ISCAS’98 paper

Design and Implementation of an ATM Segmentation Engine with PCI Interface

Chan Kim, Jong-Arm Jun, Kyou-Ho Lee, Hyup-Jong Kim

Electronics and Telecommunications Research Institute
161 Kajeong-dong, Yuseong-goo, Taejon, KOREA

ABSTRACT

This paper describes the design and implementation of an ATM Segmentation Engine including PCI interface. This engine, which was designed for an ASIC called ASAH-NIC, contains DMA read machine and segmentation machine. It uses local memory where control information is stored in schedule table, VC table, buffer descriptor and status queue. It has several spepcial features like processing split packet buffers and automatic alignment and packing of transmit data.

1. INTRODUCTION

This paper describes the implementation of an ATM segmentation engine with PCI bus which is part of an ASIC called ASAH-NIC which performs segmentation for any arbitrary number of connections at the same time with 155Mbps performance(with reassembly engine in one chip). This engine can also tightly control the CBR, VBR burst cell generation using scheduler, and also supports UBR traffic with timer control. This paper discusses the segmentation part with its DMA master read function. Figure 1 shows the block diagram of the ASAH-NIC chip. The DMA slave and local memory interface and re-assembly function will not be discussed.

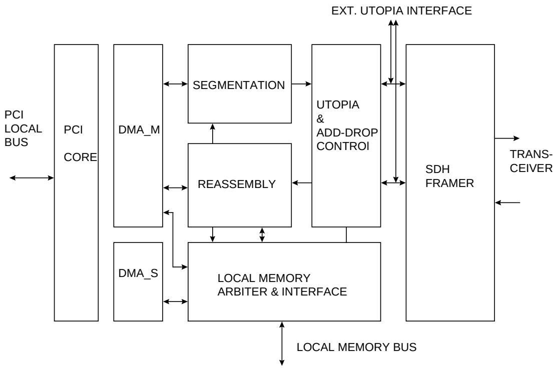

Figure 1. ASAH-NIC ASIC’s Block Diagram

2. BASIC OPERATION

The basic operation relies on some data structure maintained in the local memory. They are VC table, buffer descriptor table, buffer descriptor queue, status queue, and schedule table.

VC Table contains the control information for each connection such as the AAL type, whether it’s buffers contain complete cells, and whether it’s UBR or not. It also keeps the CURR_DSCR and LAST_DSCR which point to the current and last element of the buffer descriptor linked list connected to the VC table. It also keeps the BUFF_RD_CNT which shows how many bytes have been read from the current buffer. It is to be noted that only the VC table is updated and written back to memory after a connection’s service thus reducing local memory usage. The VC table also has UBR next pointer to link the VC table belonging to UBR into a cyclic linked list and timer control values. Partial CRC32 value is also kept in the VC table.

When there is data to send, the host CPU writes buffer descriptor for that data to the descriptor queue in the local memory while incrementing the write pointer. Buffer descriptor contains the starting address and the size of the buffer in byte. The buffer descriptor should also contain packet trailer and EOP(end of packet) indication. It also has the VCC index which shows what connection the data belongs to. Since VCC index is the order of the VC table in the pre-assigned SRAM address space, we can get the start address of the corresponding VC table with the VCC index. The buffer descriptors sent to the descriptor queue is taken out and linked to the corresponding VC table. When linked, the VCC index field of the buffer descriptor is used as pointer to form linked list.

The order of service for each connection is determined by the schedule table. The schedule table’s entry contains the start address of the VC table to be serviced. Only the CBR or VBR connection is registered in this schedule table.

The segmentation engine also writes status information into status queue and the host CPU reads from the status queue. The status information includes the finished buffer address and VCC index, and other status information.

Figure 2 shows the free buffer list and method of linking buffer descriptors to the VC table.

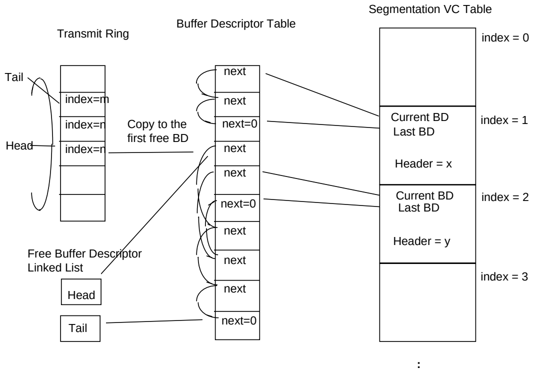

Figure 2 method of buffer descriptor linking.

3. HARDWARE ARCHITECTURE

3. 1 Segmentation Processor

This section describes each processing block's action.

3. 1. 1 Buffer Descriptor Link Manager(BLM)

The descriptor queue level is increased when the host CPU writes one or more BD(s) to the BDQ and decremented when a descriptor is taken out and linked to a VC table. The BLM continues linking the BD in the BDQ while the BDQ level is not zero.

The BLM reads one BD from the BDQ and copies the data into a new free BD's address which was fetched from FBM. This new address is linked to the VC table corresponding to the VCC index using the current and last descriptor pointer of the VC table and next pointer of the BD. Figure 3 shows a typical linking action when there was two BDs already linked.

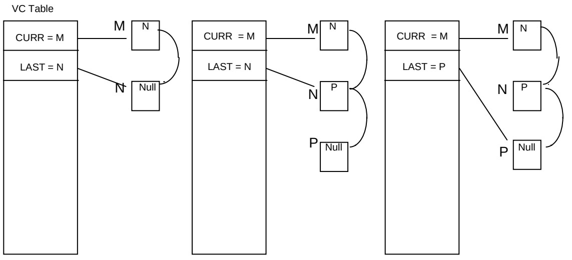

Figure 3. Example of a Buffer Linking Procedure

3.1.2. Free Buffer Descriptor Manager(FBM)

The free buffer descriptor manager manages the linked list of free buffer descriptor. When requested from the BLM, it gives the head pointer of the free list and updates the head pointer to the next free one. The next address is always pre-fetched to get the next pointer before it is overwritten with null by the BLM. In addition, when requested by the schedule and segmentation circuit, the FBM appends the address of the used descriptor to the tail of the free linked list.

3. 1. 3 Scheduling and Segmentation Manager(SSM)

SSM comprises of VC table and buffer descriptor register files, main controller, and segmentation cell buffer. The segmentation cell buffer has 10 cell buffers and writes DMA read cell data or unassigned cells into the FIFO as requested and sends the cell to the UTOPIA interface when there is at least one cell in the FIFO. Because the write controller and the main controller monitors the FIFO state before any (DMA) cell writing, FIFO overflow is prevented.

VC table and buffer descriptor register file keeps the VC table and buffer descriptor read from the memory and supplies various signals for the controller and requests DMA with retrieved parameters when commanded by the controller. The VC table is updated at the end of the DMA.

The number of remaining bytes is obtained by subtracting BUFF_RD_CNT from the buffer size. For AAL 5, if the remaining bytes are more than 48, 48 bytes are requested. If the EOP bit of the buffer descriptor is set and the number of remaining bytes is less than 40, DMA is requested for these remaining bytes with EOP cell indication set. (Figure 4 (a)) But if the remaining bytes are more than 40 and less than or equal to 48, the remaining bytes are requested with EOP cell indication deasserted. In this case, the number of bytes requested for the next DMA will be zero to indicate all padding cell.(Figure 4. (b))

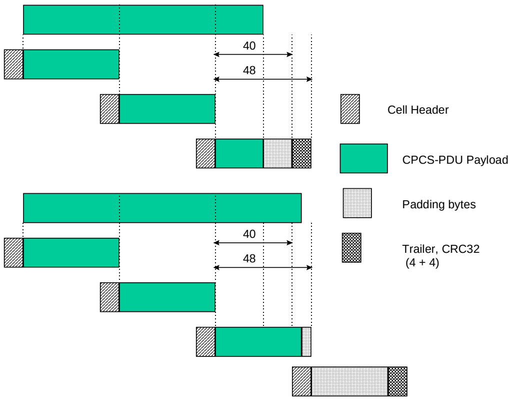

Figure 4. Typical AAL 5 segmentation
(a) above : remaining <= 40
(b) below : 40 < remaining <= 48

As a special case, if EOP is not set for the buffer and the number of remaining bytes is less than 48 but there is a following buffer already linked to the VC table, it assumes that the following buffer has the needed data and two DMA action is performed for one cell. We call this 'split DMA' in this engine. This can be called a data gathering function for segmentation.

Figure 5 shows the simplified state diagram of the main segmentation controller. It should be understood that VC table information for the decision is updated at the end of DMA completion. And data availability includes the split DMA case.

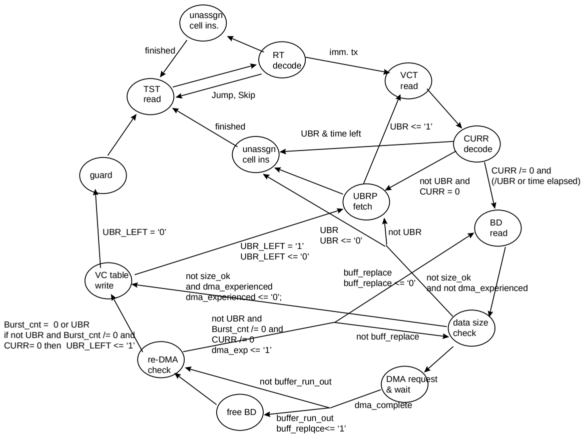
unassgn cell ins.
finished
RT decode
imm. tx
VCT read
TST read
Jump, Skip
finished
UBR & time left
UBR <= '1'
CURR decode
unassgn cell ins
guard
CURR /= 0 and (/UBR or time elapsed)
not UBR and CURR = 0
UBRP fetch
BD read
UBR_LEFT = '0'
UBR UBR <= '0'
not UBR
UBR_LEFT = '1' UBR_LEFT <= '0'
VC table write
not size_ok and dma_experienced dma_experienced <= '0';
buff_replace buff_replace <= '0'
not size_ok and not dma_experienced
Burst_cnt = 0 or UBR if not UBR and Burst_cnt /= 0 and CURR= 0 then UBR_LEFT <= '1'
not UBR and Burst_cnt /= 0 and CURR /= 0 dma_exp <= '1'
data size check
not buff_replace
re-DMA check
not buffer_run_out
DMA request & wait
dma_complete
free BD
buffer_run_out buff_replqce<= '1'

Figure 5. State Diagram of the Main Controller

The controller starts reading schedule table processing "insert unassigned cell", and "jump" and "pass" operations. If the OP code is "immediate transmit", the controller reads the VC table pointed to by the schedule table. If the CURRENT value is not zero, (meaning at least one BD is linked to the VC table), it loads the buffer descriptor but if it is not, the controller jumps to UBR pointer fetch state. If the UBR pointer is available, it switches to UBR service but if it is not, it commands insertion of 1 unassigned cell and proceeds to next entry of the schedule table.

After loading the VC table and BD, if it is decoded that there's enough data for DMA, the controller commands DMA request. At the end of DMA completion, the VC table is updated with prepared values and new information is available. In all cases when a VC table is newly loaded, the burst count is preset to the defined burst value and the count is decremented at every cell transferring DMA completion and checked for burst continuation. And if the current buffer data is running out by the DMA, the controller frees the current descriptor.

After every DMA completion, the controller checks if another DMA should be performed for the connection. The first case is for split DMA and the second case is when the burst counter is not zero. For each case the controller jumps to appropriate states of reading BD or writing VC table or checking size. When the burst ended due to the lack of data, it returns to UBR service after writing back the VC table.

During the UBR service, timer value is also checked before reading buffer descriptor and the controller jumps directly to unassigned cell insertion if data is not available for DMA. And after a DMA completion, it goes to VC table write back state without looking at burst counter.

The minimum cell generation time distance control for UBR traffic is done using a timer and two fields of the VC table. The timer is incremented every given period of, for instance, 1 usec. Whenever a cell is transmitted, the current timer value plus time distance value for that connection is added and written into the VC table. To generate a cell for any UBR traffic, the timer should be equal to or greater than this time set at its last transmission time.

3.1.4. UBR Pointer Manager(UPM)

The UBR pointer manager searches the UBR linked list VC table and get ready for the UBR pointer having buffer. When UBR pointer is fetched by the SSB it starts searching again. The UPM's pointer searching is enabled only when there is at least one buffer descriptor linked to UBR traffic.

3. 1. 5. Status Report Manager(SRM)

When the status report request bit of the buffer is set, the main control requests status report to the status report manager providing status information. The status report manager latches and writes the status information into the status queue and generates interrupt.

3. 2 DMA Master READ Processor

Basically, the DMA read block receives DMA request with DMA parameters and after reading requested data from the PCI memory of the PCI core, returns the ATM cell with enable signal.

DMA parameter includes cell_request signal, EOP cell indication, and other control signals as well as the start address and size of the DMA read, cell header, AAL type, trailer, CRC32, FCT indication. For AAL5, partial CRC32 value is also returned to the segmentation circuit at the end of DMA.

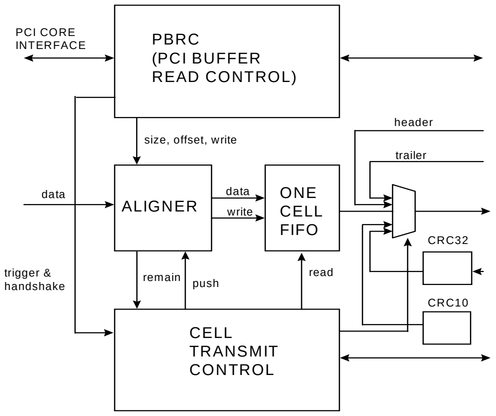

Figure 6. DMA Master Read Circuit

Figure. 6 is the diagram of DMA READ processing block. **PCI read controller** transfers the DMA parameter of start address and size to the PCI core and requests memory read. When passing the parameter to PCI core, the PCI read controller changes the start byte address and byte size to word equivalent values so that all the requested bytes can be contained in the words requested to the PCI core.

Whenever the FIFO is not empty it repeats reading the PCI FIFO until all the requested data is written to the alignment circuit. When sending the unaligned words to the alignment circuit, the PCI read controller calculates the start location(offset) and number(size) of the bytes to be taken from the word and supplies this information with enable signal. Figure 7 shows the offset and size values for a split DMA case where 17 bytes from PCI address BA0045 and 16 bytes from PCI address 79205C is read to form 33 bytes for the cell payload(remaining bytes are filled with padding bytes).

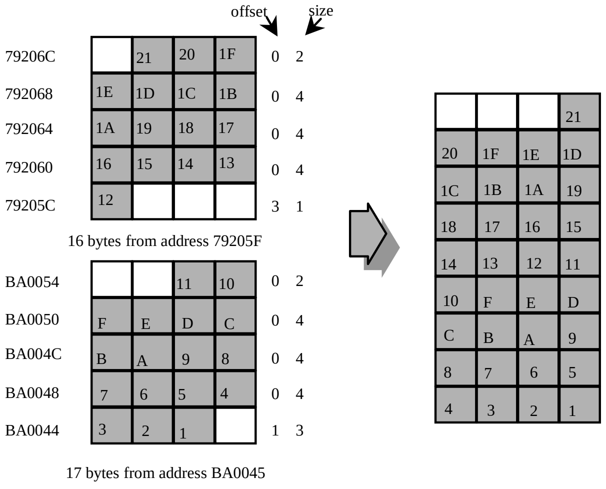

Figure 7. Word Alignment Circuit's Input and Output

Figure 8 shows the block diagram of alignment circuit. The residue register keeps the bytes which are not yet assembled into an aligned word. When a new word has been assembled, the word register latches the aligned word and the strobe signal is pulsed. The controller controls the amount of shift and select and latch actions according to the number of residual bytes and the offset, and size of the input word.

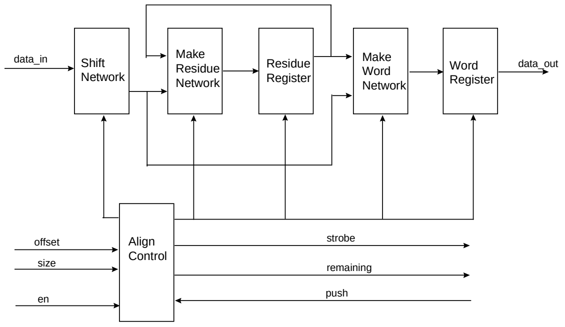

Figure. 8 Alignment Circuit

After the data has been moved to the FIFO, the cell transmit controller reads the data and makes ATM cell by multiplexing ATM cell header, FIFO output data, CRC10, CRC32, and trailer. Before starting cell forming routine, it pushes the occasional remaining word in the alignment circuit out to the FIFO. The padding is done automatically by reading the FIFO 10 or 12 times regardless of the amount of data in the FIFO which is designed to output padding pattern when empty.

4. CONCLUSION

The segmentation engine designed for an ASIC called ASAH-NIC, performs all the protocol processing using local SRAM and reads directly from host or PCI memory to make ATM cells thus relieving the host CPU from processing burdens. Since there is no limitation on the start address and the size , there is no need to move the data to a separate location after processing a higher layer application processing. This engine can tightly control the CBR, VBR bursts and UBR cell generation using schedule and timer.

The ASAH-NIC ASIC has PCI function, SAR and ATM functions for various AAL types, and 155Mbps physical layer function with external UTOPIA interface and provides one-chip solution for ATM adapter.

Acknowledgments

We would like to thank Dr. Jae-geun Kim, Dr. Munkee Choi and Yeong-won Hwang, for their valuable advices and encouragement

References

[1] ITU-T Rec. I.432, I.361, I.363, I.371

[2] PCI Local Bus Specification, Rev 2.1. June. 1995