal data interfaces. The Slave UTOPIA interface connects the CN8237 to a cell-switched backplane.
- Receives cell synchronization logic, which validates cell boundaries in the incoming byte stream, strips off the HEC byte from the ATM header, and formats the remaining 52 bytes into thirteen 32-bit words before passing them to the incoming cell FIFO buffer. The receive cell synchronization logic ensures that only complete cells are passed down to the remainder of the reassembly controller.
- Transmits cell synchronization logic, which converts the 32-bit data read from the transmit cell FIFO buffer into 8- or 16-bit (plus parity) streams, generates appropriate cell delineation pulses for use by the transmit ATM UTOPIA interface, and inserts the blank HEC byte into the ATM header of each cell prior to transferring it to the UTOPIA interface.
- Generates and checks odd parity on the octet transmit and receive data buses.

13.2 ATM UTOPIA Interface Logic

The CN8237 ATM UTOPIA interface logic consists of the I/O drivers required to communicate with the external framer device, together with adaptation logic required to convert between either the UTOPIA or slave UTOPIA interface protocol, and the internal byte streams. Configuration pins FRCFG and UTOPIA1 determine whether the UTOPIA or slave UTOPIA protocol is used and whether it is level 1 or level 2.

13.3 ATM Physical Interface I/O Pins

The operational mode desired is indicated to the CN8237 by appropriately driving the FRCFG and UTOPIA1 inputs according to Tables 13-1 and 13-3.

Table 13-1. ATM Physical Interface Mode Select (FRCFG)

FRCFG	ATM Physical Interface Mode
0	Slave UTOPIA
1	UTOPIA

Table 13-2. ATM Physical Interface Mode Select (UTOPIA1)

UTOPIA1	ATM Physical Interface Mode
0	UTOPIA Level 2
1	UTOPIA Level 1

The interpretation of the ATM UTOPIA interface pins on the CN8237 and the actual signals generated or received by the framer in UTOPIA mode are shown in Table 13-3. Both the TxCLK and RxCLK signals of the UTOPIA interface can be derived from the CLKD3 output of the CN8237.

Table 13-3. UTOPIA Mode Signals

CN8237 Signal	PHY Signal	Active Polarity	CN8237 Direction
TxDATA[15:0]	TxDATA[15:0]	—	Out
TxPAR	TxPRTY	—	Out
TxSOC (TxMARK)	TxSOC (TxMARK)	High	Out
TxEN*	TxENB*	Low	Out
TxCLAV (TxFLAG*)	TxFULL*	Low	In
TxADD[4:0]	TxADDR[4:0]	—	Out
TxCLK	TxCLK	Rising Edge	In
RxDATA[15:0](1)	RxDATA[15:0]	—	In
RxPAR	RxPRTY	—	In
RxSOC (RxMARK)	RxSOC (RxMARK)	High	In
RxEN*	RxENB*	Low	Out
RxCLAV (RxFLAG*)	RxEMPTY*	Low	In
RxCLK	RxCLK/TxCLK	Rising Edge	In
RxADD[4:0]	RxADDR[4:0]	—	Out

NOTE(S):
(1) When operating the CN8237 in 8-bit mode, pull-ups need to be added to the unused high order eight bits (RxDATA[15:8]).

The interpretation of the ATM UTOPIA interface pins on the CN8237 and the actual signals generated or received by the framer in slave UTOPIA mode is shown in Table 13-4.

Table 13-4. Slave UTOPIA Mode Interface Signals

CN8237 Signal	PHY Signal	Active Polarity	CN8237 Direction
TxDATA[15:0]	TxDATA[15:0]	—	Out
TxPAR	TxPRTY	—	Out
TxSOC (TxMARK)	TxSOC (TxMARK)	High	Out
TxEN*	TxENB*	Low	In
TxCLAV (TxFLAG*)	TxEMP*	Low	Out
RxDATA[15:0](1)	RxDATA[15:0]	—	In
RxPAR	RxPRTY	—	In
RxSOC (RxMARK)	RxSOC (RxMARK)	High	In
RxEN*	RxENB*	Low	In
RxCLAV (RxFLAG*)	RxFULL*	Low	Out
TxCLK	TxCLK	Rising Edge	In
RxCLK	RxCLK	Rising Edge	In
TxADDR[4:0]	TxADDR[4:0]	—	In
RxADDR[4:0]	RxADDR[4:0]	—	In

NOTE(S):
When operating the CN8237 in 8-bit mode, pull-ups need to be added to the unused high order eight bits (RxDATA[15:8]).

13.3.1 UTOPIA Interface

In addition to the current UTOPIA level 1 support, this interface provides complete support for UTOPIA level 2, 8/16 bit mode in both master and slave modes, including support for multi-PHY in both master and slave modes. Multi-PHY port shaping is provided by tunnels through one TxFIFO buffer. A 3-bit PORT_ID provides port identification for up to eight ports in master mode (UTOPIA address 0 through 7) and 32 ports in slave mode, where the SAR is programmed for one port I.D. The UTOPIA output drives are 8 mA to meet the multi-PHY specification.

Multi-PHY mode is enabled by setting CONFIG1(MULTI_PHY) to a logic high. In master mode, the PORT_ID field is used as a port identification. On the segmentation side, the appropriate port identification is written in the PORT_ID field in the VCC table entry. The maximum allowable PORT_ID is CONFIG1(NUM_PORTS). On the reassembly side, the UTOPIA interface polls addresses 0 through CONFIG1(NUM_PORTS) for a cell. When a cell is detected in a PHY, the cell along with the PORT_ID is transferred to the reassembly block. An expanded lookup mechanism is used to find the appropriate state table entry. See Section 5.2.2.4 for more details.

In slave mode, the UTOPIA block responds only when the UTOPIA address equals CONFIG1(SLAVE_ADDR). In non-multi-PHY master mode, CONFIG1(SLAVE_ADDR) is output on the RxADDR and TxADDR busses.

Also, routing tag prepending is supported with programmable cell size up to 64 bytes in single PHY master mode and single/multi-PHY slave mode. On the reassembly side, the tag is discarded by the UTOPIA interface.

When F4 PMOAM operation or VP ABR operation is enabled, the routing tag content of all cells including OAM within the VP must be identical.

13.4 UTOPIA Level 2 Interface

The CN8237 supports both UTOPIA Level 1 and Level 2 interfaces. These are described in general terms below with an emphasis on their differences.

- UTOPIA Level 1—This is an 8- or 16-bit interface designed for data rates up to 200 Mbps at a clock rate of 25 MHz. Both Octet-level and Cell Level handshaking are supported.
- UTOPIA Level 2—This interface defines the Multi-port support features that allow up to 31 UTOPIA devices to multiplex on one UTOPIA bus. The SAR only supports up to eight PHY devices. It allows either 8- or 16-bit data buses and uses only cell level handshaking. It can transfer up to 800 Mbps when running at 50 MHz in the 16-bit mode.

Both Level 1 and Level 2 support odd parity over the width of the data bus.

13.4.1 Cell Tagging

In addition to the standard 53-octet cell formats, given in Tables 13-5 and 13-6, the CN8237 supports user programmable cell sizes for Routing Tag applications. This allows a maximum cell size of 64 bytes. These cell formats are shown in Tables 13-7 and 13-8. The number of octets added for the tagging function is programmed into the TAG_SIZE bits of CONFIG1.

Table 13-5. Cell Format 8-bit Mode

Bit 7 ... 0
Header 1
Header 2
Header 3
Header 4
UDF1 (HEC) (byte 5)
Payload 1
:
Payload 48

Table 13-6. Cell Format 16-bit Mode

Bit 15 ... Bit 8	Bit 7 ... 0
Header 1	Header 2
Header 3	Header 4
UDF1 (HEC) (byte 5)	UDF2 (0) (byte 6)
Payload 1	Payload 2
⋮	⋮
Payload 47	Payload 48

Table 13-7. Cell Format, Tagging Enabled, 8-bit Mode

7 0	
Tag 1	1st byte of Cell
⋮	
⋮	
Tag n	
Header 1	
Header 2	
Header 3	
Header 4	
HEC	
Payload 1	
Payload 2	
⋮	
⋮	
⋮	
Payload 48	Last byte of Cell

Table 13-8. Cell Format, Tagging Enabled, 16-bit Mode

15 8	7 0	
Tag 1	Tag 2	1st word of Cell
⋮	⋮	
⋮	⋮	
Tag n-1	Tag n-2	
Header 1	Header 2	
Header 3	Header 4	
HEC	Not Used	
Payload 1	Payload 2	
⋮	⋮	
⋮	⋮	
Payload 47	Payload 48	Last word of Cell

13.4.2 UTOPIA Configuration Control

Most options for the CN8237 UTOPIA interface are configured by control bits in the CONFIG1 registers. However, the selection of UTOPIA Level 1 versus Level 2 and of Master or Slave mode is controlled by the input pins FRCFG and UTOPIA1. Normally, these inputs are hardwired as required by the system design.

UTOPIA Level 1/Level 2—Tying the UTOPIA1 input pin to ground selects UTOPIA level 2, connecting it to +3.3 V selects UTOPIA Level 1. When selecting UTOPIA Level 2 be sure to program the UTOPIA_MODE bit to logic high to enable Cell level handshaking (octet level handshaking is not valid in UTOPIA Level 2).

Master/Slave selection—Controlled by the FRCFG hardware input as defined in Table 13-1.

Octet or Cell Handshaking—This is controlled by the UTOPIA_MODE bit in the CONFIG1 register. Setting this bit high selects cell-level handshaking. In this mode, both the CN8237 and the selected port only begins a transfer when it has room in its FIFO buffer's for an entire cell. When UTOPIA Level 2 mode is selected this bit MUST be set to logic high.

Multi-port operations—By default, the CN8237 expects a single device on the UTOPIA bus. By setting the MULTI_PHY bit of CONFIG1 to logic high, up to eight devices can share the bus.

UTOPIA data bus width—The width of the UTOPIA data bus is selectable by the UTOP16 bit of the CONFIG1 register. Setting this bit high enables a 16-bit data path.

NOTE:

1. Number of ports on the bus—When running in multi-port mode the total number of ports to poll is programmed into the NUM_PORTS bits of CONFIG1.
2. Since port numbering begins with zero, this value equals the total number of ports minus one.

Slave address—When the CN8237 is configured as a slave on the UTOPIA bus its port address must be programmed into the SLAVE_ADDR bits of CONFIG1. Software must be aware that all CN8237s on the bus will default to the same address, (00), and therefore, must reprogram each device to a unique value.

13.4.3 UTOPIA Level 2 Multi-Port Operation

The CN8237 supports up to eight multi-port operations as described in the *UTOPIA Level 2 Specification af-phy-0039.000* (see the ATM forum web site for details: www.atmforum.com). Two primary functions are involved in transferring data on the UTOPIA bus: polling the ports to determine which ones have data ready and then selecting which port transfers its data. The receive side is discussed below. The reader can refer to the above referenced web page for more details.

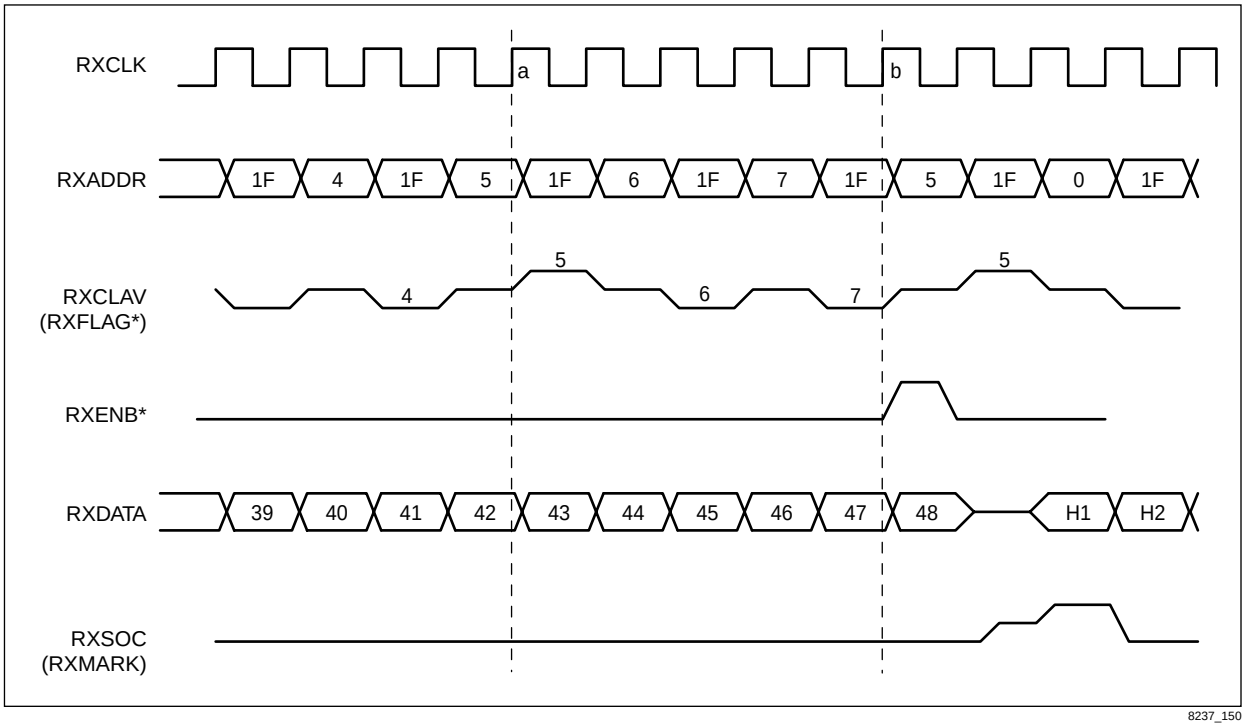
The CN8237 starts by polling port 0 and continues to NUM_PORTS as stored in the CONFIG1 register. Polling is accomplished by outputting the desired port number on the UTOPIA address bus. If that port has data ready, it asserts the RxClav (RxFLAG*) line. The controller ignores this line and continues polling other ports or it initiates the data transfer by holding the port address on the address lines while RxEnb* is deasserted. Once the data transfer has begun, the CN8237 continues polling other ports.

The polling and selection process waveforms are shown in Figure 13-1. This diagram assumes that eight ports are present. At clock cycle “a”, port 5 has just been polled. It asserts the RxClav (RxFLAG*) line to indicate that it has a complete cell to send. The CN8237 continues to poll ports 6 and 7 (both indicate that they do not have cells at this time). During this entire polling operation, data is being transferred across the data bus.

At clock cycle “b”, the CN8237 has again output port 5 on the UTOPIA address bus but this time has raised the RxEnb* line. This selects port 5. Port 5 acknowledges that it is ready by again asserting RxClav (RxFLAG*). Port 5 then asserts the RxSOC (RxMARK) line and begins to transfer data.

NOTE: CN8237 does not poll on the transmit side.

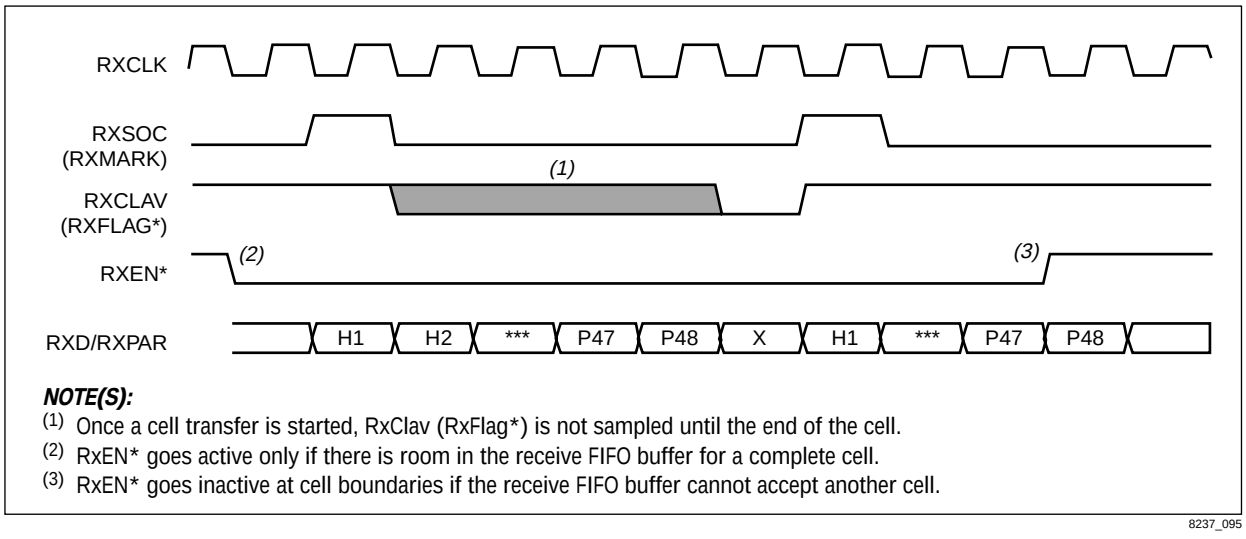
Figure 13-1. UTOPIA Level 2 Receive Timing



13.5 UTOPIA Level 1 Mode Cell Handshake Timing

If high, the UTOPIA_MODE bit in the CONFIG0 register selects cell-level handshaking. Received data is latched from the RxDATA[15:0] and RxPar lines on the rising edge of RxClk, after RxEN* is sampled active (see Figure 13-2). The odd parity computed over the RxData[15:0] lines is compared to the RxPar input. If in error, the FR_PAR_ERR bit is set in the HOST_ISTAT0 register. Data is discarded upon a parity error if the RSM_PHALT bit in the RSM_CTRL register is set to a logic high and the reassembly coprocessor halts. The RxSOC (RxMARK*) signals to the CN8237 the start of cell. The RxClav (RxFLAG*) input is the physical layer FIFO buffer empty signal. When it is active, a complete cell is not present in the physical receive FIFO buffer. The physical layer device sets RxClav (RxFLAG*) active when it has a complete cell to transfer. The CN8237 sets RxEN* to a logic low if it can accept a complete cell. On the clock cycle after the last octet of a cell is transferred, the CN8237 samples the RxClav (RxFLAG*) input. If low, the physical device does not have a cell to transfer. If RxClav (RxFLAG*) is high, the physical device has another cell to transfer and the CN8237 immediately starts receiving the next cell if it can accept a complete cell.

Figure 13-2. Receive Timing in UTOPIA Level 1 Mode with Cell Handshake



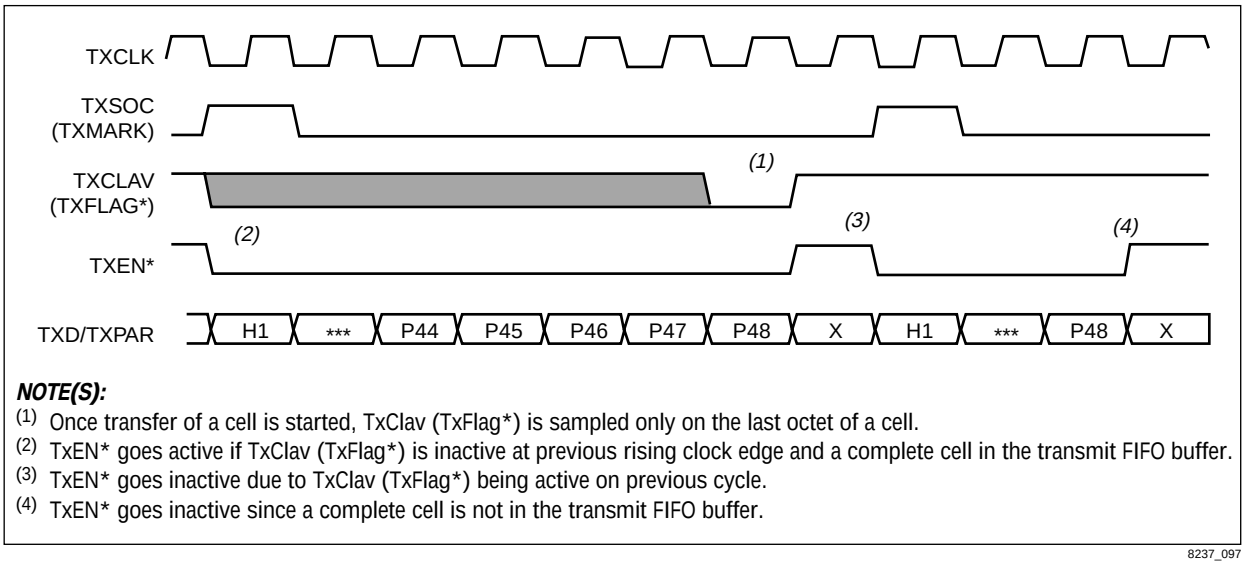
NOTE(S):
(1) Once a cell transfer is started, RxClav (RxFlag*) is not sampled until the end of the cell.
(2) RxEN* goes active only if there is room in the receive FIFO buffer for a complete cell.
(3) RxEN* goes inactive at cell boundaries if the receive FIFO buffer cannot accept another cell.

8237_095

Transmit data is driven on TxDATA[15:0] on the rising edge of TxCLK when TxEN* is asserted. TxEN* is only asserted when there is data in the CN8237 transmit FIFO buffer. Simultaneously, the odd parity computed over the TxDATA[15:0] lines is driven on to the TxPar output. The TxSOC (TxMARK) line is driven by the SAR to indicate start of cell. If the TxCLAV (TxFLAG*) input is asserted by the framer device, the framer device is full and another cell is not transmitted to the physical framer. See Figure 13-3.

In UTOPIA mode, the TxClk input can be connected to the CN8237 CLKD3 output, which is a 50% duty cycle, two-thirds of RCLK0.

Figure 13-3. Transmit Timing in UTOPIA Level 1 Mode with Cell Handshake



NOTE(S):
(1) Once transfer of a cell is started, TxClav (TxFlag*) is sampled only on the last octet of a cell.
(2) TxEN* goes active if TxClav (TxFlag*) is inactive at previous rising clock edge and a complete cell in the transmit FIFO buffer.
(3) TxEN* goes inactive due to TxClav (TxFlag*) being active on previous cycle.
(4) TxEN* goes inactive since a complete cell is not in the transmit FIFO buffer.

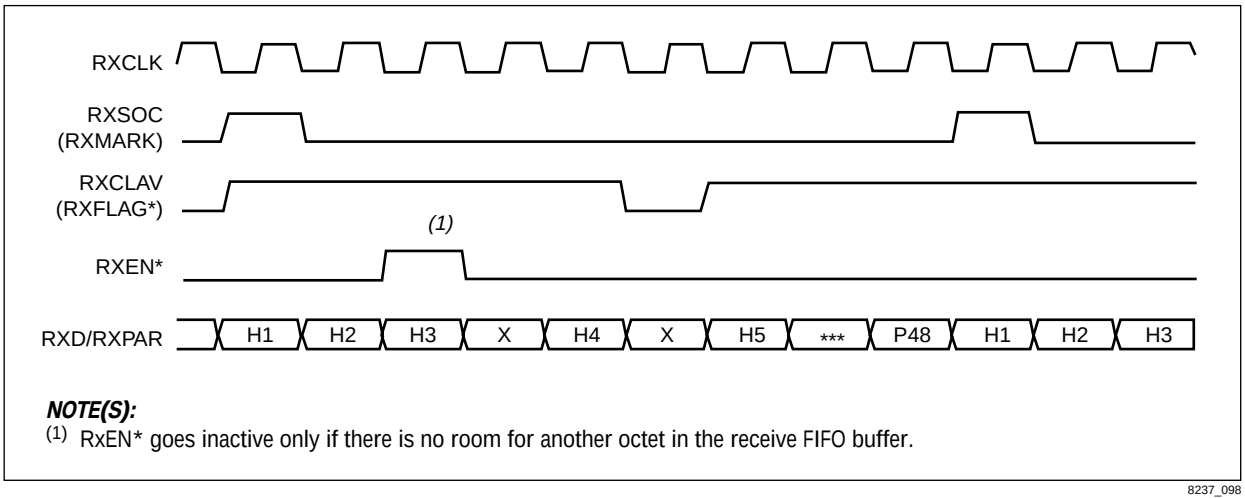
8237_097

13.6 UTOPIA Level 1 Mode Octet Handshake Timing

If low, the UTOPIA_MODE bit in the CONFIG0 register selects octet-level handshaking. Received data is latched from the RxDATA[15:0] and RxPAR lines on the rising edge of RxCLK after RxEN* is sampled active (see Figure 13-4). The odd parity computed over the RxDATA[15:0] lines is compared to the RxPAR input. If in error, FR_PAR_ERR in the HOST_ISTAT0 register is set. Data will be discarded upon a parity if the RSM_PHALT bit in the RSM_CTRL register is set to a logic high and the reassembly coprocessor halts.

The RxSOC (RxMARK) signals the start of cell to the CN8237. The RxCLAV (RxFLAG*) input is the physical layer FIFO buffer empty signal. When it is active, no data is present in the physical receive FIFO buffer. The physical layer device sets RxCLAV (RxFLAG*) inactive when it has an octet to transfer. The CN8237 sets RxEN* to a logic low if it can accept an octet in the next clock cycle.

Figure 13-4. Receive Timing in UTOPIA Level 1 Mode with Octet Handshake



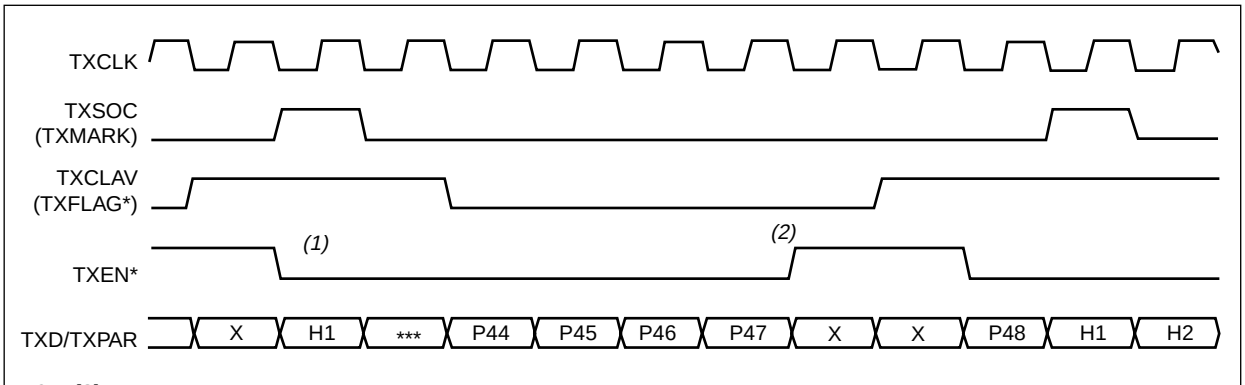
NOTE(S):
(1) RxEN* goes inactive only if there is no room for another octet in the receive FIFO buffer.

8237_098

Transmit data is driven on TxDATA[15:0] on the rising edge of TxCLK when TxEN* is asserted. TxEN* is only asserted when there is data in the CN8237 transmit FIFO buffer. Simultaneously, odd parity computed over the TxDATA[15:0] lines is driven on to the TxPAR output. The TxSOC (TxMARK) line is driven by the SAR to indicate start of cell. If the TxCLAV (TxFLAG*) input is asserted by the framer device, the framer device is full and can accept only one to four more octets. (See Figure 13-5.)

In UTOPIA mode, the TxClk input can be connected to the CN8237 CLKD3 output, which is a 50% duty cycle, two-thirds of RCLK0.

Figure 13-5. Transmit Timing in UTOPIA Level 1 Mode with Octet Handshake



NOTE(S):
(1) TxEN* goes active if TxClav (TxFlag*) is inactive at previous rising clock edge and a complete cell in the transmit FIFO buffer.
(2) TxEN* goes inactive one to four clock cycles after TxClav goes active.

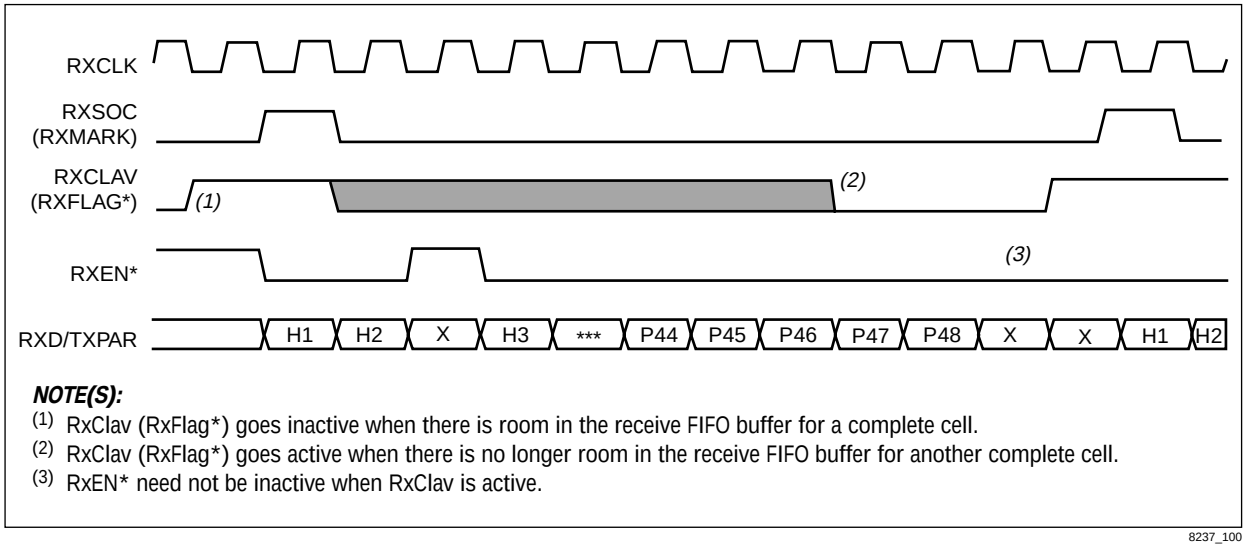
8237_099

13.7 Slave Level 1 UTOPIA Mode

The slave UTOPIA mode is similar to the UTOPIA mode, except the direction of the enable signals and FIFO buffer flags are reversed. This allows a switch fabric or backplane to directly control the physical port. The transmit and receive enable signals are generated by the physical layer instead of the CN8237. The TxFULL* signal is changed to the TxEMPTY* signal and is an output of the CN8237. The RxEMPTY* signal is changed to the RxFULL* signal, and is also an output of the CN8237. This mode supports only a cell-level handshake protocol.

Received data is latched from the RxDATA[15:0] and RxPAR lines on the rising edge of RxCLK when RxEN* is active (see Figure 13-6). The odd parity computed over the RxDATA[15:0] lines is compared to the RxPAR input. If there is a parity error, the FR_PAR_ERR bit is set in the HOST_ISTAT0 register. Data is discarded upon a parity error if the RSM_PHALT bit in the RSM_CTRL register is set to a logic high. If so, the reassembly coprocessor halts upon a parity error. The RxSOC (RxMARK) signals to the CN8237 the start of cell. The RxCLAV (RxFLAG*) output is the receive FIFO buffer full signal. When it is active, the CN8237 cannot accept another cell. The CN8237 sets RxCLAV (RxFLAG*) inactive when it has room in the receive FIFO buffer for another cell. The physical device sets RxEN* to a logic low if it can transfer an octet.

Figure 13-6. Receive Timing in Slave UTOPIA Level 1 Mode

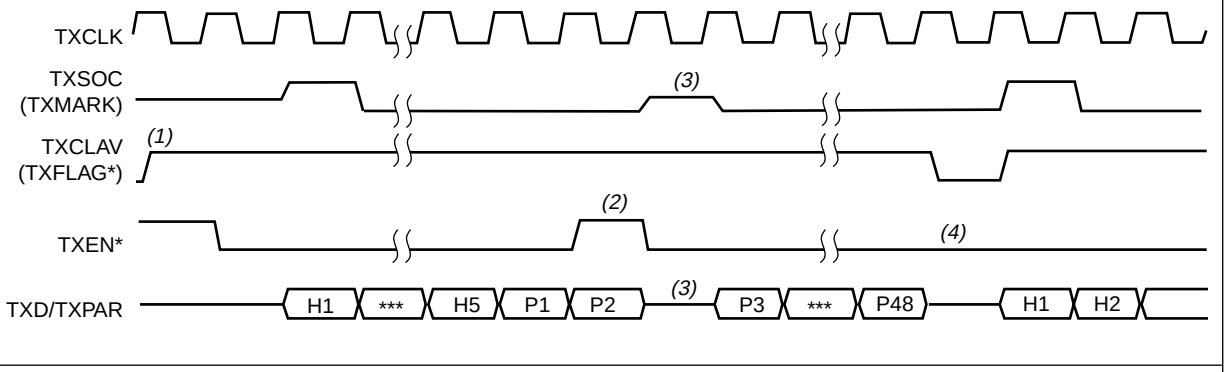


NOTE(S):
(1) RxClav (RxFlag*) goes inactive when there is room in the receive FIFO buffer for a complete cell.
(2) RxClav (RxFlag*) goes active when there is no longer room in the receive FIFO buffer for another complete cell.
(3) RxEN* need not be inactive when RxClav is active.

8237_100

Transmit data is driven on TxDATA[15:0] on the rising edge of TxCLK after TxEN* is sampled asserted. Simultaneously, the odd parity computed over the TxDATA[15:0] lines is driven onto the TxPar output. The TxSOC (TxMARK) line is driven by the SAR to indicate start of cell. If the TxCLAV (TxFLAG*) output is asserted by the CN8237, the transmit FIFO buffer does not contain a complete cell. See Figure 13-7.

Figure 13-7. Transmit Timing in Slave UTOPIA Level 1 Mode



NOTE(S):
(1) TXCLAV goes active when a complete cell is in the transmit FIFO buffer.
(2) Example of physical device deactivating TXEN*.
(3) The TXD/TXPAR and TXSOC lines are three-stated (floated) when TXEN* is sampled high (de-asserted).
(4) Physical device can keep TXEN* active while TXCLAV is inactive.

8237_101

13.8 Loopback Mode

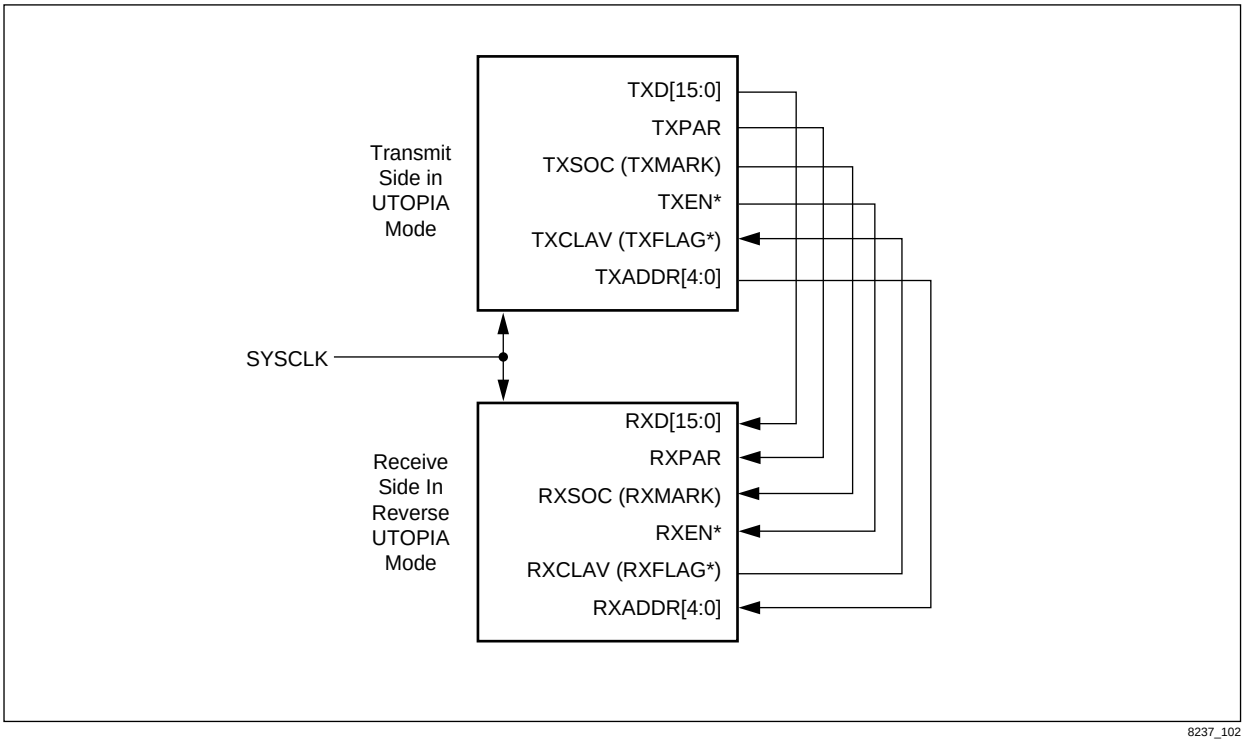
The ATM UTOPIA interface can be internally looped by setting the FR_LOOP bit in the CONFIG1 register to a logic high. This mode uses the internal system clock for operation; therefore, a framer clock is not needed during loopback operation.

When the FR_LOOP signal equals 1, the interface loops back the data and control signals, so the path from the segmentation coprocessor to the reassembly coprocessor can be tested. The internal connections of the ATM interface signals are connected, as Figure 13-8 illustrates. The transmit side is put into the UTOPIA Mode and the receive side is put into the Reverse UTOPIA Mode.

In this mode, the outputs of the chip, TxDATA, TxPAR, TxSOC (TxMARK), TxEN*, and RxCLAV (RxFLAG*) are three-stated, so there are no output conflicts with other devices connected to these signals.

NOTE: A reset of the reassembly coprocessor is required after changing the FR_LOOP bit, because this changes the source of the clock to the ATM UTOPIA interface circuitry. To accomplish this reset, assert RSM_CTRL0 (RSM_RESET).

Figure 13-8. Source Loopback Mode Diagram



13.9 Receive Cell Synchronization Logic

The receive cell synchronization logic accepts a stream of octets (together with error and cell boundary indications) from the receive ATM UTOPIA interface and performs the following functions:

- Maintains a sequence counter that marks the various components of an ATM cell: the 5-byte ATM header, the 1-byte HEC field within the header, and the 48-byte payload. The sequence counter is also used by the ATM UTOPIA interface to check cell boundary synchronization.
- Extracts and discards the HEC byte from each 53-byte ATM cell, leaving 52 bytes of cell data.
- Formats consecutive 4-byte segments into 32-bit words; thus, the header forms the first word, the first four bytes of the payload form the next word, and so on. A total of thirteen 32-bit words are created from each 52-byte cell after the HEC byte has been removed. The bytes within each word are left-justified (big-endian format), that is, the first byte received is the MSB of the word.
- Ensures that a complete cell (exactly 52 bytes) is always written to the FIFO buffer. If a synchronization error occurs, the FR_SYNC_ERR bit in the HOST_ISTAT0 register is set. The ATM UTOPIA interface attempts to resynchronize with the data stream.
- Sets the RSM_OVFL bit in the HOST_ISTAT0 register if an octet could not be transferred due to the receive FIFO buffer being full.

13.10 Transmit Cell Synchronization Logic

The transmit cell synchronization logic copies cell data from the transmit cell FIFO buffer to the transmit ATM UTOPIA interface while performing the following functions:

- Reads 32-bit words from the transmit cell FIFO buffer and converts them to a stream of octets, with the MSB of each 32-bit word corresponding to the first byte derived from that word (big-endian format).
- Maintains a sequence counter that delineates the various components of each ATM cell (4-byte header, 48-byte payload) in the outgoing byte stream.
- Inserts a blank (all-0) HEC byte, used as a placeholder, into the outgoing byte stream representing each ATM cell. The HEC placeholder is inserted after the first four bytes (the ATM header) have been transferred.
- Generates appropriate cell delineation pulses to the transmit ATM UTOPIA interface logic, for use in generating the TxSOC (TxMARK) output, and also in verifying synchronization with the framer device.
- Sets the SEG_UNFL bit in the HOST_ISTAT0 register if the ATM UTOPIA transmit interface runs out of cells to transmit.

The transmit cell synchronization logic supplies a continuous stream of octets to the transmit ATM UTOPIA interface unit, with cell delineation pulses at the starting byte of every cell. Only complete 53-byte cells are supplied to the ATM UTOPIA interface. If the transmit cell FIFO buffer is empty, the transmit cell synchronization logic indicates that no more data can be transferred to the framer.

13.10.1 Head of Line Flushing (HoLF)

If the SAR is set up as a UTOPIA Master in multi-PHY operations, the PHY device indicates that it can receive another cell by asserting its UTOPIA CLAV signal. Cells are then transmitted to the PHY device. The UTOPIA Master waits until it receives the CLAV signal from the PHY device, and therefore blocks the transmission of all other cells in the transmit FIFO. If this PHY device stops working, all other PHY devices are blocked. This is called head of the line blocking.

In order to avoid head of the line blocking in the transmit FIFO, a Head of Line Flushing (HoLF) mechanism flushes the blocking cell out of the transmit FIFO. To enable this mechanism, set the TX_FIFO_FLUSH_EN bit to a one in the CONFIG1 register.

When the UTOPIA Master puts out the address of a PHY device, a counter is reset to 0, and increased based on the UTOPIA TX_CLK. Once the counter reaches the values TX_CNTR set by the user in the CSR register, the cell is discarded, and the bit corresponding to the blocking PHY device is set in the TX_PORT_STATUS register. The counter is reset automatically. If any of the eight bits in the corresponding mask bit EN_TX_DISCARD in HOST_IMASK1 is set, an interrupt is generated. The TX_DISCARD is cleared once the TX_STATUS register is read by the host.

NOTE: The cell discard does not disable the port.

If the user decides to disable a specific port on the UTOPIA interface, then all cells belonging to this port are discarded or flushed. Ports are disabled by setting the corresponding bit in the TX_PORT_CTRL register. If a port x is disabled and a cell pertaining to port x is discarded, bit x in TX_STATUS is not set, and therefore no interrupt is generated.

For fault tolerant multi-PHY operations, a maximum TX _CNTR value of 65,535 is recommended. For specific DSL applications with variable rate PHY devices, a value between 50 and 100 is suggested.

14.0 CN8237 Registers

14.1 Control and Status Registers

Detailed control and status register (CSR) descriptions of the CN8237 are listed in Table 14-2.

NOTE: The byte-enables are ignored by the CN8237 when writing to control and status registers.

The access type terminology given in Table 14-2 applies to all registers in this section. Each register description is prefaced with the appropriate abbreviated access types described in Table 14-1.

Table 14-1. Access Type Abbreviation Description

Access Type Abbreviation	Description
R/W	Read and write access for host processor
R/O	Read only access

14.1.1 Register Terminology

Table 14-2. CN8237 Control and Status Registers (1 of 2)

Address	Name	Type	Description
0x00	CLOCK	R/W	Real-Time Clock Register
0x08	ALARM1	R/W	Alarm Register 1
0x10	Reserved	—	Not Implemented
0x18	SYS_STAT	R/O	System Status Register
0x20	Reserved	—	Not Implemented
0x28	CONFIG0	R/W	Basic Configuration and Control Register 0
0x30	CONFIG1	R/W	Basic Configuration and Control Register 1
0x38	INT_DELAY	R/W	Interrupt Delay Register
0x40–0xf8	Reserved	—	Not Implemented
0x100	SEG_CTRL	R/W	Segmentation Control Register
0x108	SEG_VBASE	R/W	Seg VCC Table and Schedule Table Base Address Register
0x110	SEG_PMBASE	R/W	Seg PM Table and Bucket Table Base Address Register
0x118	SEG_TXBASE	R/W	Segmentation Transmit Queue Base Register
0x120	SEG_TAGBASE	R/W	Base Address of Routing Tag Table
0x128–0x138	Reserved	—	Not Implemented
0x140	SCH_PRI	R/W	Schedule Priority Register
0x148	SCH_PRI_2	R/W	Schedule Priority Register 2
0x150	SCH_SIZE	R/W	Schedule Size Register
0x158	SCH_CTRL	R/W	Schedule Control Register
0x160	SCH_ABR_ MAX	R/W	Maximum ABR VCC_INDEX Register
0x168	SCH_ABR_CON	R/W	Schedule ABR Constant Register
0x170	SCH_ABRBASE	R/W	ABR Decision Table Lookup Base Register
0x178	SCH_CNG	R/W	ABR Congestion Notification Register
0x180	PCR_QUE_INT01	R/W	PCR Queue Interval 0 and 1 Register
0x188	PCR_QUE_INT23	R/W	PCR Queue Interval 2 and 3 Register
0x190–0xd8	Reserved	—	Not Implemented
0x1e0	RSM_CTRL0	R/W	Reassembly Control Register 0
0x1e8	RSM_CTRL1	R/W	Reassembly Control Register 1
0x1f0	RSM_FQBASE	R/W	Reassembly Free Buffer Queue Base Register
0x1f8	RSM_FQCTRL	R/W	Reassembly Free Buffer Queue Control Register
0x200	RSM_TBASE	R/W	Reassembly Table Base Register
0x208	RSM_TO	R/W	Reassembly Time-out Register

Table 14-2. CN8237 Control and Status Registers *(2 of 2)*

Address	Name	Type	Description
0x210	RS_QBASE	R/W	Reassembly/Segmentation Queue Base Register
0x218	ERS_BASE	R/W	Reassembly ER_SHIFT Tables Base Register
0x220	VPI_SIZE	R/W	Variable VPI Size Register
0x228	TX_FIFO_CTRL	R/W	Transmit Port Register
0c230	TX_PORT_CTRL	R/W	Transmit Port Control Register
0x238–0x2b8	Reserved	—	Not Implemented
0x2c0	CELL_XMIT_CNT	R/O	ATM Cells Transmitted Counter
0x2c8	CELL_RCVD_CNT	R/O	ATM Cells Received Counter
0x2d0	CELL_DSC_CNT	R/O	ATM Cells Discarded Counter
0x2d8	AAL5_DSC_CNT	R/O	AAL5 PDUs Discarded Counter
0x2c0–0x2f8	Reserved	—	Not Implemented
0x300	SCH_HD_TL	R/O	Scheduler Head/Tail Pointer Register
0x308–0x340	Reserved	—	Not Implemented
0x348	HOST_ST_WR	R/O	Host Status Write Register
0x350	Reserved	—	Not Implemented
0x358	TX_STATUS	R/O	Tx Port Cell Discard Status Register
0x360–0x378	Reserved	—	Not Implemented
0x380	HOST_ISTAT0	R/O	Host Interrupt Status Reserved 0
0x388	HOST_ISTAT1	R/O	Host Interrupt Status Reserved 1
0x390–0x398	Reserved	—	Not Implemented
0x3a0	HOST_IMASK0	R/W H	Host Interrupt Mask Reserved 0
0x3a8	HOST_IMASK1	R/W H	Host Interrupt Mask Reserved 1
0x3b0–0x3f8	Reserved	—	Not Implemented

14.2 System Registers

0x00—Real-Time Clock Register (CLOCK)

This register contains the 32-bit real time clock. It is incremented by the system clock (SYSCLK), which has been divided by the DIVIDER[6:0] field in the CONFIG0 register and then divided by two. This register is used by the Interrupt Delay Feature. It can be written to any value.

Bit	Field Size	Name	Description
31–0	32	CLOCK[31:0]	Real-time clock value.

0x08—Alarm Register 1 (ALARM1)

This register contains a 32-bit value which is compared against the CLOCK register. Used by the Interrupt Delay Feature.

Bit	Field Size	Name	Description
31–0	32	ALARM1[31:0]	ALARM1 comparison value.

0x18—System Status Register (SYS_STAT)

The System Status register provides read-only system status. This register reflects the device ID and version information for the part, and pin programmable options that otherwise might not be visible to the processors. It also contains expanded information for the status located in the HOST_ISTAT0 register.

Bit	Field Size	Name	Description
31–17	15	Reserved	Not implemented at this time.
16–12	5	PCI_BUS_STATUS [4:0]	The status bits are as follows: 4 = Target Abort 3 = Master Abort 2 = Parity Error 1 = Interface Disabled 0 = Internal Failure Reflects corresponding error bits in the PCI Configuration register. Bits are reset by either a write to the PCI Configuration register by the host, or by setting CONFIG0 (PCI_ERR_RESET) bit.
11–9	3	Reserved	Not implemented at this time.
8	1	FRCFG	Reflects the state of the FRCFG input pin.
7–4	4	VERSION[3:0]	Version number for the CN8237: set to 0 for Rev A and set to 1 for Rev B.
3–0	4	DEVICE[3:0]	Device ID for the CN8237; set to 0.

0x28—Configuration Register 0 (CONFIG0)

This register provides all of the control and configuration bits that are not associated with the reassembly and segmentation coprocessors. The majority of these configuration bits are set at initialization time and are not changed dynamically. The assertion of the HRST* system reset pin clears all of the bits in the CONFIG0 register except for MEMCTRL, which will be set high.

Bit	Field Size	Name	Description
31	1	Reserved	Always set to 0.
30	1	GLOBAL_RESET	When set, this bit causes reset of the segmentation and reassembly coprocessors as well as all latched status.
29	1	PCI_MSTR_RESET	When set, this bit resets the PCI master logic. Once active, this bit must stay active for 16 cycles of the HCLK input signal.
28	1	PCI_ERR_RESET	When set, resets all PCI error bits in the PCI configuration, including RMA, RTA, DPR, INTF_DIS, INT_FAIL, and MERROR. This also re-enables PCI master operation.
27	1	Reserved	Always set to 0.
26	1	INT_LBANK	When set, allows only byte 0 and 1 writes to odd quad word addresses in address space 0x1000 – 0x10ff and 0x1400 – 0x14ff. This allows endian neutral access of the Status Queue Base Table READ_UD field by the host processor.
25	1	MEMTYPE	Logic high selects non-zbt memory. Logic low selects zbt memory.
24	1	MEMCTRL	Selects 0 or 1 wait states SRC shared memory (1 or 2 cycle). Reset to a logic high (1 wait state operation).
23–21	3	BANKSIZE[2:0]	Selects size of memory banks for contiguous memory support. 111 – 8 MB 110 – 4 MB 101 – 2 MB 100 – 1 MB 011 – 512 KB 010 – 256 KB 001 – 128 KB 000 – Reserved
20–19	2	NUMBANKS[1:0]	Selects the number of RSM and SEG memory banks that are active. 0 – 1 bank active 1 – 2 banks active 2 – 4 banks active 3 – not used
18	1	PCI_READ_MULTI	When this bit is set, the SAR's PCI Master implements the PCI Read Multiple Command. Otherwise, the PCI Master implements the PCI Read Command.
17–16	2	PCI_ARB[1:0]	Selects PCI Master arbitration scheme. 00 – read priority over write 01 – round robin 10 – write priority over read 11 – not used

Bit	Field Size	Name	Description
15	1	ENDIAN	Selects between Little and Big Endian host data structures. 0 = Little Endian 1 = Big Endian
14	1	FORCE64	When logic high, the PCI master will always perform 64 bit transfers.
13–12	2	Reserved	Always set to 0.
11–7	5	STATMODE[4:0]	Selects which internal status to output on the STAT[3:0] output pins.
6–0	7	DIVIDER[6:0]	Prescaler for SYSCLK which advances the counter in the CLOCK register**.** SYSCLK is divided by the divider value; if 0, divided by 128.
NOTE(S): If one wait state is selected, PWAIT function does not work.			

0x30—Configuration Register 1 (CONFIG1)

This register provides system control and configuration bits that are not directly associated with the reassembly and segmentation coprocessors. The majority of these configuration bits are set at initialization time and are not changed dynamically. The assertion of the HRST* system reset pin clears all bits in the CONFIG1 register except for UTOPIA_MODE, UTOPIA 16, and PHY_RST bit which will be set.

Bit	Field Size	Name	Description
31–30	2	Reserved	Set to 0.
29	1	PHY_STROBE_I0	Set to logic low for operation with Mindspeed framers.
28–27	2	PHY_WAIT	Set to logic low for operation with Mindspeed framers.
26	1	PHY_RST	When logic high, sets PRST* to logic low. Set to logic 1 upon reset.
25	1	Reserved	Always set to 0.
24	1	FR_LOOP	When set, this bit enables loopback of cells at the ATM physical interface.
23	1	UTOPIA_MODE	Selects octet or cell UTOPIA handshake mode. Set to logic 1 upon reset. 0 = Octet 1 = Cell handshake
22	1	MULTI_PHY	If logic high, multi-PHY operation is enabled.
21	1	UTOP16	If logic high, UTOPIA 16 bit interface is enabled; otherwise 8-bit interface. Set to logic 1 upon reset.
20–18	3	NUM_PORTS[2:0]	Number of PHY ports to poll when in Master UTOPIA Mode starting with address 0. Number of ports = (NUM_PORTS + 1)
17–13	5	SLAVE_ADDR[4:0]	When in Slave UTOPIA Multi-PHY Mode, this is the UTOPIA device address of the SAR. When in Master Non-Multi-PHY Mode, this is the address present on both TxADDR and RxADDR.
12–9	4	TAG_SIZE[3:0]	Select Tag size. When UTOP16 is logic high, valid range is 0 to 10 in even increments. When UTOP16 is a logic low, valid range is 0 to 11.
8–4	5	PHYBANK[4:0]	Physical Chip Bank Select. This value is placed on PADDR[12:8] when a PHY control access occurs.
3	1	Reserved	Always set to 0.
2	1	TX_FIFO_FLUSH_EN	Enables Tx FIFO flush mechanism. This mechanism is only valid in UTOPIA master and multiphy mode.
1	1	INCFIFO_SZ	Incoming DMA FIFO buffer size. Logic high sets the FIFO buffer to 8 KB, logic low to 2 KB.
0	1	NEW_PMOAM	When a logic high, the new PM_OAM mechanism is enabled per *ITU-T Recommendation I.610*, June 98.

0x38—Interrupt Delay Register (INT_DELAY)

Bit	Field Size	Name	Description
31–10	22	Reserved	Set to 0.
9	1	EN_TIMER	Enable status queue interrupt timer delay.
8	1	EN_STAT_CNT	Enable status queue interrupt counter delay.
7–0	8	STAT_CNT[7:0]	Number of status queue entries written before allowing interrupt to propagate to output pin.

14.3 Segmentation Registers

0x100—Segmentation Control Register (SEG_CTRL)

This register contains general control bits for the segmentation coprocessor. The assertion of the HRST* system reset pin or GLOBAL_RESET bit in the CONFIG0 register causes the clearing of the SEG_ENABLE control bit.

Bit	Field Size	Name	Description
31	1	SEG_ENABLE	Segmentation Enable—enables segmentation coprocessor. If disabled, the segmentation coprocessor halts on a cell boundary.
30	1	SEG_RESET	Segmentation Reset—resets segmentation coprocessor and pointers.
29–27	3	VBR_OFFSET	Offset from schedule slot priority to general priority. (VBR_OFFSET + (# VBR / ABR priorities) ≤ 7.) Not active if USE_SCH_CTRL is asserted.
26	1	SEG_GFC	Enable segmentation GFC processing. The segmentation machine is disabled when the SAR receives cells with GFC halt set. GFC priority queues (set in the SCH_PRI register) are active for one cell for each received cell with GFC SET_A bit = 1.
25	1	DBL_SLOT	Each schedule slot occupies two words. Not active if USE_SCH_CTRL is asserted.
24	1	CBR_TUN	Use first entry in each schedule slot for CBR/tunnel traffic.
23	1	ADV_ABR_TMPLT	Advanced ABR template mode. When logic high, per-connection MCR and ICR enabled. When logic low, per-template MCR and ICR enabled.
22	1	USE_SCH_CTRL	Activate the use of SLOT_DEPTH, the 4-bit VBR_OFFSET field, and TUN_PRI0_OFFSET from the SCH_CTRL register. De-activate the use of DBL_SLOT and the 3-bit VBR_OFFSET field from the SEG_CTRL register.
21–16	6	Reserved	Program and read as 0.

Bit	Field Size	Name	Description
15–12	4	TX_FIFO_LEN	Depth of transmit FIFO buffer in cells. Valid range is 3–9. To ensure optimum performance, the depth of the FIFO buffer should be at least three.
11	1	CLP0_EOM	Set CLP in ATM header to 0 on last cell of CPCS-PDU.
10–6	5	OAM_STAT_ID	Status queue ID for buffer descriptors with OAM_STAT set.
5	1	SEG_ST_HALT	Enables a status queue entry for a VCC halted with a partially segmented packet.
4	1	Reserved	Program and read as 0.
3	1	SEG_HS_DIS	Segmentation Host Status Disable—Disable segmentation check for PCI memory status queue full condition. If this bit is not set, the segmentation coprocessor will not segment any cells for a VCC assigned to a full PCI memory status queue. This bit can be used to disable overflow checking when the queues are sized large enough to prevent overflow.
2	1	TX_RND	Set for transmit queue servicing in round-robin order. Clear for transmit queue servicing in priority order (31 is highest priority).
1–0	2	TR_SIZE[1:0]	Number of entries in each transmit queue. 00 = 64 01 = 256 10 = 1,024 11 = 4,096

0x108—Segmentation VCC Table and Schedule Table Base Address Register (SEG_VBASE)

The SEG_VBASE register sets the base address in SAR SEG-shared memory for the segmentation VCC table and the schedule table. Both base addresses are 128-byte aligned, and only the 16 most significant bits of the address are specified in the SEG_VBASE register.

NOTE: The PCI address of these structures is (REGISTER_VALUE × 0x80) + SEG_MEMORY_BASE_ADDRESS.

Bit	Field Size	Name	Description
31–16	16	SEG_SCHB[15:0]	Base address for the schedule table.
15–0	16	SEG_VCCB[15:0]	Base address for the segmentation VCC table.

0x110—Segmentation PM Base Register (SEG_PMBASE)

The SEG_PMBASE register sets the base address in SAR SEG-shared memory for the VBR bucket table and the performance monitoring table. Both base addresses are 128-byte aligned, and only the 16 most significant bits of the address are specified in the SEG_PMBASE register.

NOTE: The PCI address of these structures is (REGISTER_VALUE × 0x80) + SEG_MEMORY_BASE_ADDRESS.

Bit	Field Size	Name	Description
31–16	16	SEG_BCKB[15:0]	Base address for the VBR bucket table. (See Section 6.2.4.4, for details on loading bucket table entries.)
15–0	16	SEG_PMB[15:0]	Base address for the performance monitoring table.

0x118—Segmentation Transmit Queue Base Register (SEG_TXBASE)

The SEG_TXBASE register sets the base address in SAR-shared memory for the transmit queues and enables the individual queues. The base address is 128-byte aligned and only the 16 most significant bits of the address are specified in the SEG_TXBASE register.

NOTE: The PCI address of these structures is (REGISTER_VALUE × 0x80) + SEG_MEMORY_BASE_ADDRESS.

Bit	Field Size	Name	Description
31–16	16	SEG_TXB[15:0]	Base address for the transmit queues.
15–13	3	Reserved	Program and read as 0.
12–5	8	XMIT_INTERVAL[7:0]	Interval for transmit queue READ_UD_PNTR update.
4–0	5	TX_EN[4:0]	Transmit queues 0-TX_EN are enabled.

0x120—Segmentation Tag Base Register (SEG_TAGBASE)

The SEG_TAGBASE register sets the base address in segmentation SAR-shared memory for the Routing Tag table. The base address is 128-byte aligned and only the 16 most significant bits of the address are specified in the SEG_TAGBASE register.

NOTE: The PCI address of these structures is (REGISTER_VALUE × 0x80) + SEG_MEMORY_BASE_ADDRESS.

Bit	Field Size	Name	Description
31–16	16	Reserved	Always set to 0.
15–0	16	SEG_TAGB[15:0]	Base address for the routing tag table.

0x210—Reassembly/Segmentation Queue Base Address Register (RS_QBASE)

This register contains the 128-byte aligned base address of the reassembly/segmentation queue structure.

Bit	Field Size	Name	Description
31–18	14	Reserved	Program and read as 0.
17–16	2	RS_SIZE[1:0]	Size of the RS Queue. Each RS Queue entry is 16 octets in length. 00 = 256 01 = 1,024 10 = 4,096 11 = 16,384
15–0	16	RS_QBASE[15:0]	Base address of the reassembly/segmentation queue.

14.4 Scheduler Registers

0x140—Schedule Priority Register (SCH_PRI)

This register specifies the use of each global priority pointer for priority queues zero through seven.

Bit	Field Size	Name	Description
31	1	QPCR_ENA7	Enable PCR limits on global priority pointer 7.
30	1	QPCR_ENA6	Enable PCR limits on global priority pointer 6.
29	1	QPCR_ENA5	Enable PCR limits on global priority pointer 5.
28	1	QPCR_ENA4	Enable PCR limits on global priority pointer 4.
27	1	QPCR_ENA3	Enable PCR limits on global priority pointer 3.
26	1	QPCR_ENA2	Enable PCR limits on global priority pointer 2.
25	1	QPCR_ENA1	Enable PCR limits on global priority pointer 1.
24	1	QPCR_ENA0	Enable PCR limits on global priority pointer 0.
23	1	Reserved	Program and read as 0.
22	1	TUN_ENA7	Enable tunnel on global priority pointer 7.
21	1	GFC7	Enable GFC on priority pointer 7.
20	1	Reserved	Program and read as 0.
19	1	TUN_ENA6	Enable tunnel on global priority pointer 6.
18	1	GFC6	Enable GFC on priority pointer 6.

Bit	Field Size	Name	Description
17	1	Reserved	Program and read as 0.
16	1	TUN_ENA5	Enable tunnel on global priority pointer 5.
15	1	GFC5	Enable GFC on priority pointer 5.
14	1	Reserved	Program and read as 0.
13	1	TUN_ENA4	Enable tunnel on global priority pointer 4.
12	1	GFC4	Enable GFC on priority pointer 4.
11	1	Reserved	Program and read as 0.
10	1	TUN_ENA3	Enable tunnel on global priority pointer 3.
9	1	GFC3	Enable GFC on priority pointer 3.
8	1	Reserved	Program and read as 0.
7	1	TUN_ENA2	Enable tunnel on global priority pointer 2.
6	1	GFC2	Enable GFC on priority pointer 2.
5	1	Reserved	Program and read as 0.
4	1	TUN_ENA1	Enable tunnel on global priority pointer 1.
3	1	GFC1	Enable GFC on priority pointer 1.
2	1	Reserved	Program and read as 0.
1	1	TUN_ENA0	Enable tunnel on global priority pointer 0.
0	1	GFC0	Enable GFC on priority pointer 0.

0x148—Schedule Priority Control Register 2 (SCH_PRI_2)

The SCH_PRI_2 register sets the enables for GFC, CBR tunneling and PCR limits, on global priority queues 8–15.

Bit	Field Size	Name	Description
31	1	QPCR_ENA_15	Enable PCR limits on global priority pointer 15.
30	1	QPCR_ENA_14	Enable PCR limits on global priority pointer 14.
29	1	QPCR_ENA_13	Enable PCR limits on global priority pointer 13.
28	1	QPCR_ENA_12	Enable PCR limits on global priority pointer 12.
27	1	QPCR_ENA_11	Enable PCR limits on global priority pointer 11.
26	1	QPCR_ENA_10	Enable PCR limits on global priority pointer 10.
25	1	QPCR_ENA_9	Enable PCR limits on global priority pointer 9.
24	1	QPCR_ENA_8	Enable PCR limits on global priority pointer 8.
23	1	Reserved	Program and read as 0.
22	1	TUN_ENA_15	Enable tunnel on global priority pointer 15.

Bit	Field Size	Name	Description
21	1	GFC15	Enable GFC on global priority pointer 15.
20	1	Reserved	Program and read as 0.
19	1	TUN_ENA_14	Enable tunnel on global priority pointer 14.
18	1	GFC14	Enable GFC on global priority pointer 14.
17	1	Reserved	Program and read as 0.
16	1	TUN_ENA_13	Enable tunnel on global priority pointer 13.
15	1	GFC13	Enable GFC on global priority pointer 13.
14	1	Reserved	Program and read as 0.
13	1	TUN_ENA_12	Enable tunnel on global priority pointer 12.
12	1	GFC12	Enable GFC on global priority pointer 12.
11	1	Reserved	Program and read as 0.
10	1	TUN_ENA_11	Enable tunnel on global priority pointer 11.
9	1	GFC11	Enable GFC on global priority pointer 11.
8	1	Reserved	Program and read as 0.
7	1	TUN_ENA_10	Enable tunnel on global priority pointer 10.
6	1	GFC10	Enable GFC on global priority pointer 10.
5	1	Reserved	Program and read as 0.
4	1	TUN_ENA_9	Enable tunnel on global priority pointer 9.
3	1	GFC9	Enable GFC on global priority pointer 9.
2	1	Reserved	Program and read as 0.
1	1	TUN_ENA_8	Enable tunnel on global priority pointer 8.
0	1	GFC8	Enable GFC on global priority pointer 8.

0x150—Schedule Size Register (SCH_SIZE)

The SCH_SIZE register sets the size of the schedule table in schedule slots and the period of a schedule slot in system clocks.

Bit	Field Size	Name	Description
31-16	16	TBL_SIZE[15:0]	Size of schedule table in schedule slots.
15–14	2	Reserved	Program and read as 0.
13–0	14	SLOT_PER[13:0]	Number of system clocks per schedule slot. The value written to the register should be (SLOT_PER – 1). Minimum bound for SLOT_PER value = 40.

0x158—Scheduler Control Register (SCH_CTRL)

The SCH_CTRL register defines the configured schedule slot and priority and VBR offsets, when 16 priority queues are used.

Bit	Field Size	Name	Description
31–28	4	HD_TL_PRI_SEL	Selects priority queue head/tail pointers to be read in the SCH_HD_TL register.
27	1	SCHREFx4	When logic high, three extra clock edges are generated internally for each rising edge of SCHREF. This allows a 622 cell clock to be generated from quad peak 7.
26	1	USE_SCHREF	If logic high, the SCHREF input is used as the clock for defining a schedule table slot period in conjunction with SLOT_PER. If logic low, SYSCLK is used. ***NOTE:*** Must be set to 0 during initialization unless using an internal control register.
25	1	Reserved	Program and read as 0.
24	1	NCR_EN_DEST	Global enable for destination ACR notification.
23	1	NCR_EN_SRC	Global enable for source ACR notification.
22–18	5	NCR_STAT_ID	Identifies the status queue to be used for both source and destination ACR/ER notification when EN_NCR_STAT is asserted.
17	1	EN_NCR_STAT	Enable global status queue for both source and destination ACR/ER notification.
16–15	2	Reserved	Program and read as 0.
14–12	3	SLOT_DEPTH	Depth of the schedule slot is set to 1 + SLOT_DEPTH words. Active only if USE_SCH_CTRL is asserted.
11–10	2	Reserved	Program and read as 0.
9–6	4	TUN_PRI0_OFFSET	Offset from the TUN_PRI_0 field in the schedule table and CBR VCC table. Active only if USE_SCH_CTRL is asserted.
5–4	2	Reserved	Program and read as 0.
3–0	4	VBR_OFFSET	Offset from schedule slot priority to general priority. Active only if USE_SCH_CTRL is asserted.

0x160—Maximum ABR VCC_INDEX Register (SCH_ABR_MAX)

The SCH_ABR_MAX register defines the configured maximum ABR VCC index, and specifies the MAX_BEHIND value. The MAX_BEHIND value specifies the number of slots in the schedule table, the schedule table entry pointer can fall behind. MAX_BEHIND should be set to 0 for all applications, except special multi-PHY DSL applications.

Bit	Field Size	Name	Description
31–24	8	Reserved	Program and read as 0.
23–16	8	MAX_BEHIND	Number of slots Scheduler can fall behind. For normal operations, this value should be 0. (A value of 0 results in the default value of 255 internally used). For multi-PHY DSL operations, this value should be set to the number of port used *2, for example, if 8 ports are used, this MAX_BEHIND should be set to 16.
15-0	16	VCC_MAX	Maximum ABR VCC index.

0x168—Schedule ABR Constant Register (SCH_ABR_CON)

The SCH_ABR_CON register sets the ABR TRM and ADTF timeout. Timeout values are in units of 64 cell slots.

Bit	Field Size	Name	Description
31–16	16	ABR_TRM	ABR TRM parameter. Parameter value is SYSCLK period × SLOT_PER × ABR_TRM × 64.
15–0	16	ABR_ADTF	ABR ADTF parameter. Parameter values is SYSCLK period × SLOT_PER × ABR_ADTF × 64.

0x170—ABR Decision Table Lookup Base Register (SCH_ABRBASE)

The SCH_ABRBASE register sets the base address in SAR-shared memory for the ABR decision table. This address is 128-byte aligned, and only the 16 most significant bits of the address are specified in the SCH_ABRBASE register.

NOTE: The PCI address of these structures is (REGISTER_VALUE × 0x80) + SEG_MEMORY_BASE_ADDRESS.

Bit	Field Size	Name	Description
31–29	3	Reserved	Program and read as 0.
28	1	OOR_EN	Enable ABR out-of-rate Forward RM cell generation.
27–16	12	OOR_INT	ABR out-of-rate Forward RM cell interval. A VCC is examined for an out-of-rate cell every OOR_INT schedule slots.
15–0	16	SCH_ABRB[15:0]	Base address for the ABR decision table.

0x178—ABR Congestion Register (SCH_CNG)

The SCH_CNG register sets each reassembly free buffer queue to a congested or non-congested state for transmitted reverse RM cells.

Bit	Field Size	Name	Description
31–0	32	FBQ_CNG[31:0]	Congestion state for each free buffer queue.

0x180—PCR Queue Interval 0 and 1 Register (PCR_QUE_INT01)

The PCR_QUE_INT01 register fields are used to store the two lower PCR values used in PCR shaping on priority queues that have been enabled for PCR shaping by setting the QPCR_ENAx bits in the SCH_PRI register. QPCR_INTx values are used in order: 3, 2, 1, and 0; highest to lowest. PCR is determined by 1 / (QPCR_INTx × SYSCLK_period × SLOT_PER).

Bit	Field Size	Name	Description
31–16	16	QPCR_INT1	The assigned PCR interval, entered as number of schedule table slots, used to map to the 3rd-highest priority queue that is enabled for PCR shaping (using the QPCR_ENAx bits in the SCH_PRI register).
15–0	16	QPCR_INT0	The assigned PCR interval, entered as number of schedule table slots, used to map to the 4th-highest priority queue that is enabled for PCR shaping (using the QPCR_ENAx bits in the SCH_PRI register).

0x188—PCR Queue Interval 2 and 3 Register (PCR_QUE_INT23)

The PCR_QUE_INT23 register fields are used to store the two highest PCR values used in PCR shaping on priority queues that have been enabled for PCR shaping by setting the QPCR_ENAx bits in the SCH_PRI register. QPCR_INTx values are used in order: 3, 2, 1, and 0; highest to lowest. PCR is determined by 1 / (QPCR_INTx × SYSCLK_PERIOD × SLOT_PER).

Bit	Field Size	Name	Description
31–16	16	QPCR_INT3	The assigned PCR interval, entered as number of schedule table slots, used to map to the highest priority queue that is enabled for PCR shaping (using the QPCR_ENAx bits in the SCH_PRI register).
15–0	16	QPCR_INT2	The assigned PCR interval, entered as number of schedule table slots, used to map to the 2nd-highest priority queue that is enabled for PCR shaping (using the QPCR_ENAx bits in the SCH_PRI register).

14.5 Reassembly Registers

0x1e0—Reassembly Control Register 0 (RSM_CTRL0)

The Reassembly Control register 0 contains the general control bits for the reassembly coprocessor. The assertion of the HRST* system reset pin or the GLOBAL_RESET bit in the CONFIG0 register clears the RSM_ENABLE control bit.

Bit	Field Size	Name	Description
31	1	RSM_ENABLE	Reassembly enable. Initiates an incoming transfer if set, and halts it if reset. If this bit is reset while the reassembly coprocessor is running, it temporarily suspends the activities of the reassembly coprocessor logic. Suspension takes place on a cell boundary, that is, between the completion of all processing and transfers required for the current cell, and the start of processing for the next cell. The hold can be removed and the transfer resumed by setting the RSM_ENABLE bit. This bit will also be set low internally on certain reassembly error conditions. This includes parity error with PHALT_EN. In this case, the error condition should be corrected and the RSM_ENABLE bit set high to resume operation.
30	1	RSM_RESET	Reassembly reset. Forces a hardware reset of the reassembly coprocessor when asserted. It must be deasserted before the reassembly coprocessor will resume normal operation.
29	1	Reserved	Program and read as 0.

Bit	Field Size	Name	Description
28	1	VPI_MASK	VPI Mask enable. Used to select UNI/NNI header operation in the direct index method. When a logic high, the four MSBs of the header are masked for UNI operation. This also controls the Index table size. A UNI table is 256 entries and a NNI table is 4,096.
27–24	4	Reserved	Program and read as 0.
23–18	6	Reserved	Program and read as 0.
17	1	RSM_PHALT	Reassembly coprocessor halt on parity error detect. The reassembly coprocessor will halt the incoming channel logic if a parity error is detected and the RSM_PHALT bit is set.
16	1	PREPEND_INDEX	Prepend VCC Index. When logic high, the VCC_INDEX is prepended to the BOM cell transfer.
15	1	FWALL_EN	Firewall enable. Enables free buffer queue firewalling of user cells. If set, this bit enables the per connection free buffer queue firewall. Each connection that firewall is active in must have the FW_EN bit set to a logic high in the VCC table.
14	1	RSM_FBQ_DIS	Free Buffer Queue Underflow Protection Disable. When a logic high, the reassembly coprocessor ignores the VLD bit in the free buffer queue when a new buffer is required. The periodic writing of the read index pointer to host/SAR-shared memory is also disabled.
13	1	RSM_STAT_DIS	Status Queue Overflow Protection Disable. When a logic high, the reassembly coprocessor ignores the READ_UD pointer.
12	1	GTO_EN	Global Time-out Enable. When a logic high, the automatic reassembly time-out function is enabled.
11–5	7	MAX_LEN	Maximum Length. MAX_LEN × 1,024 is the maximum allowable size in bytes of an AAL5 CPCS-PDU, including overhead.
4–0	5	GDIS_PRI	Global Discard Priority. Used by the Frame Relay and CLP discard functions. If the individual channel priority number is less than or equal to GDIS_PRI, PDUs on that channel can be discarded.

0x1e8—Reassembly Control Register 1 (RSM_CTRL1)

The Reassembly Control register 1 contains additional general control bits for the reassembly coprocessor.

Bit	Field Size	Name	Description
31	1	EN_PROG_BLK_SZ	Enable Programmable Block Size. When a logic high, the programmable block size VPI/VCI lookup method is enabled. This method consists of programmable block sizes, and an additional BLK_EN bit in the VCI Index table.
30–28	3	VCI_IT_BLK_SZ	VCC table/VCI Index table Block Size. This field determines the number of RSM VCC entries accessed per VCI Index table entry.
27	1	EN_VPI_SIZE	Enable Programmable VPI size via VPI_SIZE register. When a logic high, an expanded more flexible VPI Index Table configuration is enabled. The VPI_SIZE register and PORT_ID from the framer are used to calculate an index into the VPI Index table.
26–24	3	Reserved	Program and read as 0.
23–20	4	Reserved	Program and read as 0.
19–17	3	Reserved	Program and read as 0.
16–13	4	Reserved	Program and read as 0.
12	1	OAM_FF_DSC	OAM FIFO Buffer Full Discard. When a logic high and OAM_QU_EN is a logic high, an OAM cell will be discarded if the incoming DMA FIFO buffer is almost full.
11	1	OAM_EN	OAM Enable. Enables detection and processing of OAM cells.
10	1	OAM_QU_EN	OAM Buffer/Status Queue Enable. When a logic high, an OAM cell uses the global OAM free buffer queue and status queue instead of per-channel resources.
9–5	5	OAM_BFR_QU	OAM Free Buffer Queue. When OAM_QU_EN is a logic high, OAM cells use buffers from the free buffer queue identified by OAM_BFR_QU.
4–0	5	OAM_STAT_QU	OAM Status Queue. When OAM_QU_EN is a logic high, OAM cells posts status to the status queue identified by OAM_STAT_QU.

VALUE	VCC BLOCK_SIZE
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	Not valid

0x1f0—Reassembly Free Buffer Queue Base Register (RSM_FQBASE)

This register determines the base address of both banks of contiguous free buffer queue spaces. The base address is a 16-bit number. Since both banks reside in SAR-shared memory (23-bits of byte addressing), the structures can start on 128 byte boundaries. Bank 0 has additional boundary requirements if the buffer return mechanism is enabled.

Bit	Field Size	Name	Description
31–16	16	FBQ1_BASE	Free Buffer Queue Bank 1 base address.
15–0	16	FBQ0_BASE	Free Buffer Queue Bank 0 base address.

0x1f8—Reassembly Free Buffer Queue Control Register (RSM_FQCTRL)

This register contains free buffer queue control information.

Bit	Field Size	Name	Description
31–16	16	Reserved	Not implemented at this time.
15–14	2	FBQ_SIZE	Free Buffer Queue Size. Selects the size of all free buffer queues. 0 = 64 1 = 256 2 = 1,024 3 = 4,096
13	1	FWD_RND	Buffer Return Processing Priority Selection. When a logic low, buffer return entries are processed from queues in priority fashion with queue 15 having the highest priority. When a logic high, round-robin arbitration is used.
12	1	FBQ0_RTN	Free Buffer Queue 0 Buffer Return Enable. When a logic high, bank 0 is enabled to process buffer return for firewall operation. When this bit is set, queue entries 0–15 are four words independent of the value of FWD_EN; otherwise, they are two words.
11–8	4	FWD_EN	Forward Processing Enable. Selects the number of free buffer queues in bank 0 that have buffer return processing enabled. Starting with free buffer queue 0, a value of 0 in FWD_EN selects only one queue, and a value of 15 selects 16 queues.
7–0	8	FBQ_UD_INT	Free Buffer Queue Update Interval. This value determines how many buffers are taken off the free buffer queue before the reassembly coprocessor writes the current read index pointer to host memory.

0x200—Reassembly Table Base Register (RSM_TBASE)

This register consists of a 16-bit address pointer that points to the beginning of the VPI Index table, and a 16-bit address pointer that points to the beginning of the RSM VCC table. Since both tables reside in SAR-shared memory, the tables start on 128-byte boundaries. The size of the VPI Index table used in the direct index method is dependent upon the setting of RSM_CTRL0(VPI_MASK).

Bit	Field Size	Name	Description
31–16	16	RSM_VCCB	Reassembly VCC Table Base Address.
15–0	16	RSM_ITB	VPI Index Table Base Address.

0x208—Reassembly Time-out Register (RSM_TO)

Bit	Field Size	Name	Description
31–16	16	RSM_TO_PER	Reassembly Time-out Interrupt Period. The value in this register determines the number of SYSCLK periods for each time-out interrupt. A value of 0 divides by 65,538.
15–0	16	RSM_TO_CNT	Reassembly Time-out Counter. The value in this register plus one determines the number of time-out interrupts that occur in each pass through the RSM VCC table.

0x218—Reassembly ER_SHIFT Tables Base Register (ERS_BASE)

This register sets the base address in SRC local memory for the ER_SHIFT tables used in implicit host congestion ER reduction. The base addresses are 128-byte aligned, and only the 16 most significant bits of the address are specified in the ERS_BASE register.

Bit	Field Size	Name	Description
31–16	16	Reserved	Program and read as 0.
15–0	16	ER_SHIFT_B[15:0]	Base address for the ER_SHIFT tables.

0x220—Variable VPI Size Register (VPI_SIZE)

This register determines the maximum VPI allowed per port.

Bit	Field Size	Name	Description
31–28	4	MAX_VPI7	Maximum VPI for port. Value — max VPI value 0 — OFF 1 — 1 2 — 3 3 — 7 4 — 15 5 — 31 6 — 63 7 — 127 8 — 255 9 — 511 a — 1,023 b — 2,047 c — 4,095 d — not used e — not used f — not used
27—24	4	MAX_VPI6	Maximum VPI for port 6.
23–20	4	MAX_VPI5	Maximum VPI for port 5.
19–16	4	MAX_VPI4	Maximum VPI for port 4.
15–12	4	MAX_VPI3	Maximum VPI for port 3.
11–8	4	MAX_VPI2	Maximum VPI for port 2.
7–4	4	MAX_VPI1	Maximum VPI for port 1.
3–0	4	MAX_VPI0	Maximum VPI for port 0.

0x228—TX_FIFO_CTRL (TX_FIFO_CTRL)

The TX_FIFO_CTRL register sets a counter compare value that is used for the head of line flushing mechanism of the transmit FIFO. If head of line flushing is enabled in configuration register 1, a counter is reset when the UTOPIA master in multi-PHY mode puts out the address of a PHY device, and the counter is increased based on UTOPIA TX_CLK. Once the counter reaches the TX_CNTR value and no UTOPIA CLAV signal has been received from the PHY device, the cell in the FIFO is discarded.

NOTE: TX_CNTR must not have a value of 0, if the head of the line flushing mechanism is enabled.

Bit	Field Size	Name	Description
31–16	16	Reserved	Always set to 0.
15–0	16	TX_CNTR	Counter values for TX_FIFO flush mechanism.

0x230—TX_PORT_CTRL (TX_PORT_CTRL)

This register disables UTOPIA ports, if the TX_FIFO_FLUSH_EN bit in configuration register 1 is set. If bit x of the TX_PORT_DIS bitmap is set, cells belonging to PHY x are being discarded.

NOTE: No TX_STATUS bit is set.

Bit	Field Size	Name	Description
31–24	24	Reserved	Always set to 0.
7–0	8	TX_PORT_DIS	Disables PHY port x.

14.6 Counters and Status Registers

0x2c0—ATM Cells Transmitted Counter (CELL_XMIT_CNT)

This register counts the number of user data cells transmitted by the segmentation coprocessor. The counter is reset to 0 by an assertion of either the HRST* system reset pin or the GLOBAL_RESET bit in the CONFIG0 register. Optionally, an interrupt can be programmed when the counter rolls over.

Bit	Field Size	Name	Description
31–0	32	CELL_XMIT_CNT	Count of user data cells transmitted.

0x2c8—ATM Cells Received Counter (CELL_RCVD_CNT)

This register counts the number of assigned cells received by the reassembly coprocessor. The counter is reset to 0 by an assertion of either the HRST* system reset pin or the GLOBAL_RESET bit in the CONFIG0 register. Optionally, an interrupt can be programmed when the counter rolls over.

Bit	Field Size	Name	Description
31-0	32	CELL_RCVD_CNT	Count of assigned received cells.

0x2d0—ATM Cells Discarded Counter (CELL_DSC_CNT)

This register counts the number of unassigned cells (failed VCC look-up) received by the reassembly coprocessor. The counter is reset to 0 by an assertion of either the HRST* system reset pin or the GLOBAL_RESET bit in the CONFIG0 register. Optionally, an interrupt can be programmed when the counter rolls over.

Bit	Field Size	Name	Description
31–0	32	CELL_DSC_CNT	Count of unassigned cells dropped.

0x2d8—AAL5 PDUs Discarded Counter (AAL5_DSC_CNT)

This register counts the number of AAL5 CPCS-PDUs discarded due to buffer firewall, buffer underflow, or status overflow. The counter is reset to 0 by an assertion of either the HRST* system reset pin or the GLOBAL_RESET bit in the CONFIG0 register. Optionally, an interrupt can be programmed when the counter rolls over.

Bit	Field Size	Name	Description
31–16	16	Reserved	Not implemented at this time.
15–0	16	AAL5_DSC_CNT	AAL5 PDUs discarded by the reassembly coprocessor.

0x300—Scheduler Head Tail Register (SCH_HD_TL)

This register indicates the current value of the scheduler priority queue head and tail pointers. The queue displayed is determined by the value in the SCH_CTRL (HD_TL_PRI_SEL) register.

Bit	Field Size	Name	Description
31–16	16	SCH_HEAD	Scheduler Priority Queue Head Pointer.
15–0	16	SCH_TAIL	Scheduler Priority Queue Tail Pointer.

0x358—TX_STATUS (TX_STATUS)

This register indicates the status of a PHY device. If the head of line flushing mechanism is enabled (TX_FIFO_FLUSH_EN set in configuration register 1), this register indicates if a cell has been discarded due to the head of the line flushing mechanism. If a bit x of the TX-STAT bitmap is set, PHY x discarded a cell. If one or more bits of TX_STAT is set, an interrupt is generated (if enabled). The TX_STAT bitmap is latched until the TX_STATUS register is read by the host.

Bit	Field Size	Name	Description
31–8	24	Reserved	Always set to 0.
7–0	8	TX_STAT[7:0]	Status of TX port x.

0x348—Host Status Write Register (HOST_ST_WR)

This register indicates if a reassembly or segmentation host-located status queue has been written. Only queues 0–15 are supported. All bits are latched until read by the host. The RSM_HS_WRITE[15:0] bits are ORed together into HOST/LP_ISTAT0(RSM_HS_WRITE), and the SEG_HW_WRITE[15:0] bits are ORed together into HOST/LP_ISTAT0(SEG_HS_WRITE).

Bit	Field Size	Name	Description
31–16	16	RSM_HS_WRITE[15:0]	Indication that a host-located reassembly status queue entry has been written. Only queues 0 through 15 are supported.
15–0	16	SEG_HS_WRITE[15:0]	Indication that a host-located segmentation status queue entry has been written. Only queues 0 through 15 are supported.

14.6.1 Host Interrupt Status Registers

These two registers contain all interruptible status bits for the host processor. The corresponding interrupt enables are located in the HOST_IMASKx registers.

0x380—Host Processor Interrupt Status Register 0 (HOST_ISTAT0)

Bit	Field Size	Type(1)	Name	Description
31	1	L	GPI	This bit reflects the state of the GPI input pin.
30	1	L	PHY_INTR	PHY_INTR can be connected to a PHY interrupt source.
29–24	6	—	Reserved	Read as 0.
23	1	—	Reserved	Read as 0. Reserved for future status page expansion.
22	1	L	HSTAT1	This bit is set when any bit in HOST_ISTAT1 is set.
21–19	3	—	Reserved	Read as 0.
18	1	E	GFC_LINK	Set when three consecutive received cells have GFC SET_A, SET_B, or HALT bits set.
17	1	L	RSM_RUN	Set when the reassembly machine is running. Will be high when the RSM Coprocessor is processing a cell.
16	1	L	RSM_HS_WRITE	Indicates reassembly host status has been written by CN8237 to status queues 0–15. For queue number, read HOST_ST_WR which must be read in order to clear status bit.
15–12	4	—	Reserved	Read as 0.
11	1	L	SEG_RUN	Set when the segmentation machine is running. Will be high when SEG_ENABLE bit in SEG_CTRL is high or when processing the last cell after SEG_ENABLE is low.
10	1	L	SEG_HS_WRITE	Indicates segmentation host status has been written by the CN8237 to status queues 0–15. For queue number, read HOST_ST_WR which must be read in order to clear status bit.
9–4	6	—	Reserved	Read as 0.
3	1	E	AAL5_DSC_RLOVR	Set on the occurrence of an AAL5_DSC_CNT rollover.
2	1	E	CELL_DSC_RLOVR	Set on the occurrence of a CELL_DSC_CNT rollover.
1	1	E	CELL_RCVD_RLOVR	Set on the occurrence of a CELL_RCVD_CNT rollover.
0	1	E	CELL_XMT_RLOVR	Set on the occurrence of a CELL_XMIT_CNT rollover.

NOTE(S):
1. L = Level-sensitive status—A logic 1 on the status bit causes an interrupt when enabled by the corresponding IMASK bit. Reading the status does not clear the status or interrupt. The source of the condition causing the status must be cleared before the status or interrupt is cleared.
2. E = Event driven status—A 0 > 1 transition on the status bit causes an interrupt when enabled. Reading the status register clears the status bit and the interrupt.
3. D = Dual Event status—A 0 > 1 and 1 > 0 transition on the status bit can be enabled to cause an interrupt. Reading the status register clears the status bit and the interrupt.
4. Only host reads reset the status bits in the HOST_ISTAT0 register.

0x388—Host Processor Interrupt Status Register 1 (HOST_ISTAT1)

Bit	Field Size	Type(1)	Name	Description
31	1	L	PCI_BUS_EROR	This bit is set if the MERROR bit in the PCI configuration register is set. The MERROR bit is reset by either writing a logic 1 to the MERROR bit in the PCI configuration register, or setting the CONFIG0(PCI_ERR_RESET) bit to a logic high.
30	1	—	Reserved	Read as 0.
29	1	L	TX_DISCARD	Set when any bit in TX_STATUS is a logic 1. TX_DISCARD is reset to 0 after the Host reads the TX_STATUS register.
28–27	2	—	Reserved	Read as 0.
26	1	E	DMA_AFULL	Set when the incoming DMA burst FIFO becomes almost full.
25	1	E	FR_PAR_ERR	Set on the occurrence of a parity error on the reassembly ATM physical interface.
24	1	E	FR_SYNC_ERR	Set on the occurrence of a synchronization error on the reassembly ATM physical interface.
23–16	8	—	Reserved	Read as 0.
15	1	E	RS_QUEUE_FULL	Reassembly/segmentation queue full condition.
14	1	E	RSM_OVFL	Reassembly overflow. Indicates that a cell was lost due to a FIFO buffer full condition.
13	1	E	RSM_HS_FULL	Set on the occurrence of a host status queue full condition.
12	1	E	Reserved	Read as 0.
11	1	E	RSM_HF_EMPT	Set on the occurrence of a host free buffer queue empty condition.
10–3	8	—	Reserved	Read as 0.
2	1	E	SEG_UNFL	Segmentation underflow indicates that a scheduled cell could not be sent due to lack of PCI bandwidth.
1	1	E	SEG_HS_FULL	Indicates that the segmentation host status queue is full.
0	1	—	Reserved	Read as 0.

NOTE(S):

1. L = Level-sensitive status—A logic 1 on the status bit causes an interrupt when enabled by the corresponding IMASK bit. Reading the status does not clear the status or interrupt. The source of the condition causing the status must be cleared before the status or interrupt is cleared.
2. E = Event driven status—A 0 > 1 transition on the status bit causes an interrupt when enabled. Reading the status register clears the status bit and the interrupt.
3. D = Dual Event status—A 0 > 1 and 1 > 0 transition on the status bit can be enabled to cause an interrupt. Reading the status register clears the status bit and the interrupt.
4. Only host reads reset the status bits in the HOST_ISTAT0 register.

0x3a0—Host Interrupt Mask Register 0 (HOST_IMASK0)

This register contains the interrupt enables that correspond to the status bits in the HOST_ISTAT0 register. The assertion of the HRST* system reset pin clears all of the HOST_IMASK0 interrupt enables.

Bit	Field Size	Name	Description
31	1	EN_GPI	Enables interrupt when CPI status is a logic 1.
30	1	EN_PHY_INTR	Enables interrupt when PHY_INTR status is a logic 1.
29–24	6	Reserved	Set to 0.
23	1	Reserved	Set to 0. Reserved for future status page expansion.
22	1	EN_HSTAT1	Global interrupt enable for HOST_ISTAT1 status register. Individual interrupts in HOST_ISTAT1 are enabled in HOST_IMASK1.
21–19	3	Reserved	Set to 0.
18	1	EN_GFC_LINK	Enables interrupt when GFC_LINK status is a logic 1.
17	1	EN_RSM_RUN	Enables interrupt when RSM_RUN status is a logic 1.
16	1	EN_RSM_HS_WRITE	Enables interrupt when RSM_HS_WRITE status is a logic high.
15–12	4	Reserved	Set to 0.
11	1	EN_SEG_RUN	Enables interrupt when SEG_RUN status is a logic high.
10	1	EN_SEG_HS_WRITE	Enables interrupt when SEG_HS_WRITE status is a logic high.
9–4	6	Reserved	Set to 0.
3	1	EN_AAL5_DSC_RLOVR	Enables an interrupt when AAL5_DSC_RLOVR status is a logic high.
2	1	EN_CELL_DSC_RLOVR	Enables an interrupt when CELL_DSC_RLOVR status is a logic high.
1	1	EN_CELL_RCVD_RLOVR	Enables an interrupt when CELL_RCVD_RLOVR status is a logic high.
0	1	EN_CELL_XMIT_RLOVR	Enables an interrupt when CELL_XMIT_RLOVR status is a logic high.

0x3a8—Host Interrupt Mask Register 1 (HOST_IMASK1)

This register contains the interrupt enables that correspond to the status in the HOST_ISTAT1 register. The assertion of the HRST* system reset pin clears all of the HOST_IMASK1 interrupt enables.

Bit	Field Size	Name	Description
31	1	EN_PCI_BUS_ERROR	Enables interrupt when PCI_BUS_ERROR status is a logic 1.
30	1	Reserved	Set to 0.
29	1	EN_TX_DISCARD	Enables interrupt when TX_DISCARD status is a logic 1.
28–27	2	Reserved	Set to 0.
26	1	EN_DMA_AFULL	Enables interrupt when DMA_AFULL status is a logic 1.
25	1	EN_FR_PAR_ERR	Enables interrupt when FR_PAR_ERR status is a logic 1.
24	1	EN_FR_SYNC_ERR	Enables interrupt when FR_SYNC_ERR status is a logic 1.
23–16	8	Reserved	Set to 0.
15	1	EN_RSQUEUE_FULL	Enables interrupt when RSQUEUE_FULL status is a logic 1.
14	1	EN_RSM_OVFL	Enables interrupt when RSM_OVFL status is a logic 1.
13	1	EN_RSM_HS_FULL	Enables interrupt when RSM_HS_FULL status is a logic high.
12	1	Reserved	Set to 0.
11	1	EN_RSM_HF_EMPT	Enables interrupt when RSM_HF_EMPT status is a logic high.
10–3	8	Reserved	Set to 0.
2	1	EN_SEG_UNFL	Enables interrupt when SEG_UNFL status is a logic high.
1	1	EN_SEG_HS_FULL	Enables interrupt when SEG_HS_FULL status is a logic high.
0	1	Reserved	Set to 0.

14.7 PCI Bus Interface Registers

In accordance with the *PCI Bus Specification, Revision 2.1*, the SAR PCI bus interface implements a 128-byte configuration register space. These configuration registers are used by the host processor to initialize, control, and monitor the PCI bus interface logic. The complete definitions of these registers and the relevant fields within them are given in the PCI bus specification, and are not repeated here. The implementation of the configuration space registers in the CN8237 is shown in Table 14-3. Table 14-4 provides descriptions of fields and other registers within the PCI configuration space.

Table 14-3. PCI Configuration Register

Byte Addr	PCI Configuration Register Layout																														
	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
0x00	DEVICE_ID (0x8237)																VENDOR_ID (0x14F1)															
0x04	STATUS																COMMAND															
0x08	CLASS_CODE (0x020300)																								REV_ID (0x00)							
0x0c	Reserved								HEADER_TYPE (0x00)								LAT_TIMER								CACHE_LINE_SIZE (0x00)							
0x10	BASE_ADDRESS_REGISTER_0																															
0x14-0x28	Reserved																															
0x2C	SUBSYSTEM_ID (SID)																SUBSYSTEM_VENDOR_ID (SVID)															
0x30	Reserved																															
0x34	Reserved																								CAPABILITY_PTR (0x50)							
0x38	Reserved																															
0x3C	MAX_LATENCY (0x05)								MIN_GRANT (0x02)								INTERRUPT_PIN (0x01)								INTERRUPT_LINE (0x02)							
0x40	SPECIAL_STATUS_REGISTER																															
0x44	MASTER_READ_ADDR																															
0x48	MASTER_WRITE_ADDR																															
0x4C	EEPROM_REGISTER																															
0x50	PM_CAPABILITY (0x0001)																NEXT_CAP_PTR (0x58)								CAPABILITY_ID (0x01)							
0x54	PM_DATA								Reserved								PMCSR															
0x58	Reserved								HS_CSR								NEXT_CAP_PT (0x00)								CAPABILITY_ID (0x06)							
0x5C–0x7C	Reserved (0x00000000)																															

Table 14-6 details the PCI Command register and Table 14-5 shows the PCI Status register breakdown. The PCI Special Status register is detailed in Table 14-7. Table 14-8 details the EPROM register.

Table 14-4. PCI Configuration Field Descriptions *(1 of 2)*

Field	Description
DEVICE_ID	16-bit device identifier. Serves to uniquely identify the SAR to the host operating system. Set to 0x8237.
VENDOR_ID	16-bit vendor identifier code, allocated on a global basis by the PCI Special Interest Group. Set to 0x14F1.
STATUS	PCI bus interface status register. The PCI host can monitor its operation using the STATUS field. This field is further divided into subfields. These bits can be reset by writing a logic high to the appropriate bit. See Table 14-5 for a description of the bits in the register.
COMMAND	PCI bus interface control/command register. The PCI host can configure the SAR bus interface logic using the COMMAND field. This field is further divided into subfields. Active HRST* input causes all bits to be a logic 0. See Table 14-6 for a description of the bits in the register.
CLASS_CODE	The CLASS_CODE register is read-only and is used to identify the generic function of the device. See *PCI Bus Specification, Revision 2.1* for the specific allowed settings for this field. Set to 0x020300 (indicates a network controller, specifically an ATM controller).
REV_ID	Revision ID code for the CN8237 chip; 0=Rev A, 1=Rev B, 2=Rev C. Modify per bond-out option.
HEADER_TYPE	This field identifies the layout of the second part of the predefined header of the PCI Configuration space (beginning at 0x10). See *PCI Local Bus Specification*, *Revision 2.1* for the specific possible settings for this field. Set to 0x00.
LAT_TIMER	Latency timer. Value after HRST* active is 0x00. All bits are writable. The suggested value is 0x10 in order to allow the complete transfer of a cell.
CACHE_LINE_SIZE	This read/write register specifies the system cacheline size in units of 32-bit words. Must be initialized to 0x00 at initialization and reset.
BASE_ADDRESS _REGISTER_0	Base address of PCI address space occupied by the CN8237 (as seen and assigned by the host processor). Value after HRST* active is 0x00000000.
SUBSYSTEM_ID (SID)	This register value is used to uniquely identify the add-in board or subsystem where the PCI device resides. Thus, it provides a mechanism for add-in card vendors to distinguish their cards from one another even though the cards can have the same PCI controller on them (and therefore the same DEVICE_ID).
SUBSYSTEM _VENDOR_ID (SVID)	This register value is used to uniquely identify the vendor of an add-in board or subsystem where the PCI device resides. Thus, it provides a mechanism for add-in card vendors to distinguish their cards from one another even though the cards can have the same PCI controller on them (and therefore the same VENDOR_ID).
CAPABILITY_PTR	This field provides an offset into the PCI Configuration space for the location of the first item in the Capabilities Linked List. Set to 0x50.
MAX_LATENCY	This read-only register specifies how often the CN8237 device needs to gain access to the PCI bus, assuming a clock rate of 33 MHz. Set to 0x05 (a period of time in units of 1/4 microseconds).
MIN_GRANT	This read-only register specifies how long of a burst period the CN8237 device needs to gain access to the PCI bus. Set to 0x02 (a period of time in units of 1/4 microseconds).
INTERRUPT_PIN	This read-only register tells which interrupt pin the device (or device function) uses. Set to 0x01 (corresponds to interrupt pin INTA*).

Table 14-4. PCI Configuration Field Descriptions *(2 of 2)*

Field	Description
INTERRUPT_LINE	Interrupt line identifier. The value in this register tells which input of the system interrupt controller(s) the device's interrupt pin is connected to. The device itself does not use this value; rather, it is used by device drivers and operating systems.
SPECIAL_STATUS _REGISTER	Device status not defined by the PCI specification. The field is further subdivided into subfields as shown in Table 14-7. Detailed descriptions of these subfields can be found in the PCI bus specification. The configuration registers are accessed starting from byte address 0 in the configuration space allotted to an adapter card containing the SAR chip. Access to the configuration registers is available only to the PCI host CPU, and is independent of all other SAR logic.
MASTER_READ _ADDR	Current read target address used by PCI bus master (read only).
MASTER_WRITE _ADDR	Current write target address used by PCI bus master (read only).
EEPROM_REGISTER	A 32-bit register controlling access to the Serial EEPROM. See Table 14-8 for a description of the specific fields in the EEPROM_REGISTER.
PM_CAPABILITY	Power Management Capabilities register. A 16-bit read-only register which provides information on the capabilities of the function related to Power Management. See *PCI Bus Power Management Interface Specification, Revision 1.0* for specific information related to this register.
NEXT_CAP_PTR	Next Item Pointer register. This field provides an offset into the PCI Configuration space pointing to the location of the next item of the linked capability list. If there are no additional items in the Capabilities List, this register is set to 0x58.
CAPABILITY_ID	Capability Identifier. When set to 0x01, indicates that the linked list item being pointed to is the PCI Power Management registers. Default value is 0x01.
PM_DATA	Power Management Data register. This 8-bit read-only register provides a mechanism for the Power Management function to report state-dependent operating data, such as power consumed or heat dissipation. See *PCI Bus Power Management Interface Specification,* Revision 1.0 for specific information related to this register.
PMCSR	Power Management Control/Status register. This 16-bit register is used to manage the PCI function's power management state, as well as to enable and monitor power management events. See *PCI Bus Power Management Interface Specification, Revision 1.0* for specific information related to this register.
HS_CSR	CompactPCI Hot Swap Control/Status register.

Table 14-5. PCI Status Register (0x04)

Bit	Field Size	Name	Description
31	1	DPE	Parity Error Detected.
30	1	SSE	Signalled System Error. (Device asserted SERR*.)
29	1	RMA	Received Master Abort. (Device Master aborted transfer.)
28	1	RTA	Received Target Abort. (Detected target abort as master.)
27	1	—	Target aborted as slave.
26–25	2	—	DEVSEL Speed (01 Medium).
24	1	DPR	Reported Data Parity Error. (Parity error detected as master).
23	1	—	Fast Back-to-back supported as Slave. Set to 1.
22	1	—	UDF Support. Set to 0.
21	1	—	66 MHz Support. Set to 0.
20	1	NEW_CAP	Capability List Support. This bit indicates whether this function implements a list of extended capabilities defined in *PCI Bus Specification, Revision 2.1*, such as PCI Power Management. When set to 1, indicates the presence of New Capabilities. A value of 0 indicates that this function does not implement New Capabilities. Set to 1.(1)
19–16	4	Reserved	Set to 0x0.

NOTE(S):
(1) May be set to 0 from EEPROM.

Table 14-6. PCI Command Register (0x04) (1 of 2)

Bit	Field Size	Name	Description
15–10	6	Reserved	Set to 0x0b000000.
9	1	FB_EN	Master Fast Back-to-back enable across target.
8	1	SE_EN	SERR* pin output enable.
7	1	Reserved	Set to 0.

Table 14-6. PCI Command Register (0x04) *(2 of 2)*

Bit	Field Size	Name	Description
6	1	PE_EN(1)	Parity Error detection and report enable.
5–3	3	Reserved	Set to 0.
2	1	M_EN(1) (2)	Master enable. M_EN must be asserted (that is, set to 1) before the CN8237 can act as master on the PCI bus.
1	1	MS_EN(1) (2)	Memory Space enable. MS_EN must be asserted (that is, set to 1) before the CN8237 address space (registers and memory) can be accessed across the PCI interface.
0	1	—	I/O Space enable. (Not used.) Set to 0.

NOTE(S):
(1) May be loaded from EEPROM.
(2) Should only be set after the PCI Base Address register has been set.

Table 14-7. PCI Special Status Register (0x40) *(1 of 2)*

Bit	Field Size	Name	Description
31	1	EN_SID_WR	Enable SVID/SID Write. Default = 0.
30	1	MSTR_CTRL_SWAP(1)	Enable byte swapping on control words that the SAR writes. Default = low.
29	1	SLAVE_SWAP(1)	Enable byte swapping on control words of a slave write or read access. Default = low.
28	1	SLAVE_DWORD(1)	Slave DWORD conversion.
27	1	MSTR_DATA_DWORD(1)	Master Data DWORD conversion.
26	1	MSTR_CTRL_DWORD(1)	Master Control DWORD conversion. ***NOTE(S):*** Set to 1 for normal operation.
25	1	Reserved	Set to 0.
24	1	PCI_EMPTY_MODE	Fast Empty Flag mode. Logic low is default on reset.
23–16	8	Memory Size Mask(1)	An 8-bit mask which sets one of a range of values for the size of the PCI address space. Default=0000000. 00000000 = 16 M 10000000 = 8 M

Table 14-7. PCI Special Status Register (0x40) (2 of 2)

Bit	Field Size	Name	Description
15–12	4	PLL_MUL(1)	PLL_MUL, PLL Multiplier value. Only 0x4 through 0xa (multiplier values of 4 through 10) are valid. 0000 = 16 0001 = 17 0010 = 2 0011 = 3 0100 = 4 0101 = 5 0110 = 6 0111 = 7 1000 = 8 1001 = 9 1010 = 10 1011 = 11 1100 = 12 1101 = 13 1110 = 14 1111 = 15 Upon an active reset, HRST* = 0, this register is set to 0x4 if HM66EN = 1 and 0x8 if HM66EN = 0.
11	1	INTF_DIS	Master Interface Disabled. If the M_EN bit in the COMMAND field is a logic low, any attempt by the CN8237 to perform a DMA transaction to the PCI bus will result in an error. INTF_DIS and MERROR bits set to a logic high. This bit can be reset by writing a logic high to itself.
10	1	INT_FAIL	Master Interface Failed. Set to a logic high when an internal PCI/DMA synchronization error has occurred. The MERROR bit is also set to a logic high. This bit can be reset by writing a logic high to itself.
9	1	MERROR	Memory Error. Indicates that the PCI bus master has encountered a fatal error and therefore has halted operation. Set when either RTA, RMA, DPR, INTF_DIS, or INT_FAIL errors occur. This bit can be reset by writing a logic high to itself.
8	1	MRD	Error on Master Read/Write. If a logic high, indicates that the errored transaction was a read and the address of the read is located in the MASTER_READ_ADDR field. If a logic low, indicates the errored transaction was a write and the corresponding address is located in the MASTER_WRITE_ADDR field.
7–0	8	Reserved	Set to 0x00.

NOTE(S):
(1) Can be loaded from EEPROM.

Table 14-8. **EEPROM Register *(0x4C)***

Bit	Field Size	Name	Description
31	1	BUSY	Indicates that an EEPROM operation is currently in progress. This bit must be read as a 0 before initiating an EEPROM transfer.
30	1	NO_ACK	When set to a 1, indicates that the previous transfer failed (that is, no hardware address response).
29–24	6	Reserved	Reserved for future use.
23–17	7	HARDWARE_ADDR	The 7-bit hardware address for a transfer that indicates the target of the transfer. The EEPROM has the hardware address of b1010000.
16	1	READ_WRITE	Indicates the desired operation. 0=Write; 1=Read.
15–8	8	BYTE_ADDR	The desired byte address of the EEPROM.
7–0	8	DATA	For writes, the data to be written to the EEPROM. For reads, the data read from the EEPROM.

Table 14-9. Power Management Capabilities Register

Bit	R/W	Description
15	R	PME can be asserted from D3 cold (0)
14	R	PME can be asserted from D3 hot (0)
13	R	PME can be asserted from D2 (0)
12	R	PME can be asserted from D1 (0)
11	R	PME can be asserted from D0 (0)
10	R	D2 Support (0)
9	R	D1 Support (0)
8:6	R	Reserved (000)
5	R	Device Specific Initialization Required (0)
4	R	Auxiliary Power Source (0)
3	R	PME Clock (0)
2–0	R	Power Management Version (001)

Table 14-10. Power Management Control/Status Register

Bit	R/W	Description
15	R	PME Status (0) For more details, refer to Section 3.2.4 in *PCI Bus Power Management Interface Specification*, *Version 1.0 S*.
14–13	R	Data Scale (00) For more details, refer to Section 3.2.6 in *PCI Bus Power Management Interface Specification*, *Version 1.0 S*.
12–9	R/W	Data Select (0000) For more details, refer to Section 3.2.6 in *PCI Bus Power Management Interface Specification*, *Version 1.0 S*.
7–2	R	Reserved (000000)
1–0	R/W	Power State (00) 00 = D0, 01 = D1, 10 = D2, 11 = D3hot (SYSCLK_OFF)

Table 14-11. Hot Swap Control/Status Register

Bit	R/W	Description
7	R/W	INS: ENUM# Status—Insertion 1—ENUM# asserted 0—not asserted
6	R/W	EXT: ENUM# Status—Extraction 1—ENUM# asserted 0—not asserted
5-4	R	Reserved (00)
3	R/W	LOO: LED ON/OFF 1—LED ON 2—LED OFF
2	R	Reserved (0)
1	R/W	EIM: ENUM# Signal Mask 1—Mask Signal 0—Enable Signal
0	R	Reserved (0)

15.0 SAR Initialization—Example Tables

NOTE: Initializing the CN8237 SAR is a complex procedure. Mindspeed offers an optional (ANSI C) software package which performs SAR device initialization and programmability. This is the recommended approach for SAR initialization.

The following tables provide an *example* CN8237 SAR initialization of control registers, internal memory control structures, and external memory control structures.

15.1 Segmentation Initialization

15.1.1 Segmentation Control Registers

Before segmentation is enabled, the host must allocate and initialize all of the segmentation control registers. Table 15-1 lists the initialized values for each field.

> ***NOTE:***
> The PCI address of these structures is (Register_Value × 0x80) + SEG_MEMORY_BASE_ADDRESS.

15.1.2 Segmentation Internal Memory Control Structures

Table 15-1. Table of Values for Segmentation Control Register Initialization

Register	Field	Initialized Value	Notes
SEG_CTRL (Segmentation Control Register)	SEG_ENABLE	0–1	Must be set to a logic low until initialization of all segmentation structures is complete. Set to a logic high to commence segmentation process.
	SEG_RESET	0	Use CONFIG0(GLOBAL_RESET) to initialize the SAR.
	VBR_OFFSET	0	Schedule slot priority equals general priority.
	SEG_GFC	0	Disable segmentation GFC processing.
	DBL_SLOT	1	Enable two word schedule slot entries.
	CBR_TUN	1	Enable CBR traffic scheduling.
	ADV_ABR_TMPLT	0	Per-template MCR and ICR ABR parameters.
	USE_SCH_CTRL	0	SCH_CTRL register disabled.
	TX_FIFO_LEN	0x4	Set Transmit FIFO buffer depth to four cells.
	CLP0_EOM	0	Set CLP to zero on last cell of CPCS-PDU.
	OAM_STAT_ID	0x10	OAM global status queue set to 16.
	SEG_ST_HALT	0	Disable status queue entry for a VCC halted with a partially segmented buffer.
	SEG_HS_DIS	0	Enable host status queue full check.
	TX_RND	1	Round-robin transmit queue processing selected.
	TR_SIZE	0x0	Transmit queue size set to 64 entries.
SEG_VBASE (SEG Virtual Channel Connection Base Address Register)	SEG_SCHB	0x1A5	Schedule table starts at 0xD280 in SEG memory.
	SEG_VCCB	0x17D	SEG VCC table starts at 0xBE80 in SEG memory.
SEG_PMBASE (SEG PM Base Address Register)	SEG_BCKB	0x219	VBR bucket table starts at 0x10C80 in SEG memory.
	SEG_PMB	0x1AD	PM table starts at 0xD680 in SEG memory.
SEG_TXBASE (Segmentation Transmit Queue Base Register)	SEG_TXB	0x13D	Transmit queues start at 0x9E80 in SEG memory.
	XMIT_INTERVAL	0x20	Transmit queue update interval set to 32.
	TX_EN	0x8	Transmit queues 0–8 are enabled.

Before segmentation is enabled, the host must allocate and initialize all of the segmentation internal memory control structures. Table 15-2 lists the initialized values for each field.

Table 15-2. Table of Values for Segmentation Internal Memory Initialization

Table	Field	Initialized Value	Notes
SEG_SQ_BASE Table Entry 0 (SEG Status Queue Base Table Entry 0)	BASE_PNTR	0x1C00	Base address of status queue 0x1C00.
	SIZE	0x0	Size of status queue 0 is 64 entries.
	WRITE	0x0	Must be initialized to 0.
	READ_UD	0x0	Must be initialized to 0.
	Reserved	0x0	Must be initialized to 0.
SEG_TQ_BASE Table Entry 0 (SEG Transmit Queue Base Table Entry 0)	READ_UD_PNTR	0x1000	Location of READ_UD.
	UPDATE	0x0	Must be initialized to 0.
	READ	0x0	Must be initialized to 0.
	Reserved	0x0	Must be initialized to 0.

15.1.3 Segmentation SAR Shared Memory Control Structures

Before segmentation is enabled, the host must allocate and initialize all of the segmentation SAR-shared memory control structures. Table 15-3 lists the initialized values for each field.

Table 15-3. Table of Values for Segmentation SAR Shared Memory Initialization *(1 of 2)*

Table	Field	Initialized Value	Notes
SEG VCC Table Entry 0 – (Words 0 – 6)	PM_INDEX	0x0	PM table index = 0.
	LAST_PNTR	0x0	Must be initialized to 0.
	ATM_HEADER	0x00100100	ATM Header, VPI = 0x1, VCI = 0x10.
	CRC_REM	0xFFFF_FFFF	Must be initialized to 0xFFFF_FFFF for AAL5 channels.
	STM_MODE	0	Status Message Mode enabled.
	STAT	0x2	Channel will use status queue 2.
	PM_EN	1	PM OAM processing enabled.
	CURR_PNTR	0x0	Must be initialized to 0.
	VPC	0	VCC connection.
	GFR_PRI	0x2	GFR UBR priority is 2.
	SCH_MODE	0x4	Channel configured for VBR traffic.
	PRI	0x3	Schedule priority is 3.
	SCH_OPT	1	Send maximum burst.
	Reserved	0x0	Must be initialized to 0.
SEG Buffer Descriptors	—	—	(No initialization required.)
SEG Transmit Queue Entries	VLD	0	Must be initialized to 0.
	LINK_HEAD	0	Must be initialized to 0.
	FND_CHAIN	0	Must be initialized to 0.
	SEG_BD_PNTR	0x0	Must be initialized to 0.
	Reserved	0x0	Must be initialized to 0.

Table 15-3. Table of Values for Segmentation SAR Shared Memory Initialization *(2 of 2)*

Table	Field	Initialized Value	Notes
SEG Status Queue Entries	USER_PNTR	0x0	Must be initialized to 0.
	VLD	0	Must be initialized to 0.
	STOP	0	Must be initialized to 0.
	DONE	0	Must be initialized to 0.
	SINGLE	0	Must be initialized to 0.
	OVFL	0	Must be initialized to 0.
	I_EXP	0x0	Must be initialized to 0.
	I_MAN	0x0	Must be initialized to 0.
	SEG_VCC_INDEX	0x0	Must be initialized to 0.
	Reserved	0x0	Must be initialized to 0.
SEG_PM Table Entry 0	ATM_HEADER	0x00100108	VPI = 0x1, VCI = 0 x 10, PTI = 4 (F5 segment).
	FWD_TUC0	0x0	Must be initialized to 0.
	FWD_TUC01	0x0	Must be initialized to 0.
	BCK_MSN	0x0	Must be initialized to 0.
	BCK_TUC0	0x0	Must be initialized to 0.
	BCK_TUC01	0x0	Must be initialized to 0.
	TRCC0	0x0	Must be initialized to 0.
	TRCC0+1	0x0	Must be initialized to 0.
	BIP	0x0	Must be initialized to 0.
	FWD_MON	1	Forward Monitoring cell generation enabled.
	BLOCK_SIZE	10	PM OAM block size of 512.
	FWD_MSN	0x3	Initial sequence number is 3.
	Reserved	0x0	Must be initialized to 0.

15.2 Scheduler Initialization

15.2.1 Scheduler Control Registers

Before segmentation is enabled, the host must allocate and initialize all of the Scheduler control registers. Table 15-4 lists the initialized values for each field.

Table 15-4. Table of Values for Scheduler Control Register Initialization

Register	Field	Initialized Value	Notes
SCH_PRI (Schedule Priority Register)	TUN_ENA7-0	0	No tunnels enabled.
	GFC7-0	0	No GFC priorities enabled.
	QPCR_ENA7-0	0x04	Enable PCR limits on queue 2.
SCH_SIZE (Schedule Size Register)	TBL_SIZE	0x80	Schedule table consists of 128 entries.
	SLOT_PER	0x2E	Schedule slot period is 46 SYSCLK periods.
SCH_ABR_MAX (Schedule Maximum ABR Register)	VCC_MAX	0x63	Enable 50 channels of ABR processing.
PCR_QUE_INT01 (PCR Queue Interval 0 and 1 Register)	QPCR_INT0	0x0	Not used since only one global PCR queue enabled.
	QPCR_INT1	0x0	Not used since only one global PCR queue enabled.
PCR_QUE_INT_23 (PCR Queue Interval 2 and 3 Register)	QPCR_INT2	0x0	Not used since only one global PCR queue enabled.
	QPCR_INT3	0x16C	PCR interval = 364 schedule slots.
SCH_ABR_CON (Schedule ABR Constant Register)	ABR_TRM	0x23C	Set TRM to *TM 4.0* default of 100 ms.
	ABR_ADTF	0xB2E	Set ADTF to *TM 4.0* default of 0.5 second.
SCH_ABRBASE (ABR Decision Table Lookup Base Reg)	OOR_ENA	1	Enable out-of-rate ABR RM cells.
	OOR_INT	0x1C9D	Produces an out-of-rate interval of one cell per second.
	SCH_ABRB	0x1D1	ABR table starts at 0xE880 in SAR-shared memory.
SCH_CNG (ABR Congestion Register)	FBQ_CNG	0x0	No congestion experienced.

15.2.2 Scheduler Internal Memory Control Structures

There are no internal memory scheduler structures that need to be initialized.

15.2.3 Scheduler SAR Shared Memory Control Structures

Before segmentation is enabled, the host must allocate and initialize all of the scheduler SAR-shared memory control structures. Table 15-5 lists the initialized values for each field.

Table 15-5. Table of Values for Sch SAR Shared Memory Initialization *(1 of 2)*

Table	Field	Initialized Value	Notes
Schedule Table (CBR slot)	CBR	1	Slot will schedule a CBR channel.
	CBR_VCC_INDEX	0x0	Schedule seg_vcc_index = 0 channel as CBR.
	Reserved	0xF	Set all reserved bits to 1.
Schedule Table (Tunnel slot)	CBR	0	Slot will schedule a tunnel.
	PRI	0x3	Tunnel Priority Queue = 3.
	Reserved	0xF	Set all reserved bits to 1.
Schedule Table (DEFAULT)	Reserved	0xF	Set all reserved bits to 1.
SEG VCC Table Entry for VBR (Words 7 – 8)	BUCKET2	0x4	Offset of four into Bucket table for VBR2 parameters.
	L1_EXP	0xB	—
	L1_MAN	0x0	GCRA L = 2.
	I1_EXP	0xB	—
	I1_MAN	0x100	GCRA I = 3.
	Reserved	0x0	Must be initialized to 0.
Bucket Table Entry	L2_EXP	0xA	—
	L2_MAN	0x0	GCRA L = 1.
	I2_EXP	0xB	—
	I2_MAN	0x0	GCRA I = 2.
	Reserved	0x0	Must be initialized to 0.
SEG VCC Table Entry for GFR (Words 7 – 9) (NOTE: Set PRI_GFR lower than PRI in word 6.)	MCRLIM_IDX	0x3	Offset of three into GFR MCR_Limit table.
	L1_EXP	0xF	—
	L1_MAN	0x0	GCRA L = 32.
	I1_EXP	0xF	—
	I1_MAN	0x120	GCRA I = 50. (Provides MCR ~ 3 Mbps.)
	Reserved	0x0	Must be initialized to 0.

***Table 15-5. Table of Values for Sch SAR Shared Memory Initialization** (2 of 2)*

Table	Field	Initialized Value	Notes
SEG VCC Table Entry for ABR (Words 7 – 19)	CONG_ID	0x1	Channel associated with RSM free buffer queue 1 for host congestion indication.
	OOR_PRI	0x2	Out-of-rate RM cells assigned to priority queue 2.
	CRM	0x14	*TM 4.0* CRM parameter set to 20.
	MCR_INDEX	—	Output of ABR template.
	ICR_INDEX	—	Output of ABR template.
	FWD_ID	0x01	Forward RM cell ID field set to 1.
	FWD_DIR	0	Forward RM cell DIR field set to 0.
	FWD_BN	0	Forward RM cell BN field set to 0.
	FWD_CI	0	Forward RM cell CI field set to 0.
	FWD_NI	0	Forward RM cell NI field set to 0.
	FWD_RA	0	Forward RM cell RA field set to 0.
	FWD_ER	0x64BF	Forward RM cell ER field set to approximately 360,000 cells per second.
	Reserved	0x0	Must be initialized to 0.
	(ALL OTHER FIELDS)	—	Direct output of ABR template.
ABR Cell Decision Block	(ALL FIELDS)	—	Direct output of ABR template.
ABR Rate Decision Block	(ALL FIELDS)	—	Direct output of ABR template.
Exponent Table	(ALL FIELDS)	—	Direct output of ABR template.

15.3 Reassembly Initialization

15.3.1 Reassembly Control Registers

Before reassembly is enabled, the host must allocate and initialize all of the reassembly control registers. Table 15-6 lists the initialization values for each field.

Table 15-6. Table of Values for Reassembly Control Register Initialization *(1 of 2)*

Register	Field	Initialized Value	Notes
RSM_CTRL0 (Reassembly Control Register 0)	RSM_ENABLE	0–1	Must be set to a logic low until initialization of all reassembly structures is complete. Set to a logic high to commence reassembly process.
	RSM_RESET	0	Use CONFIG0(GLOBAL_RESET) to initialize the SAR.
	VPI_MASK	1	UNI VPI space selected.
	RSM_PHALT	0	RSM halt on receive parity error disabled.
	PREPEND_INDEX	0	Disabled
	FWALL_EN	1	Firewall function enabled.
	RSM_FBQ_DIS	0	Free buffer queue underflow protection enabled.
	RSM_STAT_DIS	0	Status queue overflow protection enabled.
	GTO_EN	1	Enable internal time-out interrupt.
	MAX_LEN	0x10	AAL5 max CPCS-PDU length = 16,384 bytes.
	GDPRI	0x4	Global Service Discard Priority = 4.
	Reserved	0x0	Must be initialized to 0.
RSM_CTRL1 (Reassembly Control Register 1)	EN_PROG_BLK_SZ	1	Disabled
	VCI_IT_BK_SZ	0	Disabled
	EN_VPI_SIZE	0	Disabled
	OAM_FF_DSC	1	OAM cells discarded when Incoming DMA FIFO buffer almost full.
	OAM_EN	1	OAM detection enabled.
	OAM_QU_EN	1	Global OAM FB and Stat queues enabled.
	OAM_BFR_QU	0x4	Global OAM free buffer queue = 4.
	OAM_STAT_QU	0x4	Global OAM status queue = 4.
	Reserved	0x0	Must be initialized to 0.

Table 15-6. Table of Values for Reassembly Control Register Initialization *(2 of 2)*

Register	Field	Initialized Value	Notes
RSM_FQBASE (RSM Free Buffer Queue Base Register)	FBQ1_BASE	0xB0	Free buffer queue bank 1 starts at 0x5800 in SAR-shared memory.
	FBQ0_BASE	0x30	Free buffer queue bank 0 starts at 0x1800 in SAR-shared memory.
RSM_FQCTRL (RSM Free Buffer Queue Control Register)	FBQ_SIZE	0x0	Free buffer queue size is 64 entries.
	FWD_RND	1	Free buffer queues are processed in round-robin fashion for credit return.
	FBQ0_RTN	0	Free buffer queue bank 0 selected for buffer return processing.
	FWD_EN	0x4	Free buffers queues 0 through 4 enabled for buffer return processing.
	FBQ_UD_INT	0x20	Free buffer queue update interval set to 32.
	Reserved	0x0	Must be initialized to 0.
RSM_TBASE (RSM Table Base Register)	RSM_VCCB	0x105	RSM VCC table starts at 0x8280 in SAR-shared memory.
	RSM_ITB	0xF0	VPI Index table starts at 0x7800 in SAR-shared memory.
RSM_TO (RSM Time-Out Register)	RSM_TO_PER	0x100	Internal time-out interrupt every 256 SYSCLK periods.
	RSM_TO_CNT	0x10	RSM VCC table entries 0 through 16 enabled for time-out processing.
RS_QBASE (RSM/SEG Queue Base Register)	RS_SIZE	0x0	RSM/SEG Queue size is 256 entries.
	RS_QBASE	0x12D	RSM/SEG Queue starts at 0x9680 in SAR-shared memory.
	Reserved	0x0	Must be initialized to 0.

15.3.2 Reassembly Internal Memory Control Structures

Before reassembly is enabled, the host must allocate and initialize all of the reassembly internal memory control structures. Table 15-7 lists the initialized values for each field.

Table 15-7. Table of Values for Reassembly Internal Memory Initialization

Table	Field	Initialized Value	Notes
RSM_SQ_BASE Table Entry 0 (RSM Status Queue Base Table Entry 0)	BASE_PNTR	0x1800	Base address of RSM status queue 0.
	SIZE	0x0	Size of status queue 0 is 64 entries.
	WRITE	0x0	Must be initialized to 0.
	READ_UD	0x0	Must be initialized to 0.
	Reserved	0x0	Must be initialized to 0.
RSM_FBQ_BASE Table Entry 0 (RSM Free Buffer Queue Base Table Entry 0)	READ_UD_PNTR	0x0	Location of READ_UD is at 0x0.
	EMPT	0	Must be initialized to 0.
	UPDATE	0x0	Must be initialized to 0.
	READ	0x0	Must be initialized to 0.
	FORWARD	0x20	32 free buffers initially put on free buffer queue.
	LENGTH	0x200	Buffer lengths in free buffer queue 0 are 200 bytes.
	Reserved	0x0	Must be initialized to 0.
Global Time-Out Table	TERM_TOCNT0	0x400	Provides a 133 ms time-out period.
	TERM_TOCNT1	0xFFFF	Provides an 8.5 sec time-out period.
	TERM_TOCNT2	0x400	Provides a 133 ms time-out period.
	TERM_TOCNT3	0x400	Provides a 133 ms time-out period.
	TERM_TOCNT4	0x400	Provides a 133 ms time-out period.
	TERM_TOCNT5	0x400	Provides a 133 ms time-out period.
	TERM_TOCNT6	0x400	Provides a 133 ms time-out period.
	TERM_TOCNT7	0x400	Provides a 133 ms time-out period.
	TO_VCC_INDEX	0x0	Must be initialized to 0.
	Reserved	0x0	Must be initialized to 0.

15.3.3 Reassembly SAR Shared Memory Control Structures

Before reassembly is enabled, the host must allocate and initialize all of the reassembly SAR-shared memory control structures. Table 15-8 lists the initialized values for each field.

Table 15-8. Table of Values for Reassembly SAR Shared Memory Initialization *(1 of 2)*

Table	Field	Initialized Value	Notes
VPI Index Table Entry 0	VP_EN	1	VPI = 0 enabled. NOTE: All entries in the VPI Index table must be initialized. If VPI is disabled, set VP_EN = 0.
	VCI_RANGE	0x10	Allowable VCI range is 0 to 0x43F.
	VCI_IT_PNTR	0x2014	VCI Index table is located at 0x8050 in SAR-shared memory.
VCI Index Table Entry 0	VCC_BLOCK_PNTR	0x0	Initial block of 64 VCC table entries is 0.
	Reserved	0x0	Must be initialized to 0.
RSM VCC Table Entry 0	FF_DSC	1	Enable FIFO buffer Full EPD.
	VC_EN	1	Table Entry is enabled. NOTE: All VCC table entries that have a path through the VPI/VCI lookup space must be initialized. If entry is disabled, set VC_EN = 0.
	AAL_TYPE	0x0	Channel configured for AAL5.
	DPRI	0x2	Channel service discard priority set to 2.
	TO_INDEX	0x1	Point to TERM_TOCNT1 global time-out value.
	PM_INDEX	0x2	Channel used entry 2 of PM_OAM table.
	AAL_EN	0x082	Message Status mode and Frame Relay Discard enabled.
	TO_LAST	0	Not last VCC entry processed for time-out.
RSM VCC Table Entry 0	TO_EN	1	Time-Out processing enabled on this channel.
	CUR_TOCNT	0x0	Must be initialized to 0.
	ABR_CTRL	0x0	No ABR service enabled on this channel.
	PDU_FLAGS	0x002	Must be initialized to 2.
	TOT_PDU_LEN	0x0	Must be initialized to 0.
	CRC_REM	0xFFFF_FFFF	Must be initialized to 0xFFFF_FFFF for AAL5 channels.

Table 15-8. Table of Values for Reassembly SAR Shared Memory Initialization *(2 of 2)*

Table	Field	Initialized Value	Notes
RSM VCC Table Entry 0	STAT	0x1	Channel uses status queue 1.
	BFR1	0x11	Channel uses free buffer queue 17 for COM buffers.
	BFR0	0x1	Channel uses free buffer queue 1 for BOM buffers.
	SEG_VCC_INDEX	0x4	Corresponding SEG channel = 4 for PM_OAM and ABR service.
	SERV_DIS	0x0	Must be initialized to 0.
	RX_COUNTER	0x100	Initial firewall credit = 256.
	Reserved	0x0	Must be initialized to 0.
RSM Buffer Descriptors	NEXT_PTR	0x0	Initialization optional.
	BUFF_PNTR	0x2000	Buffer address of 0x2000.
RSM Free Buffer Queue Entries	VLD	0 – 1	Free buffer queue can be initialized with several free buffers. Valid entries should have VLD = 1, and invalid entries should have VLD = 0.
	BUFFER_PNTR	0x2000	Buffer address of 0x2000.
	BD_PNTR	0x80	Buffer descriptor address of 0x80.
	FWD_VLD	0	Must be initialized to 0.
	VCC_INDEX	0	Must be initialized to 0.
	Reserved	0x0	Must be initialized to 0.
RSM Status Queue Entries	VLD	0	Must be initialized to 0.
	(ALL OTHER ENTRIES)	0	Must be initialized to 0.
LECID Table	LECID0-31	0x20	LANE LECID = 32.
RSM_PM Table Entry 0	BCNT	0x0	Must be initialized to 0.
	BIP16	0x0	Must be initialized to 0.
	MSN	0x4	First expected MSN in forward monitoring cell is four.
	Reserved	0x0	Must be initialized to 0.
	TRCC0	0x0	Must be initialized to 0.
	TRCC01	0x0	Must be initialized to 0.

15.4 General Initialization

15.4.1 General Control Registers

Before the SAR is enabled, the host must allocate and initialize all of the general SAR control registers. Table 15-9 lists the initialized values for each field.

Table 15-9. Table of Values for General Control Register Initialization *(1 of 3)*

Register	Field	Initialized Value	Notes
CONFIG0 (Configuration Register 0)	GLOBAL_RESET	1–0	This must be toggled to a logic high and back to a logic low after completion of all initialization, but before RSM and SEG coprocessors are enabled.
	PCI_MSTR_RESET	0	Use GLOBAL_RESET to reset SAR.
	PCI_ERR_RESET	0	Must be initialized to 0.
	INT_LBANK	0 – 1	Should be set to 0 during initialization, but set to 1 after system reset.
	MEMTYPE	0	Selects zbt memory.
	MEMCTRL	0	Selects 0 wait state SRC shared memory.
	BANKSIZE[2:0]	100	Selects 1 MB memory bank.
	NUMBANKS[1:0]	10	For both RSM and SEG.
	PCI_READ_MULTI	1	PCI Read Multiple Command used.
	PCI_ARB	01	Round-robin arbitration of internal read/write PCI master.
	ENDIAN	1	Big Endian mode selected.
	FORCE64	1	PCI master will always perform 64-bit transfers
	STATMODE	0x0	Selects BOM sync hardware mode.
	DIVIDER	0x0	Divide by 128 selected for CLOCK prescaler.
	Reserved	0x0	Must be initialized to 0.

Table 15-9. Table of Values for General Control Register Initialization *(2 of 3)*

Register	Field	Initialized Value	Notes
CONFIG1 (Configuration Register 1)	PHY_STROBE_IO	0	Low for operation with Mindspeed framers.
	PHY_WAIT	0	Low for operation with Mindspeed framers.
	PHY_RST	0	Sets PRST* to logic high.
	FR_LOOP	0	Loopback disabled.
	UTOPIA_MODE	1	Cell handshake.
	MULTI_PHY	0	Multi-PHY operation disabled.
	UTOP16	1	UTOPIA 16-bit interface is enabled.
	NUM_PORTS[2:0]	0	Multi-PHY disabled.
	SLAVE_ADDR[4:0]	0	When in Master non-Multi-PHY mode, this is the address present on both TxADDR and RxADDR.
	TAG_SIZE[3:0]	0	N/A
	PHYBANK[4:0]	0	Physical Chip Bank Select. This value is placed on PADDR[12:8] when a PHY control access occurs.
	TX_FIFO_FLUSH_EN	0	Tx FIFO flush mechanism disabled.
	INCFIFO_SZ	1	Incoming DMA FIFO buffer size enables to 8 KB.
	NEW_PMOAM	1	Logic high enables the new PM OAM mechanism is enabled per *ITU-T Recommendation I.610*, June 98.
INT_DELAY (Interrupt Delay Register)	TIMER_LOC	0	Interrupt hold-off timer used with HINT*.
	EN_TIMER	0	Disable status queue interrupt timer delay.
	EN_STAT_CNT	1	Enable status queue interrupt counter delay.
	STAT_CNT[7:0]	0x35	Interrupt delay counter set to 53.
HOST_ST_WR (Host Status Write Register)	RSM_HS_WRITE	—	Read twice after SAR reset.
HOST_ISTAT1 (Host Interrupt Status Register 1)	(ALL)	—	Read twice after HOST_ST_WR reset.
HOST_ISTAT0 (Host Interrupt Status Register 0)	(ALL)	—	Read twice after HOST_ISTAT1 reset.
LP_ISTAT1 (Local Interrupt Status Register 1)	(ALL)	—	Read twice after SAR reset.
LP_ISTAT0 (Local Interrupt Status Register 0)	(ALL)	—	Read twice after LP_ISTAT1 reset.
HOST_IMASK1 (Host Interrupt Mask Register 1)	(ALL)	0x8700FC07	Enable all errors to cause an interrupt.

Table 15-9. Table of Values for General Control Register Initialization (3 of 3)

Register	Field	Initialized Value	Notes
HOST_IMASK0 (Host Interrupt Mask Register 0)	(ALL)	0x4040000F	Enable interrupts in errors, counter relievers and framer interrupt.
PCI Configuration Space	COMMAND	0x0346	Enable all functions of PCI interface.
	LAT_TIMER	0x10	Latency timer = 16 clock periods.
	BASE_ADDRESS_ REGISTER_0	0x0100_0000	Base address of SAR device in PCI memory space is 0x0100_0000.
	(ALL OTHER FIELDS)	—	Hard-coded reads only.
PCI COMMAND Register	FB_EN	1	Enable master fast back-to-back across target.
	SE_EN	1	Enable SERR* output pin.
	PE_EN	1	Enable parity error detection and report.
	M_EN	1	Enable CN8237 device master on the PCI bus.
	MS_EN	1	Enable CN8237 memory space access across the PCI bus.
PCI STATUS Register	—	—	(No initialization required.)
PCI SPECIAL_STATUS Register	—	—	(No initialization required.)
PCI EEPROM Register	—	—	(No initialization required.)

16.0 Electrical and Mechanical Specifications

16.1 Timing

16.1.1 PCI Bus Interface Timing

All PCI bus interface signals are synchronous to the PCI bus clock, CLK, except for HRST* and HINT*. Table 16-1 provides the clock specifications and Table 16-2 provides the PCI bus interface timing parameters. Figure 16-1 and Figure 16-2 illustrate this timing.

Table 16-1. Clock Specifications

		66 MHz		33 MHz		
Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Units**
t_{cyc}	CLK Cycle Time	15	30	30	—	ns
t_{high}	CLK High Time	6	—	11	—	ns
t_{low}	CLK Low Time	6	—	11	—	ns
—	CLK Slew Rate	1.5	4	1	4	V/ns

Table 16-2. 66 MHz and 33 MHz Timing Parameters

		66 MHz		33 MHz			
Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Units**	**Notes**
t_{val}	CLK to Signal Valid Delay—bused signals	2	6	2	11	ns	1, 4
$t_{val}(ptp)$	CLK to Signal Valid Delay—point to point signals	2	6	2	12	ns	1, 4
t_{on}	Float to Active Delay	2	—	2	—	ns	4, 5
t_{off}	Active to Float Delay	—	14	—	28	ns	5
t_{su}	Input Set up Time to CLK—bused signals	3	—	7	—	ns	1
$t_{su}(ptp)$	Input Set up Time to CLK—point-to-point signals	5	—	10, 12	—	ns	1
t_h	Input Hold Time from CLK	0	—	0	—	ns	
t_{rst}	Reset Active Time after power stable	1	—	1	—	ms	2
$t_{rst-clk}$	Reset Active Time after CLK stable	100	—	100	—	µs	2
$t_{rst-off}$	Reset Active to output float delay	—	40	—	40	ns	2, 3
t_{rrsu}	REQ64* to RST* setup time	$10T_{cyc}$	—	$10T_{cyc}$	—	ns	—
t_{rrh}	RST* to REQ64* hold time	0	50	0	50	ns	—

NOTE(S):

1. REQ* and GNT* are point-to-point signals and have different input setup times than do bused signals. GNT# and REC# have a setup of 5 ns at 66 MHz. All other signals are bused.
2. RST* is asserted and deasserted asynchronously with respect to CLK.
3. All output drivers must be floated when RST* is active.
4. When M66EN is asserted, the minimum specification for $T_{val}^{(min)}$, $T_{val}^{(ptp)(min)}$, and T_{on} may be reduced to 1 ns if a mechanism is provided to guarantee a minimum value of 2 ns when M66EN is deasserted.
5. For purposes of Active/Float timing measurements, the Hi-Z or "off" state is defined to be when the total current delivered through the component pin is less than or equal to the leakage current specification.

Figure 16-1. PCI Bus Input Timing Measurement Conditions

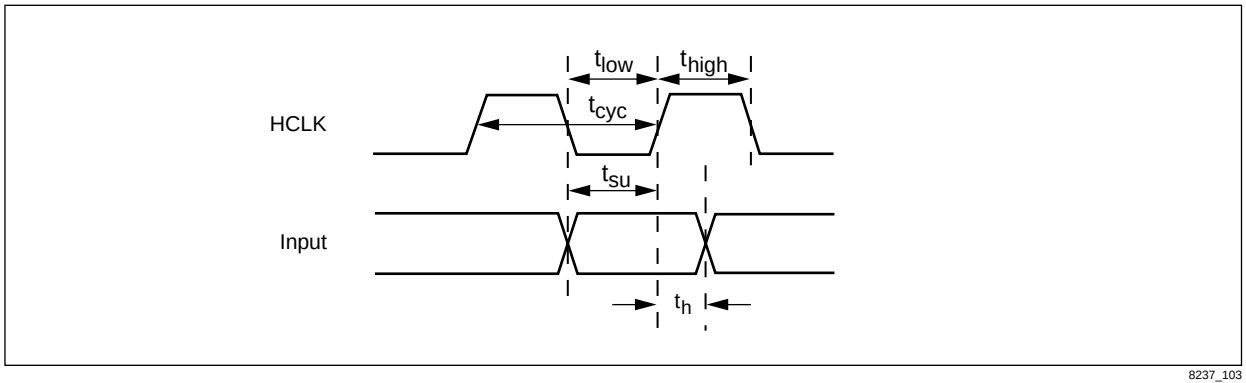
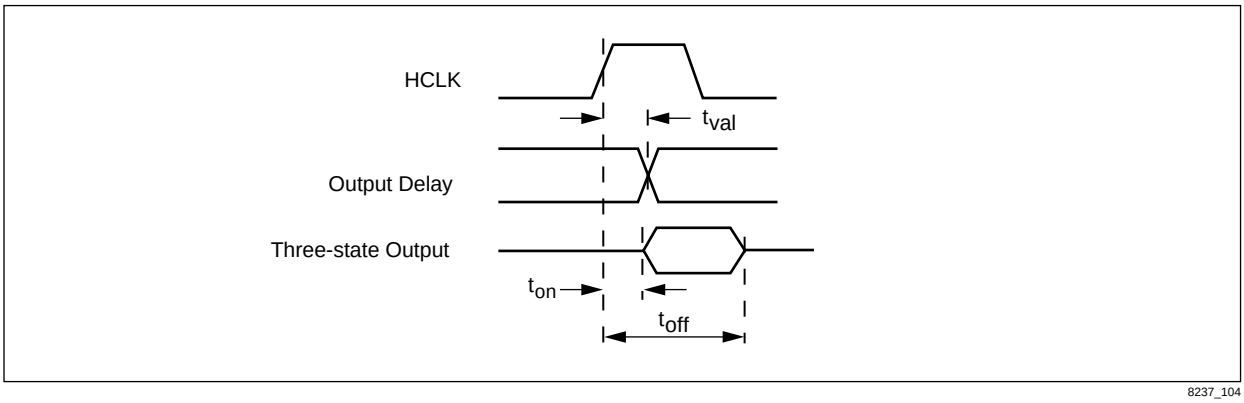


Figure 16-2. PCI Bus Output Timing Measurement Conditions



16.1.2 ATM Physical Interface Timing—UTOPIA and Slave UTOPIA

All ATM physical interface signals are synchronous to the interface clocks, TxCLK and RxCLK, except for TxCLAV (TxFLAG*) and RxCLAV (RxFLAG*) in the slave UTOPIA mode. Timing parameters for the UTOPIA interface are provided in Table 16-3. Table 16-4 provides the timing parameters for the slave UTOPIA interface. Timing diagrams for both interfaces are provided in Figures 16-3 and 16-4.

NOTE: When operating in UTOPIA 8-bit mode, RXDATA[8:15] are gated off. However, these input signals are not terminated and float. These floating signals may induce noise into the SAR and corrupt data received on the UTOPIA Rx Interface. To fix this condition, add 50 kΩ pull-ups to RxDATA[8:15].

Table 16-3. UTOPIA Interface Timing Parameters *(1 of 2)*

Symbol	Parameter	Min	Max	Units
t_{cyc}	TxCLK/RxCLK Cycle Time(1)	20	—	ns
t_{high}	TxCLK/RxCLK High Time(1) (% of tcyc)	40	60	%
t_{low}	TxCLK/RxCLK Low Time(1) (% of tcyc)	40	60	%
t_{su}	RxDATA Input Setup Time to RxCLK(1)	4	—	ns
	RxPAR Input Setup Time to RxCLK(1)	4	—	ns
	RxSOC (RxMARK) Input Setup Time to RxCLK(1)	4	—	ns
	RxCLAV (RxFLAG*) Input Setup Time to RxCLK(1)	4	—	ns
	TxCLAV (TxFLAG*) Input Setup Time to TxCLK(1)	4	—	ns

***Table 16-3. UTOPIA Interface Timing Parameters** (2 of 2)*

Symbol	Parameter	Min	Max	Units
t_h	RxDATA Input Hold Time from RxCLK(1)	1	—	ns
	RxPAR Input Hold Time from RxCLK(1)	1	—	ns
	RxSOC (RxMARK) Input Hold Time from RxCLK(1)	1	—	ns
	RxCLAV (RxFLAG*) Input Hold Time from RxCLK(1)	1	—	ns
	TxCLAV (TxFLAG*) Input Hold Time from TxCLK(1)	1	—	ns
t_{val}	TxCLK to TxDATA Valid Delay(2)	2	12	ns
	TxCLK to TxPAR Valid Delay(2)	2	12	ns
	TxCLK to TxSOC (TxMARK) Valid Delay(2)	2	12	ns
	TxCLK to TxEN* Valid Delay(2)	2	12	ns
	RxCLK to RxEN* Valid Delay(2l	2	12	ns

NOTE(S):
(1) See Figure 16-3 for waveforms and definitions.
(2) See Figure 16-4 for waveforms and definitions. The output delays are measured with a 50 pF load.

***Table 16-4. Slave UTOPIA Interface Timing Parameters** (1 of 2)*

Symbol	Parameter	Min	Max	Units
t_{cyc}	TxCLK/RxCLK Cycle Time(1)	20	—	ns
t_{high}	TxCLK/RxCLK High Time(1) (% of tcyc)	40	60	%
t_{low}	TxCLK/RxCLK Low Time(1) (% of tcyc)	40	60	%
t_{su}	RxDATA Input Setup Time to RxCLK(1)	4	—	ns
	RxPAR Input Setup Time to RxCLK(1)	4	—	ns
	RxSOC (RxMARK) Input Setup Time to RxCLK(1)	4	—	ns
	RxEN* Input Setup Time to RxCLK(1)	4	—	ns
	TxEN* Input Setup Time to TxCLK(1)	4	—	ns
t_h	RxDATA Input Hold Time from RxCLK(1)	1	—	ns
	RxPAR Input Hold Time from RxCLK(1)	1	—	ns
	RxSOC (RxMARK) Input Hold Time from RxCLK(1)	1	—	ns
	RxEN* Input Hold Time from RxCLK(1)	1	—	ns
	TxEN* Input Hold Time from TxCLK(1)	1	—	ns

Table 16-4. Slave UTOPIA Interface Timing Parameters *(2 of 2)*

Symbol	Parameter	Min	Max	Units
t_{val}	TxCLK to TxDATA Valid Delay(2)	2	12	ns
	TxCLK to TxPAR Valid Delay(2)	2	12	ns
	TxCLK to TxCLAV (TxFLAG*) Valid Delay(2)	2	12	ns
	RxCLK to RxCLAV (RxFLAG*) Valid Delay(2)	2	12	ns
	TxCLK to TxSOC (TxMARK) Valid Delay(2)	2	12	ns

NOTE(S):
(1) See Figure 16-3 for waveforms and definitions.
(2) See Figure 16-4 for waveforms and definitions. The output delays are measured with a 50 pF load.

Figure 16-3. UTOPIA and Slave UTOPIA Input Timing Measurement Conditions

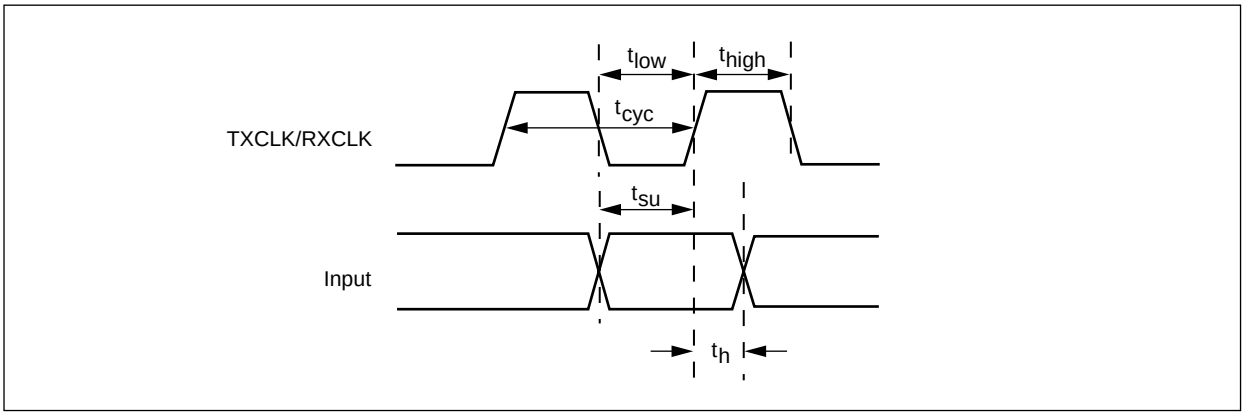
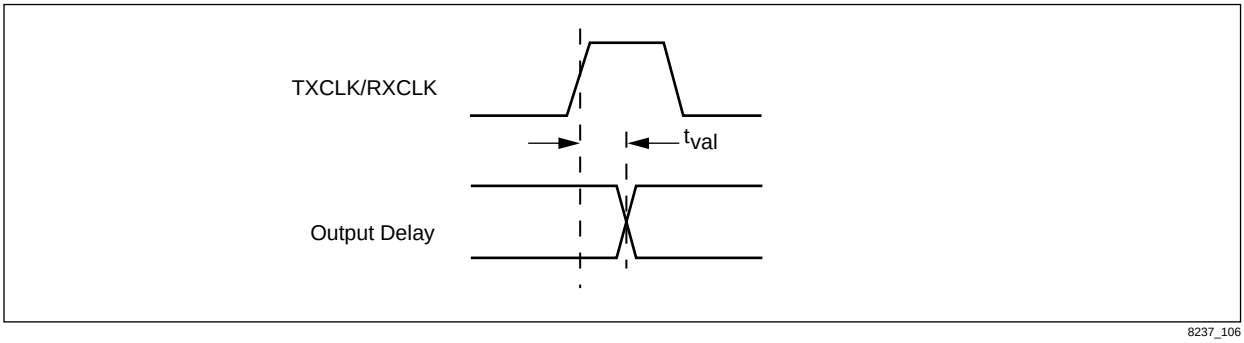


Figure 16-4. UTOPIA and Slave UTOPIA Output Timing Measurement Conditions



16.1.3 CN8237 Local Memory Interface Timing

CN8237 one cycle memory interface timing is provided in Table 16-5. Two cycle memory interface timing is provided in Table 16-6. See Section 10.2 for details of memory bank organization.

Table 16-5. One Cycle Memory Timing Parameter Characteristics

Symbol	Parameter	Min	Max	Units
$t_{s\ rd}$	Data setup time	3	—	ns
$t_{h\ rd}$	Data hold time	0	—	ns
$t_{pd\ a}$	Address propagation delay	3/4T + 1.25	3/4T + 3.75	ns
$t_{pd\ cs}$	Chip select propagation delay	3/4T + 1.25	3/4T + 3.75	ns
$t_{pd\ rnw}$	Read/write propagation delay	3/4T + 1.25	3/4T + 3.75	ns
$t_{pd\ bw}$	Byte write propagation delay	3/4T + 1.25	3/4T + 3.75	ns
$t_{en\ d}$	Data enable	3/4T + 1.25	3/4T + 3.75	ns
$t_{dis\ d}$	Data disable	T/4 + 0.25	T/4 + 1.25	ns

NOTE(S):
(1) 40 pF capacitive load
(2) T = SCLK or RCLK period

Figure 16-5. ZBT Flow Through SRAM Timing Diagram—One Cycle Memory

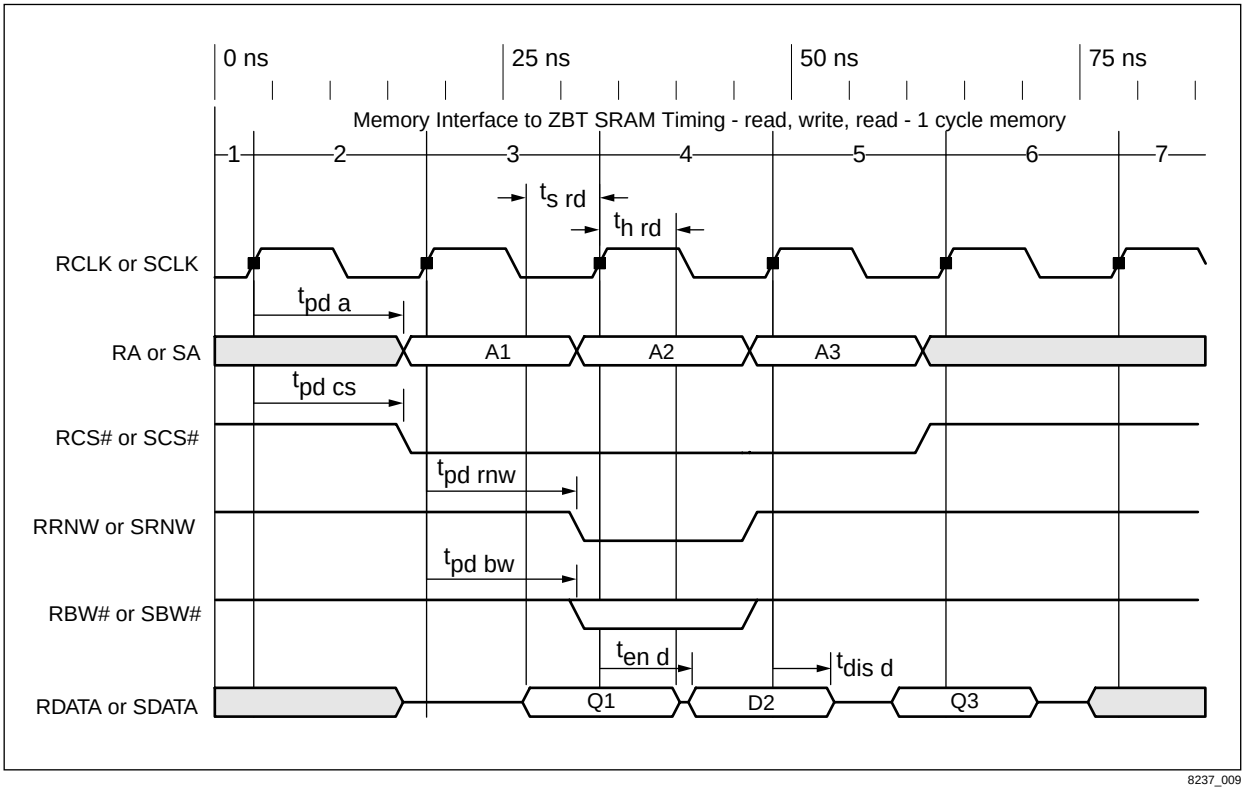


Table 16-6. Two Cycle Memory Timing Parameter Characteristics

Symbol	Parameter	Min	Max	Units
$t_{s\ rd}$	Data setup time	T/4 + 3	—	ns
$t_{h\ rd}$	Data hold time	0	—	ns
$t_{pd\ a}$	Address propagation delay	T/2 + 1.25	T/2 + 3.75	ns
$t_{pd\ cs}$	Chip select propagation delay	T/2 + 1.25	T/2 + 3.75	ns
$t_{pd\ rnw}$	Read/write propagation delay	T/2 + 1.25	T/2 + 3.75	ns
$t_{pd\ bw}$	Byte write propagation delay	T/2 + 1.25	T/2 + 3.75	ns
$t_{en\ d}$	Data enable	T/2 + 1.25	T/2 + 3.75	ns
$t_{dis\ d}$	Data disable	T/4 + 0.25	T/4 + 1.25	ns

NOTE(S):
(1) 40 pF capacitive load
(2) T = SCLK or RCLK period

Figure 16-6. ZBT Flow Through SRAM Memory Timing Diagram—Two Cycle Memory

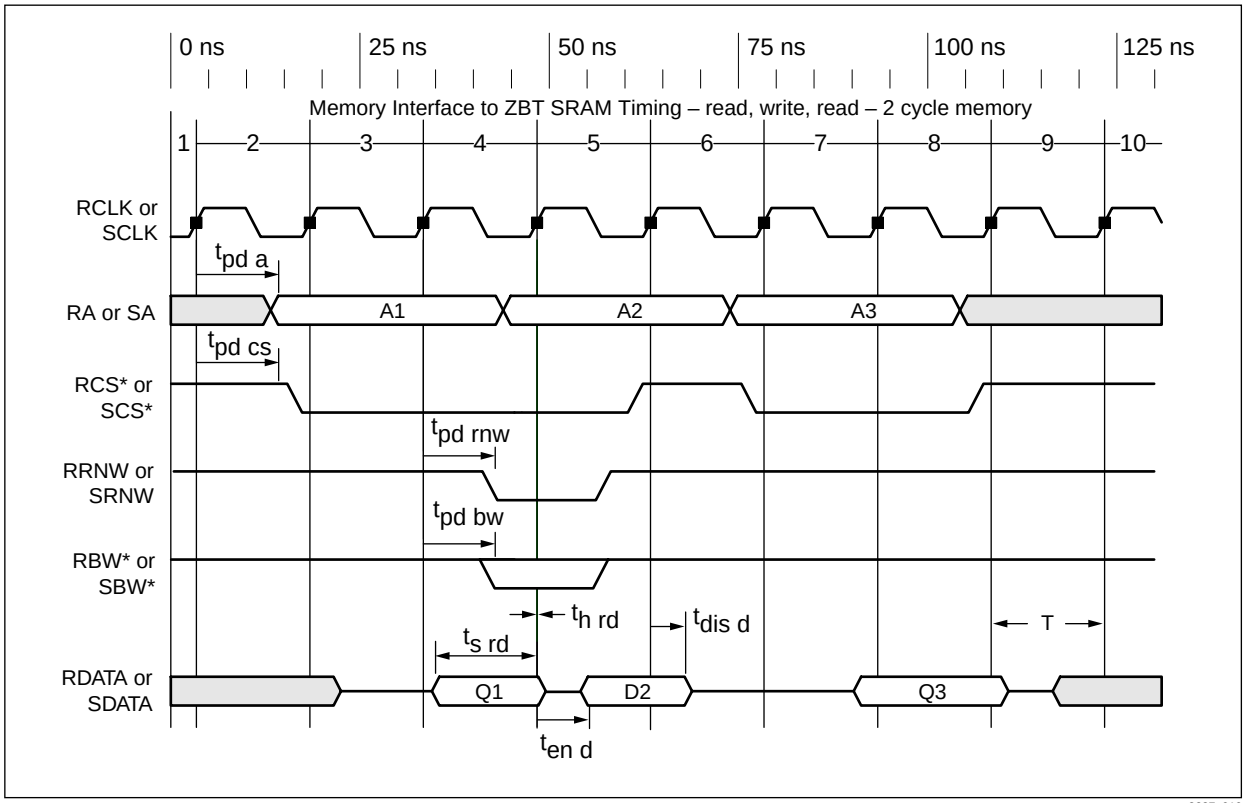


Figure 16-7. Memory Interface to Syncburst Flow Through SRAM Timing—One Cycle Memory

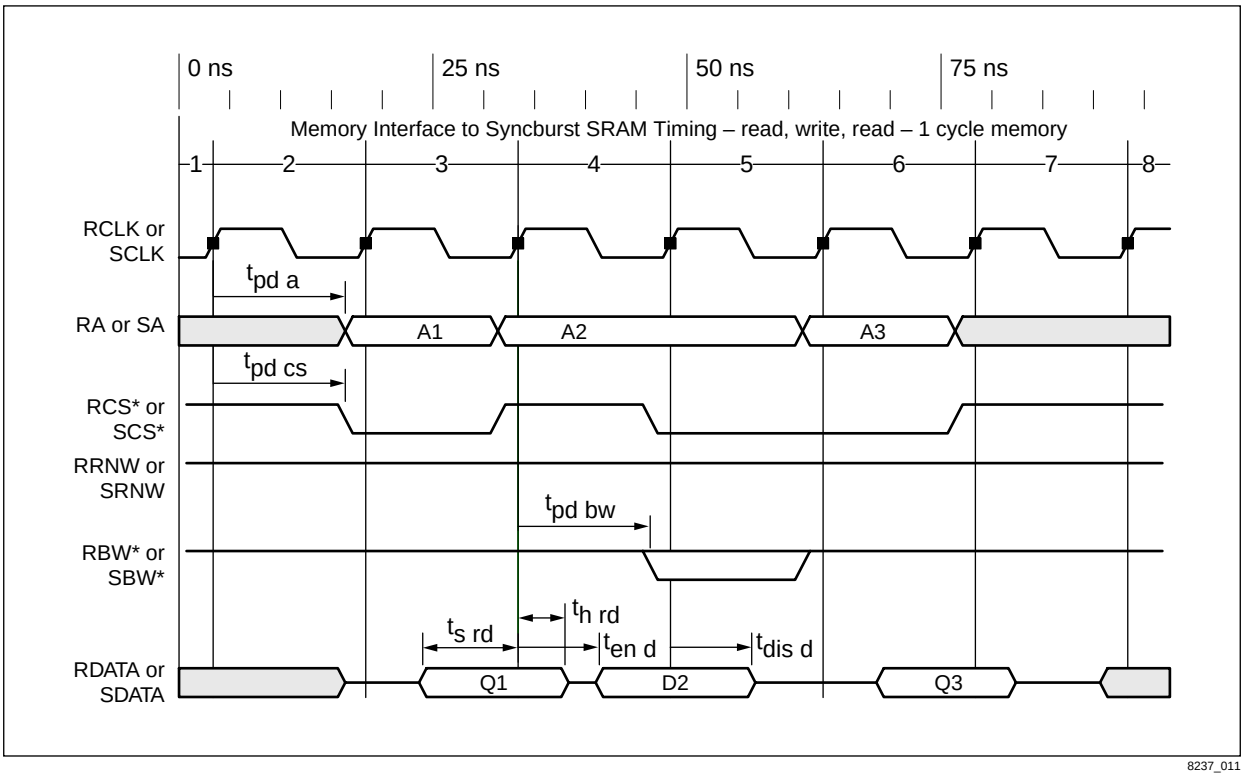
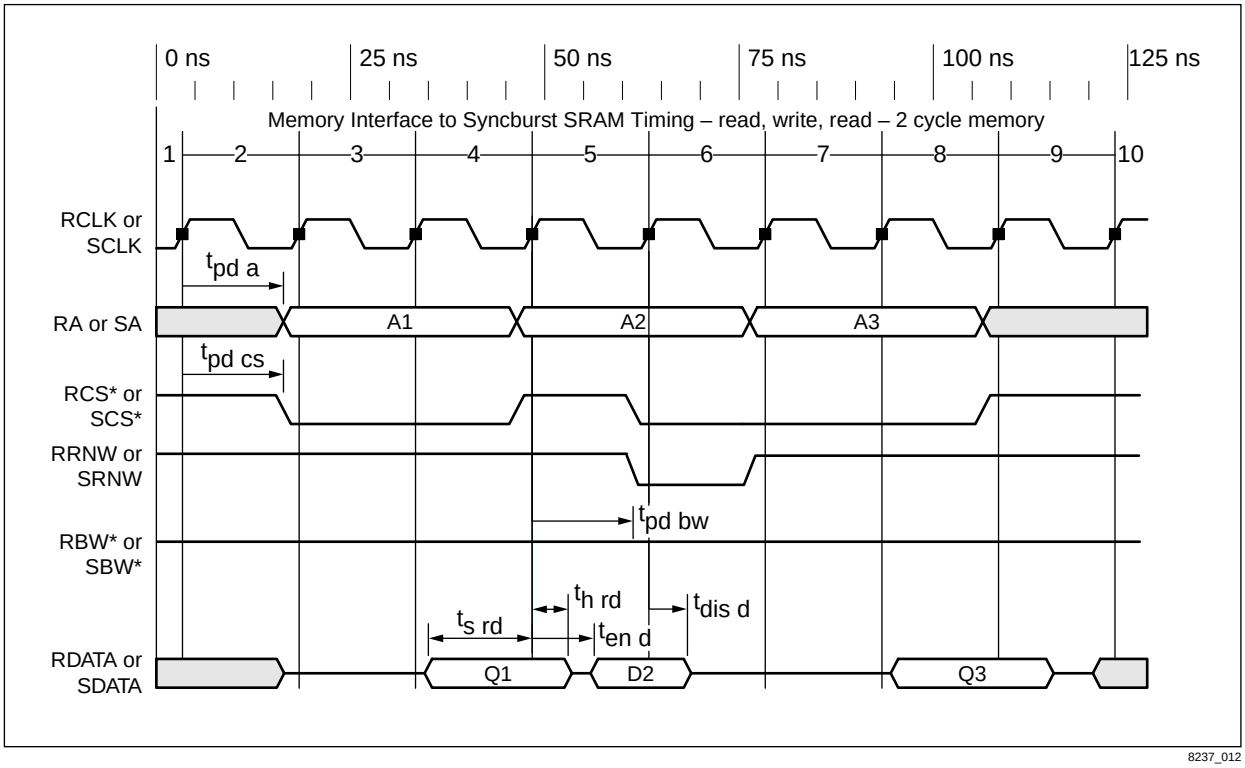


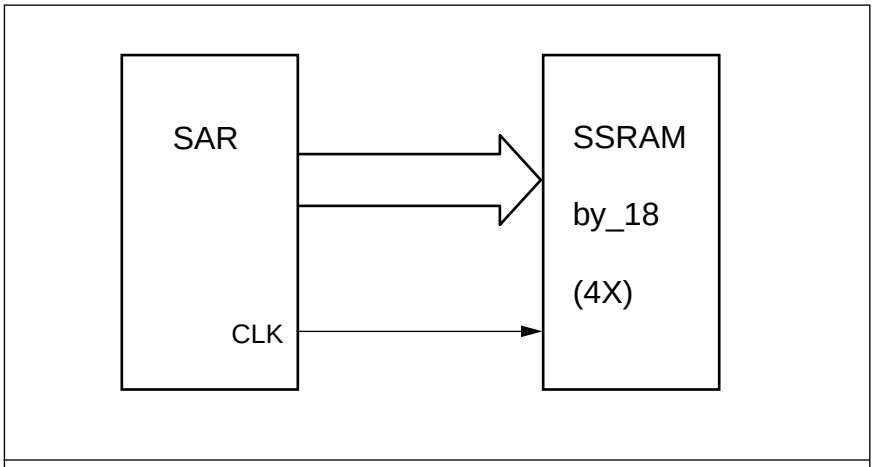
Figure 16-8. Memory Interface to Syncburst Flow Through SRAM Timing—Two Cycle Memory



16.1.3.1 Local Memory Interface Design Details

Figure 16-9 is a simplified drawing of the SAR's Local Memory interface. This applies to both the RSM and SEG coprocessor's Local Memory interfaces. The memory used is flow-through ZBT synchronous SRAM (Samsung KM718V847T-8) or equivalent. The capacitive load seen by the SAR is estimated at 40 pF.

Figure 16-9. SAR's Local Memory Interface



NOTE(S):
Not shown are series terminators located close to the source on all signals.

8237_161

The SAR source driver impedances for Address, xBW#, xRNW and xCLK are 32 ohms (12 ma). The PCB trace impedances are approximately 50 ohms. The series terminators were selected to be 18 ohms so that the SAR driver impedance matches the board trace impedance.

The local memory interface design is divided into the four steps that follow.

Step 1 – Calculating the ADDR/CNTRL/DATA Set-up to the SSRAMs

Calculate the ADDR/CNTRL/DATA set-up to the SSRAM using the SAR's absolute maximum signal propagation delay:

T – (3/4T + 3.75) = Set-up Time

T equals the clock period

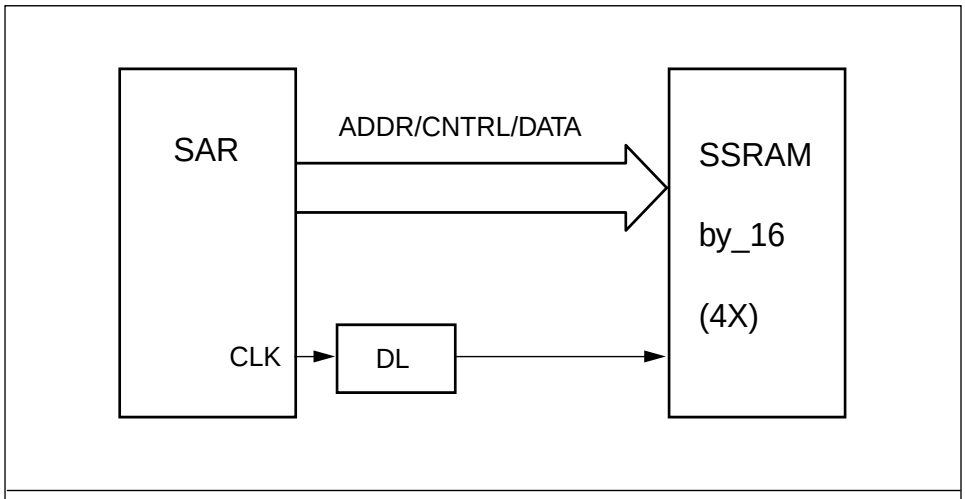
NOTE: The PCB traces are assumed to be of similar length; therefore, signal propagation delays or Flight Times (FT) will be similar.

Using a clock frequency of 66.66 MHz, T = 15.0 ns

The Set-up Time will be equal to 0 ns.

The synchronous SRAMs require a minimum of 2.0 ns (as devices increase in speed, the minimum set-up time is reduced) for set-up time. This design therefore requires that a delay of at least 2.0 ns be added to the clock lines. An initial approach was to adjust the series terminators for the clocks upwards to achieve the equivalent of a 2 ns delay. This yielded a Set-up Time of 2.0 ns for the SSRAMs. Eighteen ohm resistors were used on all signals except the clock lines which had 70 ohm resistors. The large series (70 ohm) terminators on the clock lines cause the signal fidelity to degrade (i.e., rounding of the square wave, slow rise and fall times, and a voltage shift above ground). A better solution is to use 18 ohm series terminators on the clock lines and then add a Delay Line (DL) on the output of the series terminator which needs to provide an absolute minimum delay of 2.0 ns. (See Figure 16-10.) To accomplish a 2.5 ns Delay Line, Thin-Film DS1L5DJ250S was selected which has a tolerance of +/- 5% (DL range is 2.375 ns to 2.625 ns) and an impedance of 50 ohms.

Figure 16-10. Add a Delay Line



NOTE(S):
Not shown are series terminators located close to the source on all signals.

8237_162

Step 2 – Calculating the ADDR/CNTRL/DATA Hold Time to the SSRAMs

Calculate the ADDR/CNTRL/DATA hold time to the SSRAM using the SAR's absolute minimum signal propagation delay(s):

T/4 + 0.25 ns – DL = HOLD TIME (this applies to the DATA; ADDR and CNTRL are held longer on the bus 3/4T + 1.25 ns)

15/4 + 0.25 ns – 2.375 ns = 1.625 ns

The hold time for the SSRAMs is 0.5 ns (or less) and this design provides for 1.625 ns. This gives a margin of 1.125 ns.

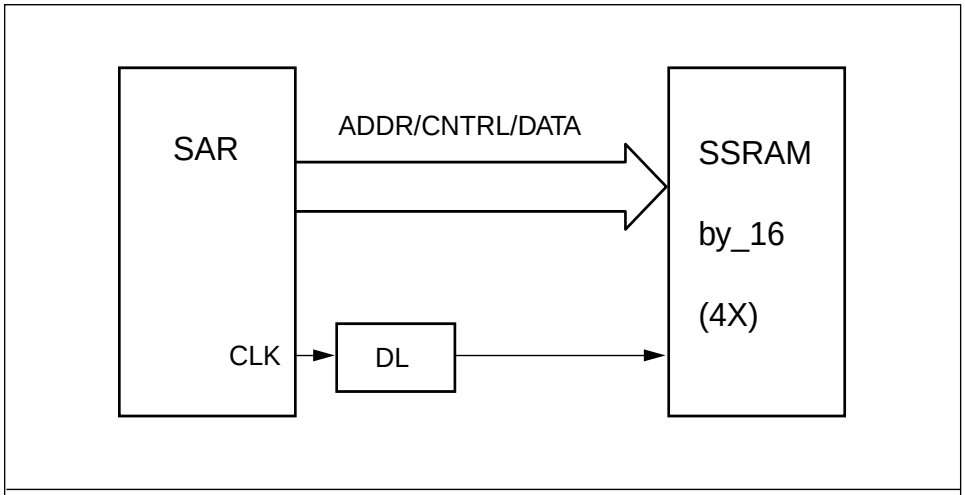
Step 3 – Calculating the SSRAM Access Speed

Calculate the SSRAM access speed in order for the SAR to have a minimum of 3 ns of data set-up:

T – DL – FT1 – FT2 – 3 ns = Max. access speed for SSRAM

FTn represents the signal flight time or signal propagation delay from source to destination. The CN8237EVM card was measured for PCB signal propagation delay which came in at 3.1 ns for 12 inches (or 30.5 cm) of trace. All local memory signal traces are less than 3 inches (or 7.5 cm). The Flight Time for FT1 and FT2 was set at 0.8 ns each. (See Figure 16-11.)

Figure 16-11. Flight Time



NOTE(S):
Not shown are series terminators located close to the source on all signals.

8237_162

Using the above design numbers yields the following:

15 ns – 2.625 ns – 0.8 ns – 0.8 ns – 3 ns = 7.775 ns (maximum access speed for SSRAMs)

The Samsung parts used on the CN8237EVM card have a maximum access speed of 8.5 ns.

NOTE: This gives the design some exposure to failure should the memory chips exceed an access speed of 7.775 ns.

NOTE: A reduction in Flight Time gives direct relief to SSRAM access speed.

In this example, if all Local Memory PCB trace lengths are kept less than 2 inches (or 5.08 cm), then the maximum SSRAM access speed could be 8.3 ns.

Step 4 – Calculating the Data Hold Time to the SAR

Calculate the data hold time to the SAR:

DL + FT1 + FT2 + (SSRAM data disable time) = data hold time to SAR

Assume the SSRAM minimum data disable time to be 0 ns.

2.375 ns + 0.8 ns + 0.8 ns = 3.975 ns

The SAR has a 0 ns minimum data hold time requirement; therefore, there is plenty of margin in this design.

Conclusions and Recommendations

As is evident by the data presented thus far, the SAR (CN8237) Local Memory interface requires attention to detail. The Local Memory interface is tightly timed and requires that the clock signal(s) be delayed a minimum of 2.0 ns. The clock delay may be accomplished with the use of a delay line (either active or passive). Keeping all Local Memory PCB trace lengths to an absolute minimum and the same relative length yields the optimal timing. This requires the memory chips to be placed as close to the SAR device as possible. All of the Local Memory PCB traces need to be of similar length (i.e., +/– 10%) with exception to the clock line(s) which must be equal to the maximum Local Memory PCB trace length.

Figures 16-12 and 16-13 show a recommended PCB trace layout scheme when using by_16 SSRAM parts and populating two banks (per coprocessor). Two of the SSRAM devices are placed on the same side of the board as the SAR and the other two are placed on the back side of the board under the first two SSRAM devices. The memory chips are connected to the pads at the ends of the L2 PCB trace segments. The series terminator (Rs) should be placed as close to the (SAR) source pin as is possible. The total signal trace length will be L1 + L2 with the bulk of the trace length attributed to the L1 segment. In other words, route the traces such that the L2 segments are kept to a minimum and insure that the two L2 segments are of equal length.

Figure 16-12. Recommended PCB Trace Layout Scheme

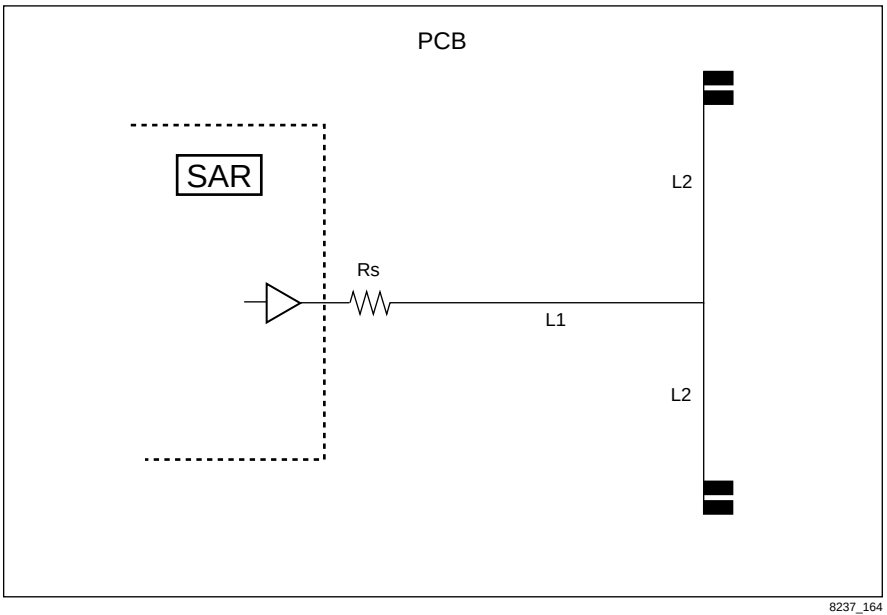
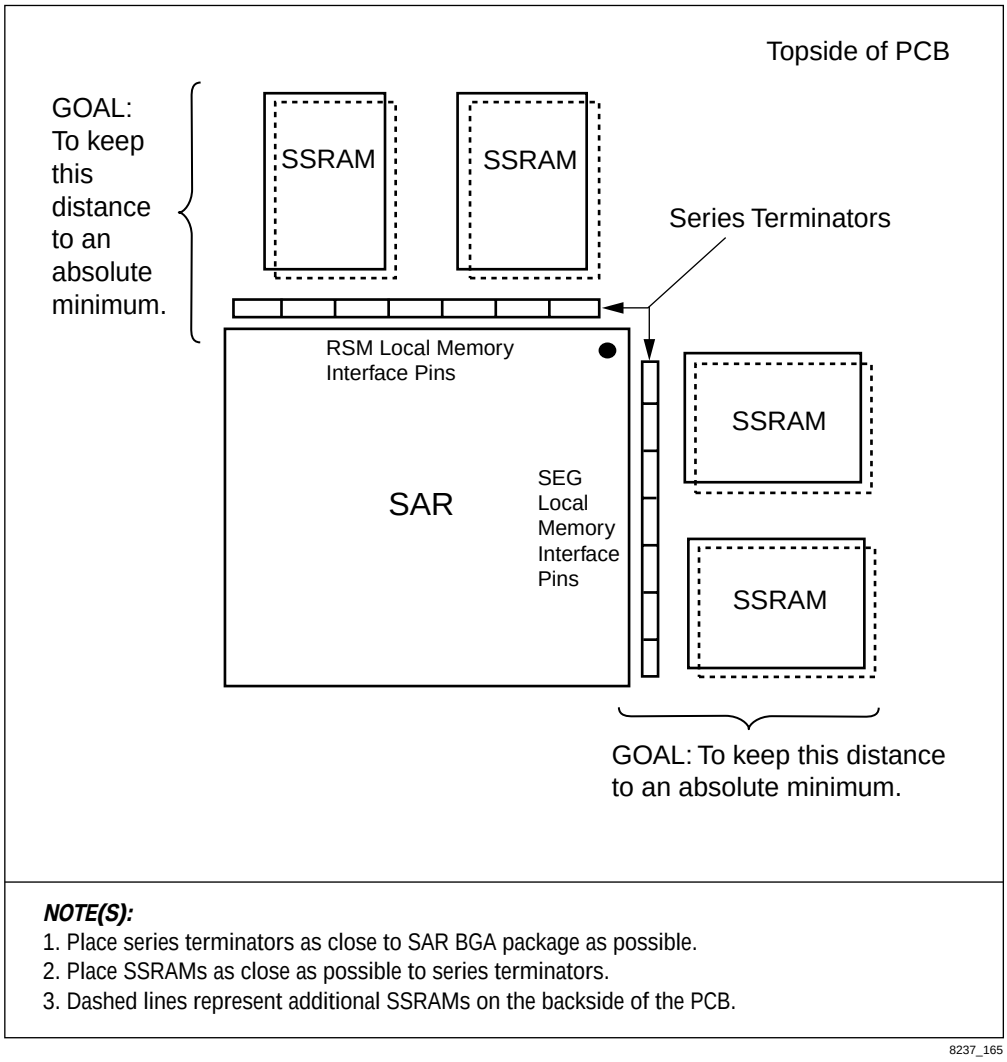


Figure 16-13. Recommended PCB Layout



NOTE(S):
1. Place series terminators as close to SAR BGA package as possible.
2. Place SSRAMs as close as possible to series terminators.
3. Dashed lines represent additional SSRAMs on the backside of the PCB.

8237_165

A daisy chain routing approach may be more practical for some PCB designs. The most important point is to keep all PCB traces to a minimum and of equivalent length. Following a careful timing analysis and careful board layout keeps the design within all minimum and maximum design parameters.

If the application requires more than two banks of memory, then by_36 SSRAM chips should be used. This allows PCB trace routing to be similar to that of the by_16 SSRAM devices and also keeps the capacitive loading the same.

16.1.4 PHY Interface Timing

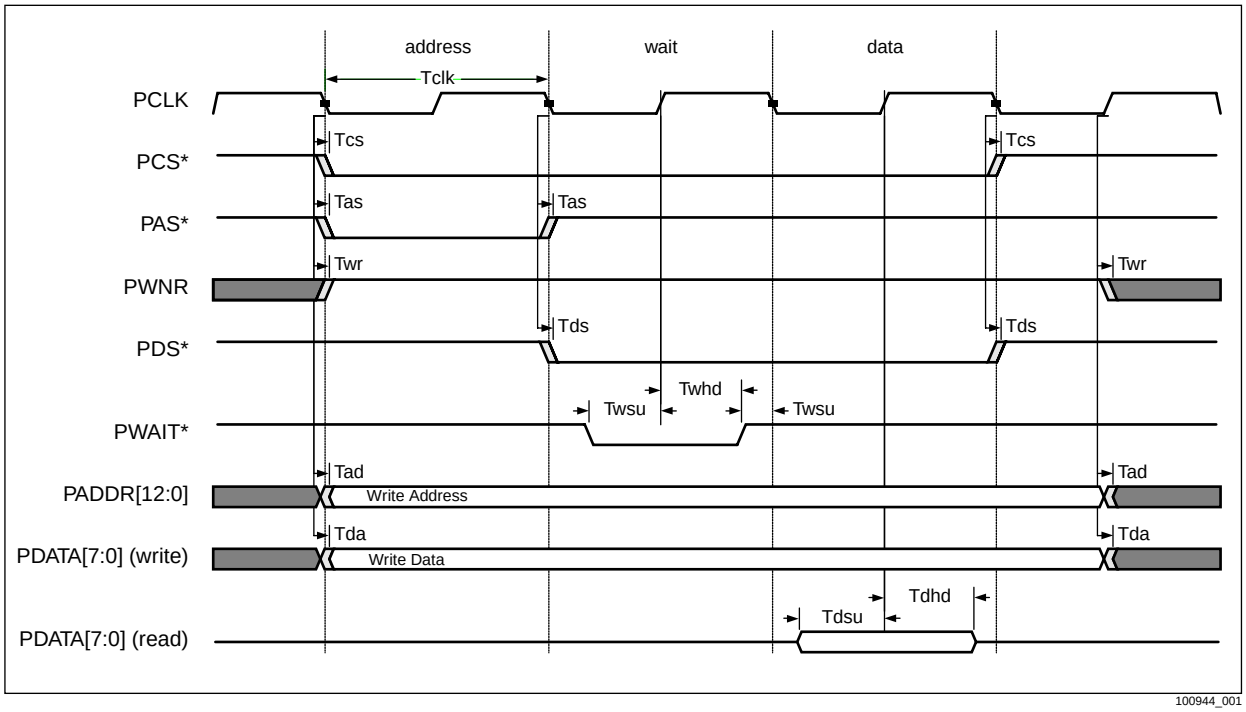
CN8237 PHY interface timing for the normal mode is provided in Table 16-7.

Table 16-7. Interface Timing—Normal Mode

Symbol	Parameter	Min	Max	Units
Tclk	PCLK period(1)	30.2	—	ns
Tclkds	PCLK duty cycle	45	55	%
Tcs	PCLK fall to PCS*	–4	4	ns
Tas	PCLK fall to PAS*	–4	4	ns
Twr	PCLK fall to PWNR*	–4	4	ns
Tds	PCLK fall to PDS*	–4	4	ns
Tad	PCLK fall to PADDR	–4	4	ns
Tda	PCLK fall to PDATA (Write)	–4	4	ns
Twsu	PWAIT* setup to PCLK rise or fall	3	—	ns
Twhd	PWAIT* hold after PCLK rise or fall	1	—	ns
Tdasu	PDATA setup before PCLK rise (Read)	8	—	ns
Tdahd	PDATA hold after PCLK rise (Read)	1	—	ns

NOTE(S): All outputs are measured driving a 50 pF load.
(1) PCLK frequency is a function of the PCI CLK (HCLK) and the PLL divider that generates the internal SYSCLK.

Figure 16-14. CN8237/PHY Functional Timing with Inserted Wait States (Normal Mode)



Timing for strobe mode is shown in Figure 16-15. Timing numbers not found in this diagram are the same as those in the previous normal mode timing diagram. A single internally generated wait state is shown for illustrative purposes.

Figure 16-15. Interface Timing Diagram—Strobed Mode

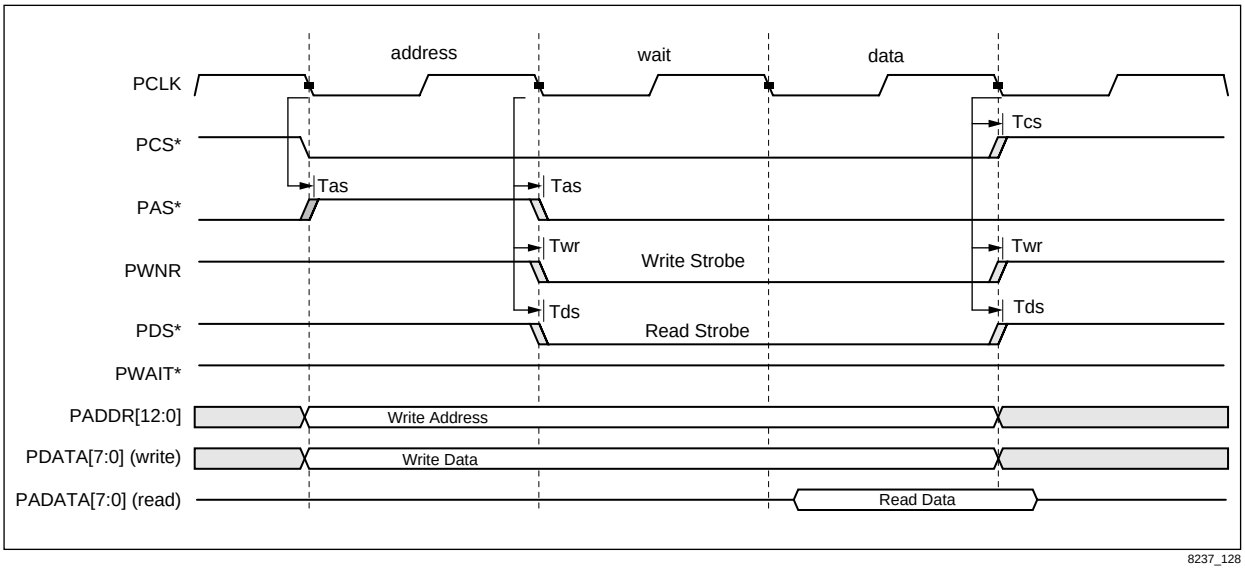


Table 16-8. Interface Timing—Strobed Mode

Symbol	Parameter	Min	Max	Units
Tas	PCLK fall to PAS* (ALE)	–4	4	ns
Twr	PCLK fall to PWNR (WRB)	–4	4	ns
Tds	PCLK fall to PDS* (RDB)	–4	4	ns
Tcs	PCLK fall to PCS*	–4	4	ns
NOTE(S): All outputs are measured driving a 50 pF load. Other timing properties are the same as Normal Mode.				

16.2 Package I/O Description

The package interface (I/O) pins and descriptions are given in Table 16-9. In normal mode, the PCS*, PAS*, PDS*, and PWNR pins are outputs providing chip select, address strobe, data strobe and write/read control to the PHY device. When the PDS* output is active, it indicates the cycles in which the data transaction occurs. In strobed mode, PAS*, PDS*, and PWNR become address, read and write strobes, respectively.

Table 16-9. Package Interface Pins

Signal	Dir(1)	Description (Normal Mode)	Description (Strobed Mode)
PRST*	O	Normally active low reset to PHY device. Set by a control register and HRST*.	
PCLK	O	Processor clock to the PHY device. All interface control signals are output at the falling edge of this clock. Inputs are sampled at the rising edge.	
PCS*	O	Active low chip select output to the PHY device.	
PAS*	O	Active low address strobe to the PHY device.	Active high address strobe.
PDS	O	Active low data strobe to the PHY device.	Active low read strobe.
PWNR	O	Read/Write select. A logic 1 on the output indicates a write cycle, while a logic 0 indicates a read cycle.	Active low write strobe.
PWAIT*	O	Active low wait input from the PHY device. Allows extension of the data phase of the interface cycle if the interface finds this input low. Sampled by PCLK rising and falling edges.	
PDATA[7:0]	B	Bidirectional data bus between the SAR and the PHY device.	
PADDR[12:0]	O	Address bus to the PHY device.	
PINT*	I	Active low interrupt input from PHY device.	

NOTE(S):
(1) Direction given with respect to the SAR.

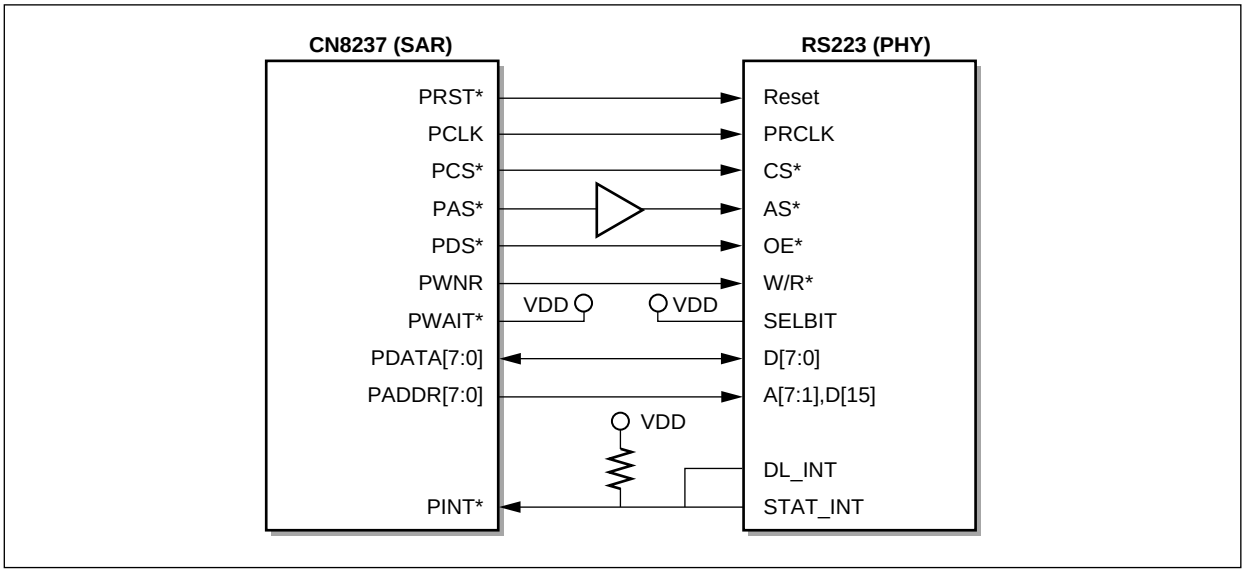
16.3 CN8223 PHY Device Connection

Connection to the Mindspeed RS8223 PHY device is shown in Figure 16-16. Note that RS8223 requires an active high reset which is opposite from the normal sense of the PRST* output. Activating HRST_ through the PCI port sets PRST* low. Once HRST_ returns high, the PHY_RST control must be manipulated by control register writes to set PHY_RST and PRST* to the desired state. This can be used to set the PRST* output to the appropriate reset logic sense.

The PAS* signal needs to be stable during PCLK high to avoid a race inside the RS8223. Allowing PAS* to transition while PCLK is transitioning could cause a glitch in the RS8223 resulting in erroneous operation of the interface. This timing requirement is met by adding an external delay to the PAS* signal. Care needs to be taken in the PC layout to insure this condition is met.

With the RS8223, a wait state needs to be inserted during reads from odd addresses. This is due to the use of the 8-bit RS8223 interface mode which adds a single clock delay in presenting the read data on the data bus during reads to odd addresses only. This is not documented in the RS8223 data sheet. Setting the PHY_WAIT control to one provides for this requirement. Note that all accesses take an additional cycle with this control set, not just reads to odd addresses.

Figure 16-16. RS8223 PHY Device Connection



16.4 Absolute Maximum Ratings

Stresses above those listed as absolute maximum ratings (Table 16-10) can cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in other sections of this document is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability. This device should be handled as an ESD-sensitive device. Voltage on any signal pin exceeding 5.5 V can induce destructive latchup.

Table 16-10. Absolute Maximum Ratings

Parameter	Value	Unit
Supply Voltage (VDD)	–0.5 to +4	V
Input Voltage	–0.5 to +5.5	V
Output Voltage	–0.5 to (VDD + 0.3)	V
Theta Ja (No Air Flow)	23.4	C/W
Storage Temperature	–40 to 125	C
Maximum Current @ Max Clock Frequencies(1)	830	mA
Moisture Sensitivity Level (MSL)	4	—

16.5 DC Characteristics

The DC electrical characteristics are listed in Table 16-11.

Table 16-11. DC Characteristics *(1 of 2)*

Parameter	Conditions	Min	Max	Unit
Operating Supply Voltage (VDD)	—	3.135	3.63	V
Output Voltage High	I_{OH} = –500 µA (for all 3.3 V PCI signaling)	0.9*VDD	—	V
	I_{OH} = –2 mA (for all 5 V PCI signaling)	2.4	—	V
	I_{OH} = 4.0 mA (for all other signals)	2.4	—	V
Output Voltage Low	I_{OL} = 1500 µA (for all 3.3 V PCI signals)	—	0.1*VDD	V
	I_{OH} = 3 mA, 6 mA(1) (for all 5 V PCI signaling)	—	0.55	V
	I_{OL} = 4.0 mA (for all other signals)	—	0.4	V
Input Voltage High (PCI)	(for 3.3 V PCI signaling)	0.5*VDD	5.25	V
	(for 5 V PCI signaling)	2.0	5.25	V
Input Voltage High (RxCLK, TxCLK)	—	0.7*VDD	5.25	V
Input Voltage High (TCLK, TRST_)	—	2.52	5.25	V
Input Voltage High (all others)	—	2.0	5.25	V
Input Voltage Low (PCI)	(for 3.3 V PCI signaling)	–0.5	0.25*VDD	V
	(for 5 V PCI signaling)	–0.5	0.8	V
Input Voltage Low (RxCLK, TxCLK)	—	0	0.3*VDD	V
Input Voltage Low (TCLK)	—	0	0.4	V
Input Voltage Low (all others)	—	0	0.8	V
Input Leakage Current	Vin = VDD or GND	–10	10	µA
Three-state Output Leakage Current	VOUT = VDD or GND	–10	10	µA
Pullup Current	RxEN*, TxEN*, TRST*, TMS, TDI	30	100	µA
Pulldown Current	RxSOC (RxMARK), RxCLAV (RxFLAG*), TxCLAV (TxFLAG*)	30	100	µA

Table 16-11. DC Characteristics *(2 of 2)*

Parameter	Conditions	Min	Max	Unit
Input Capacitance	—	—	7	pF
Output Capacitance	—	—	7	pF

NOTE(S):
All outputs are CMOS drive levels, and can be used with CMOS or TTL logic.
(1) Per PCI Specification Rev 2.1, signals without pullup resistors must have 3 mA low output current. Signals requiring pullup must have 6 mA. The latter include: FRAME*, TRDY*, IRDY*, DEVSEL*, STOP*, SERR*, PERR*, and LOCK*.

16.6 Mechanical Specifications

The CN8237 456-pin BGA package is illustrated in Figure 16-17. Figure 16-18 illustrates a pinout configuration of the CN8237 and pin listings are provided in Figure 16-17.

Figure 16-17. 456-Pin Ball Grid Array Package (BGA)

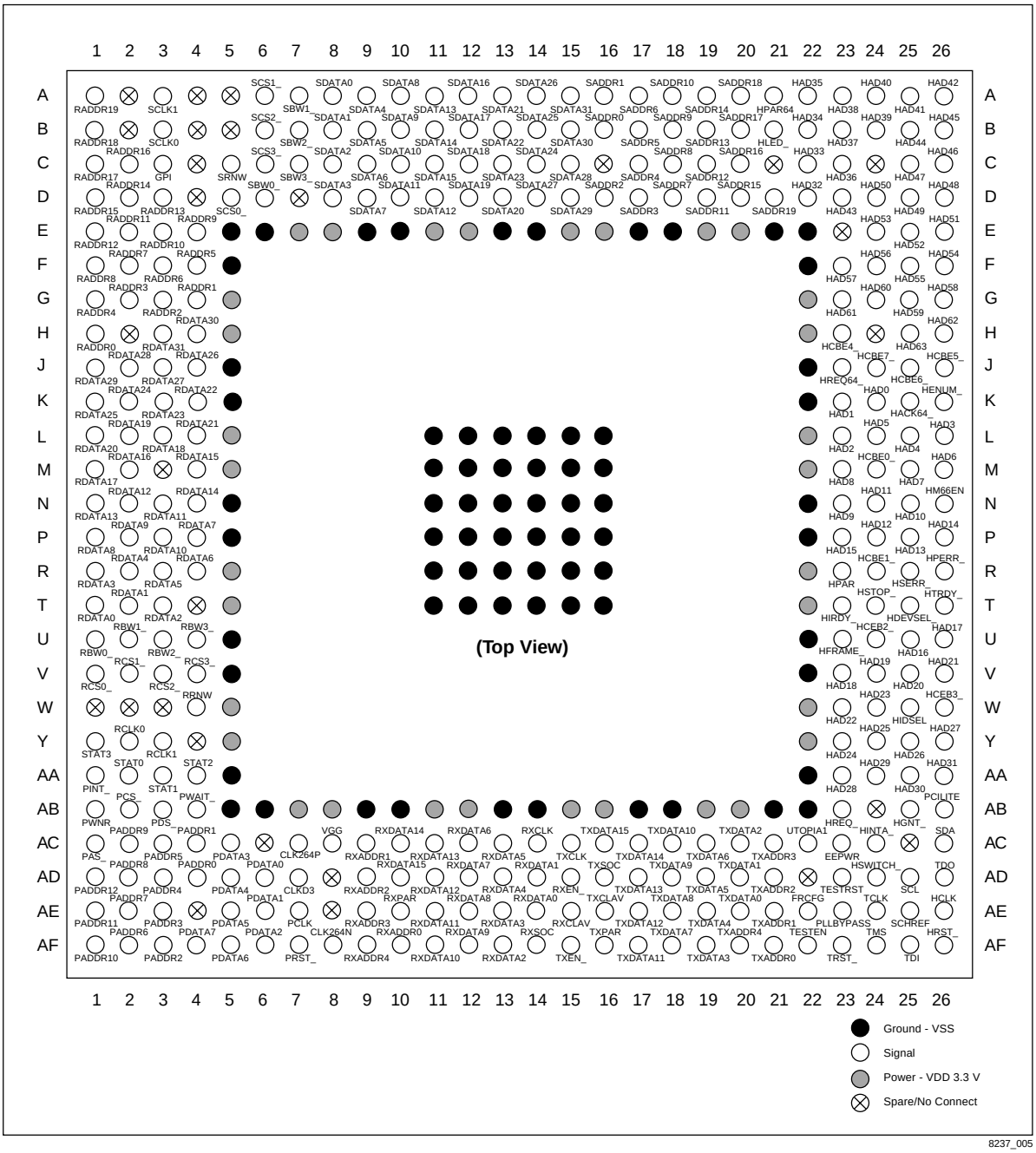


Figure 16-18. CN8237 Package Dimensions

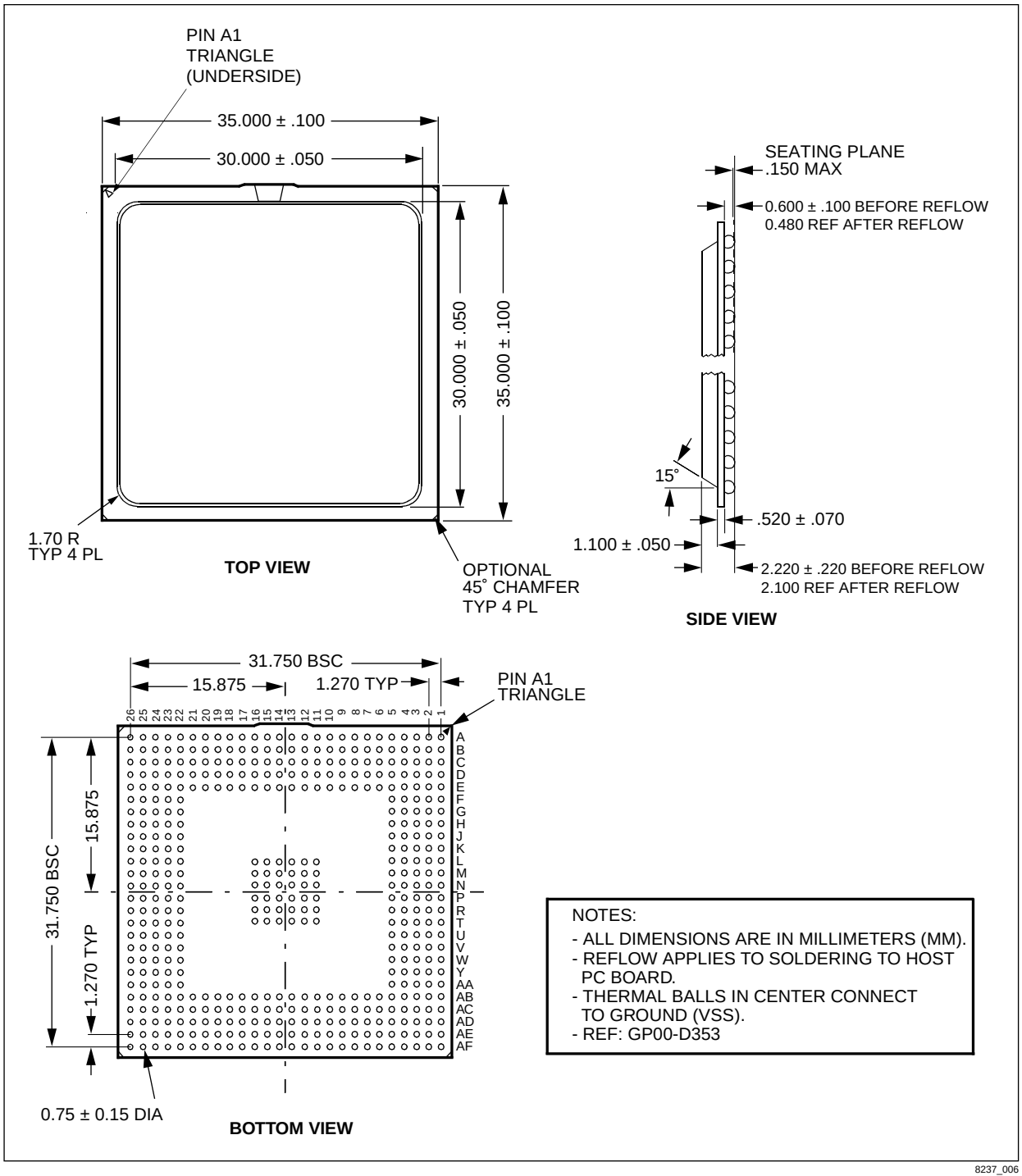


Table 16-12. Listing of Pin Numbers and Labels (Numeric Order) (1 of 5)

Pin Number	Pin Label	I/O(1)
A1	RADDR19	O
A2	SPARE_A2	—
A3	SCLK1	O
A4	SPARE_A4	—
A5	SPARE_A5	—
A6	SCS1*	O
A7	SBW1*	O
A8	SDATA0	I/O
A9	SDATA4	I/O
A10	SDATA8	I/O
A11	SDATA13	I/O
A12	SDATA16	I/O
A13	SDATA21	I/O
A14	SDATA26	I/O
A15	SDATA31	I/O
A16	SADDR1	O
A17	SADDR6	O
A18	SADDR10	O
A19	SADDR14	O
A20	SADDR18	O
A21	HPAR64	I/O
A22	HAD35	I/O
A23	HAD38	I/O
A24	HAD40	I/O
A25	HAD41	I/O
A26	HAD42	I/O
B1	RADDR18	O
B2	SPARE_B2	—
B3	SCLK0	O
B4	SPARE_B4	—
B5	SPARE_B5	—
B6	SCS2*	O
B7	SBW2*	O
B8	SDATA1	I/O
B9	SDATA5	I/O
B10	SDATA9	I/O
B11	SDATA14	I/O
B12	SDATA17	I/O
B13	SDATA22	I/O
B14	SDATA25	I/O
B15	SDATA30	I/O
B16	SADDR0	O
B17	SADDR5	O
B18	SADDR9	O
B19	SADDR13	O
B20	SADDR17	O
B21	HLED*	OD
B22	HAD34	I/O
B23	HAD37	I/O
B24	HAD39	I/O
B25	HAD44	I/O
B26	HAD45	I/O
C1	RADDR17	O
C2	RADDR16	O
C3	GPI	I
C4	SPARE_C4	—
C5	SRNW	O
C6	SCS3*	O
C7	SBW3*	O
C8	SDATA2	I/O
C9	SDATA6	I/O
C10	SDATA10	I/O
C11	SDATA15	I/O
C12	SDATA18	I/O
C13	SDATA23	I/O
C14	SDATA24	I/O
C15	SDATA28	I/O
C16	SPARE_C16	—
C17	SADDR4	O
C18	SADDR8	O
C19	SADDR12	O
C20	SADDR16	O
C21	SPARE_C21(2)	—
C22	HAD33	I/O
C23	HAD36	I/O
C24	SPARE_C24	—
C25	HAD47	I/O
C26	HAD46	I/O
D1	RADDR15	O
D2	RADDR14	O
D3	RADDR13	O
D4	SPARE_D4(2)	—
D5	SCS0*	O
D6	SBW0*	O
D7	SPARE_D7	—
D8	SDATA3	I/O
D9	SDATA7	I/O
D10	SDATA11	I/O
D11	SDATA12	I/O
D12	SDATA19	I/O
D13	SDATA20	I/O
D14	SDATA27	I/O
D15	SDATA29	I/O
D16	SADDR2	O
D17	SADDR3	O
D18	SADDR7	O
D19	SADDR11	O
D20	SADDR15	O
D21	SADDR19	O
D22	HAD32	I/O
D23	HAD43	I/O
D24	HAD50	I/O
D25	HAD49	I/O
D26	HAD48	I/O
E1	RADDR12	O

Table 16-12. Listing of Pin Numbers and Labels (Numeric Order) (2 of 5)

Pin Number	Pin Label	I/O(1)
E2	RADDR11	O
E3	RADDR10	O
E4	RADDR9	O
E5	VSS	—
E6	VSS	—
E7	VDD	—
E8	VDD	—
E9	VSS	—
E10	VSS	—
E11	VDD	—
E12	VDD	—
E13	VSS	—
E14	VSS	—
E15	VDD	—
E16	VDD	—
E17	VSS	—
E18	VSS	—
E19	VDD	—
E20	VDD	—
E21	VSS	—
E22	VSS	—
E23	SPARE_E23	—
E24	HAD53	I/O
E25	HAD52	I/O
E26	HAD51	I/O
F1	RADDR8	O
F2	RADDR7	O
F3	RADDR6	O
F4	RADDR5	O
F5	VSS	—
F22	VSS	—
F23	HAD57	I/O
F24	HAD56	I/O
F25	HAD55	I/O
F26	HAD54	I/O
G1	RADDR4	O
G2	RADDR3	O
G3	RADDR2	O
G4	RADDR1	O
G5	VDD	—
G22	VDD	—
G23	HAD61	I/O
G24	HAD60	I/O
G25	HAD59	I/O
G26	HAD58	I/O
H1	RADDR0	O
H2	SPARE_H2	—
H3	RDATA31	I/O
H4	RDATA30	I/O
H5	VDD	—
H22	VDD	—
H23	HCBE4*	I/O
H24	SPARE_H24	—
H25	HAD63	I/O
H26	HAD62	I/O
J1	RDATA29	I/O
J2	RDATA28	I/O
J3	RDATA27	I/O
J4	RDATA26	I/O
J5	VSS	—
J22	VSS	—
J23	HREQ64*	I/O
J24	HCBE7*	I/O
J25	HCBE6*	I/O
J26	HCBE5*	I/O
K1	RDATA25	I/O
K2	RDATA24	I/O
K3	RDATA23	I/O
K4	RDATA22	I/O
K5	VSS	—
K22	VSS	—
K23	HAD1	I/O
K24	HAD0	I/O
K25	HACK64*	I/O
K26	HENUM*	OD
L1	RDATA20	I/O
L2	RDATA19	I/O
L3	RDATA18	I/O
L4	RDATA21	I/O
L5	VDD	—
L11	VSS	—
L12	VSS	—
L13	VSS	—
L14	VSS	—
L15	VSS	—
L16	VSS	—
L22	VDD	—
L23	HAD2	I/O
L24	HAD5	I/O
L25	HAD4	I/O
L26	HAD3	I/O
M1	RDATA17	I/O
M2	RDATA16	I/O
M3	SPARE_M3	—
M4	RDATA15	I/O
M5	VDD	—
M11	VSS	—
M12	VSS	—
M13	VSS	—
M14	VSS	—
M15	VSS	—
M16	VSS	—
M22	VDD	—
M23	HAD8	I/O
M24	HCBE0*	I/O

Table 16-12. Listing of Pin Numbers and Labels (Numeric Order) (3 of 5)

Pin Number	Pin Label	I/O(1)
M25	HAD7	I/O
M26	HAD6	I/O
N1	RDATA13	I/O
N2	RDATA12	I/O
N3	RDATA11	I/O
N4	RDATA14	I/O
N5	VSS	—
N11	VSS	—
N12	VSS	—
N13	VSS	—
N14	VSS	—
N15	VSS	—
N16	VSS	—
N22	VSS	—
N23	HAD9	I/O
N24	HAD11	I/O
N25	HAD10	I/O
N26	HM66EN	I
P1	RDATA8	I/O
P2	RDATA9	I/O
P3	RDATA10	I/O
P4	RDATA7	I/O
P5	VSS	—
P11	VSS	—
P12	VSS	—
P13	VSS	—
P14	VSS	—
P15	VSS	—
P16	VSS	—
P22	VSS	—
P23	HAD15	I/O
P24	HAD12	I/O
P25	HAD13	I/O
P26	HAD14	I/O
R1	RDATA3	I/O
R2	RDATA4	I/O
R3	RDATA5	I/O
R4	RDATA6	I/O
R5	VDD	—
R11	VSS	—
R12	VSS	—
R13	VSS	—
R14	VSS	—
R15	VSS	—
R16	VSS	—
R22	VDD	—
R23	HPAR	I/O
R24	HCBE1*	I/O
R25	HSERR*	OD
R26	HPERR*	I/O
T1	RDATA0	I/O
T2	RDATA1	I/O
T3	RDATA2	I/O
T4	SPARE_T4	—
T5	VDD	—
T11	VSS	—
T12	VSS	—
T13	VSS	—
T14	VSS	—
T15	VSS	—
T16	VSS	—
T22	VDD	—
T23	HIRDY*	I/O
T24	HSTOP*	I/O
T25	HDEVSEL*	I/O
T26	HTRDY*	I/O
U1	RBW0*	O
U2	RBW1*	O
U3	RBW2*	O
U4	RBW3*	O
U5	VSS	—
U22	VSS	—
U23	HFRAME*	I/O
U24	HCBE2*	I/O
U25	HAD16	I/O
U26	HAD17	I/O
V1	RCS0*	O
V2	RCS1*	O
V3	RCS2*	O
V4	RCS3*	O
V5	VSS	—
V22	VSS	—
V23	HAD18	I/O
V24	HAD19	I/O
V25	HAD20	I/O
V26	HAD21	I/O
W1	SPARE_W1	—
W2	SPARE_W2	—
W3	SPARE_W3	—
W4	RRNW	O
W5	VDD	—
W22	VDD	—
W23	HAD22	I/O
W24	HAD23	I/O
W25	HIDSEL	I
W26	HCBE3*	I/O
Y1	STAT3	O
Y2	RCLK0	O
Y3	RCLK1	O
Y4	SPARE_Y4	—
Y5	VDD	—
Y22	VDD	—
Y23	HAD24	I/O
Y24	HAD25	I/O
Y25	HAD26	I/O

Table 16-12. Listing of Pin Numbers and Labels (Numeric Order) (4 of 5)

Pin Number	Pin Label	I/O(1)
Y26	HAD27	I/O
AA1	PINT*	I
AA2	STAT0	O
AA3	STAT1	O
AA4	STAT2	O
AA5	VSS	—
AA22	VSS	—
AA23	HAD28	I/O
AA24	HAD29	I/O
AA25	HAD30	I/O
AA26	HAD31	I/O
AB1	PWNR	O
AB2	PCS*	O
AB3	PDS*	O
AB4	PWAIT*	I
AB5	VSS	—
AB6	VSS	—
AB7	VDD	—
AB8	VDD	—
AB9	VSS	—
AB10	VSS	—
AB11	VDD	—
AB12	VDD	—
AB13	VSS	—
AB14	VSS	—
AB15	VDD	—
AB16	VDD	—
AB17	VSS	—
AB18	VSS	—
AB19	VDD	—
AB20	VDD	—
AB21	VSS	—
AB22	VSS	—
AB23	HREQ*	I/O
AB24	SPARE_AB24	—
AB25	HGNT*	I
AB26	PCILITE	I
AC1	PAS*	O
AC2	PADDR9	O
AC3	PADDR5	O
AC4	PADDR1	O
AC5	PDATA3	I/O
AC6	SPARE_AC6	—
AC7	CLK264P	I
AC8	VGG	I
AC9	RxADDR1	I/O
AC10	RxDATA14	I
AC11	RxDATA13	I
AC12	RxDATA6	I
AC13	RxDATA5	I
AC14	RxCLK	I
AC15	TxCLK	I
AC16	TxDATA15	O
AC17	TxDATA14	O
AC18	TxDATA10	O
AC19	TxDATA6	O
AC20	TxDATA2	O
AC21	TxADDR3	I/O
AC22	UTOPIA1	I
AC23	EEPWR	O
AC24	HINTA*	OD
AC25	SPARE_AC25	—
AC26	SDA	I/O
AD1	PADDR12	O
AD2	PADDR8	O
AD3	PADDR4	O
AD4	PADDR0	O
AD5	PDATA4	I/O
AD6	PDATA0	I/O
AD7	CLKD3	O
AD8	SPARE_AD8	—
AD9	RxADDR2	I/O
AD10	RxDATA15	I
AD11	RxDATA12	I
AD12	RxDATA7	I
AD13	RxDATA4	I
AD14	RxDATA1	I
AD15	RxEN*	I/O
AD16	TxSOC (TxMARK)	O
AD17	TxDATA13	O
AD18	TxDATA9	O
AD19	TxDATA5	O
AD20	TxDATA1	O
AD21	TxADDR2	I/O
AD22	SPARE_AD22	—
AD23	TESTRST	I
AD24	HSWITCH*	I
AD25	SCL	O
AD26	TDO	O
AE1	PADDR11	O
AE2	PADDR7	O
AE3	PADDR3	O
AE4	SPARE_AE4	—
AE5	PDATA5	I/O
AE6	PDATA1	I/O
AE7	PCLK	O
AE8	SPARE_AE8	—
AE9	RxADDR3	I/O
AE10	RxPAR	I
AE11	RxDATA11	I
AE12	RxDATA8	I
AE13	RxDATA3	I
AE14	RxDATA0	I
AE15	RxCLAV (RxFLAG*)	I/O

Table 16-12. Listing of Pin Numbers and Labels (Numeric Order) (5 of 5)

Pin Number	Pin Label	I/O(1)
AE16	TxCLAV (TxFLAG*)	I/O
AE17	TxDATA12	O
AE18	TxDATA8	O
AE19	TxDATA4	O
AE20	TxDATA0	O
AE21	TxADDR1	I/O
AE22	FRCFG	I
AE23	PLLBYPASS	I
AE24	TCLK	I
AE25	SCHREF	I
AE26	HCLK	I
AF1	PADDR10	O
AF2	PADDR6	O
AF3	PADDR2	O
AF4	PDATA7	I/O
AF5	PDATA6	I/O
AF6	PDATA2	I/O
AF7	PRST*	O
AF8	CLK264N	—
AF9	RxADDR4	I/O
AF10	RxADDR0	I/O
AF11	RxDATA10	I
AF12	RxDATA9	I
AF13	RxDATA2	I
AF14	RxSOC (RxMARK)	I
AF15	TxEN*	I/O
AF16	TxPAR	O
AF17	TxDATA11	O
AF18	TxDATA7	O
AF19	TxDATA3	O
AF20	TxADDR4	I/O
AF21	TxADDR0	I/O
AF22	TESTEN	I
AF23	TRST*	I
AF24	TMS	I
AF25	TDI	I
AF26	HRST*	I

NOTE(S):

(1)Key:

I = Input
O = Output
OD = Open Drain Output
The symbol (*) indicates Active Low.

(2)Spare pins reserved for future local memory expansion:

C21 - SADDR20
D4 - RADDR20

Table 16-13. Listing of Pin Numbers and Labels (Alphabetic Order) (1 of 5)

Pin Number	Pin Label	I/O(1)
AF8	CLK264N	—
AC7	CLK264P	I
AD7	CLKD3	O
AC23	EEPWR	O
AE22	FRCFG	I
C3	GPI	I
K25	HACK64*	I/O
K24	HAD0	I/O
K23	HAD1	I/O
N25	HAD10	I/O
N24	HAD11	I/O
P24	HAD12	I/O
P25	HAD13	I/O
P26	HAD14	I/O
P23	HAD15	I/O
U25	HAD16	I/O
U26	HAD17	I/O
V23	HAD18	I/O
V24	HAD19	I/O
L23	HAD2	I/O
V25	HAD20	I/O
V26	HAD21	I/O
W23	HAD22	I/O
W24	HAD23	I/O
Y23	HAD24	I/O
Y24	HAD25	I/O
Y25	HAD26	I/O
Y26	HAD27	I/O
AA23	HAD28	I/O
AA24	HAD29	I/O
L26	HAD3	I/O
AA25	HAD30	I/O
AA26	HAD31	I/O
D22	HAD32	I/O
C22	HAD33	I/O
B22	HAD34	I/O
A22	HAD35	I/O
C23	HAD36	I/O
B23	HAD37	I/O
A23	HAD38	I/O
B24	HAD39	I/O
L25	HAD4	I/O
A24	HAD40	I/O
A25	HAD41	I/O
A26	HAD42	I/O
D23	HAD43	I/O
B25	HAD44	I/O
B26	HAD45	I/O
C26	HAD46	I/O
C25	HAD47	I/O
D26	HAD48	I/O
D25	HAD49	I/O
L24	HAD5	I/O
D24	HAD50	I/O
E26	HAD51	I/O
E25	HAD52	I/O
E24	HAD53	I/O
F26	HAD54	I/O
F25	HAD55	I/O
F24	HAD56	I/O
F23	HAD57	I/O
G26	HAD58	I/O
G25	HAD59	I/O
M26	HAD6	I/O
G24	HAD60	I/O
G23	HAD61	I/O
H26	HAD62	I/O
H25	HAD63	I/O
M25	HAD7	I/O
M23	HAD8	I/O
N23	HAD9	I/O
M24	HCBE0*	I/O
R24	HCBE1*	I/O
U24	HCBE2*	I/O
W26	HCBE3*	I/O
H23	HCBE4*	I/O
J26	HCBE5*	I/O
J25	HCBE6*	I/O
J24	HCBE7*	I/O
AE26	HCLK	I
T25	HDEVSEL*	I/O
K26	HENUM*	OD
U23	HFRAME*	I/O
AB25	HGNT*	I
W25	HIDSEL	I
AC24	HINTA*	OD
T23	HIRDY*	I/O
B21	HLED*	OD
N26	HM66EN	I
R23	HPAR	I/O
A21	HPAR64	I/O
R26	HPERR*	I/O
AB23	HREQ*	I/O
J23	HREQ64*	I/O
AF26	HRST*	I
R25	HSERR*	OD
T24	HSTOP*	I/O
AD24	HSWITCH*	I
T26	HTRDY*	I/O
AD4	PADDR0	O
AC4	PADDR1	O
AF1	PADDR10	O
AE1	PADDR11	O
AD1	PADDR12	O
AF3	PADDR2	O

Table 16-13. Listing of Pin Numbers and Labels (Alphabetic Order) (2 of 5)

Pin Number	Pin Label	I/O(1)
AE3	PADDR3	O
AD3	PADDR4	O
AC3	PADDR5	O
AF2	PADDR6	O
AE2	PADDR7	O
AD2	PADDR8	O
AC2	PADDR9	O
AC1	PAS*	O
AB26	PCILITE	I
AE7	PCLK	O
AB2	PCS*	O
AD6	PDATA0	I/O
AE6	PDATA1	I/O
AF6	PDATA2	I/O
AC5	PDATA3	I/O
AD5	PDATA4	I/O
AE5	PDATA5	I/O
AF5	PDATA6	I/O
AF4	PDATA7	I/O
AB3	PDS*	O
AA1	PINT*	I
AE8	SPARE_AE8	—
AD8	SPARE_AD8	—
AE23	PLLBYPASS	I
AF7	PRST*	O
AB4	PWAIT*	I
AB1	PWNR	O
H1	RADDR0	O
G4	RADDR1	O
E3	RADDR10	O
E2	RADDR11	O
E1	RADDR12	O
D3	RADDR13	O
D2	RADDR14	O
D1	RADDR15	O

Pin Number	Pin Label	I/O(1)
C2	RADDR16	O
C1	RADDR17	O
B1	RADDR18	O
A1	RADDR19	O
G3	RADDR2	O
G2	RADDR3	O
G1	RADDR4	O
F4	RADDR5	O
F3	RADDR6	O
F2	RADDR7	O
F1	RADDR8	O
E4	RADDR9	O
U1	RBW0*	O
U2	RBW1*	O
U3	RBW2*	O
U4	RBW3*	O
Y2	RCLK0	O
Y3	RCLK1	O
V1	RCS0*	O
V2	RCS1*	O
V3	RCS2*	O
V4	RCS3*	O
T1	RDATA0	I/O
T2	RDATA1	I/O
P3	RDATA10	I/O
N3	RDATA11	I/O
N2	RDATA12	I/O
N1	RDATA13	I/O
N4	RDATA14	I/O
M4	RDATA15	I/O
M2	RDATA16	I/O
M1	RDATA17	I/O
L3	RDATA18	I/O
L2	RDATA19	I/O
T3	RDATA2	I/O

Pin Number	Pin Label	I/O(1)
L1	RDATA20	I/O
L4	RDATA21	I/O
K4	RDATA22	I/O
K3	RDATA23	I/O
K2	RDATA24	I/O
K1	RDATA25	I/O
J4	RDATA26	I/O
J3	RDATA27	I/O
J2	RDATA28	I/O
J1	RDATA29	I/O
R1	RDATA3	I/O
H4	RDATA30	I/O
H3	RDATA31	I/O
R2	RDATA4	I/O
R3	RDATA5	I/O
R4	RDATA6	I/O
P4	RDATA7	I/O
P1	RDATA8	I/O
P2	RDATA9	I/O
W4	RRNW	O
AF10	RxADDR0	I/O
AC9	RxADDR1	I/O
AD9	RxADDR2	I/O
AE9	RxADDR3	I/O
AF9	RxADDR4	I/O
AE15	RxCLAV (RxFLAG*)	I/O
AC14	RxCLK	I
AE14	RxDATA0	I
AD14	RxDATA1	I
AF11	RxDATA10	I
AE11	RxDATA11	I
AD11	RxDATA12	I
AC11	RxDATA13	I
AC10	RxDATA14	I

Table 16-13. Listing of Pin Numbers and Labels (Alphabetic Order) (3 of 5)

Pin Number	Pin Label	I/O(1)
AD10	RxDATA15	I
AF13	RxDATA2	I
AE13	RxDATA3	I
AD13	RxDATA4	I
AC13	RxDATA5	I
AC12	RxDATA6	I
AD12	RxDATA7	I
AE12	RxDATA8	I
AF12	RxDATA9	I
AD15	RxEN*	I/O
AE10	RxPAR	I
AF14	RxSOC (RxMARK)	I
B16	SADDR0	O
A16	SADDR1	O
A18	SADDR10	O
D19	SADDR11	O
C19	SADDR12	O
B19	SADDR13	O
A19	SADDR14	O
D20	SADDR15	O
C20	SADDR16	O
B20	SADDR17	O
A20	SADDR18	O
D21	SADDR19	O
D16	SADDR2	O
D17	SADDR3	O
C17	SADDR4	O
B17	SADDR5	O
A17	SADDR6	O
D18	SADDR7	O
C18	SADDR8	O
B18	SADDR9	O
D6	SBW0*	O
A7	SBW1*	O
B7	SBW2*	O
C7	SBW3*	O
AE25	SCHREF	I
AD25	SCL	O
B3	SCLK0	O
A3	SCLK1	O
D5	SCS0*	O
A6	SCS1*	O
B6	SCS2*	O
C6	SCS3*	O
AC26	SDA	I/O
A8	SDATA0	I/O
B8	SDATA1	I/O
C10	SDATA10	I/O
D10	SDATA11	I/O
D11	SDATA12	I/O
A11	SDATA13	I/O
B11	SDATA14	I/O
C11	SDATA15	I/O
A12	SDATA16	I/O
B12	SDATA17	I/O
C12	SDATA18	I/O
D12	SDATA19	I/O
C8	SDATA2	I/O
D13	SDATA20	I/O
A13	SDATA21	I/O
B13	SDATA22	I/O
C13	SDATA23	I/O
C14	SDATA24	I/O
B14	SDATA25	I/O
A14	SDATA26	I/O
D14	SDATA27	I/O
C15	SDATA28	I/O
D15	SDATA29	I/O
D8	SDATA3	I/O
B15	SDATA30	I/O
A15	SDATA31	I/O
A9	SDATA4	I/O
B9	SDATA5	I/O
C9	SDATA6	I/O
D9	SDATA7	I/O
A10	SDATA8	I/O
B10	SDATA9	I/O
A2	SPARE_A2	—
A4	SPARE_A4	—
A5	SPARE_A5	—
AB24	SPARE_AB24	—
AC25	SPARE_AC25	—
AC6	SPARE_AC6	—
AD22	SPARE_AD22	—
AE4	SPARE_AE4	—
B2	SPARE_B2	—
B4	SPARE_B4	—
B5	SPARE_B5	—
C16	SPARE_C16	—
C21	SPARE_C21(2)	—
C24	SPARE_C24	—
C4	SPARE_C4	—
D4	SPARE_D4(2)	—
D7	SPARE_D7	—
E23	SPARE_E23	—
H2	SPARE_H2	—
H24	SPARE_H24	—
M3	SPARE_M3	—
T4	SPARE_T4	—
W1	SPARE_W1	—
W2	SPARE_W2	—
W3	SPARE_W3	—
Y4	SPARE_Y4	—
C5	SRNW	O

Table 16-13. Listing of Pin Numbers and Labels (Alphabetic Order) (4 of 5)

Pin Number	Pin Label	I/O(1)
AA2	STAT0	O
AA3	STAT1	O
AA4	STAT2	O
Y1	STAT3	O
AE24	TCLK	I
AF25	TDI	I
AD26	TDO	O
AF22	TESTEN	I
AD23	TESTRST	I
AF24	TMS	I
AF23	TRST*	I
AF21	TxADDR0	I/O
AE21	TxADDR1	I/O
AD21	TxADDR2	I/O
AC21	TxADDR3	I/O
AF20	TxADDR4	I/O
AE16	TxCLAV (TxFLAG*)	I/O
AC15	TxCLK	I
AE20	TxDATA0	O
AD20	TxDATA1	O
AC18	TxDATA10	O
AF17	TxDATA11	O
AE17	TxDATA12	O
AD17	TxDATA13	O
AC17	TxDATA14	O
AC16	TxDATA15	O
AC20	TxDATA2	O
AF19	TxDATA3	O
AE19	TxDATA4	O
AD19	TxDATA5	O
AC19	TxDATA6	O
AF18	TxDATA7	O
AE18	TxDATA8	O
AD18	TxDATA9	O
AF15	TxEN*	I/O
AF16	TxPAR	O
AD16	TxSOC (TxMARK)	O
AC22	UTOPIA1	I
E7	VDD	—
E8	VDD	—
E11	VDD	—
E12	VDD	—
E15	VDD	—
E16	VDD	—
E19	VDD	—
E20	VDD	—
G5	VDD	—
G22	VDD	—
H5	VDD	—
H22	VDD	—
L5	VDD	—
L22	VDD	—
M5	VDD	—
M22	VDD	—
R5	VDD	—
R22	VDD	—
T5	VDD	—
T22	VDD	—
W5	VDD	—
W22	VDD	—
Y5	VDD	—
Y22	VDD	—
AB7	VDD	—
AB8	VDD	—
AB11	VDD	—
AB12	VDD	—
AB15	VDD	—
AB16	VDD	—
AB19	VDD	—
AB20	VDD	—
AC8	VGG	I
E5	VSS	—
E6	VSS	—
E9	VSS	—
E10	VSS	—
E13	VSS	—
E14	VSS	—
E17	VSS	—
E18	VSS	—
E21	VSS	—
E22	VSS	—
F5	VSS	—
F22	VSS	—
J5	VSS	—
J22	VSS	—
K5	VSS	—
K22	VSS	—
L11	VSS	—
L12	VSS	—
L13	VSS	—
L14	VSS	—
L15	VSS	—
L16	VSS	—
M11	VSS	—
M12	VSS	—
M13	VSS	—
M14	VSS	—
M15	VSS	—
M16	VSS	—
N5	VSS	—
N11	VSS	—
N12	VSS	—
N13	VSS	—

Table 16-13. Listing of Pin Numbers and Labels (Alphabetic Order) (5 of 5)

Pin Number	Pin Label	I/O(1)
N14	VSS	—
N15	VSS	—
N16	VSS	—
N22	VSS	—
P5	VSS	—
P11	VSS	—
P12	VSS	—
P13	VSS	—
P14	VSS	—
P15	VSS	—
P16	VSS	—
P22	VSS	—
R11	VSS	—
R12	VSS	—
R13	VSS	—
R14	VSS	—
R15	VSS	—
R16	VSS	—
T11	VSS	—
T12	VSS	—
T13	VSS	—
T14	VSS	—
T15	VSS	—
T16	VSS	—
U5	VSS	—
U22	VSS	—
V5	VSS	—
V22	VSS	—
AA5	VSS	—
AA22	VSS	—
AB5	VSS	—
AB6	VSS	—
AB9	VSS	—
AB10	VSS	—
AB13	VSS	—
AB14	VSS	—
AB17	VSS	—
AB18	VSS	—
AB21	VSS	—
AB22	VSS	—

NOTE(S):

(1) Key:
I = Input
O = Output
OD = Open Drain Output
The symbol (*) indicates Active Low.

(2) Spare pins reserved for future local memory expansion:
C21 - SADDR20
D4 - RADDR20

Appendix A : Boundary Scan

The CN8237 supports boundary scan testing conforming to *IEEE Standard 1149.1-1990* and *Supplement B, 1994*. This appendix is intended to assist the customer in developing boundary scan tests for printed circuit boards and systems that use the CN8237. It is assumed that the reader is familiar with boundary scan terminology.

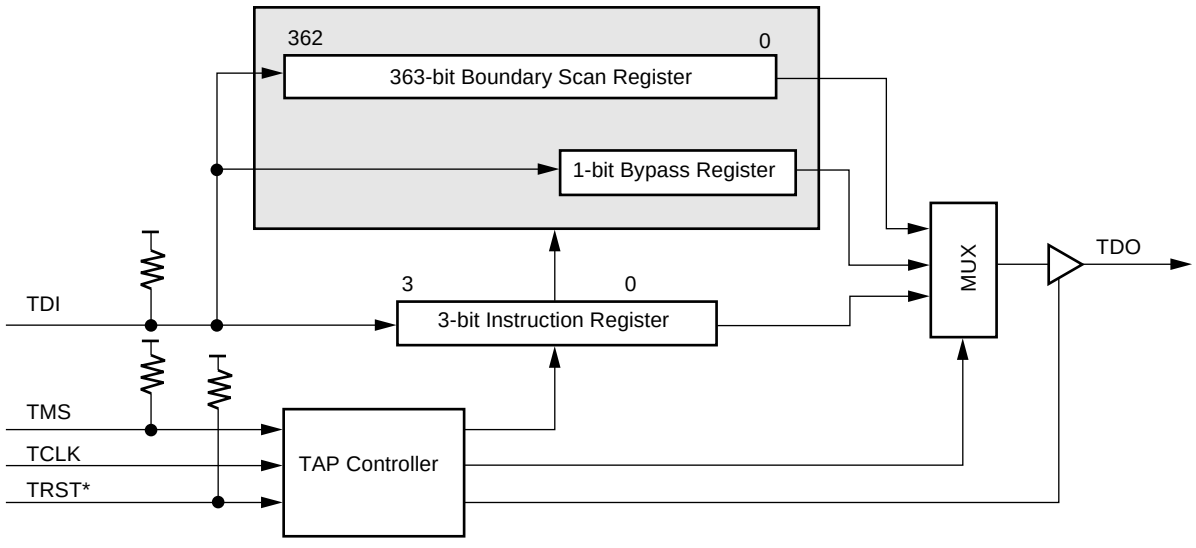
The Boundary Scan section of the CN8237 provides access to all external I/O signals of the device for board and system-level testing. This circuitry also conforms to *IEEE Std 1149.1-1990*. The boundary scan test logic is accessed through five dedicated pins on the CN8237 (see Table A-1).

Table A-1. Boundary Scan Signals

Pin Name	Signal Name	I/O	Definition
TRST*	Test Logic Reset	I	When at a logic low, this signal asynchronously resets the boundary scan test circuitry and puts the test controller into the reset state. This state allows normal system operation.
TCLK	Test Clock	I	Test clocking is generated externally by the system board or by the tester. TCLK can be stopped in either the high state or the low state.
TMS	Test Mode Select	I	Decoded to control test operations.
TDO	Serial Test Data Output	O	Outputs serial test pattern data.
TDI	Serial Test Data Input	I	Input for serial test pattern data.

The test circuitry includes the Boundary Scan register, a BYPASS register, an Instruction register (IR), and the Tap Access Port (TAP) controller (see Figure A-1).

Figure A-1. Test Circuitry Block Diagram



NOTE(S):
If the boundary scan circuitry of the CN8237 is not being used, tie TCLK and TRST* to ground.

8237_112

A.1 Instruction Register

The Instruction register is a 3-bit register with no parity. When the boundary scan circuitry is reset, the IR is loaded with the binary value b111, which is equivalent to the BYPASS Instruction value. The Capture-IR binary value is b001 and is shifted out TDO as the IR is loaded.

The 16 instructions include three *IEEE 1149.1* mandatory public instructions (BYPASS, EXTEST, and SAMPLE/PRELOAD) and five private instructions for manufacturing use only. Bit-0 (LSB) is shifted into the instruction register first. Table A-2 shows the bit settings for this register.

NOTE: The implementation of this register as described is applicable only to Rev. A of the CN8237 device.

Table A-2. IEEE Std. 1149.1 Instructions

Bit 2	Bit 1	Bit 0	Instruction	Register Accessed
0	0	0	EXTEST	Boundary Scan
0	0	1	SAMPLE/PRELOAD	Boundary Scan
0	1	0	IDCODE–Device ID	Device ID
0	1	1	RSTHIGH–Private	—
1	0	0	HIGH2–Private	—
1	0	1	RUNBIST–Private	—
1	1	0	RSVD1–Private	—
1	1	1	BYPASS	Bypass

A.2 BYPASS Register

The BYPASS register is a 1-bit shift register used to pass TDI data to TDO to facilitate the testing of other devices in the scan path without having to shift the data patterns through the complete Boundary Scan register of the CN8237.

A.3 Boundary Scan Register

The Boundary Scan register consists of two different types of registers corresponding to *IEEE Std. 1149.1a-1993* attributes and definitions. These register names are *STD_1149_1_1993* Standard Boundary Cell names BC-1 and BC-7.

A.4 Electrical Characteristics

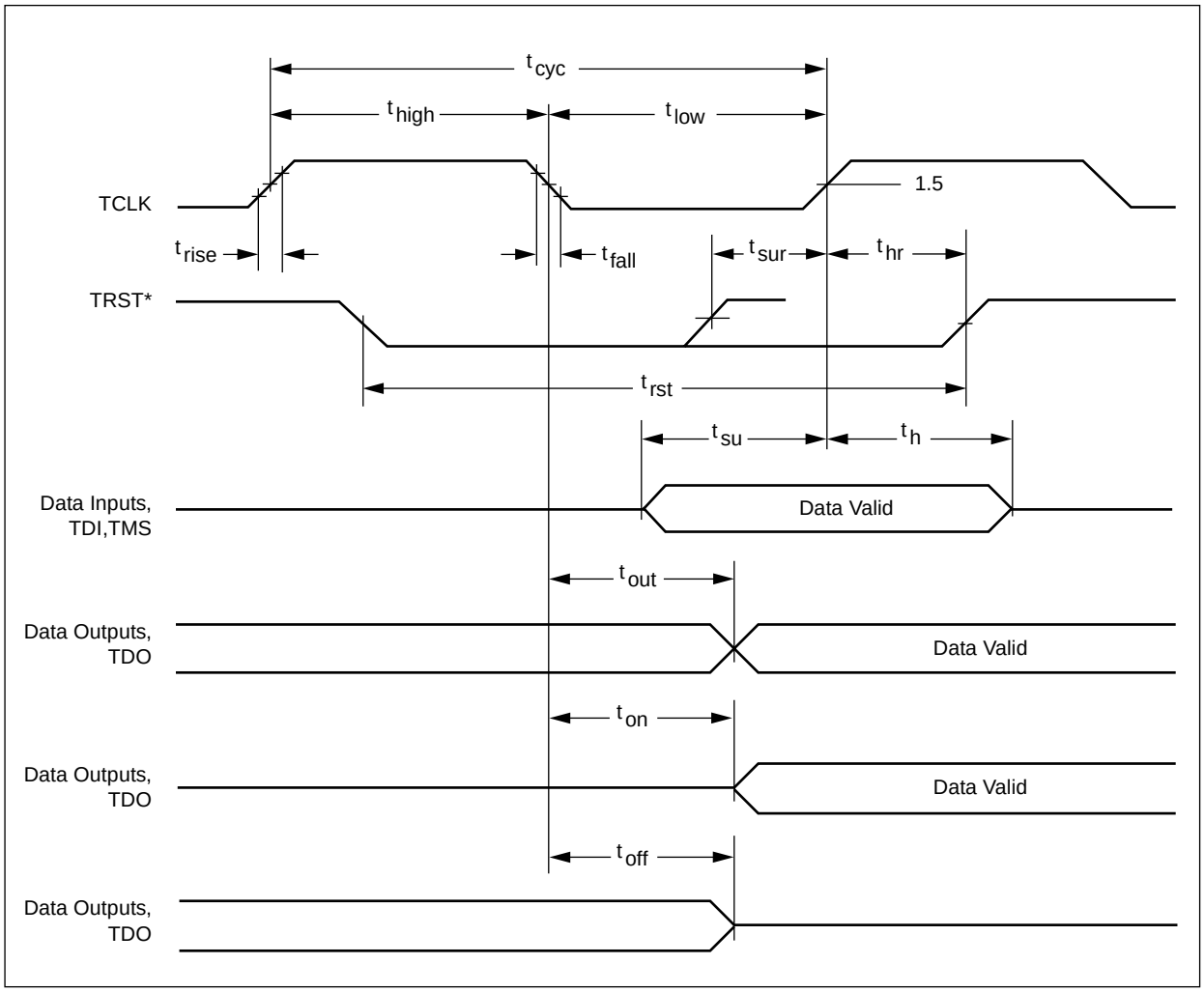
This section describes the electrical characteristics for boundary scan. Table A-3 provides timing specifications and Figure A-2 shows a timing diagram.

Table A-3. Timing Specifications

Symbol	Description	Min	Max	Unit
—	TCLK Frequency	—	20	MHz
t_{cyc}	TCLK Cycle	50	—	ns
t_{high}	TCLK High Pulse Width	20	30	ns
t_{low}	TCLK Low Pulse Width	20	30	ns
t_{rise}	TCLK Rise	1	4	ns
t_{fall}	TCLK Fall	1	4	ns
t_{sur}	TRST* Inactive to TCLK Rising Edge(1)	5	—	ns
t_{hr}	TRST* Inactive after TCLK Rising Edge(1)	5	—	ns
t_{rst}	TRST* Active	1	—	t_{cyc}
t_{su}	TDI, TMS, and Data Inputs Setup	5	—	ns
t_{h}	TDI, TMS, and Data Inputs Hold	5	—	ns
t_{out}	TDO and Data Outputs Valid	—	10	ns
t_{on}	TDO and Data Outputs Float to Valid	—	10	ns
t_{off}	TDO and Data Outputs Valid to Float	—	10	ns

NOTE(S):
(1) TRST* going inactive timing only required if TMS = 0 at the rising edge of TCLK.

Figure A-2. Timing Diagram



A.5 Boundary Scan Description Language (BSDL) File

To obtain an electronic copy of this file, go to Mindspeed web site at www.mindspeed.com.

```
--------------------------------------------------------------------------
-- Boundary Scan Description Language (IEEE 1149.1b)
-- BSDL for Device CN8237
-- BSDL File Created by 'topgen' Revision 2.1
-- (c) Mindspeed, Inc.
--------------------------------------------------------------------------

entity CN8237 is  generic (PHYSICAL_PIN_MAP : string := "BGA_456");

  port (
    CLK264N                 : linkage  bit;
    CLK264P                 : in       bit;
    CLKD3                   : out      bit;
    EEPWR                   : out      bit;
    FRCFG                   : in       bit;
    GPI                     : in       bit;
    HACK64_NEG              : inout    bit;
    HAD0                    : inout    bit;
    HAD1                    : inout    bit;
    HAD10                   : inout    bit;
    HAD11                   : inout    bit;
    HAD12                   : inout    bit;
    HAD13                   : inout    bit;
    HAD14                   : inout    bit;
    HAD15                   : inout    bit;
    HAD16                   : inout    bit;
    HAD17                   : inout    bit;
    HAD18                   : inout    bit;
    HAD19                   : inout    bit;
    HAD2                    : inout    bit;
    HAD20                   : inout    bit;
    HAD21                   : inout    bit;
    HAD22                   : inout    bit;
    HAD23                   : inout    bit;
    HAD24                   : inout    bit;
    HAD25                   : inout    bit;
    HAD26                   : inout    bit;
    HAD27                   : inout    bit;
    HAD28                   : inout    bit;
    HAD29                   : inout    bit;
    HAD3                    : inout    bit;
    HAD30                   : inout    bit;
    HAD31                   : inout    bit;
    HAD32                   : inout    bit;
    HAD33                   : inout    bit;
    HAD34                   : inout    bit;
```

```
HAD35                   : inout    bit;
HAD36                   : inout    bit;
HAD37                   : inout    bit;
HAD38                   : inout    bit;
HAD39                   : inout    bit;
HAD4                    : inout    bit;
HAD40                   : inout    bit;
HAD41                   : inout    bit;
HAD42                   : inout    bit;
HAD43                   : inout    bit;
HAD44                   : inout    bit;
HAD45                   : inout    bit;
HAD46                   : inout    bit;
HAD47                   : inout    bit;
HAD48                   : inout    bit;
HAD49                   : inout    bit;
HAD5                    : inout    bit;
HAD50                   : inout    bit;
HAD51                   : inout    bit;
HAD52                   : inout    bit;
HAD53                   : inout    bit;
HAD54                   : inout    bit;
HAD55                   : inout    bit;
HAD56                   : inout    bit;
HAD57                   : inout    bit;
HAD58                   : inout    bit;
HAD59                   : inout    bit;
HAD6                    : inout    bit;
HAD60                   : inout    bit;
HAD61                   : inout    bit;
HAD62                   : inout    bit;
HAD63                   : inout    bit;
HAD7                    : inout    bit;
HAD8                    : inout    bit;
HAD9                    : inout    bit;
HCBE0_NEG               : inout    bit;
HCBE1_NEG               : inout    bit;
HCBE2_NEG               : inout    bit;
HCBE3_NEG               : inout    bit;
HCBE4_NEG               : inout    bit;
HCBE5_NEG               : inout    bit;
HCBE6_NEG               : inout    bit;
HCBE7_NEG               : inout    bit;
HCLK                    : in       bit;
HDEVSEL_NEG             : inout    bit;
HENUM_NEG               : inout    bit;
HFRAME_NEG              : inout    bit;
HGNT_NEG                : in       bit;
HIDSEL                  : in       bit;
HINTA_NEG               : inout    bit;
HIRDY_NEG               : inout    bit;
HLED_NEG                : out      bit;
HM66EN                  : in       bit;
HPAR                    : inout    bit;
HPAR64                  : inout    bit;
HPERR_NEG               : inout    bit;
HREQ64_NEG              : inout    bit;
HREQ_NEG                : inout    bit;
HRST_NEG                : in       bit;
```

```
HSERR_NEG               : inout    bit;
HSTOP_NEG               : inout    bit;
HSWITCH_NEG             : in       bit;
HTRDY_NEG               : inout    bit;
PADDR0                  : out      bit;
PADDR1                  : out      bit;
PADDR10                 : out      bit;
PADDR11                 : out      bit;
PADDR12                 : out      bit;
PADDR2                  : out      bit;
PADDR3                  : out      bit;
PADDR4                  : out      bit;
PADDR5                  : out      bit;
PADDR6                  : out      bit;
PADDR7                  : out      bit;
PADDR8                  : out      bit;
PADDR9                  : out      bit;
PAS_NEG                 : out      bit;
PCILITE                 : in       bit;
PCLK                    : out      bit;
PCS_NEG                 : out      bit;
PDATA0                  : inout    bit;
PDATA1                  : inout    bit;
PDATA2                  : inout    bit;
PDATA3                  : inout    bit;
PDATA4                  : inout    bit;
PDATA5                  : inout    bit;
PDATA6                  : inout    bit;
PDATA7                  : inout    bit;
PDS_NEG                 : out      bit;
PINT_NEG                : in       bit;
PLLBYPASS               : in       bit;
PLL_VDD                 : linkage  bit;
PLL_VSS                 : linkage  bit;
PRST_NEG                : out      bit;
PWAIT_NEG               : in       bit;
PWNR                    : out      bit;
RADDR0                  : out      bit;
RADDR1                  : out      bit;
RADDR10                 : out      bit;
RADDR11                 : out      bit;
RADDR12                 : out      bit;
RADDR13                 : out      bit;
RADDR14                 : out      bit;
RADDR15                 : out      bit;
RADDR16                 : out      bit;
RADDR17                 : out      bit;
RADDR18                 : out      bit;
RADDR19                 : out      bit;
RADDR2                  : out      bit;
RADDR3                  : out      bit;
RADDR4                  : out      bit;
RADDR5                  : out      bit;
RADDR6                  : out      bit;
RADDR7                  : out      bit;
RADDR8                  : out      bit;
RADDR9                  : out      bit;
RBW0_NEG                : out      bit;
RBW1_NEG                : out      bit;
```

```
RBW2_NEG                : out      bit;
RBW3_NEG                : out      bit;
RCLK0                   : out      bit;
RCLK1                   : out      bit;
RCS0_NEG                : out      bit;
RCS1_NEG                : out      bit;
RCS2_NEG                : out      bit;
RCS3_NEG                : out      bit;
RDATA0                  : inout    bit;
RDATA1                  : inout    bit;
RDATA10                 : inout    bit;
RDATA11                 : inout    bit;
RDATA12                 : inout    bit;
RDATA13                 : inout    bit;
RDATA14                 : inout    bit;
RDATA15                 : inout    bit;
RDATA16                 : inout    bit;
RDATA17                 : inout    bit;
RDATA18                 : inout    bit;
RDATA19                 : inout    bit;
RDATA2                  : inout    bit;
RDATA20                 : inout    bit;
RDATA21                 : inout    bit;
RDATA22                 : inout    bit;
RDATA23                 : inout    bit;
RDATA24                 : inout    bit;
RDATA25                 : inout    bit;
RDATA26                 : inout    bit;
RDATA27                 : inout    bit;
RDATA28                 : inout    bit;
RDATA29                 : inout    bit;
RDATA3                  : inout    bit;
RDATA30                 : inout    bit;
RDATA31                 : inout    bit;
RDATA4                  : inout    bit;
RDATA5                  : inout    bit;
RDATA6                  : inout    bit;
RDATA7                  : inout    bit;
RDATA8                  : inout    bit;
RDATA9                  : inout    bit;
RRNW                    : out      bit;
RXADDR0                 : inout    bit;
RXADDR1                 : inout    bit;
RXADDR2                 : inout    bit;
RXADDR3                 : inout    bit;
RXADDR4                 : inout    bit;
RXCLAV                  : inout    bit;
RXCLK                   : in       bit;
RXDATA0                 : in       bit;
RXDATA1                 : in       bit;
RXDATA10                : in       bit;
RXDATA11                : in       bit;
RXDATA12                : in       bit;
RXDATA13                : in       bit;
RXDATA14                : in       bit;
RXDATA15                : in       bit;
RXDATA2                 : in       bit;
RXDATA3                 : in       bit;
RXDATA4                 : in       bit;
```

```
RXDATA5                 : in       bit;
RXDATA6                 : in       bit;
RXDATA7                 : in       bit;
RXDATA8                 : in       bit;
RXDATA9                 : in       bit;
RXEN_NEG                : inout    bit;
RXPAR                   : in       bit;
RXSOC                   : in       bit;
SADDR0                  : out      bit;
SADDR1                  : out      bit;
SADDR10                 : out      bit;
SADDR11                 : out      bit;
SADDR12                 : out      bit;
SADDR13                 : out      bit;
SADDR14                 : out      bit;
SADDR15                 : out      bit;
SADDR16                 : out      bit;
SADDR17                 : out      bit;
SADDR18                 : out      bit;
SADDR19                 : out      bit;
SADDR2                  : out      bit;
SADDR3                  : out      bit;
SADDR4                  : out      bit;
SADDR5                  : out      bit;
SADDR6                  : out      bit;
SADDR7                  : out      bit;
SADDR8                  : out      bit;
SADDR9                  : out      bit;
SBW0_NEG                : out      bit;
SBW1_NEG                : out      bit;
SBW2_NEG                : out      bit;
SBW3_NEG                : out      bit;
SCHREF                  : in       bit;
SCL                     : inout    bit;
SCLK0                   : out      bit;
SCLK1                   : out      bit;
SCS0_NEG                : out      bit;
SCS1_NEG                : out      bit;
SCS2_NEG                : out      bit;
SCS3_NEG                : out      bit;
SDA                     : inout    bit;
SDATA0                  : inout    bit;
SDATA1                  : inout    bit;
SDATA10                 : inout    bit;
SDATA11                 : inout    bit;
SDATA12                 : inout    bit;
SDATA13                 : inout    bit;
SDATA14                 : inout    bit;
SDATA15                 : inout    bit;
SDATA16                 : inout    bit;
SDATA17                 : inout    bit;
SDATA18                 : inout    bit;
SDATA19                 : inout    bit;
SDATA2                  : inout    bit;
SDATA20                 : inout    bit;
SDATA21                 : inout    bit;
SDATA22                 : inout    bit;
SDATA23                 : inout    bit;
SDATA24                 : inout    bit;
```

```
SDATA25                 : inout    bit;
SDATA26                 : inout    bit;
SDATA27                 : inout    bit;
SDATA28                 : inout    bit;
SDATA29                 : inout    bit;
SDATA3                  : inout    bit;
SDATA30                 : inout    bit;
SDATA31                 : inout    bit;
SDATA4                  : inout    bit;
SDATA5                  : inout    bit;
SDATA6                  : inout    bit;
SDATA7                  : inout    bit;
SDATA8                  : inout    bit;
SDATA9                  : inout    bit;
SPARE_A2                : linkage  bit;
SPARE_A4                : linkage  bit;
SPARE_A5                : linkage  bit;
SPARE_AB24              : linkage  bit;
SPARE_AC25              : linkage  bit;
SPARE_AC6               : linkage  bit;
SPARE_AD22              : linkage  bit;
SPARE_AE4               : linkage  bit;
SPARE_B2                : linkage  bit;
SPARE_B4                : linkage  bit;
SPARE_B5                : linkage  bit;
SPARE_C16               : linkage  bit;
SPARE_C21               : linkage  bit;
SPARE_C24               : linkage  bit;
SPARE_C4                : linkage  bit;
SPARE_D4                : linkage  bit;
SPARE_D7                : linkage  bit;
SPARE_E23               : linkage  bit;
SPARE_H2                : linkage  bit;
SPARE_H24               : linkage  bit;
SPARE_M3                : linkage  bit;
SPARE_T4                : linkage  bit;
SPARE_W1                : linkage  bit;
SPARE_W2                : linkage  bit;
SPARE_W3                : linkage  bit;
SPARE_Y4                : linkage  bit;
SRNW                    : out      bit;
STAT0                   : out      bit;
STAT1                   : out      bit;
STAT2                   : out      bit;
STAT3                   : out      bit;
TCLK                    : in       bit;
TDI                     : in       bit;
TDO                     : out      bit;
TESTEN                  : in       bit;
TESTRST                 : in       bit;
TMS                     : in       bit;
TRST_NEG                : in       bit;
TXADDR0                 : inout    bit;
TXADDR1                 : inout    bit;
TXADDR2                 : inout    bit;
TXADDR3                 : inout    bit;
TXADDR4                 : inout    bit;
TXCLAV                  : inout    bit;
TXCLK                   : in       bit;
```

```
TXDATA0                 : out      bit;
TXDATA1                 : out      bit;
TXDATA10                : out      bit;
TXDATA11                : out      bit;
TXDATA12                : out      bit;
TXDATA13                : out      bit;
TXDATA14                : out      bit;
TXDATA15                : out      bit;
TXDATA2                 : out      bit;
TXDATA3                 : out      bit;
TXDATA4                 : out      bit;
TXDATA5                 : out      bit;
TXDATA6                 : out      bit;
TXDATA7                 : out      bit;
TXDATA8                 : out      bit;
TXDATA9                 : out      bit;
TXEN_NEG                : inout    bit;
TXPAR                   : out      bit;
TXSOC                   : out      bit;
UTOPIA1                 : in       bit;
VGG                     : linkage  bit;
VDD_AB11                : linkage  bit;
VDD_AB12                : linkage  bit;
VDD_AB15                : linkage  bit;
VDD_AB16                : linkage  bit;
VDD_AB19                : linkage  bit;
VDD_AB20                : linkage  bit;
VDD_AB7                 : linkage  bit;
VDD_AB8                 : linkage  bit;
VDD_E11                 : linkage  bit;
VDD_E12                 : linkage  bit;
VDD_E15                 : linkage  bit;
VDD_E16                 : linkage  bit;
VDD_E19                 : linkage  bit;
VDD_E20                 : linkage  bit;
VDD_E7                  : linkage  bit;
VDD_E8                  : linkage  bit;
VDD_G22                 : linkage  bit;
VDD_G5                  : linkage  bit;
VDD_H22                 : linkage  bit;
VDD_H5                  : linkage  bit;
VDD_L22                 : linkage  bit;
VDD_L5                  : linkage  bit;
VDD_M22                 : linkage  bit;
VDD_M5                  : linkage  bit;
VDD_R22                 : linkage  bit;
VDD_R5                  : linkage  bit;
VDD_T22                 : linkage  bit;
VDD_T5                  : linkage  bit;
VDD_W22                 : linkage  bit;
VDD_W5                  : linkage  bit;
VDD_Y22                 : linkage  bit;
VDD_Y5                  : linkage  bit;
VSS_AA22                : linkage  bit;
VSS_AA5                 : linkage  bit;
VSS_AB10                : linkage  bit;
VSS_AB13                : linkage  bit;
VSS_AB14                : linkage  bit;
VSS_AB17                : linkage  bit;
```

```
VSS_AB18                : linkage  bit;
VSS_AB21                : linkage  bit;
VSS_AB22                : linkage  bit;
VSS_AB5                 : linkage  bit;
VSS_AB6                 : linkage  bit;
VSS_AB9                 : linkage  bit;
VSS_E10                 : linkage  bit;
VSS_E13                 : linkage  bit;
VSS_E14                 : linkage  bit;
VSS_E17                 : linkage  bit;
VSS_E18                 : linkage  bit;
VSS_E21                 : linkage  bit;
VSS_E22                 : linkage  bit;
VSS_E5                  : linkage  bit;
VSS_E6                  : linkage  bit;
VSS_E9                  : linkage  bit;
VSS_F22                 : linkage  bit;
VSS_F5                  : linkage  bit;
VSS_J22                 : linkage  bit;
VSS_J5                  : linkage  bit;
VSS_K22                 : linkage  bit;
VSS_K5                  : linkage  bit;
VSS_L11                 : linkage  bit;
VSS_L12                 : linkage  bit;
VSS_L13                 : linkage  bit;
VSS_L14                 : linkage  bit;
VSS_L15                 : linkage  bit;
VSS_L16                 : linkage  bit;
VSS_M11                 : linkage  bit;
VSS_M12                 : linkage  bit;
VSS_M13                 : linkage  bit;
VSS_M14                 : linkage  bit;
VSS_M15                 : linkage  bit;
VSS_M16                 : linkage  bit;
VSS_N11                 : linkage  bit;
VSS_N12                 : linkage  bit;
VSS_N13                 : linkage  bit;
VSS_N14                 : linkage  bit;
VSS_N15                 : linkage  bit;
VSS_N16                 : linkage  bit;
VSS_N22                 : linkage  bit;
VSS_N5                  : linkage  bit;
VSS_P11                 : linkage  bit;
VSS_P12                 : linkage  bit;
VSS_P13                 : linkage  bit;
VSS_P14                 : linkage  bit;
VSS_P15                 : linkage  bit;
VSS_P16                 : linkage  bit;
VSS_P22                 : linkage  bit;
VSS_P5                  : linkage  bit;
VSS_R11                 : linkage  bit;
VSS_R12                 : linkage  bit;
VSS_R13                 : linkage  bit;
VSS_R14                 : linkage  bit;
VSS_R15                 : linkage  bit;
VSS_R16                 : linkage  bit;
VSS_T11                 : linkage  bit;
VSS_T12                 : linkage  bit;
VSS_T13                 : linkage  bit;
```

```
  VSS_T14                 : linkage  bit;
  VSS_T15                 : linkage  bit;
  VSS_T16                 : linkage  bit;
  VSS_U22                 : linkage  bit;
  VSS_U5                  : linkage  bit;
  VSS_V22                 : linkage  bit;
  VSS_V5                  : linkage  bit
);

use STD_1149_1_1994.all;

attribute COMPONENT_CONFORMANCE of CN8237 : entity is
  "STD_1149_1_1993";

attribute PIN_MAP of CN8237 : entity is PHYSICAL_PIN_MAP;

constant BGA_456: PIN_MAP_STRING :=
  "CLK264N                : AF8," &
  "CLK264P                : AC7," &
  "CLKD3                  : AD7," &
  "EEPWR                  : AC23," &
  "FRCFG                  : AE22," &
  "GPI                    : C3," &
  "HACK64_NEG             : K25," &
  "HAD0                   : K24," &
  "HAD1                   : K23," &
  "HAD10                  : N25," &
  "HAD11                  : N24," &
  "HAD12                  : P24," &
  "HAD13                  : P25," &
  "HAD14                  : P26," &
  "HAD15                  : P23," &
  "HAD16                  : U25," &
  "HAD17                  : U26," &
  "HAD18                  : V23," &
  "HAD19                  : V24," &
  "HAD2                   : L23," &
  "HAD20                  : V25," &
  "HAD21                  : V26," &
  "HAD22                  : W23," &
  "HAD23                  : W24," &
  "HAD24                  : Y23," &
  "HAD25                  : Y24," &
  "HAD26                  : Y25," &
  "HAD27                  : Y26," &
  "HAD28                  : AA23," &
  "HAD29                  : AA24," &
  "HAD3                   : L26," &
  "HAD30                  : AA25," &
  "HAD31                  : AA26," &
  "HAD32                  : D22," &
  "HAD33                  : C22," &
  "HAD34                  : B22," &
  "HAD35                  : A22," &
  "HAD36                  : C23," &
  "HAD37                  : B23," &
  "HAD38                  : A23," &
  "HAD39                  : B24," &
  "HAD4                   : L25," &
```

```
"HAD40                  : A24," &
"HAD41                  : A25," &
"HAD42                  : A26," &
"HAD43                  : D23," &
"HAD44                  : B25," &
"HAD45                  : B26," &
"HAD46                  : C26," &
"HAD47                  : C25," &
"HAD48                  : D26," &
"HAD49                  : D25," &
"HAD5                   : L24," &
"HAD50                  : D24," &
"HAD51                  : E26," &
"HAD52                  : E25," &
"HAD53                  : E24," &
"HAD54                  : F26," &
"HAD55                  : F25," &
"HAD56                  : F24," &
"HAD57                  : F23," &
"HAD58                  : G26," &
"HAD59                  : G25," &
"HAD6                   : M26," &
"HAD60                  : G24," &
"HAD61                  : G23," &
"HAD62                  : H26," &
"HAD63                  : H25," &
"HAD7                   : M25," &
"HAD8                   : M23," &
"HAD9                   : N23," &
"HCBE0_NEG              : M24," &
"HCBE1_NEG              : R24," &
"HCBE2_NEG              : U24," &
"HCBE3_NEG              : W26," &
"HCBE4_NEG              : H23," &
"HCBE5_NEG              : J26," &
"HCBE6_NEG              : J25," &
"HCBE7_NEG              : J24," &
"HCLK                   : AE26," &
"HDEVSEL_NEG            : T25," &
"HENUM_NEG              : K26," &
"HFRAME_NEG             : U23," &
"HGNT_NEG               : AB25," &
"HIDSEL                 : W25," &
"HINTA_NEG              : AC24," &
"HIRDY_NEG              : T23," &
"HLED_NEG               : B21," &
"HM66EN                 : N26," &
"HPAR                   : R23," &
"HPAR64                 : A21," &
"HPERR_NEG              : R26," &
"HREQ64_NEG             : J23," &
"HREQ_NEG               : AB23," &
"HRST_NEG               : AF26," &
"HSERR_NEG              : R25," &
"HSTOP_NEG              : T24," &
"HSWITCH_NEG            : AD24," &
"HTRDY_NEG              : T26," &
"PADDR0                 : AD4," &
"PADDR1                 : AC4," &
```

```
"PADDR10                : AF1," &
"PADDR11                : AE1," &
"PADDR12                : AD1," &
"PADDR2                 : AF3," &
"PADDR3                 : AE3," &
"PADDR4                 : AD3," &
"PADDR5                 : AC3," &
"PADDR6                 : AF2," &
"PADDR7                 : AE2," &
"PADDR8                 : AD2," &
"PADDR9                 : AC2," &
"PAS_NEG                : AC1," &
"PCILITE                : AB26," &
"PCLK                   : AE7," &
"PCS_NEG                : AB2," &
"PDATA0                 : AD6," &
"PDATA1                 : AE6," &
"PDATA2                 : AF6," &
"PDATA3                 : AC5," &
"PDATA4                 : AD5," &
"PDATA5                 : AE5," &
"PDATA6                 : AF5," &
"PDATA7                 : AF4," &
"PDS_NEG                : AB3," &
"PINT_NEG               : AA1," &
"PLLBYPASS              : AE23," &
"PLL_VDD                : AE8," &
"PLL_VSS                : AD8," &
"PRST_NEG               : AF7," &
"PWAIT_NEG              : AB4," &
"PWNR                   : AB1," &
"RADDR0                 : H1," &
"RADDR1                 : G4," &
"RADDR10                : E3," &
"RADDR11                : E2," &
"RADDR12                : E1," &
"RADDR13                : D3," &
"RADDR14                : D2," &
"RADDR15                : D1," &
"RADDR16                : C2," &
"RADDR17                : C1," &
"RADDR18                : B1," &
"RADDR19                : A1," &
"RADDR2                 : G3," &
"RADDR3                 : G2," &
"RADDR4                 : G1," &
"RADDR5                 : F4," &
"RADDR6                 : F3," &
"RADDR7                 : F2," &
"RADDR8                 : F1," &
"RADDR9                 : E4," &
"RBW0_NEG               : U1," &
"RBW1_NEG               : U2," &
"RBW2_NEG               : U3," &
"RBW3_NEG               : U4," &
"RCLK0                  : Y2," &
"RCLK1                  : Y3," &
"RCS0_NEG               : V1," &
"RCS1_NEG               : V2," &
```

```
"RCS2_NEG               : V3," &
"RCS3_NEG               : V4," &
"RDATA0                 : T1," &
"RDATA1                 : T2," &
"RDATA10                : P3," &
"RDATA11                : N3," &
"RDATA12                : N2," &
"RDATA13                : N1," &
"RDATA14                : N4," &
"RDATA15                : M4," &
"RDATA16                : M2," &
"RDATA17                : M1," &
"RDATA18                : L3," &
"RDATA19                : L2," &
"RDATA2                 : T3," &
"RDATA20                : L1," &
"RDATA21                : L4," &
"RDATA22                : K4," &
"RDATA23                : K3," &
"RDATA24                : K2," &
"RDATA25                : K1," &
"RDATA26                : J4," &
"RDATA27                : J3," &
"RDATA28                : J2," &
"RDATA29                : J1," &
"RDATA3                 : R1," &
"RDATA30                : H4," &
"RDATA31                : H3," &
"RDATA4                 : R2," &
"RDATA5                 : R3," &
"RDATA6                 : R4," &
"RDATA7                 : P4," &
"RDATA8                 : P1," &
"RDATA9                 : P2," &
"RRNW                   : W4," &
"RXADDR0                : AF10," &
"RXADDR1                : AC9," &
"RXADDR2                : AD9," &
"RXADDR3                : AE9," &
"RXADDR4                : AF9," &
"RXCLAV                 : AE15," &
"RXCLK                  : AC14," &
"RXDATA0                : AE14," &
"RXDATA1                : AD14," &
"RXDATA10               : AF11," &
"RXDATA11               : AE11," &
"RXDATA12               : AD11," &
"RXDATA13               : AC11," &
"RXDATA14               : AC10," &
"RXDATA15               : AD10," &
"RXDATA2                : AF13," &
"RXDATA3                : AE13," &
"RXDATA4                : AD13," &
"RXDATA5                : AC13," &
"RXDATA6                : AC12," &
"RXDATA7                : AD12," &
"RXDATA8                : AE12," &
"RXDATA9                : AF12," &
"RXEN_NEG               : AD15," &
```

```
"RXPAR                  : AE10," &
"RXSOC                  : AF14," &
"SADDR0                 : B16," &
"SADDR1                 : A16," &
"SADDR10                : A18," &
"SADDR11                : D19," &
"SADDR12                : C19," &
"SADDR13                : B19," &
"SADDR14                : A19," &
"SADDR15                : D20," &
"SADDR16                : C20," &
"SADDR17                : B20," &
"SADDR18                : A20," &
"SADDR19                : D21," &
"SADDR2                 : D16," &
"SADDR3                 : D17," &
"SADDR4                 : C17," &
"SADDR5                 : B17," &
"SADDR6                 : A17," &
"SADDR7                 : D18," &
"SADDR8                 : C18," &
"SADDR9                 : B18," &
"SBW0_NEG               : D6," &
"SBW1_NEG               : A7," &
"SBW2_NEG               : B7," &
"SBW3_NEG               : C7," &
"SCHREF                 : AE25," &
"SCL                    : AD25," &
"SCLK0                  : B3," &
"SCLK1                  : A3," &
"SCS0_NEG               : D5," &
"SCS1_NEG               : A6," &
"SCS2_NEG               : B6," &
"SCS3_NEG               : C6," &
"SDA                    : AC26," &
"SDATA0                 : A8," &
"SDATA1                 : B8," &
"SDATA10                : C10," &
"SDATA11                : D10," &
"SDATA12                : D11," &
"SDATA13                : A11," &
"SDATA14                : B11," &
"SDATA15                : C11," &
"SDATA16                : A12," &
"SDATA17                : B12," &
"SDATA18                : C12," &
"SDATA19                : D12," &
"SDATA2                 : C8," &
"SDATA20                : D13," &
"SDATA21                : A13," &
"SDATA22                : B13," &
"SDATA23                : C13," &
"SDATA24                : C14," &
"SDATA25                : B14," &
"SDATA26                : A14," &
"SDATA27                : D14," &
"SDATA28                : C15," &
"SDATA29                : D15," &
"SDATA3                 : D8," &
```

```
"SDATA30                : B15," &
"SDATA31                : A15," &
"SDATA4                 : A9," &
"SDATA5                 : B9," &
"SDATA6                 : C9," &
"SDATA7                 : D9," &
"SDATA8                 : A10," &
"SDATA9                 : B10," &
"SPARE_A2               : A2," &
"SPARE_A4               : A4," &
"SPARE_A5               : A5," &
"SPARE_AB24             : AB24," &
"SPARE_AC25             : AC25," &
"SPARE_AC6              : AC6," &
"SPARE_AD22             : AD22," &
"SPARE_AE4              : AE4," &
"SPARE_B2               : B2," &
"SPARE_B4               : B4," &
"SPARE_B5               : B5," &
"SPARE_C16              : C16," &
"SPARE_C21              : C21," &
"SPARE_C24              : C24," &
"SPARE_C4               : C4," &
"SPARE_D4               : D4," &
"SPARE_D7               : D7," &
"SPARE_E23              : E23," &
"SPARE_H2               : H2," &
"SPARE_H24              : H24," &
"SPARE_M3               : M3," &
"SPARE_T4               : T4," &
"SPARE_W1               : W1," &
"SPARE_W2               : W2," &
"SPARE_W3               : W3," &
"SPARE_Y4               : Y4," &
"SRNW                   : C5," &
"STAT0                  : AA2," &
"STAT1                  : AA3," &
"STAT2                  : AA4," &
"STAT3                  : Y1," &
"TCLK                   : AE24," &
"TDI                    : AF25," &
"TDO                    : AD26," &
"TESTEN                 : AF22," &
"TESTRST                : AD23," &
"TMS                    : AF24," &
"TRST_NEG               : AF23," &
"TXADDR0                : AF21," &
"TXADDR1                : AE21," &
"TXADDR2                : AD21," &
"TXADDR3                : AC21," &
"TXADDR4                : AF20," &
"TXCLAV                 : AE16," &
"TXCLK                  : AC15," &
"TXDATA0                : AE20," &
"TXDATA1                : AD20," &
"TXDATA10               : AC18," &
"TXDATA11               : AF17," &
"TXDATA12               : AE17," &
"TXDATA13               : AD17," &
```

```
"TXDATA14               : AC17," &
"TXDATA15               : AC16," &
"TXDATA2                : AC20," &
"TXDATA3                : AF19," &
"TXDATA4                : AE19," &
"TXDATA5                : AD19," &
"TXDATA6                : AC19," &
"TXDATA7                : AF18," &
"TXDATA8                : AE18," &
"TXDATA9                : AD18," &
"TXEN_NEG               : AF15," &
"TXPAR                  : AF16," &
"TXSOC                  : AD16," &
"UTOPIA1                : AC22," &
"VGG                    : AC8," &
"VDD_AB11               : AB11," &
"VDD_AB12               : AB12," &
"VDD_AB15               : AB15," &
"VDD_AB16               : AB16," &
"VDD_AB19               : AB19," &
"VDD_AB20               : AB20," &
"VDD_AB7                : AB7," &
"VDD_AB8                : AB8," &
"VDD_E11                : E11," &
"VDD_E12                : E12," &
"VDD_E15                : E15," &
"VDD_E16                : E16," &
"VDD_E19                : E19," &
"VDD_E20                : E20," &
"VDD_E7                 : E7," &
"VDD_E8                 : E8," &
"VDD_G22                : G22," &
"VDD_G5                 : G5," &
"VDD_H22                : H22," &
"VDD_H5                 : H5," &
"VDD_L22                : L22," &
"VDD_L5                 : L5," &
"VDD_M22                : M22," &
"VDD_M5                 : M5," &
"VDD_R22                : R22," &
"VDD_R5                 : R5," &
"VDD_T22                : T22," &
"VDD_T5                 : T5," &
"VDD_W22                : W22," &
"VDD_W5                 : W5," &
"VDD_Y22                : Y22," &
"VDD_Y5                 : Y5," &
"VSS_AA22               : AA22," &
"VSS_AA5                : AA5," &
"VSS_AB10               : AB10," &
"VSS_AB13               : AB13," &
"VSS_AB14               : AB14," &
"VSS_AB17               : AB17," &
"VSS_AB18               : AB18," &
"VSS_AB21               : AB21," &
"VSS_AB22               : AB22," &
"VSS_AB5                : AB5," &
"VSS_AB6                : AB6," &
"VSS_AB9                : AB9," &
```

```
"VSS_E10                : E10," &
"VSS_E13                : E13," &
"VSS_E14                : E14," &
"VSS_E17                : E17," &
"VSS_E18                : E18," &
"VSS_E21                : E21," &
"VSS_E22                : E22," &
"VSS_E5                 : E5," &
"VSS_E6                 : E6," &
"VSS_E9                 : E9," &
"VSS_F22                : F22," &
"VSS_F5                 : F5," &
"VSS_J22                : J22," &
"VSS_J5                 : J5," &
"VSS_K22                : K22," &
"VSS_K5                 : K5," &
"VSS_L11                : L11," &
"VSS_L12                : L12," &
"VSS_L13                : L13," &
"VSS_L14                : L14," &
"VSS_L15                : L15," &
"VSS_L16                : L16," &
"VSS_M11                : M11," &
"VSS_M12                : M12," &
"VSS_M13                : M13," &
"VSS_M14                : M14," &
"VSS_M15                : M15," &
"VSS_M16                : M16," &
"VSS_N11                : N11," &
"VSS_N12                : N12," &
"VSS_N13                : N13," &
"VSS_N14                : N14," &
"VSS_N15                : N15," &
"VSS_N16                : N16," &
"VSS_N22                : N22," &
"VSS_N5                 : N5," &
"VSS_P11                : P11," &
"VSS_P12                : P12," &
"VSS_P13                : P13," &
"VSS_P14                : P14," &
"VSS_P15                : P15," &
"VSS_P16                : P16," &
"VSS_P22                : P22," &
"VSS_P5                 : P5," &
"VSS_R11                : R11," &
"VSS_R12                : R12," &
"VSS_R13                : R13," &
"VSS_R14                : R14," &
"VSS_R15                : R15," &
"VSS_R16                : R16," &
"VSS_T11                : T11," &
"VSS_T12                : T12," &
"VSS_T13                : T13," &
"VSS_T14                : T14," &
"VSS_T15                : T15," &
"VSS_T16                : T16," &
"VSS_U22                : U22," &
"VSS_U5                 : U5," &
"VSS_V22                : V22," &
```

```
  "VSS_V5                 : V5 ";

-- TAP Port Name Attributes
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_CLOCK of TCLK     : signal is (1.0e+07, BOTH);
attribute TAP_SCAN_RESET of TRST_NEG : signal is true;

-- Instruction Register Attributes
attribute INSTRUCTION_LENGTH of CN8237 : entity is 3;

attribute INSTRUCTION_OPCODE of CN8237 : entity is
  "BYPASS    (111)," &
  "SAMPLE    (001)," &
  "EXTEST    (000)," &
  "HIGHZ     (100)," &
  "IDCODE    (010)";

attribute INSTRUCTION_CAPTURE of CN8237 : entity is "001";

attribute IDCODE_REGISTER of CN8237 : entity is
  "0001" &              -- Version
  "1000001000110111" &  -- Part Number
  "00000010011" &       -- Manufacturer's ID
  "1";

attribute REGISTER_ACCESS of CN8237 : entity is
  "BYPASS    (BYPASS, HIGHZ)," &
  "BOUNDARY  (EXTEST, SAMPLE)," &
  "DEVICE_ID (IDCODE)";

-- Boundary Register Attributes
attribute BOUNDARY_LENGTH of CN8237 : entity is 363;

attribute BOUNDARY_REGISTER of CN8237 : entity is
-- num   cell   port                function  safe  ccell  dsval  rslt
  "0    (BC_4,  GPI           ,       input,    X)," &
  "1    (BC_1,  SCLK1         ,      output3,    X,    67,    1,   Z)," &
  "2    (BC_1,  SCLK0         ,      output3,    X,    67,    1,   Z)," &
  "3    (BC_1,  SRNW          ,      output3,    X,    67,    1,   Z)," &
  "4    (BC_1,  SCS0_NEG      ,      output3,    X,    67,    1,   Z)," &
  "5    (BC_1,  SCS1_NEG      ,      output3,    X,    67,    1,   Z)," &
  "6    (BC_1,  SCS2_NEG      ,      output3,    X,    67,    1,   Z)," &
  "7    (BC_1,  SCS3_NEG      ,      output3,    X,    67,    1,   Z)," &
  "8    (BC_1,  SBW0_NEG      ,      output3,    X,    67,    1,   Z)," &
  "9    (BC_1,  SBW1_NEG      ,      output3,    X,    67,    1,   Z)," &
  "10   (BC_1,  SBW2_NEG      ,      output3,    X,    67,    1,   Z)," &
  "11   (BC_1,  SBW3_NEG      ,      output3,    X,    67,    1,   Z)," &
  "12   (BC_7,  SDATA0        ,        bidir,    X,    19,    1,   Z)," &
  "13   (BC_7,  SDATA1        ,        bidir,    X,    19,    1,   Z)," &
  "14   (BC_7,  SDATA2        ,        bidir,    X,    19,    1,   Z)," &
  "15   (BC_7,  SDATA3        ,        bidir,    X,    19,    1,   Z)," &
  "16   (BC_7,  SDATA4        ,        bidir,    X,    19,    1,   Z)," &
  "17   (BC_7,  SDATA5        ,        bidir,    X,    19,    1,   Z)," &
  "18   (BC_7,  SDATA6        ,        bidir,    X,    19,    1,   Z)," &
  "19   (BC_1,  *             ,      control,    1)," &
  "20   (BC_7,  SDATA7        ,        bidir,    X,    19,    1,   Z)," &
  "21   (BC_7,  SDATA8        ,        bidir,    X,    28,    1,   Z)," &
```

```
"22   (BC_7,  SDATA9        ,       bidir,    X,    28,    1,   Z)," &
"23   (BC_7,  SDATA10       ,       bidir,    X,    28,    1,   Z)," &
"24   (BC_7,  SDATA11       ,       bidir,    X,    28,    1,   Z)," &
"25   (BC_7,  SDATA12       ,       bidir,    X,    28,    1,   Z)," &
"26   (BC_7,  SDATA13       ,       bidir,    X,    28,    1,   Z)," &
"27   (BC_7,  SDATA14       ,       bidir,    X,    28,    1,   Z)," &
"28   (BC_1,  *              ,     control,    1)," &
"29   (BC_7,  SDATA15       ,       bidir,    X,    28,    1,   Z)," &
"30   (BC_7,  SDATA16       ,       bidir,    X,    37,    1,   Z)," &
"31   (BC_7,  SDATA17       ,       bidir,    X,    37,    1,   Z)," &
"32   (BC_7,  SDATA18       ,       bidir,    X,    37,    1,   Z)," &
"33   (BC_7,  SDATA19       ,       bidir,    X,    37,    1,   Z)," &
"34   (BC_7,  SDATA20       ,       bidir,    X,    37,    1,   Z)," &
"35   (BC_7,  SDATA21       ,       bidir,    X,    37,    1,   Z)," &
"36   (BC_7,  SDATA22       ,       bidir,    X,    37,    1,   Z)," &
"37   (BC_1,  *              ,     control,    1)," &
"38   (BC_7,  SDATA23       ,       bidir,    X,    37,    1,   Z)," &
"39   (BC_7,  SDATA24       ,       bidir,    X,    46,    1,   Z)," &
"40   (BC_7,  SDATA25       ,       bidir,    X,    46,    1,   Z)," &
"41   (BC_7,  SDATA26       ,       bidir,    X,    46,    1,   Z)," &
"42   (BC_7,  SDATA27       ,       bidir,    X,    46,    1,   Z)," &
"43   (BC_7,  SDATA28       ,       bidir,    X,    46,    1,   Z)," &
"44   (BC_7,  SDATA29       ,       bidir,    X,    46,    1,   Z)," &
"45   (BC_7,  SDATA30       ,       bidir,    X,    46,    1,   Z)," &
"46   (BC_1,  *              ,     control,    1)," &
"47   (BC_7,  SDATA31       ,       bidir,    X,    46,    1,   Z)," &
"48   (BC_1,  SADDR0        ,     output3,    X,    67,    1,   Z)," &
"49   (BC_1,  SADDR1        ,     output3,    X,    67,    1,   Z)," &
"50   (BC_1,  SADDR2        ,     output3,    X,    67,    1,   Z)," &
"51   (BC_1,  SADDR3        ,     output3,    X,    67,    1,   Z)," &
"52   (BC_1,  SADDR4        ,     output3,    X,    67,    1,   Z)," &
"53   (BC_1,  SADDR5        ,     output3,    X,    67,    1,   Z)," &
"54   (BC_1,  SADDR6        ,     output3,    X,    67,    1,   Z)," &
"55   (BC_1,  SADDR7        ,     output3,    X,    67,    1,   Z)," &
"56   (BC_1,  SADDR8        ,     output3,    X,    67,    1,   Z)," &
"57   (BC_1,  SADDR9        ,     output3,    X,    67,    1,   Z)," &
"58   (BC_1,  SADDR10       ,     output3,    X,    67,    1,   Z)," &
"59   (BC_1,  SADDR11       ,     output3,    X,    67,    1,   Z)," &
"60   (BC_1,  SADDR12       ,     output3,    X,    67,    1,   Z)," &
"61   (BC_1,  SADDR13       ,     output3,    X,    67,    1,   Z)," &
"62   (BC_1,  SADDR14       ,     output3,    X,    67,    1,   Z)," &
"63   (BC_1,  SADDR15       ,     output3,    X,    67,    1,   Z)," &
"64   (BC_1,  SADDR16       ,     output3,    X,    67,    1,   Z)," &
"65   (BC_1,  SADDR17       ,     output3,    X,    67,    1,   Z)," &
"66   (BC_1,  SADDR18       ,     output3,    X,    67,    1,   Z)," &
"67   (BC_1,  *              ,     control,    1)," &
"68   (BC_1,  SADDR19       ,     output3,    X,    67,    1,   Z)," &
"69   (BC_1,  *              ,     control,    1)," &
"70   (BC_1,  HLED_NEG       ,     output3,    X,    69,    1,   Z)," &
"71   (BC_1,  *              ,     control,    1)," &
"72   (BC_7,  HPAR64         ,       bidir,    X,    71,    1,   Z)," &
"73   (BC_7,  HAD32          ,       bidir,    X,    80,    1,   Z)," &
"74   (BC_7,  HAD33          ,       bidir,    X,    80,    1,   Z)," &
"75   (BC_7,  HAD34          ,       bidir,    X,    80,    1,   Z)," &
"76   (BC_7,  HAD35          ,       bidir,    X,    80,    1,   Z)," &
"77   (BC_7,  HAD36          ,       bidir,    X,    80,    1,   Z)," &
"78   (BC_7,  HAD37          ,       bidir,    X,    80,    1,   Z)," &
"79   (BC_7,  HAD38          ,       bidir,    X,    80,    1,   Z)," &
"80   (BC_1,  *              ,     control,    1)," &
```

```
"81   (BC_7,  HAD39          ,        bidir,    X,    80,    1,   Z)," &
"82   (BC_7,  HAD40          ,        bidir,    X,    89,    1,   Z)," &
"83   (BC_7,  HAD41          ,        bidir,    X,    89,    1,   Z)," &
"84   (BC_7,  HAD42          ,        bidir,    X,    89,    1,   Z)," &
"85   (BC_7,  HAD43          ,        bidir,    X,    89,    1,   Z)," &
"86   (BC_7,  HAD44          ,        bidir,    X,    89,    1,   Z)," &
"87   (BC_7,  HAD45          ,        bidir,    X,    89,    1,   Z)," &
"88   (BC_7,  HAD46          ,        bidir,    X,    89,    1,   Z)," &
"89   (BC_1,  *              ,      control,    1)," &
"90   (BC_7,  HAD47          ,        bidir,    X,    89,    1,   Z)," &
"91   (BC_7,  HAD48          ,        bidir,    X,    98,    1,   Z)," &
"92   (BC_7,  HAD49          ,        bidir,    X,    98,    1,   Z)," &
"93   (BC_7,  HAD50          ,        bidir,    X,    98,    1,   Z)," &
"94   (BC_7,  HAD51          ,        bidir,    X,    98,    1,   Z)," &
"95   (BC_7,  HAD52          ,        bidir,    X,    98,    1,   Z)," &
"96   (BC_7,  HAD53          ,        bidir,    X,    98,    1,   Z)," &
"97   (BC_7,  HAD54          ,        bidir,    X,    98,    1,   Z)," &
"98   (BC_1,  *              ,      control,    1)," &
"99   (BC_7,  HAD55          ,        bidir,    X,    98,    1,   Z)," &
"100  (BC_7,  HAD56          ,        bidir,    X,   107,    1,   Z)," &
"101  (BC_7,  HAD57          ,        bidir,    X,   107,    1,   Z)," &
"102  (BC_7,  HAD58          ,        bidir,    X,   107,    1,   Z)," &
"103  (BC_7,  HAD59          ,        bidir,    X,   107,    1,   Z)," &
"104  (BC_7,  HAD60          ,        bidir,    X,   107,    1,   Z)," &
"105  (BC_7,  HAD61          ,        bidir,    X,   107,    1,   Z)," &
"106  (BC_7,  HAD62          ,        bidir,    X,   107,    1,   Z)," &
"107  (BC_1,  *              ,      control,    1)," &
"108  (BC_7,  HAD63          ,        bidir,    X,   107,    1,   Z)," &
"109  (BC_7,  HCBE4_NEG      ,        bidir,    X,   112,    1,   Z)," &
"110  (BC_7,  HCBE5_NEG      ,        bidir,    X,   112,    1,   Z)," &
"111  (BC_7,  HCBE6_NEG      ,        bidir,    X,   112,    1,   Z)," &
"112  (BC_1,  *              ,      control,    1)," &
"113  (BC_7,  HCBE7_NEG      ,        bidir,    X,   112,    1,   Z)," &
"114  (BC_1,  *              ,      control,    1)," &
"115  (BC_7,  HREQ64_NEG     ,        bidir,    X,   114,    1,   Z)," &
"116  (BC_1,  *              ,      control,    1)," &
"117  (BC_7,  HENUM_NEG      ,        bidir,    X,   116,    1,   Z)," &
"118  (BC_1,  *              ,      control,    1)," &
"119  (BC_7,  HACK64_NEG     ,        bidir,    X,   118,    1,   Z)," &
"120  (BC_7,  HAD0           ,        bidir,    X,   127,    1,   Z)," &
"121  (BC_7,  HAD1           ,        bidir,    X,   127,    1,   Z)," &
"122  (BC_7,  HAD2           ,        bidir,    X,   127,    1,   Z)," &
"123  (BC_7,  HAD3           ,        bidir,    X,   127,    1,   Z)," &
"124  (BC_7,  HAD4           ,        bidir,    X,   127,    1,   Z)," &
"125  (BC_7,  HAD5           ,        bidir,    X,   127,    1,   Z)," &
"126  (BC_7,  HAD6           ,        bidir,    X,   127,    1,   Z)," &
"127  (BC_1,  *              ,      control,    1)," &
"128  (BC_7,  HAD7           ,        bidir,    X,   127,    1,   Z)," &
"129  (BC_7,  HCBE0_NEG      ,        bidir,    X,   112,    1,   Z)," &
"130  (BC_7,  HAD8           ,        bidir,    X,   138,    1,   Z)," &
"131  (BC_7,  HAD9           ,        bidir,    X,   138,    1,   Z)," &
"132  (BC_4,  HM66EN         ,        input,    X)," &
"133  (BC_7,  HAD10          ,        bidir,    X,   138,    1,   Z)," &
"134  (BC_7,  HAD11          ,        bidir,    X,   138,    1,   Z)," &
"135  (BC_7,  HAD12          ,        bidir,    X,   138,    1,   Z)," &
"136  (BC_7,  HAD13          ,        bidir,    X,   138,    1,   Z)," &
"137  (BC_7,  HAD14          ,        bidir,    X,   138,    1,   Z)," &
"138  (BC_1,  *              ,      control,    1)," &
"139  (BC_7,  HAD15          ,        bidir,    X,   138,    1,   Z)," &
```

```
"140  (BC_7,  HCBE1_NEG      ,        bidir,    X,   112,    1,   Z)," &
"141  (BC_1,  *              ,      control,    1)," &
"142  (BC_7,  HPAR           ,        bidir,    X,   141,    1,   Z)," &
"143  (BC_1,  *              ,      control,    1)," &
"144  (BC_7,  HSERR_NEG      ,        bidir,    X,   143,    1,   Z)," &
"145  (BC_1,  *              ,      control,    1)," &
"146  (BC_7,  HPERR_NEG      ,        bidir,    X,   145,    1,   Z)," &
"147  (BC_1,  *              ,      control,    1)," &
"148  (BC_7,  HSTOP_NEG      ,        bidir,    X,   147,    1,   Z)," &
"149  (BC_1,  *              ,      control,    1)," &
"150  (BC_7,  HDEVSEL_NEG    ,        bidir,    X,   149,    1,   Z)," &
"151  (BC_1,  *              ,      control,    1)," &
"152  (BC_7,  HTRDY_NEG      ,        bidir,    X,   151,    1,   Z)," &
"153  (BC_1,  *              ,      control,    1)," &
"154  (BC_7,  HIRDY_NEG      ,        bidir,    X,   153,    1,   Z)," &
"155  (BC_1,  *              ,      control,    1)," &
"156  (BC_7,  HFRAME_NEG     ,        bidir,    X,   155,    1,   Z)," &
"157  (BC_7,  HCBE2_NEG      ,        bidir,    X,   112,    1,   Z)," &
"158  (BC_7,  HAD16          ,        bidir,    X,   165,    1,   Z)," &
"159  (BC_7,  HAD17          ,        bidir,    X,   165,    1,   Z)," &
"160  (BC_7,  HAD18          ,        bidir,    X,   165,    1,   Z)," &
"161  (BC_7,  HAD19          ,        bidir,    X,   165,    1,   Z)," &
"162  (BC_7,  HAD20          ,        bidir,    X,   165,    1,   Z)," &
"163  (BC_7,  HAD21          ,        bidir,    X,   165,    1,   Z)," &
"164  (BC_7,  HAD22          ,        bidir,    X,   165,    1,   Z)," &
"165  (BC_1,  *              ,      control,    1)," &
"166  (BC_7,  HAD23          ,        bidir,    X,   165,    1,   Z)," &
"167  (BC_4,  HIDSEL         ,        input,    X)," &
"168  (BC_7,  HCBE3_NEG      ,        bidir,    X,   112,    1,   Z)," &
"169  (BC_7,  HAD24          ,        bidir,    X,   176,    1,   Z)," &
"170  (BC_7,  HAD25          ,        bidir,    X,   176,    1,   Z)," &
"171  (BC_7,  HAD26          ,        bidir,    X,   176,    1,   Z)," &
"172  (BC_7,  HAD27          ,        bidir,    X,   176,    1,   Z)," &
"173  (BC_7,  HAD28          ,        bidir,    X,   176,    1,   Z)," &
"174  (BC_7,  HAD29          ,        bidir,    X,   176,    1,   Z)," &
"175  (BC_7,  HAD30          ,        bidir,    X,   176,    1,   Z)," &
"176  (BC_1,  *              ,      control,    1)," &
"177  (BC_7,  HAD31          ,        bidir,    X,   176,    1,   Z)," &
"178  (BC_1,  *              ,      control,    1)," &
"179  (BC_7,  HREQ_NEG       ,        bidir,    X,   178,    1,   Z)," &
"180  (BC_4,  HGNT_NEG       ,        input,    X)," &
"181  (BC_4,  PCILITE        ,        input,    X)," &
"182  (BC_1,  *              ,      control,    1)," &
"183  (BC_7,  HINTA_NEG      ,        bidir,    X,   182,    1,   Z)," &
"184  (BC_1,  *              ,      control,    1)," &
"185  (BC_7,  SDA            ,        bidir,    X,   184,    1,   Z)," &
"186  (BC_1,  *              ,      control,    1)," &
"187  (BC_7,  SCL            ,        bidir,    X,   186,    1,   Z)," &
"188  (BC_4,  HCLK           ,        input,    X)," &
"189  (BC_4,  HRST_NEG       ,        input,    X)," &
"190  (BC_1,  EEPWR          ,      output3,    X,   280,    1,   Z)," &
"191  (BC_4,  HSWITCH_NEG    ,        input,    X)," &
"192  (BC_4,  SCHREF         ,        input,    X)," &
"193  (BC_4,  PLLBYPASS      ,        input,    X)," &
"194  (BC_4,  TESTRST        ,        input,    X)," &
"195  (BC_4,  TESTEN         ,        input,    X)," &
"196  (BC_4,  FRCFG          ,        input,    X)," &
"197  (BC_4,  UTOPIA1        ,        input,    X)," &
"198  (BC_7,  TXADDR0        ,        bidir,    X,   202,    1,   Z)," &
```

```
"199  (BC_7,  TXADDR1       ,       bidir,    X,   202,    1,   Z)," &
"200  (BC_7,  TXADDR2       ,       bidir,    X,   202,    1,   Z)," &
"201  (BC_7,  TXADDR3       ,       bidir,    X,   202,    1,   Z)," &
"202  (BC_1,  *             ,     control,    1)," &
"203  (BC_7,  TXADDR4       ,       bidir,    X,   202,    1,   Z)," &
"204  (BC_1,  TXDATA0       ,     output3,    X,   220,    1,   Z)," &
"205  (BC_1,  TXDATA1       ,     output3,    X,   220,    1,   Z)," &
"206  (BC_1,  TXDATA2       ,     output3,    X,   220,    1,   Z)," &
"207  (BC_1,  TXDATA3       ,     output3,    X,   220,    1,   Z)," &
"208  (BC_1,  TXDATA4       ,     output3,    X,   220,    1,   Z)," &
"209  (BC_1,  TXDATA5       ,     output3,    X,   220,    1,   Z)," &
"210  (BC_1,  TXDATA6       ,     output3,    X,   220,    1,   Z)," &
"211  (BC_1,  TXDATA7       ,     output3,    X,   220,    1,   Z)," &
"212  (BC_1,  TXDATA8       ,     output3,    X,   220,    1,   Z)," &
"213  (BC_1,  TXDATA9       ,     output3,    X,   220,    1,   Z)," &
"214  (BC_1,  TXDATA10      ,     output3,    X,   220,    1,   Z)," &
"215  (BC_1,  TXDATA11      ,     output3,    X,   220,    1,   Z)," &
"216  (BC_1,  TXDATA12      ,     output3,    X,   220,    1,   Z)," &
"217  (BC_1,  TXDATA13      ,     output3,    X,   220,    1,   Z)," &
"218  (BC_1,  TXDATA14      ,     output3,    X,   220,    1,   Z)," &
"219  (BC_1,  TXDATA15      ,     output3,    X,   220,    1,   Z)," &
"220  (BC_1,  *             ,     control,    1)," &
"221  (BC_1,  TXPAR         ,     output3,    X,   220,    1,   Z)," &
"222  (BC_1,  *             ,     control,    1)," &
"223  (BC_7,  TXCLAV        ,       bidir,    X,   222,    1,  PULL0)," &
"224  (BC_1,  *             ,     control,    1)," &
"225  (BC_1,  TXSOC         ,     output3,    X,   224,    1,   Z)," &
"226  (BC_1,  *             ,     control,    1)," &
"227  (BC_7,  TXEN_NEG      ,       bidir,    X,   226,    1, PULL1)," &
"228  (BC_1,  *             ,     control,    1)," &
"229  (BC_7,  RXCLAV        ,       bidir,    X,   228,    1, PULL0)," &
"230  (BC_1,  *             ,     control,    1)," &
"231  (BC_7,  RXEN_NEG      ,       bidir,    X,   230,    1, PULL1)," &
"232  (BC_4,  TXCLK         ,       input,    X)," &
"233  (BC_4,  RXCLK         ,       input,    X)," &
"234  (BC_4,  RXSOC         ,       input,    X)," &
"235  (BC_4,  RXDATA0       ,       input,    X)," &
"236  (BC_4,  RXDATA1       ,       input,    X)," &
"237  (BC_4,  RXDATA4       ,       input,    X)," &
"238  (BC_4,  RXDATA3       ,       input,    X)," &
"239  (BC_4,  RXDATA2       ,       input,    X)," &
"240  (BC_4,  RXDATA5       ,       input,    X)," &
"241  (BC_4,  RXDATA7       ,       input,    X)," &
"242  (BC_4,  RXDATA6       ,       input,    X)," &
"243  (BC_4,  RXDATA8       ,       input,    X)," &
"244  (BC_4,  RXDATA9       ,       input,    X)," &
"245  (BC_4,  RXDATA12      ,       input,    X)," &
"246  (BC_4,  RXDATA11      ,       input,    X)," &
"247  (BC_4,  RXDATA10      ,       input,    X)," &
"248  (BC_4,  RXDATA13      ,       input,    X)," &
"249  (BC_4,  RXDATA14      ,       input,    X)," &
"250  (BC_4,  RXDATA15      ,       input,    X)," &
"251  (BC_4,  RXPAR         ,       input,    X)," &
"252  (BC_7,  RXADDR0       ,       bidir,    X,   256,    1,   Z)," &
"253  (BC_7,  RXADDR1       ,       bidir,    X,   256,    1,   Z)," &
"254  (BC_7,  RXADDR2       ,       bidir,    X,   256,    1,   Z)," &
"255  (BC_7,  RXADDR3       ,       bidir,    X,   256,    1,   Z)," &
"256  (BC_1,  *             ,     control,    1)," &
"257  (BC_7,  RXADDR4       ,       bidir,    X,   256,    1,   Z)," &
```

```
"258  (BC_4,  CLK264P       ,       input,    X)," &
"259  (BC_1,  CLKD3         ,     output3,    X,   280,    1,   Z)," &
"260  (BC_1,  PCLK          ,     output3,    X,   280,    1,   Z)," &
"261  (BC_1,  PRST_NEG      ,     output3,    X,   280,    1,   Z)," &
"262  (BC_7,  PDATA0        ,       bidir,    X,   270,    1,   Z)," &
"263  (BC_7,  PDATA1        ,       bidir,    X,   270,    1,   Z)," &
"264  (BC_7,  PDATA2        ,       bidir,    X,   270,    1,   Z)," &
"265  (BC_7,  PDATA3        ,       bidir,    X,   270,    1,   Z)," &
"266  (BC_7,  PDATA4        ,       bidir,    X,   270,    1,   Z)," &
"267  (BC_7,  PDATA5        ,       bidir,    X,   270,    1,   Z)," &
"268  (BC_7,  PDATA6        ,       bidir,    X,   270,    1,   Z)," &
"269  (BC_1,  PADDR0        ,     output3,    X,   280,    1,   Z)," &
"270  (BC_1,  *             ,     control,    1)," &
"271  (BC_7,  PDATA7        ,       bidir,    X,   270,    1,   Z)," &
"272  (BC_1,  PADDR3        ,     output3,    X,   280,    1,   Z)," &
"273  (BC_1,  PADDR2        ,     output3,    X,   280,    1,   Z)," &
"274  (BC_1,  PADDR6        ,     output3,    X,   280,    1,   Z)," &
"275  (BC_1,  PADDR10       ,     output3,    X,   280,    1,   Z)," &
"276  (BC_1,  PADDR1        ,     output3,    X,   280,    1,   Z)," &
"277  (BC_1,  PADDR4        ,     output3,    X,   280,    1,   Z)," &
"278  (BC_1,  PADDR7        ,     output3,    X,   280,    1,   Z)," &
"279  (BC_1,  PADDR11       ,     output3,    X,   280,    1,   Z)," &
"280  (BC_1,  *             ,     control,    1)," &
"281  (BC_1,  PADDR12       ,     output3,    X,   280,    1,   Z)," &
"282  (BC_1,  PADDR8        ,     output3,    X,   280,    1,   Z)," &
"283  (BC_1,  PAS_NEG       ,     output3,    X,   280,    1,   Z)," &
"284  (BC_1,  PADDR9        ,     output3,    X,   280,    1,   Z)," &
"285  (BC_1,  PADDR5        ,     output3,    X,   280,    1,   Z)," &
"286  (BC_1,  PWNR          ,     output3,    X,   280,    1,   Z)," &
"287  (BC_1,  PCS_NEG       ,     output3,    X,   280,    1,   Z)," &
"288  (BC_1,  PDS_NEG       ,     output3,    X,   280,    1,   Z)," &
"289  (BC_4,  PWAIT_NEG     ,       input,    X)," &
"290  (BC_4,  PINT_NEG      ,       input,    X)," &
"291  (BC_1,  STAT0         ,     output3,    X,   361,    1,   Z)," &
"292  (BC_1,  STAT1         ,     output3,    X,   361,    1,   Z)," &
"293  (BC_1,  STAT2         ,     output3,    X,   361,    1,   Z)," &
"294  (BC_1,  STAT3         ,     output3,    X,   361,    1,   Z)," &
"295  (BC_1,  RCLK0         ,     output3,    X,   361,    1,   Z)," &
"296  (BC_1,  RCLK1         ,     output3,    X,   361,    1,   Z)," &
"297  (BC_1,  RRNW          ,     output3,    X,   361,    1,   Z)," &
"298  (BC_1,  RCS0_NEG      ,     output3,    X,   361,    1,   Z)," &
"299  (BC_1,  RCS1_NEG      ,     output3,    X,   361,    1,   Z)," &
"300  (BC_1,  RCS2_NEG      ,     output3,    X,   361,    1,   Z)," &
"301  (BC_1,  RCS3_NEG      ,     output3,    X,   361,    1,   Z)," &
"302  (BC_1,  RBW0_NEG      ,     output3,    X,   361,    1,   Z)," &
"303  (BC_1,  RBW1_NEG      ,     output3,    X,   361,    1,   Z)," &
"304  (BC_1,  RBW2_NEG      ,     output3,    X,   361,    1,   Z)," &
"305  (BC_1,  RBW3_NEG      ,     output3,    X,   361,    1,   Z)," &
"306  (BC_7,  RDATA0        ,       bidir,    X,   313,    1,   Z)," &
"307  (BC_7,  RDATA1        ,       bidir,    X,   313,    1,   Z)," &
"308  (BC_7,  RDATA2        ,       bidir,    X,   313,    1,   Z)," &
"309  (BC_7,  RDATA3        ,       bidir,    X,   313,    1,   Z)," &
"310  (BC_7,  RDATA4        ,       bidir,    X,   313,    1,   Z)," &
"311  (BC_7,  RDATA5        ,       bidir,    X,   313,    1,   Z)," &
"312  (BC_7,  RDATA6        ,       bidir,    X,   313,    1,   Z)," &
"313  (BC_1,  *             ,     control,    1)," &
"314  (BC_7,  RDATA7        ,       bidir,    X,   313,    1,   Z)," &
"315  (BC_7,  RDATA8        ,       bidir,    X,   322,    1,   Z)," &
"316  (BC_7,  RDATA9        ,       bidir,    X,   322,    1,   Z)," &
```

```
        "317  (BC_7,  RDATA10       ,       bidir,    X,   322,    1,   Z)," &
        "318  (BC_7,  RDATA11       ,       bidir,    X,   322,    1,   Z)," &
        "319  (BC_7,  RDATA12       ,       bidir,    X,   322,    1,   Z)," &
        "320  (BC_7,  RDATA13       ,       bidir,    X,   322,    1,   Z)," &
        "321  (BC_7,  RDATA14       ,       bidir,    X,   322,    1,   Z)," &
        "322  (BC_1,  *             ,     control,    1)," &
        "323  (BC_7,  RDATA15       ,       bidir,    X,   322,    1,   Z)," &
        "324  (BC_7,  RDATA16       ,       bidir,    X,   331,    1,   Z)," &
        "325  (BC_7,  RDATA17       ,       bidir,    X,   331,    1,   Z)," &
        "326  (BC_7,  RDATA18       ,       bidir,    X,   331,    1,   Z)," &
        "327  (BC_7,  RDATA19       ,       bidir,    X,   331,    1,   Z)," &
        "328  (BC_7,  RDATA20       ,       bidir,    X,   331,    1,   Z)," &
        "329  (BC_7,  RDATA21       ,       bidir,    X,   331,    1,   Z)," &
        "330  (BC_7,  RDATA22       ,       bidir,    X,   331,    1,   Z)," &
        "331  (BC_1,  *             ,     control,    1)," &
        "332  (BC_7,  RDATA23       ,       bidir,    X,   331,    1,   Z)," &
        "333  (BC_7,  RDATA24       ,       bidir,    X,   340,    1,   Z)," &
        "334  (BC_7,  RDATA25       ,       bidir,    X,   340,    1,   Z)," &
        "335  (BC_7,  RDATA26       ,       bidir,    X,   340,    1,   Z)," &
        "336  (BC_7,  RDATA27       ,       bidir,    X,   340,    1,   Z)," &
        "337  (BC_7,  RDATA28       ,       bidir,    X,   340,    1,   Z)," &
        "338  (BC_7,  RDATA29       ,       bidir,    X,   340,    1,   Z)," &
        "339  (BC_7,  RDATA30       ,       bidir,    X,   340,    1,   Z)," &
        "340  (BC_1,  *             ,     control,    1)," &
        "341  (BC_7,  RDATA31       ,       bidir,    X,   340,    1,   Z)," &
        "342  (BC_1,  RADDR0        ,     output3,    X,   361,    1,   Z)," &
        "343  (BC_1,  RADDR1        ,     output3,    X,   361,    1,   Z)," &
        "344  (BC_1,  RADDR2        ,     output3,    X,   361,    1,   Z)," &
        "345  (BC_1,  RADDR3        ,     output3,    X,   361,    1,   Z)," &
        "346  (BC_1,  RADDR4        ,     output3,    X,   361,    1,   Z)," &
        "347  (BC_1,  RADDR5        ,     output3,    X,   361,    1,   Z)," &
        "348  (BC_1,  RADDR6        ,     output3,    X,   361,    1,   Z)," &
        "349  (BC_1,  RADDR7        ,     output3,    X,   361,    1,   Z)," &
        "350  (BC_1,  RADDR8        ,     output3,    X,   361,    1,   Z)," &
        "351  (BC_1,  RADDR9        ,     output3,    X,   361,    1,   Z)," &
        "352  (BC_1,  RADDR10       ,     output3,    X,   361,    1,   Z)," &
        "353  (BC_1,  RADDR11       ,     output3,    X,   361,    1,   Z)," &
        "354  (BC_1,  RADDR12       ,     output3,    X,   361,    1,   Z)," &
        "355  (BC_1,  RADDR13       ,     output3,    X,   361,    1,   Z)," &
        "356  (BC_1,  RADDR14       ,     output3,    X,   361,    1,   Z)," &
        "357  (BC_1,  RADDR15       ,     output3,    X,   361,    1,   Z)," &
        "358  (BC_1,  RADDR16       ,     output3,    X,   361,    1,   Z)," &
        "359  (BC_1,  RADDR17       ,     output3,    X,   361,    1,   Z)," &
        "360  (BC_1,  RADDR18       ,     output3,    X,   361,    1,   Z)," &
        "361  (BC_1,  *             ,     control,    1)," &
        "362  (BC_1,  RADDR19       ,     output3,    X,   361,    1,   Z)";

end CN8237;
```

Appendix B : List of Acronyms

A

AAL	ATM Adaption Layer
ABR	Available Bit Rate
ACR	Allowed Cell Rate
ATM	Asynchronous Transfer Mode

B

BASIZE	Buffer Allocation Size
BGA	Ball Grid Array
BOM	Beginning of Message
BSDL	Boundary Scan Description Language
BTAG	Beginning Tag

C

CBR	Constant Bit Rate
CCR	Current Cell Rate
CDV	Cell Delay Variation
CDVT	Cell Delay Variation Tolerance
CLP	Cell Loss Priority
CLR	Cell Loss Ratio
CI	Congestion Indication
COM	Continuation of Message
CPCS	Common Part Convergence Sublayer
CPI	Common Part Indicator
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CSR	Control and Status Registers
CTD	Cell Transfer Delay

D

DE	Discard Eligibility
DMA	Direct Memory Access

E

EDC	Error Detection Code
EOM	End of Message
EPD	Early Packet Discard
ER	Explicit Rate
ETAG	Ending Tag
EVS	Evaluation System

F

FIFO	First In First Out

G

GCRA	Generic Cell Rate Algorithm
GFC	Generic Flow Control
GFR	Guaranteed Frame Rate
GPIO	General Purpose Input/Output

I

IETF	Internet Engineering Task Force
ILMI	Interim Local Management Interface

J

JTAG	Joint Test Action Group

L

LANE	LAN Emulation
LEC	LAN Emulation Client
LECID	LAN Emulation Client ID
LEN	Length
LSB	Least Significant Bit

M

Mbit	Megabit
Mbps	Megabits Per Second
MBS	Maximum Burst Size
MIB	Management Information Base
MSB	Most Significant Bit

N

NI	No Increase
NIC	Network Interface Card
NNI	Network-to-Network Interface

O

OAM	Operation And Maintenance

P

PCI	Peripheral Component Interconnect
PCR	Peak Cell Rate
PDU	Protocol Data Unit
PFE	Package Forward Engine
PHY	Physical Layer Device
PM	Performance Monitoring
PTI	Payload Type Identifier
PVC	Permanent Virtual Connections (Circuit)

Q

QoS	Quality of Service

R

RDB	Rate Decision Block
RDF	Rate Decrease Factor
RM	Resource Management
RR	Relative Rate
RSM	Reassembly Coprocessor

S

SAM	Service Access Multiplexer
SBD	Segmentation Buffer Descriptor
SCR	Sustainable Cell Rate
SDU	Service Data Unit
SEG	Segmentation Coprocessor
ServiceSAR	Service Segmentation and Reassembly Controller
SMDS	Switched Multimegabit Data Service
SNMP	Simple Network Management Protocol
SRAM	Static Random Access Memory
SRC	Segmentation and Reassembly Controller Chip
SSM	Single Segment Message
ST	Segment Type
SVC	Switched Virtual Circuit (Connection)

U

UNI	User-to-Network Interface
UTOPIA	Universal Test and Operation Physical Interface for ATM
UU	User-to-User

T

TCR	Tagged Cell Rate
TTL	Transistor-Transistor Logic

U

UBR	Unspecified Bit Rate

V

VBR	Variable Bit Rate
VC	Virtual Circuit
VCC	Virtual Channel Connection
VCI	Virtual Channel Identifier
VP	Virtual Path
VPI	Virtual Path Identifier

MINDSPEED™

www.mindspeed.com

General Information:
U.S. and Canada: (800) 854-8099
International: (949) 483-6996
Headquarters - Newport Beach
4311 Jamboree Rd. P.O. Box C
Newport Beach, CA. 92658-8902