PM7326

S/UNI™ - APEX

S/UNI-APEX

ATM/PACKET TRAFFIC MANAGER AND SWITCH

HARDWARE PROGRAMMER'S GUIDE

RELEASED

ISSUE 2: MARCH 2000

REVISION HISTORY

Issue No.	Issue Date	Details of Change
1	Dec. 1999	Original Issue
2	Mar. 2000	• Added sections 5.4.1 Configuring the Shaper and 5.4.2 Configuring Queue Context • Table 2, Loop Class memory requirements changed to 128K • Inserted Table 3 and Table 4, example memory configurations • Changed Figure 2 to reflect recommended connection setup configuration step order and re-ordered section 6.1 appropriately. • Added section 6.1.2.1 Setting WAN Port Scheduler Context • Updated Section 6.2.2 Class Tear down steps

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 DEFINITIONS

Table 1 Terminology

Term	Definition
AAL5	ATM Adaptation Layer
Any-PHY	Interoperable version of UTOPIA and SCI-PHY, with inband addressing.
ATM	Asynchronous Transfer Mode
BOM	Beginning of Message
CBI	Common Bus Interface
CBR	Constant Bit Rate
CDV	Cell Delay Variation
CDVT	Cell Delay Variation Tolerance
CLP	Cell Loss Priority
CTD	Cell Transfer Delay
DLL	Delay Locked Loop
EOM	End of Message
EPD	Early Packet Discard
GFR	Guaranteed Frame Rate
ICI	Ingress Connection Identifier
MBS	Maximum Burst Size
MCR	Minimum Cell Rate
MFS	Maximum Frame Size
OAM	Operation, Administration and Maintenance
PCR	Peak Cell Rate
PHY	Physical Layer Device
PPD	Partial Packet Discard
PTI	Payload Type Indicator
QOS	Quality of Service
SAR	Segmentation and Re-assembly

SCI-PHY	PMC-Sierra enhanced UTOPIA bus
SCR	Sustained Cell Rate
UBR	Unspecified Bit Rate
UTOPIA	Universal Test & Operations PHY Interface for ATM
VBR	Variable Bit Rate
WAN	Wide Area Network

2 REFERENCES

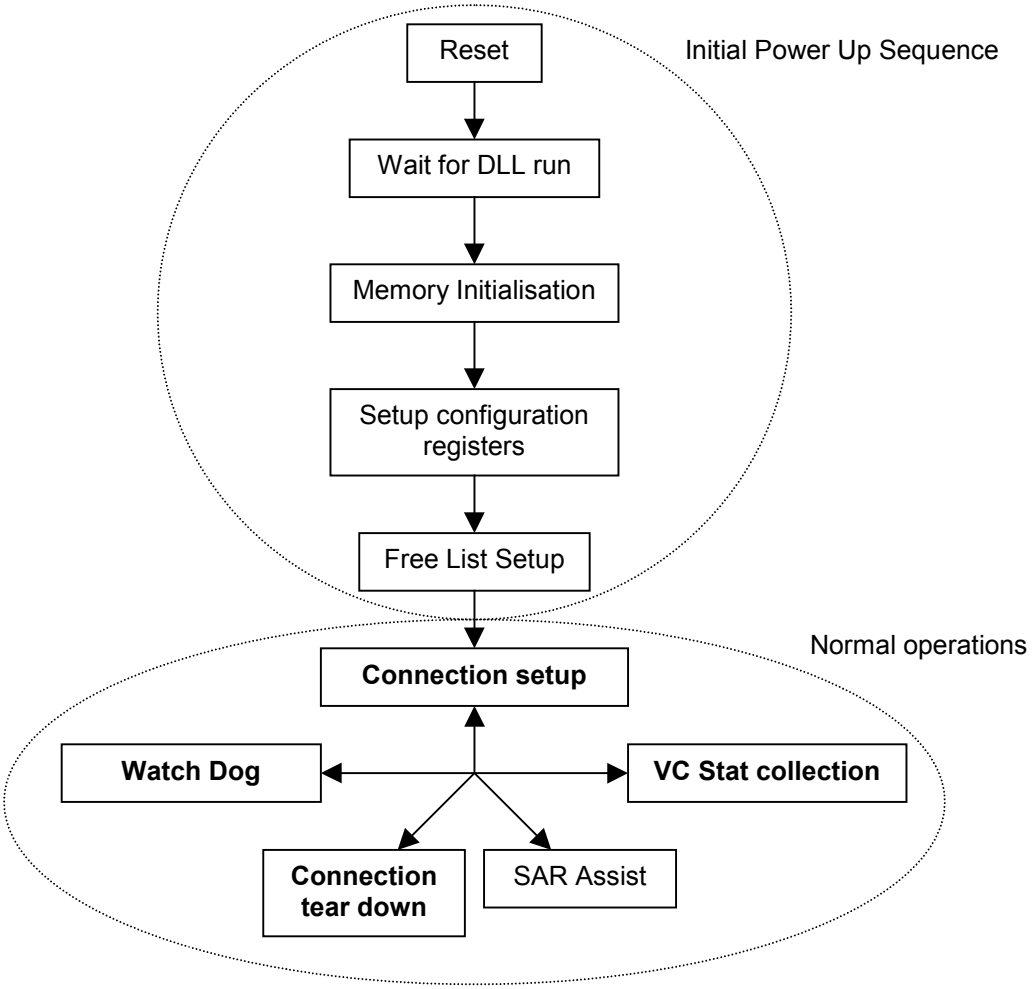
1. PMC Sierra, "S/UNI APEX Data Sheet", PMC-981224
2. PMC Sierra, "Traffic Management and Switching Using the Vortex Chip Set", PMC-981024
3. PMC Sierra, "S/UNI APEX Driver Manual", PMC-991727
4. PMC Sierra, "S/UNI APEX Driver Release Notes", PMC-991578

3 OVERVIEW

This document illustrates the necessary steps that have to be taken to bring the S/UNI APEX into an operational state. In addition, this document provides a step by step guide in using many of the features. This document also serves as a guideline in calculating the many context parameters that have to be set in order to optimize the traffic management features.

Figure 1 is a general flow chart illustrating the steps necessary to configure the S/UNI APEX and bring it into a normal operational state. Each step in the flow chart requires access to the registers and context memory. These procedures are described in the general operations section of this document. The actual steps are discussed in detail in the chapters following the general operations. Finally, steps that are associated with the built in test circuitry are described in the test operations section of the document.

Figure 1 - Operational Flow Chart



4 GENERAL OPERATIONS

4.1 CBI Register Port Access

4.1.1 Writing

Write operations to CBI registers are made through the CBI Register Port. The procedure is as follows:

1. Simply write the address, data, CBIRdWrb = 0 & CBIBusy = 1 into the CBI Register Port.

2. Command is complete when CBIBusy is read back as zero. Subsequent write access to the CBI Register Port will hold the microprocessor bus until the first CBI command is complete

4.1.2 Reading

Read operations from CBI registers are made through the CBI Register Port. The procedure is as follows:

1. Simply write the address, CBIRdWrb = 1 & CBIBusy = 1 into the CBI Register Port.

2. Data is valid on the CBI Register Port when CBIBusy = 0. Subsequent write access to the CBI Register Port will hold the microprocessor bus until the first CBI command is complete.

4.2 Memory Port Access

Memory port access is used to access all internal and external context memory. The procedures outlined below should be used when context memory access is required.

Any access to all context memory will have impact on the S/UNI APEX throughput. The reduction of throughput is a function of the frequency of context access, and the length (MPLWordEn) of each access.

4.2.1 Writing

Write operations to context memory (internal or external) can be performed in up to 4-long word bursts to the memory port. Some context memory can only support single long word access. Refer to specific context memory descriptions for restrictions. The procedure is as follows:

1. The microprocessor polls the MPBusy bit of the Memory Port Control register (or monitors the MPIdleStatus interrupt) to verify that the previous write is complete. Alternatively, this step may be skipped if the system application allows S/UNI APEX to withhold the READYB for write accesses. In this case, S/UNI APEX will delay write operations to the write burst registers and overflow register until the previous write command is complete.

2. The microprocessor writes up to 4 long words of data into the write burst register array and the overflow register (for 34-bit accesses). For non-contiguous bursts (using the long word enables), the write data is always associated with the long word enable. For example, to write words zero and two of a 4-long word region in memory, the long word enables should indicate

MPLWordEn = "0101", and the data should be loaded into the first (index 0) and third (index 2) register of the register array. For masked write, the first word and associated overflow bits contains the data, and the second word and associated overflow bits represent the bit mask. The last two words are not used.

3. The microprocessor writes a command to the memory burst control register. The command indicates the aperture, the quad-long word address in memory, the type of write (masked or unmasked), and the 4 long word enables. At this time the microprocessor sets the MPBusy bit in this register to indicate a new command is available.

4. S/UNI APEX arbitrates for the appropriate memory, performs the write to memory, and clears the MPBusy bit in the control register.

4.2.2 Reading

Reads from context memory (internal or external) can be performed in 4-long word bursts from the memory port. Some context memory can only support single long word access. Refer to specific context memory descriptions for restrictions. The procedure is as follows:

1. The microprocessor issues a read command to the Memory Port Control register. The command indicates aperture, the quad-long word address in memory, and 4 long word enables. At this time the microprocessor sets the MPBusy bit in this register to indicate a new command is available.

2. S/UNI APEX arbitrates for the appropriate memory, performs the read from memory and loads the read burst registers with the results, and clears the MPBusy bit in the control register.

3. The microprocessor polls the MPBusy bit of the Memory Port Control register status (or monitors the MPIdleStatus interrupt) to verify that the read is complete. Alternatively, this step may be skipped if the system application can tolerate long response times for read accesses. In this case, S/UNI APEX will delay read operations from the read burst registers and overflow register until the read command is complete.

4. The microprocessor reads up to 4 long words of data from the read burst register array and the overflow register (for 34-bit accesses). For non-contiguous bursts (using the long word enables), the read data is always associated with the long word enable. For example, to read words second and fourth of a 4-long word region in memory, the long word enables should indicate MPLWordEn = "1010", and the data will be loaded by S/UNI APEX into the second (index 1) and fourth (index 3) register of the register array.

5 INITIAL POWER UP SEQUENCING

5.1 Powering Up Sequence

Please refer to PMC-981224, top of the Pin Description section for details regarding power sequencing.

5.2 DLL Run

The first process that must be carried out is to observe the DLL run status bit, found in the CBI register section, indicates that the internal clocks are aligned, and that the device is ready to continue the configuration process.

5.3 Memory Initialization

Memory initialization requires access via the cell buffer diagnostic access as well as the memory port. The procedure is as follows:

1. Program the SDRAM/SSRAM Configuration register in accordance with the physical memory connected to the S/UNI APEX.

2. (Optional) Using the cell buffer diagnostic access for the SDRAM, and the memory port access for the SSRAM and internal memories, write and then read test patterns on all memories. Check that the SSRAM parity bit does not become active. Read to clear the state of the SDRAMCrcErr. The state is ignored, as it does not apply during diagnostic accesses.

3. Using the memory port access, clear all internal and external context memory. Clearing of the SDRAM is not necessary.

5.4 Setup Configuration Registers

All configuration registers have to be programmed, and are listed out in their recommend sequence below:

1. 0x10x, 0x20x Cell Interface Configuration. Required if an Any-PHY compatible device is attached to the respective pins. Each interface should be enabled prior to the external device if the S/UNI APEX is the bus master. This is to ensure that all the signals on the bus are driven correctly.

2. 0x73x Shaper Configuration. Required if the respective shaper is to be enabled. See the section Configuring the Shaper for guidelines.

3. 0x700 Queue Context Configuration. See the section Configuring Queue Context for guidance on configuring the queue context memory sizes and starting addresses.

5.4.1 Configuring the Shaper

In addition to selecting the desired port and class, the shaper's unit of time has to be programmed with the following 2 formulas satisfied:

QShpNRTRate <= f(SYSCLK) / highest SCR to be shaped.
1/SUM(1/QShpNRTRate) <= 57, where N = 1..4 and the shaper is active.

It is recommended that the smallest QShpNRTRate value be programmed. The faster the shaper operates, the higher the shaper's resolution and the lower the CDV.

If throughput is more important than fairness, or if congestion/over-subscription is not to be encountered, set

QShpSlowDownEn = 0

If fairness is more important than throughput, and congestion is expected, set

QShpSlowDownEn =1
QShpNThrshEn = 0
QShpNMeasInt = 3
QShpNRedFact = 0

QShpNMeasInt: Longer interval values reduce the chance that congestion is declared; thereby maintaining throughput but provides less fairness. Values that are too small may increase the likely hood of misinterpreting contention for congestion; hence cause the slow down to engage prematurely.

QShpNRedFact: Smaller values give better fairness. Larger values will improve throughput, but may cause instability by attempting to return to the fundamental time unit too quickly.

5.4.2 Configuring Queue Context Memory

Choosing whether to support 16K or 64K VC, or address re-mapping, and selecting the start address for the loop, shaping and cell context depends on the specific requirements as well as how much SDRAM and SSRAM is available. Using the table below, determine the amount of memory required for each context memory group. Arrange the start of each context group to the nearest Kbytes resolution without overlapping. Start over or add more memory if there is

insufficient memory. Once all the context memory requirements have been determined, the start address can then be easily calculated.

Table 2 Determining Context Memory Size

Context Memory	Space (bytes)	Comment
VC	#VC * 24	# VC = 16K or 64K
Address Map	0, or #VC * 4	Zero if address re-mapping is not enabled (VcReMapMode = 00 for all VCs). # VC = 16K or 64K
Loop Class (QLClassStartAdr)	128K	Context space required at all times, even if no traffic is destined for a loop port.
Shaper (QShpStartAdr)	0 or (128K + (#VC * 8))	Zero if shaping not enabled.
Cell (QCellStartAdr)	#Cells * 4	Upper limit for #cells determined by SDRAM size. A cell takes up 64 bytes of SDRAM memory, independent of the number of Rx & Tx prepends.

Table 3 and Table 4 illustrate two examples of possible memory configurations and their associated memory map and starting address values. Table 3 is based on 16K VC's, no address re-mapping, the shapers are off and 128K cell buffers. Table 4 is based on 64KVC's, address re-mapping, the shapers are enabled and 256K cell buffers.

Table 3 Determining Queue Start Addresses Example #1

Context Record	Size (bytes)	MPQuadAddr	Comments
VC Context	262144	0 – 16383 (0x0000 – 0x3FFF)	16K VC's
VC Statistic	131072	16384 – 24575 (0x4000 – 0x5FFF)	
VC Address			Not Enabled
Loop Class 0	32768	24576 – 26623 (0x6000 – 0x67FF)	QLClassStartAdr = 24 (0x18)
Loop Class 1	32768	26624 – 28671 (0x600 – 0x6FFF)	
Loop Class 2	32768	28672 – 30719 (0x7000 – 0x77FF)	
Loop Class 3	32768	30720 –32767 (0x7800 – 0x7FFF)	

Context Record	Size (bytes)	MPQuadAddr	Comments
Shaper 0 TxSlot Shaper 1 TxSlot Shaper 2 TxSlot Shaper 3 TxSlot Shape Rate			Not Enabled QShpStartAdr = 0
Cell Record	524288	32768 – 65535 (0x8000 – 0xFFFF)	128K Cell Buffers QCellStartAdr = 32 (0x20)

Table 4 Determining Queue Start Addresses Example #2

Context Record	Size (bytes)	MPQuadAddr	Comments
VC Context	1048576	0 – 65535 (0x00000 – 0x0FFFF)	64K VC's
VC Statistic	524288	65536 – 98303 (0x10000 – 0x17FFF)	
VC Address	262144	98304 – 114687 (0x18000 – 0x1BFFF)	Enabled
Loop Class 0 Loop Class 1 Loop Class 2 Loop Class 3	32768 32768 32768 32768	114688 – 116735 (0x1C000 – 0x1C7FF) 116736 – 118783 (0x1C800 – 0x1CFFF) 118784 – 120831 (0x1D000 – 0x1D7FF) 120832 – 122879 (0x1D800 – 0x1DFFF)	QLClassStartAdr = 112 (0x70)
Shaper 0 TxSlot Shaper 1 TxSlot Shaper 2 TxSlot Shaper 3 TxSlot Shape Rate	32768 32768 32768 32768 524288	122880 – 124927 (0x1E000 – 0x1E7FF) 124928 – 126975 (0x1E800 – 0x1EFFF) 126976 – 129023 (0x1F000 – 0x1F7FF) 129024 – 131071 (0x1F800 – 0x1FFFF) 131072 – 163839 (0x20000 – 0x27FFF)	Enabled QShpStartAdr = 120 (0x78)
Cell Record	1048576	163840 – 229375 (0x28000 – 0x37FFF)	256K Cell Buffers QCellStartAdr = 160 (0xA0)

5.5 Setup Context Memory

Once all the configuration registers have been programmed, it is then necessary to create the free list for the cell buffer, as well as initialize the remaining context information.

Procedure:

Let us assume that the desired length of the free list is MaxCellBufs. The value of MaxCellBufs depends on the size of cell buffer and context memory available. The maximum value of MaxCellBufs supported by the APEX is 256K.

1. Program first cell record's (located at CellStartAdr) CellNextPtr parameter as 0. Note that this cell record is reserved. Starting from the 2nd cell record,

program each cell record's CellNextPtr parameter with a count value that is initially 2 and increments by one after programming each CellNextPtr. The 2nd last cell in the link list will have it's CellNextPtr = (MaxCellBufs-1). Finally, the last cell's CellNextPtr should be set to 0 to indicate the end of the link list.

2. Program the FreeListHeadPtr with 1, the FreeListTailPtr with(MaxCellBufs-1), and the FreeCnt with (MaxCellBufs –1).

3. Clear the overall count context structure, congestion discard context structure, maximum congestion ID context structure, and miscellaneous error context structure.

4. Clear all records falling under the WAN Port Scheduler Context.

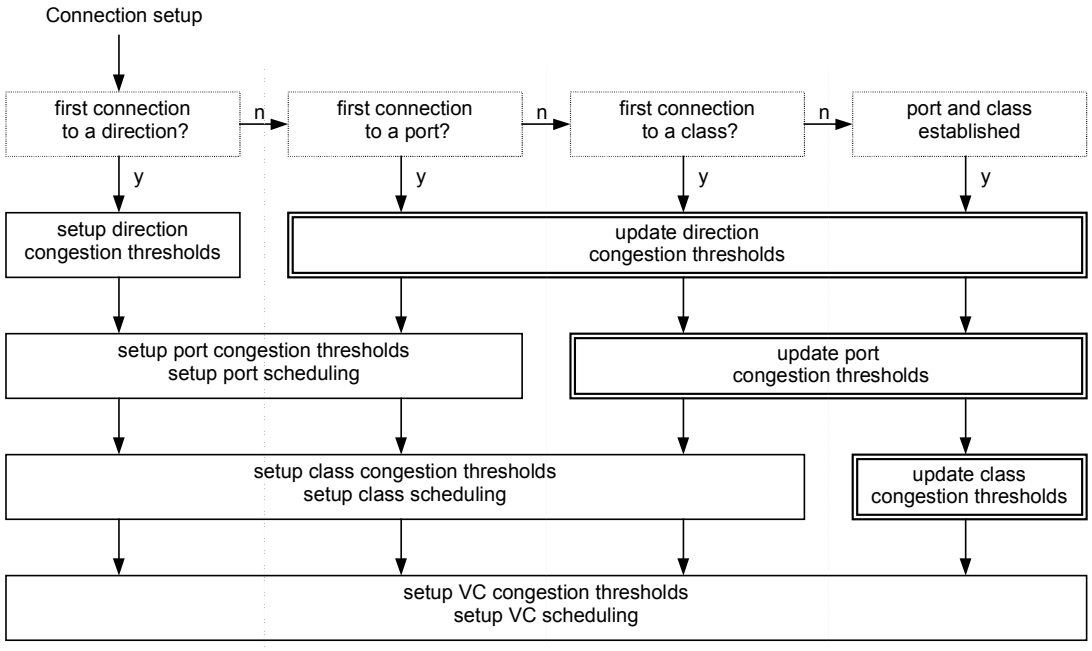
5. Clear all records falling under the Loop Port Scheduler Context.

6 NORMAL OPERATIONS

6.1 Connection Setup

Before any traffic can go through the S/UNI APEX, a connection has to be established. Figure 2 illustrates one way of establishing a connection setup. Each step is given in detail in the following sections. A section is devoted entirely to the programming of the class, port and direction congestion thresholds.

Figure 2 - Connection Setup Flow Chart



6.1.1 Direction/Port/Class Congestion Settings

Rules:

1. 256K – 1 >= initial FreeCnt >= DirMaxThrsh >= DirCLP0Thrsh >= DirCLP1Thrsh, "Dir" refers to either Loop or WAN

2. 256K – 1 >= initial FreeCnt >= PortMaxThrsh >= PortCLP0Thrsh >= PortCLP1Thrsh

3. 256K – 1 >= initial FreeCnt >= ClassMaxThrsh >= ClassCLP0Thrsh >= ClassCLP1Thrsh

ClassMaxThrsh = SUM(VcMinThrsh) * W_c
PortMaxThrsh = SUM(ClassMaxThrsh) * W_p
DirMaxThrsh = SUM(PortMaxThrsh) * W_d, where

W < 1: apply statistical MUX factor on minimum "guarantee"
W = 1: maintain guarantee
W > 1: provide some resource to UBR/ABR under heavy congestion.

ClassCLP0Thrsh = SUM(VcCLP0Thrsh) * Y_c
PortCLP0Thrsh = SUM(ClassCLP0Thrsh) * Y_p
DirCLP0Thrsh = SUM(PortCLP0Thrsh) * Y_d
ClassCLP1Thrsh = SUM(VcCLP1Thrsh) * Z_c
PortCLP1Thrsh = SUM(ClassCLP1Thrsh) * Z_p
DirCLP1Thrsh = SUM(PortCLP1Thrsh) * Z_d, where

Y,Z < 1: apply statistical MUX factor. Lower values reduces fair allocation of resources under congestion.
Y,Z = 1: no statistical MUX
Y,Z > 1: wastes resources and it not recommended.

6.1.2 Setting Up a Port

Procedure:

1. Program the class scheduler context to specify the manner in which the member classes of the port will be scheduled. The class scheduler parameters may be set up completely at this time. Alternatively, they may be updated each time a new member class is set up.

2. For WAN ports, program the WAN transmit port polling weight record, and clear the respective WANClassStat parameters in the WAN Transmit Class Status record. See the section Setting WAN Port Scheduler Context for guidance in setting the weight records

3. For loop ports, program the Loop transmit port polling weight & sequence record, and clear the respective LoopClassStat parameters in the Loop Transmit Class Status record. See the section Setting Loop Port Scheduler Context for guidance in setting the weight & sequence record.

4. Program the port threshold context followed by the port count context using the section on class/port/direction congestion settings for guidance in setting the port congestion thresholds. Make sure the PortCnt parameter is cleared. Set the PortEn bit to 1.

6.1.2.1 Setting WAN Port Scheduler Context

Setting the WAN poll weight for a given port determines when cells are scheduled into its port FIFO. The weighting value represents a relative frequency a port is serviced. Each port FIFO has an equal opportunity to be scheduled out the WAN interface in a round robin fashion. The system designer must be aware of two possible results when assigning WAN poll weights to each port.

If the APEX queue engine's available bandwidth is less then or equal to the WAN bandwidth, then the ports are serviced based on the WIRR values. Therefore, the WAN poll weight ensures scheduling fairness between ports of different rates. For example, one would weight a port that is 3 times faster than another port with a weight that would ensure the queue engine would service the first port 3 times as often. This might occur when the WAN interface is competing with the loop interface for queue engine bandwidth.

Alternatively, if the APEX available queue engine's bandwidth is more than the WAN bandwidth, i.e., the queue engine is able to keep the port FIFOs full, then the ports are serviced in a simple round robin fashion.

6.1.2.2 Setting Loop Port Scheduler Context

Setting the loop poll weight and assigning the sequence for a given port determines its place in the polling schedule relative to all other loop ports. The weighting value represents a relative frequency of the number of times a port is polled, and the corresponding relative data throughput. Lower poll weights have higher polling frequency, and give a port a higher throughput. Table 5 shows the maximum polling frequency as a function of the loop poll weight. Note that the maximum polling frequency of loop poll weights 0 & 1 are greater than the maximum data throughput of 0.56Mcells/s to a single port. This is due to the separation of the polling mechanism from the data path.

Table 5 Loop Polling Frequency, SYSCLK = 80MHz

LoopPollWght	Max. Polling Freq. (MHz)
0	1.25
1	0.625
2	0.313
3	0.156
4	0.078
5	0.039
6	0.019
7	0.01

To prevent the need to update the entire loop port scheduler context whenever a port is enabled/disabled, one should create a look up table in advance that matches a range of port data rates to a given loop poll weight.

For example, a given system has support for port rates ranging from 400Kcells/s on down. A possible look up table can be mapped as follows:

Table 6 Example look up table

LoopPollWght	Port Rate (Kcells/s)
0	400-350
1	349-300
2	299-150
3	149-75
4	74-40
5	39-20
6	19-10
7	9-0

Notice that the port rate range never exceeds the max. polling frequency. This ensures that if the number of active ports is low, all ports will be serviced at their maximum throughput.

Once the poll weight has been determined, a sequence number has to be assigned. Recall that only the n LSB of the sequence number is significant, where n is the loop poll weight. To minimize CDV impact on the higher rate ports, maintain an even and wide distribution of port sequence assignments across the valid range of sequence numbers. For example, if loop poll weight of 2 has 10 ports, then an ideal sequence distribution would be:

Table 7 Example Sequence Distribution

LoopPollSeq	# Ports assigned
0	2
1	3
2	2
3	3

In the case where only a few ports are enabled within a weight class, ensure that the sequence numbers assigned are as far apart from each other as possible. For example, for weight class 7, if only 2 ports are enabled, the optimal sequence number assignment would be 0 and 63.

6.1.3 Setting up a Class

Procedure:

1. If a Class is being recycled, make sure that the class queues are cleared of all cells from the previous setup. This can be verified by performing a Class tear down and waiting for ClassCnt = 0 before proceeding.

2. If class will be shaped, clear all the Shape TxSlot context records for the associated shaper.

3. Program the Class context record in lword 0, clear all bits for lword 1,2. See the section on class/port/direction congestion settings for guidance in setting the class congestion thresholds.

4. Update the class scheduler context parameters (OPTIONAL)

6.1.3.1 Cell and Packet Class Scheduling

To calculate the minimum bandwidth a class gets the following formula can be used:

MinClassRate = PortRate / (1+ClassXCellLmt)

Table 8 ClassXCellLmt field setting

ClassXCellLmt Field	Effective Count	% Usage
0	0	Strict priority with classes above
1	1	50.00%
2	2	33.33%
3	3	25.00%
4	4	20.00%
5	5	16.67%
6	6	14.28%
7	7	12.50%
8	8	11.11%
9	10	9.09%
10	12	7.69%
11	14	6.66%

ClassXCellLmt Field	Effective Count	% Usage
12	16	5.88%
13	20	4.76%
14	24	4.00%
15	28	3.44%

The ClassXCellLmt field has a 2 bit log, 2 bit fractional granularity. This allows the lowest rate supported by the class scheduler to be 1/28 of the port rate (1.85 Mbps for a 52 Mbps port rate).

To calculate the value of ClassXCellLmt, use the formula below to determine the effective count value, and then use Table 8 to cross reference to the ClassXCellLmt value

Effective Count Value = (PortRate / MinClassRate) – 1

For uP class scheduling, it is recommended that the class scheduler be set up to perform round robin (all ClassXCellLmt = 3). This is to ensure that cells queued on lower class queues become visible to the uP.

When ClassPacket = 1, the class scheduler must be set up to perform strict priority (all ClassXCellLmt = 0).

6.1.4 Setting up a VC

Procedure:

1. If a VC is being recycled, make sure that the VC and class queues are cleared of all cells from the previous VC. This can be verified by performing a VC tear down and waiting for VcClassQCLP01Cnt = 0 before proceeding.

2. Set VC Address Map if enabled.

3. Clear VC Statistic (lword 0, 1 or lword 2,3 depending if ICI is even or odd).

4. If shaping to the destination port/class has been enabled, program the following Shape Rate context below with the ATMF TM 4.1 parameters (SCR, PCR, MBS):

   ShpIncr = f(SYSCLK) / QShpNRTRate / SCR

ShpCdvt = ShpIncr - f(SYSCLK) / QShpNRTRate / PCR[1]
ShpLateBits = log (MBS * ShpCdvt) / log (2)

If PCR, and MBS are not defined parameters (e.g. CBR traffic), one should set ShpLateBits = 0xf, and the ShpCdvt set to the CDVT of the connection. Setting ShpCdvt to zero will effectively remove any possibility of recovering lost transmit opportunities due to contention/congestion and thereby reduce throughput.

Clear the other parameters to zero (odd lword).

5. Set VC Context configuration parameters (lword 0, lword 1) using the guidelines outlined below. Clear all the other parameters to zero (lword 2 & 3).

6.1.4.1 Per-VC Congestion Thresholds

Rules:

1. 8k – 1 >= VcMaxThrsh >= VcCLP0Thrsh >= VcMinThrsh >= VcCLP1Thrsh

2. If tagging is not applicable for the traffic flow, then VcCLP1Thrsh should be set to the same value as VcCLP0Thrsh. If tagging is applicable, then the values stated in the table below should be adhered.

Table 9 provides suggested values to be programmed into the various congestion thresholds. Specific applications and needs should take precedence over the suggested values.

Table 9 VC context

Traffic	VcCLP1 Thrsh	VcMin Thrsh	VcCLP0 Thrsh	VcMaxThrsh (EPD/PPD only)	Comment
CBR	maxCTD	maxCTD	maxCTD	maxCTD	No value storing more than the maxCTD. Tagging not applied to this traffic; hence CLP0 & CLP1 Thrsh values are identical.
rt-VBR	maxCTD	maxCTD	maxCTD	maxCTD	From congestion perspective, rt-VBR and CBR are identical. Difference lies in CDV tolerance reflected in class scheduling.

[1] Note that the inter-cell arrival times coming out of the S/UNI APEX can be affected by the 4 cell deep WAN FIFO, interaction of the other WAN ports via the WAN Port Scheduler, and interaction of the traffic destined for a loop port. PCR should be programmed with these factors in mind.

Traffic	VcCLP1 Thrsh	VcMin Thrsh	VcCLP0 Thrsh	VcMaxThrsh (EPD/PPD only)	Comment
nrt-VBR	MBS * (PCR – SCR) / PCR	MBS	MBS * n, n > 1	VcCLP0Thrsh + est. MFS	Minimum CLR is the focus. Always want sufficient resources to capture a burst. VcCLP1Thrsh set to a level where the VC is within traffic contract. VcCLP0Thrsh set to a level where the VC has some burstiness caused by the network. VcMaxThrsh set to accept the last frame permitted under VcCLP0Thrsh.
GFR	MBS * (PCR-MCR) / PCR	MBS	MBS * n, n > 1	VcCLP0Thrsh + MFS	Very similar to nrt-VBR, except packet centric.
UBR	1	0	est. MFS	VcCLP0Thrsh + est. MFS	No minimum resource guarantees.
ABR	1	0	1	est. MFS + 1	The CLP0 & 1 thresholds are kept to very small values, relying on the connection s/w to regulate traffic rates and avoid congestion.

6.1.4.2 Per VC WFQ Scheduling

WFQ is only available when the destination port/class is not shaped.

The VcWght parameter represents the number of transmit opportunities that a given WFQ VC has relative to other VCs within the same class. A maximum range of 126 units, with a granularity of 2 units is provided.

The resulting CDV is sensitive to the largest VcWght assigned in a given class. Larger values will increase the probability of clumping; however, a larger range of VcWght provides better granularity. One must consider the trade off between granularity and CDV.

Table 10 gives three possible formulas when determining the value to be programmed into the VC queue weight.

Table 10 Determining VC Queue Weight

Optimize	Queue Weight	Comment
None	126 * VC_PCR / Port_Rate	Simplest setting used when the limit of class PCR is not known in advanced. Poor CDV, poor granularity.

Optimize	Queue Weight	Comment
Granularity optimized	126 * VC_PCR / Max_Class_PCR	Provides the best granularity between VCs within a class. Can only be applied when the Max_Class_PCR is known ahead of time.
CDV optimized	126 * VC_PCR / Max_Class_PCR * Z, Z < 1	Provides the best CDV by scaling VcWght across all traffic within a class by the factor Z, where Z is a real number less than 1.

where:

VC_PCR: PCR of the VC

Port_Rate: destination port rate of the VC.

Max_Class_PCR: highest PCR value expected for a VC within the class

Min_Class_PCR: lowest PCR value expected for a VC within the class

Actual value programmed in the context parameter, VcWght:

```
if (Queue Weight = 1)
  VcWght = 0
else
  VcWght = Queue Weight / 2
```

6.1.4.3 Per VC Class Type

Selecting a class for a given connection is based primarily on connection's sensitivity to CDV. The lower a connection's tolerance to CDV, the higher its class designation should be. Table 11 recommends class designation for standard traffic types.

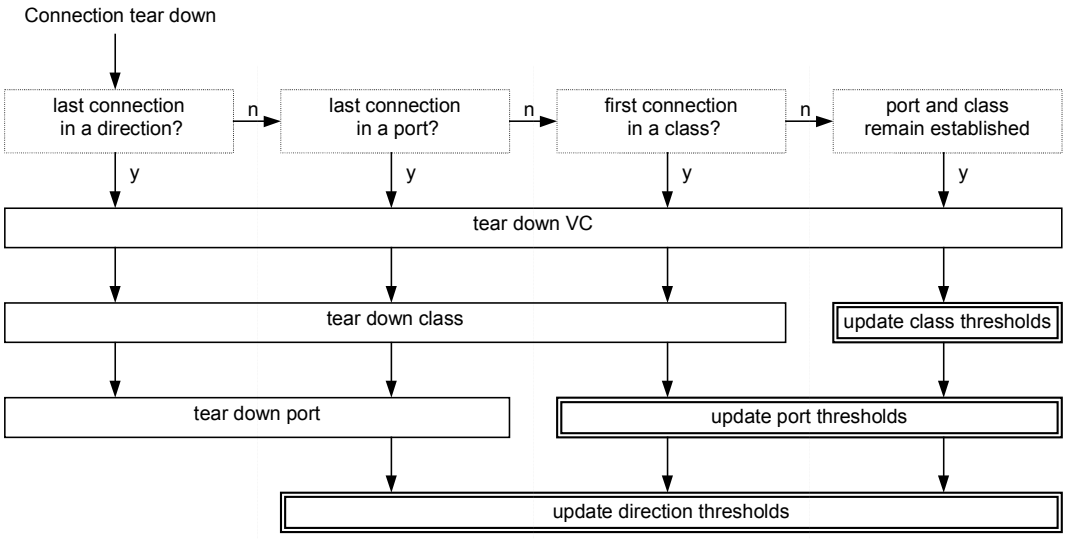
Table 11 Class recommendation for different traffic types

	CBR	rt-VBR	nrt-VBR	GFR	UBR	ABR
Class	1	2	3	3	4	4

6.2 Connection Tear Down

Once a connection is no longer required, the connection should be torn down and the resources reclaimed for other connections. Figure 3 show the steps required to tear down a connection. Each step is given in detail in the following sections. Guidance for updating of the congestion thresholds can be found under the connection setup.

Figure 3 - Connection Tear Down Flow Chart



6.2.1 Tearing Down a VC

Procedure:

Method A:

This is the fastest method to recover resources from a VC that is no longer required for service.

1. Set the VC context parameter VcEn = 0.
2. Set QTdICI register to the ICI that is to be torn down.
3. Set QTdMode = 0 & QTdActive = 1. The VC queue will be empty when QTdActive is read back as 0; however, there may be a few cells remaining in the class queue.

Method B:

This is the easiest method to recover resources from a VC that is no longer required for service. Restriction is that the destination port must still be operational. Out of service ports require method A for recovery.

1. Set the VC context parameter VcEn = 0 & VcIntDis = 1(optional). This will stop all incoming cells associated with the VC from entering the queue structure. The cells in the queue structure will drain as they normally would. Turning off or masking the interrupt will prevent unnecessary warnings caused by cells arriving at a disabled VC.

2. A VC is completely “torn down” when the VC context parameters VcQCLP01Cnt = 0 and VcClassQCLP01Cnt = 0.

6.2.2 Tearing Down a Class

Procedure:

1. For shaped class, program QShpSlowDownEn = 0 to keep the fundamental time unit constant and running at full speed.

2. For each VC associated with the class follow the procedure to tear down a VC. Method A must be used if the destination port is incapable of receiving any cells.

3. Set Port Class context parameter ClassEn = 0.

4. Set QTdClassID and QTdPortID registers. For shaped class, wait (4096*fundamental time unit) to allow the slot table to deplete itself completely into the classQ.

5. Set QTdMode = 1 & QTdActive = 1. The class queue will be empty when QTdActive is read back as 0.

6. For loop port, clear the appropriate class status bit in the internal loop context class status record. For WAN port, clear the appropriate class status bit in the internal WAN context class status record.

7. A class is completely "torn down" when the Port Class context parameter ClassCnt = 0.

6.2.3 Tearing Down a Port

Procedure:

1. For each class associated with the port, follow the procedure to tear down a class.

2. Wait for PortCnt = 0. Set the Port Count context parameter PortEn = 0.

6.3 Watch Dog

Procedure:

1. Make sure targeted VCs have the VcWDEn bit is set to one.

2. Set Reg. 0x720 Watch Dog ICI Patrol Range registers QWdEndICI & QWdStartICI.

3. Check QWdActive is zero, indicating that no watch dog patrol is in progress. Set the QWdActive bit to one. The watch dog will now walk through once the range of ICIs as selected in step 2. The earliest time period it takes for the watch dog to complete its patrol is

   Min Time for patrol = 48 * (QWdEndICI – QWdStartICI) / f(SYSCLK)

4. After a predetermined time period, repeat step 3. One should not initiate another patrol until the first patrol is complete. The period should be equal to or greater than the longest expected time period between receiving a BOM and receiving an EOM of all FCQ VC.

6.4 SAR Assist

6.4.1 Injecting Single Cells

The steps below outline the procedure for injecting single OAM or non-OAM cells.

1. Poll the SarRxRdyStatus bit in the interrupt status register to determine if the SAR buffer can accept a cell. Alternatively, the low priority interrupt pin may be monitored. This step may also be skipped entirely; however the next step will cause S/UNI APEX to use the READYB output to stall the microprocessor bus until the SAR buffer can accept data.

2. Write the CRC control information, ICI, cell header, and payload to the SAR write staging buffer. The CRC control information is located in the first word of the buffer and includes the following controls:

   - Initiate CRC-32 calculation (for start of frame).

- Overwrite the end of the payload with a CRC-32 (for end of frame).
- Overwrite the end of the payload with a CRC-10 (for OAM cells).

Note that once the last word of the payload is written, the cell is advanced in the 2-cell pipeline. Therefore, this word must be written after all other information has been loaded into the buffer (as would intuitively be the case).

If this is an OAM cell, then the Overwrite CRC-10 control should be activated.

6.4.2 Injecting AAL5 Frames

The steps below outline the procedure for injecting AAL5 frames.

1. Poll the SarRxRdyStatus bit in the interrupt status register to determine if the SAR buffer can accept a cell. Alternatively, the low priority interrupt pin may be monitored. This step may also be skipped entirely; however the next step will cause S/UNI APEX to use the READYB output to stall the microprocessor bus until the SAR buffer can accept data.
2. Write the CRC control information, ICI, cell header, and payload to the SAR write staging buffer. The CRC control information is located in the first word of the buffer and includes the following controls:
   - Initiate CRC-32 calculation (for start of frame)
   - Overwrite the end of the payload with a CRC-32 (for end of frame)
   - Overwrite the end of the payload with a CRC-10 (for OAM cells)

   Note that once the last word of the payload is written, the cell is advanced in the 2-cell pipeline. Therefore, this word must be written after all other information has been loaded into the buffer (as would intuitively be the case). Note that the cell header must be updated for the first two cells and last cell of the frame. For middle cells of a frame, the cell header does not need to be written every time.

   If this is the end of message, then the microprocessor must fill in padding and the AAL5 length in the payload portion of the buffer. This may be necessary even if this is not the end of message cell, as the 8-byte AAL5 trailer may force another cell.

   If the cell represents the start of a frame, the Initiate CRC-32 control should be activated. If this is the end of message, then the CRC-32 overwrite control should be activated.

3. If the end of message has not been reached, then repeat at step 1.

6.4.3 Reading Cells and AAL5 Frames

A similar mechanism is used for reading single cells and reading an AAL5 frame. The steps below outline the procedure.

1. Poll the SarTxRdyStatus bits in the interrupt status register to determine if any of the four SAR buffers have a cell available. Alternatively, the low priority interrupt pin may be monitored. This step may also be skipped entirely; however the next step will cause S/UNI APEX to use the READYB output to stall the microprocessor bus until the SAR buffer has valid data.

   If multiple buffers have valid data, then the microprocessor may choose which buffer to service. This allows the microprocessor to implicitly set priorities between the four read buffers (and four microprocessor class queues).

2. Read the CRC status information, ICI, cell header, and payload from a SAR read staging buffer. The CRC status information is located in the first word of the buffer and includes the following status:

   - CRC-32 error (valid at end of message, as indicated by PTI)
   - CRC-10 error (valid for OAM cells, as indicated by the cell header)

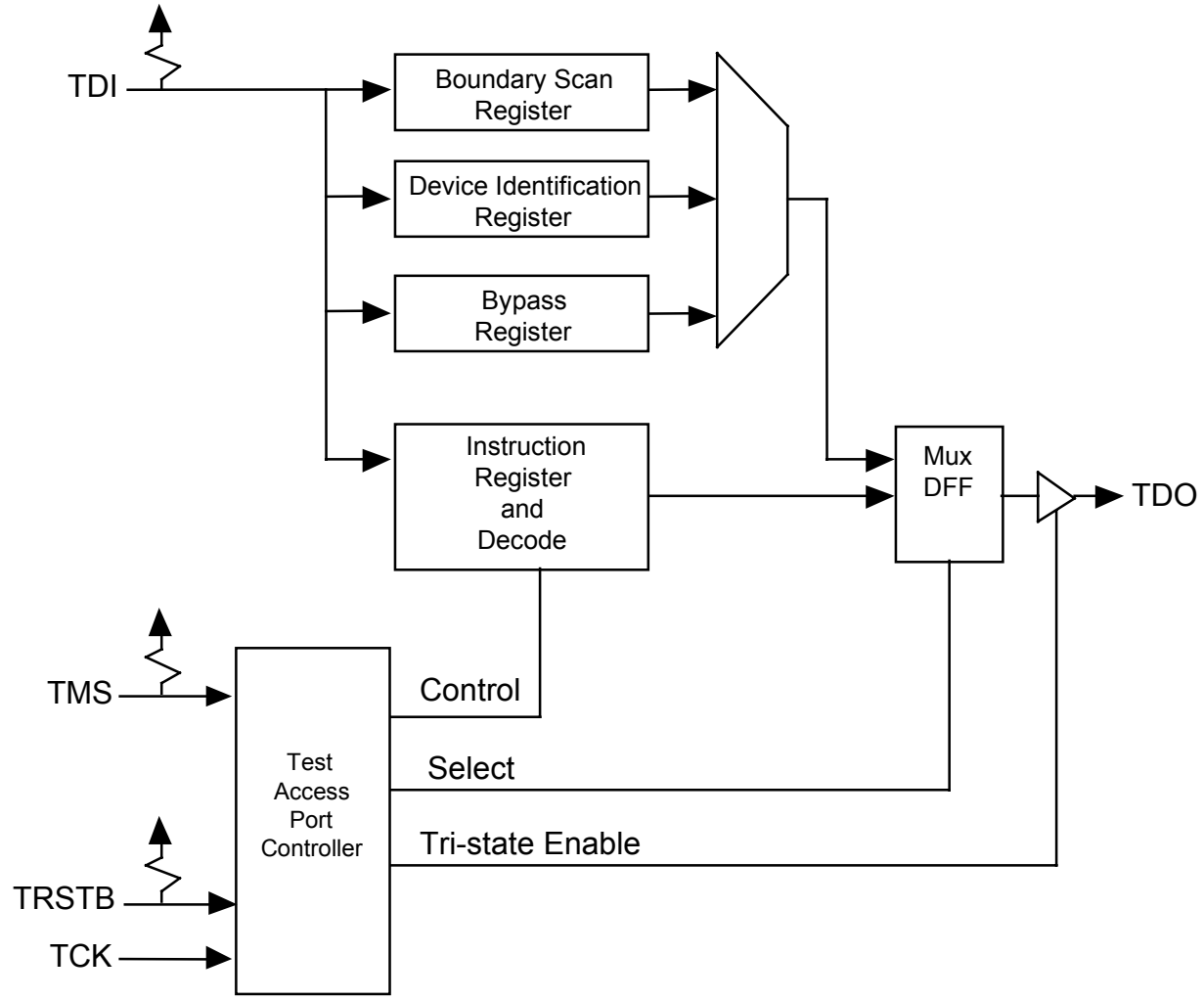
   Note that once the last word of the payload is read, the cell is advanced in the 2-cell pipeline. Therefore, this word must be read after all other information has been retrieved from the buffer (as would intuitively be the case).

7 TEST OPERATIONS

7.1 JTAG Support

The S/UNI APEX supports the IEEE Boundary Scan Specification as described in the IEEE 1149.1 standards. The Test Access Port (TAP) consists of the five standard pins, TRSTB, TCK, TMS, TDI and TDO used to control the TAP controller and the boundary scan registers. The TRSTB input is the active-low reset signal used to reset the TAP controller. TCK is the test clock used to sample data on input, TDI and to output data on output, TDO. The TMS input is used to direct the TAP controller through its states. The basic boundary scan architecture is shown below.

Figure 4 - Boundary Scan Architecture



The boundary scan architecture consists of a TAP controller, an instruction register with instruction decode, a bypass register, a device identification register and a boundary scan register. The TAP controller interprets the TMS input and generates control signals to load the instruction and data registers. The instruction register with instruction decode block is used to select the test to be executed and/or the register to be accessed. The bypass register offers a single-bit delay from primary input, TDI to primary output, TDO. The device identification register contains the device identification code.

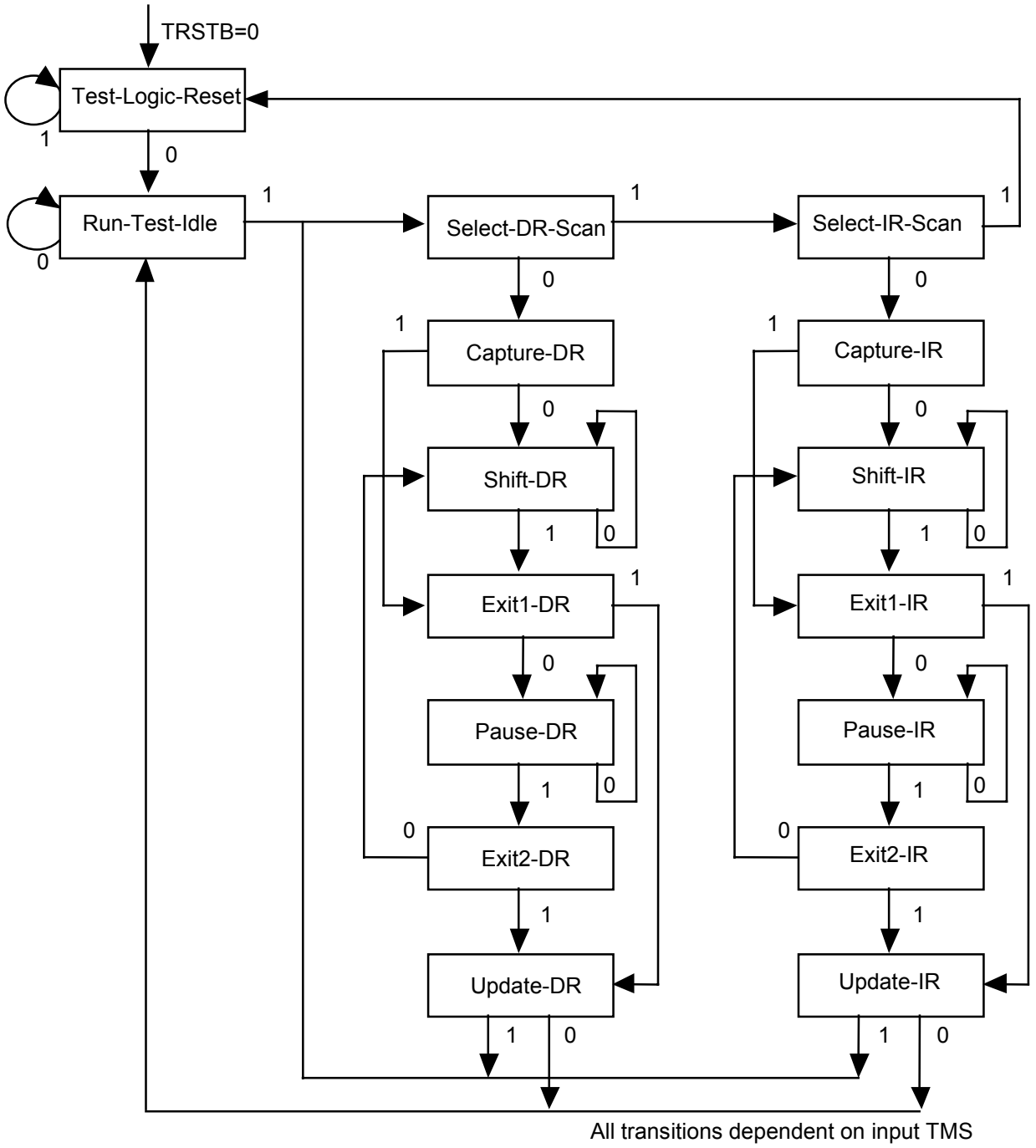
The boundary scan register allows testing of board inter-connectivity. The boundary scan register consists of a shift register place in series with device inputs and outputs. Using the boundary scan register, all digital inputs can be

sampled and shifted out on primary output, TDO. In addition, patterns can be shifted in on primary input, TDI and forced onto all digital outputs.

7.1.1 TAP Controller

The TAP controller is a synchronous finite state machine clocked by the rising edge of primary input, TCK. All state transitions are controlled using primary input, TMS. The finite state machine is described below.

Figure 5 - TAP Controller Finite State Machine



Test-Logic-Reset

The test logic reset state is used to disable the TAP logic when the device is in normal mode operation. The state is entered asynchronously by asserting input, TRSTB. The state is entered synchronously regardless of the current TAP controller state by forcing input, TMS high for 5 TCK clock cycles. While in this state, the instruction register is set to the IDCODE instruction.

Run-Test-Idle

The run test/idle state is used to execute tests.

Capture-DR

The capture data register state is used to load parallel data into the test data registers selected by the current instruction. If the selected register does not allow parallel loads or no loading is required by the current instruction, the test register maintains its value. Loading occurs on the rising edge of TCK.

Shift-DR

The shift data register state is used to shift the selected test data registers by one stage. Shifting is from MSB to LSB and occurs on the rising edge of TCK.

Update-DR

The update data register state is used to load a test register's parallel output latch. In general, the output latches are used to control the device. For example, for the EXTEST instruction, the boundary scan test register's parallel output latches are used to control the device's outputs. The parallel output latches are updated on the falling edge of TCK.

Capture-IR

The capture instruction register state is used to load the instruction register with a fixed instruction. The load occurs on the rising edge of TCK.

Shift-IR

The shift instruction register state is used to shift both the instruction register and the selected test data registers by one stage. Shifting is from MSB to LSB and occurs on the rising edge of TCK.

Update-IR

The update instruction register state is used to load a new instruction into the instruction register. The new instruction must be scanned in using the Shift-IR state. The load occurs on the falling edge of TCK.

The Pause-DR and Pause-IR states are provided to allow shifting through the test data and/or instruction registers to be momentarily paused.

Boundary Scan Instructions

The following is a description of the standard instructions. Each instruction selects a serial test data register path between input, TDI and output, TDO.

BYPASS

The bypass instruction shifts data from input, TDI to output, TDO with one TCK clock period delay. The instruction is used to bypass the device.

EXTEST

The external test instruction allows testing of the interconnection to other devices. When the current instruction is the EXTEST instruction, the boundary scan register is place between input, TDI and output, TDO. Primary device inputs can be sampled by loading the boundary scan register using the Capture-DR state. The sampled values can then be viewed by shifting the boundary scan register using the Shift-DR state. Primary device outputs can be controlled by loading patterns shifted in through input TDI into the boundary scan register using the Update-DR state.

SAMPLE

The sample instruction samples all the device inputs and outputs. For this instruction, the boundary scan register is placed between TDI and TDO. Primary device inputs and outputs can be sampled by loading the boundary scan register using the Capture-DR state. The sampled values can then be viewed by shifting the boundary scan register using the Shift-DR state.

IDCODE

The identification instruction is used to connect the identification register between TDI and TDO. The device's identification code can then be shifted out using the Shift-DR state.

STCTEST

The single transport chain instruction is used to test out the TAP controller and the boundary scan register during production test. When this instruction is the current instruction, the boundary scan register is connected between TDI and TDO. During the Capture-DR state, the device identification code is loaded into the boundary scan register. The code can then be shifted out output, TDO using the Shift-DR state.

7.2 Cell Buffer Diagnostic Access

Simultaneous reads & writes to the cell buffer in diagnostic mode is not possible.

Performing diagnostic write access while in service may corrupt existing cell data in transit.

Performing diagnostic access while in service will reduce throughput.

7.2.1 Diagnostic Writes to Cell buffer

Write operations to cell buffer are performed in 16-long word bursts to the receive SAR buffer. The procedure is as follows:

1. The microprocessor polls for the SarRxEmptyStatus = 1.
2. The microprocessor then sets SarDiagWrModeEn = 1, along with setting the address location SarDiagAddr[17:0].
3. The microprocessor writes 16 long words of data into the receive SAR buffer. Once the last word has been written, the process of writing the data into the SDRAM will be initiated.
4. If another diagnostic write operation to the cell buffer is required, repeat steps 1->3.
5. If a diagnostic write operation is no longer required, the microprocessor polls for the SarRxEmptyStatus = 1, and then clears SarDiagWrModeEn =0.

7.2.2 Diagnostic Reads from Cell buffer

Read operations from cell buffer are performed in 16-long word bursts using the class 3 transmit SAR buffer. The procedure is as follows:

1. The microprocessor sets the SarDiagRdModeEn = 1 bit in the cell buffer diagnostic control register.

2. The microprocessor should then service all cells for the class 3 transmit SAR buffer until the SarDiagRdModeLock bit in the test cell buffer control register is set, indicating that the class 3 transmit SAR buffer has been emptied and is ready for diagnostic access.

3. The microprocessor writes a command to the test cell buffer control register. The command indicates the 16-long word address. At this time the microprocessor sets the SarDiagRdBusy bit in this register to initiate the diagnostic read.

4. The microprocessor may attempt to read the SAR transmit buffer immediately (microprocessor will be held in wait state until the data is valid), or read the SAR transmit buffer after receiving the interrupt SarTxRdyStatus = 1 in the Low Priority Interrupt Status register, or read the SAR transmit buffer after polling for the SarDiagRdBusy = 0. The microprocessor reads 16 long words of data from the class 3 transmit SAR buffer. Note that the data block is not cleared from the buffer until the last word has been read.

5. The SDRAMCrcErr status in the High Priority Interrupt Status register is now valid.

6. If more blocks are to be read from cell buffer, then return to step 3.

7. If no more blocks are to be read, then the processor clears the SarDiagRdModeEn bit. The transmit buffer should be emptied before SarDiagRdModeEn bit is cleared.

CONTACTING PMC-SIERRA, INC.

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

Tel: (604) 415-6000

Fax: (604) 415-6200

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

PMC-1991454 (R2) ref PMC-1981224 (R6) Issue date: March 2000