ATM Packet Scheduler

in FLEX 8000 Devices

March 1995, ver. 1

Application Note 46

Introduction

The ATM Forum
Member

The Asynchronous Transfer Mode (ATM) is rapidly becoming the premier protocol for many communications and networking applications. With ATM installed on LAN, MAN, and WAN networks, all types of voice, data, and video traffic can operate together seamlessly. Currently, no other protocol offers this seamless integration of information, making ATM a catalyst for technological advances in telecommunications, multimedia, and other markets. With a relatively small cell size, ATM is a compromise between the long frames generated in data communications and the short, repetitive transmissions required for voice communication, video transmission, and other isochronous data transmissions. Additionally, ATM allows networks to be reconfigured to meet changing time-of-day conditions and requires users to pay for only the amount of bandwidth used.

ATM is currently a scaleable standard that does not specify requirements for transmission rates, framing, and physical layers. Broadband networks must develop guarantees on bandwidth, delay, and jitter to support a wide variety of ATM applications. These guarantees require designers to use appropriate traffic-scheduling algorithms in ATM switches to ensure that available resources are properly allocated to individual traffic streams. Not only is it difficult to design the scheduling algorithm, but the small size of the ATM cell usually rules out a software implementation of the algorithm.

This application note describes how the packet scheduler/buffer manager for an ATM switch works with Altera's FLEX 8000A devices and off-the-shelf SRAM memory devices. It includes a sample design for switch architectures with output buffering, which is based on a weighted round-robin scheduling algorithm to schedule inputs from multiple ports to one output port. The design supports up to 32 ATM virtual channel (VC) groups with a dedicated buffer of up to 1,024 cells per group.

For information on the ATM standard, refer to ***ATM Specifications***, which is available from the ATM Forum:

The ATM Forum
Worldwide Headquarters
303 Vintage Park Drive
Foster City, CA 94404-1138
TEL: (415) 578-6860
FAX: (415) 525-0182

☞ The sample design in this application note uses the following design files: **bmgr.tdf**, **sender.tdf**, **receiver.tdf**, **control.tdf**, **encoder.tdf**, **decoder.tdf**, **select.tdf**, and **ctrlel.tdf**. These files are available from Altera's electronic bulletin board service (BBS) at (408) 954-0104 or from Altera's FTP site under the name **an_46.exe**.

ATM Switch Architecture

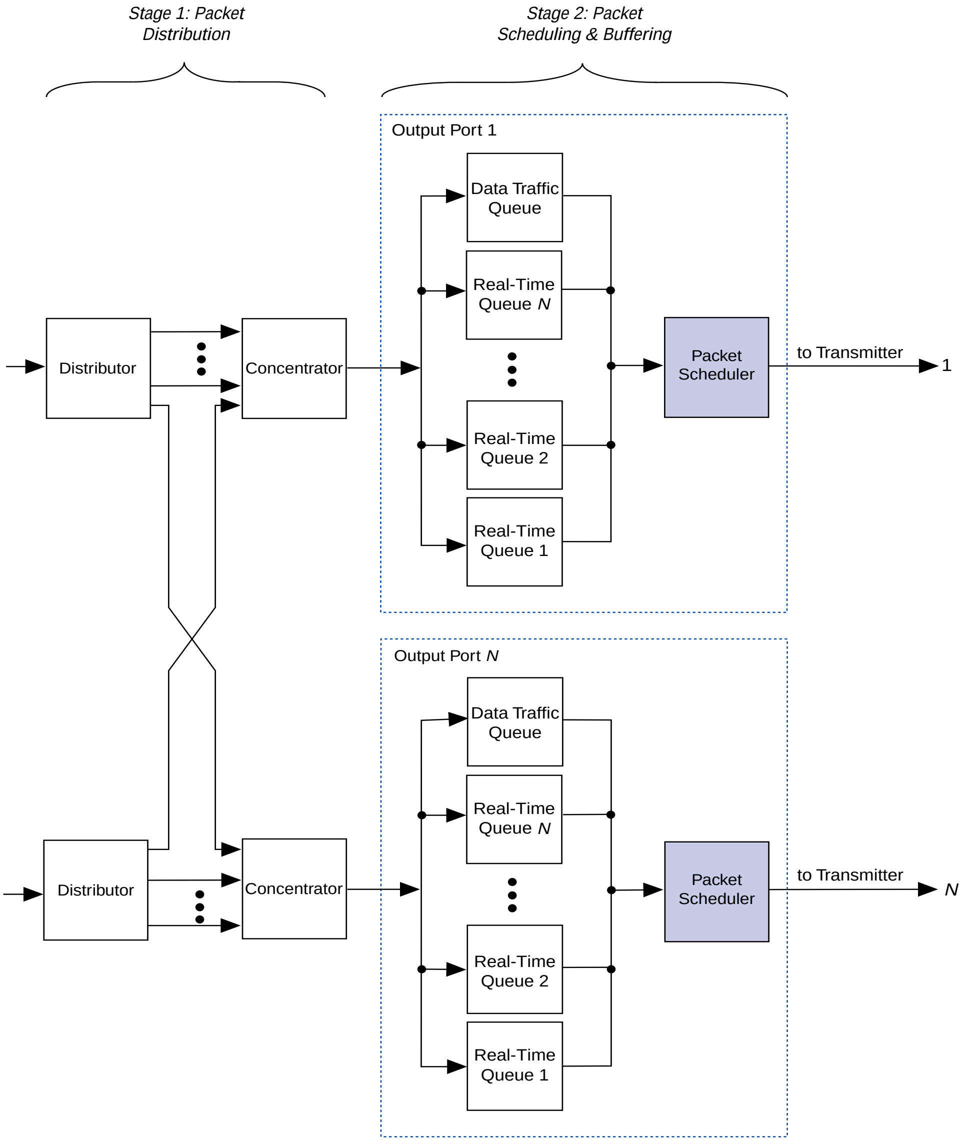
ATM switches transmit both real-time and time-insensitive data, all of which is held in queues until the scheduler selects data packets for transmission to the output port. The ATM output-buffered switch consists of two stages:

1. During the distribution stage, incoming data packets are routed to the output ports. This stage features a multiplexing function to prevent multiple packets from being sent simultaneously to one output port.

2. During the packet scheduling and buffering stage, the scheduler/buffer manager buffers the incoming packets and schedules them for transmission based on their relative priorities. Several VCs can share a queue during this stage.

Figure 1 shows the two stages of the ATM output-buffered switch. The output port prioritizes data packets during the scheduling and buffering stage.

Figure 1. ATM Output-Buffered Switch

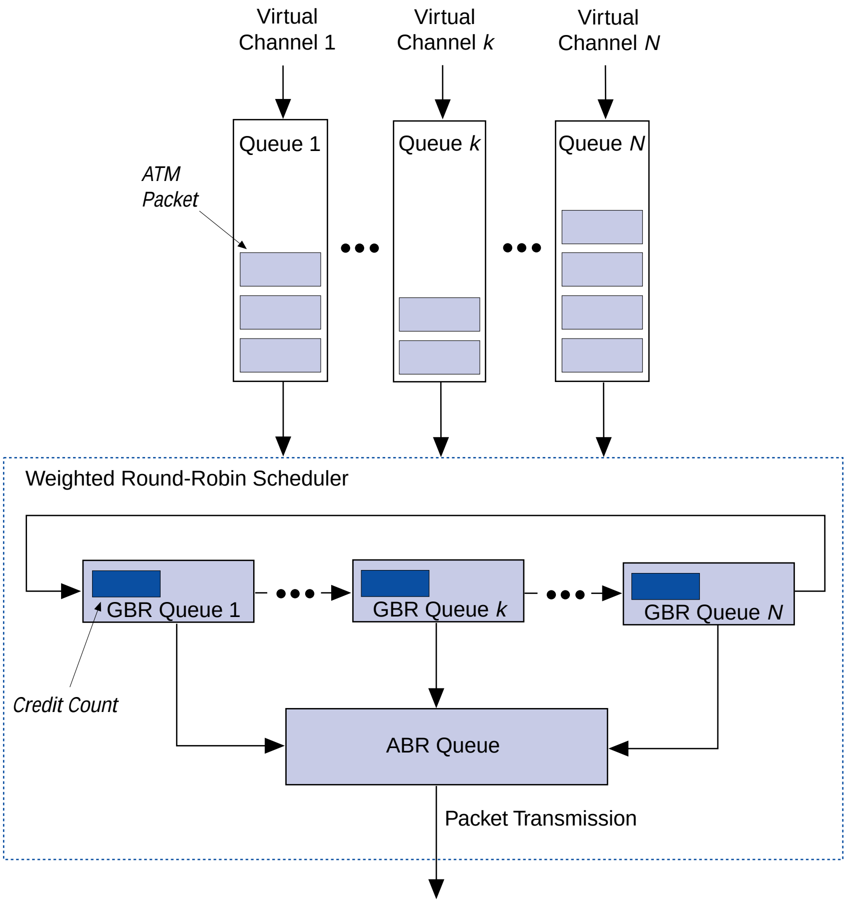


The weighted round-robin algorithm shown in Figure 2 provides a simple and efficient method for scheduling data. During each cycle, the packet scheduler selects cells from VCs in a round-robin fashion. The scheduler organizes into frames the maximum number of cells (C) that can be sent during each frame, assigning every VC group a credit (R_i) that is incremented at the beginning of the next frame. The credit count is the maximum number of cells that can be transmitted from the queue during a frame period. Whenever a cell is sent to the output port, the VC group's credit is decremented. If a VC attempts to send cells for which it has insufficient credits, the excess cells are not sent until the next frame.

VC groups can participate in the selection process only if they have available credits. The scheduler's efficient credit mechanism allocates the bandwidth for each VC group because different VCs may require different portions of the output bandwidth. VC groups can participate in the selection process only if they have available credits.

Incoming ATM cells are queued on the basis of their virtual channel identifier (VCI), which differentiates between cells belonging to guaranteed bit rate (GBR) and available bit rate (ABR) traffic.

Figure 2. Weighted Round-Robin Scheduling Algorithm



Real-time GBR traffic flows usually require bandwidth guarantees for transmission. Any leftover bandwidth can be used by traffic that does not require bandwidth guarantees, such as ABR traffic, which flows after the real-time VCs run out of credits to send their cells. Thus, the bandwidth for each VC group can be explicitly allocated.

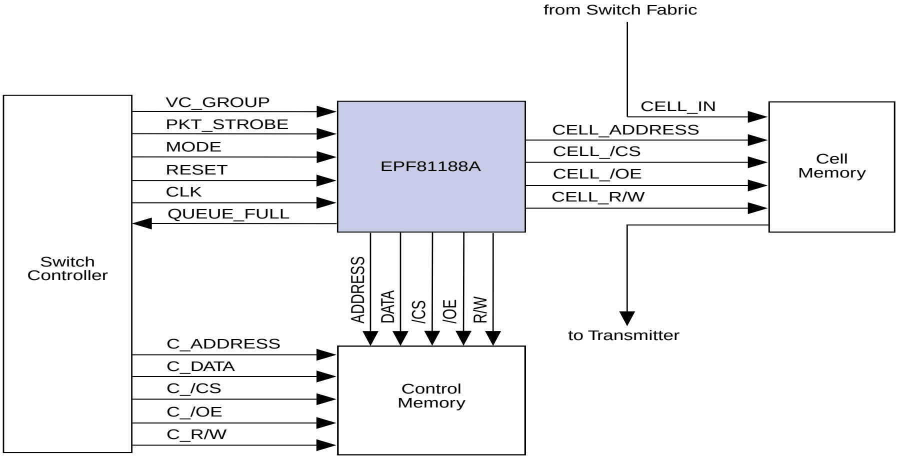
At the beginning of each frame, each GBR queue is assigned a credit count corresponding to the bandwidth allocated to it. The enable signal, called a token, allows cells to be scheduled for transmission to the outgoing link during the frame period. If a GBR queue has a non-zero credit count and an available cell to send, it can block the token and transmit a cell.

For example, a packet from GBR queue *k* is transmitted during the cell cycle. The controller from queue *k* passes the token to the next queue, (represented as an arrow in Figure 2). The token propagates through the chain until it is blocked by a queue with a non-zero credit count and an available cell to send. If a queue meets both conditions, it can transmit the next cell. If none of the queues with credits has a cell to send, the token returns to GBR queue *k*. The scheduler can transmit the next cell from queue *k* if there are available credits and cells; otherwise, the next cell is selected from the ABR queue. If the ABR queue is empty, a second round of token propagation is initiated. During any given round, non-empty GBR queues can block the token regardless of credit count, which keeps the output link busy whenever cells are available in the queues.

Packet Scheduler Architecture

Figure 3 illustrates the packet scheduler implemented with an Altera EPF81188A device and several off-the-shelf SRAM memory devices. The design supports 32 distinct VC groups for GBR traffic and provides a separate queue for ABR traffic.

Figure 3. Packet Scheduler Architecture



The incoming cells from the switch fabric are entered into the cell memory, where the queues reside for both GBR and ABR traffic. Part of the cell memory is dedicated to each queue, allowing a maximum of 1,024 cells per queue. You can modify the design to accommodate a linked-list implementation of the queues by simply adding one more field to each cell. Then, the control memory can store the head and tail pointers for the queues, as well as the counters containing the available credit for each queue and the values for counter initialization. It is generally difficult to initialize the counters at the beginning of each frame because the credit counters are implemented in SRAM devices. This design does not require counter reloading at the beginning of each frame because the scheduler only needs to access one credit counter from the control memory during each cycle.

Whenever a cell is entered into the cell memory by the switch fabric, the scheduler receives a 5-bit address (VC_GROUP) that identifies the corresponding VC group, and a strobe (PKT_STROBE) that signals the receipt of a new packet. Because the first 32 combinations of the 6-bit address signify the 32 queues used for GBR traffic, a cell is assumed to belong to ABR traffic if a significant bit of its address is already set.

The controller is responsible for decoding the VC_GROUP address and storing the cell in its corresponding queue by providing the address and control signals to the cell memory. This design requires two counters for each queue that implement head and tail pointers. These counters do not need to access sequential addresses, but can be implemented as linear-feedback shift registers (LFSRs). LFSRs are composed of very simple logic (a shift register and XOR gates), and do not require carry propagation logic.

If you design the cell memory using dual-port RAM, the next cell for transmission can be selected in parallel when the cells are loading. The small size of the ATM cells allows the scheduler to buffer all cells for at least one cycle before forwarding them to an output port. The scheduling process includes selecting a queue for transmission and updating the corresponding credit counters in the control memory. Scheduling is the most time-critical part of the design because multiple queues need to be checked during each cycle. After the process is completed, the controller sends the address and control signals to the cell memory so that the selected cell can be forwarded to the transmitter.

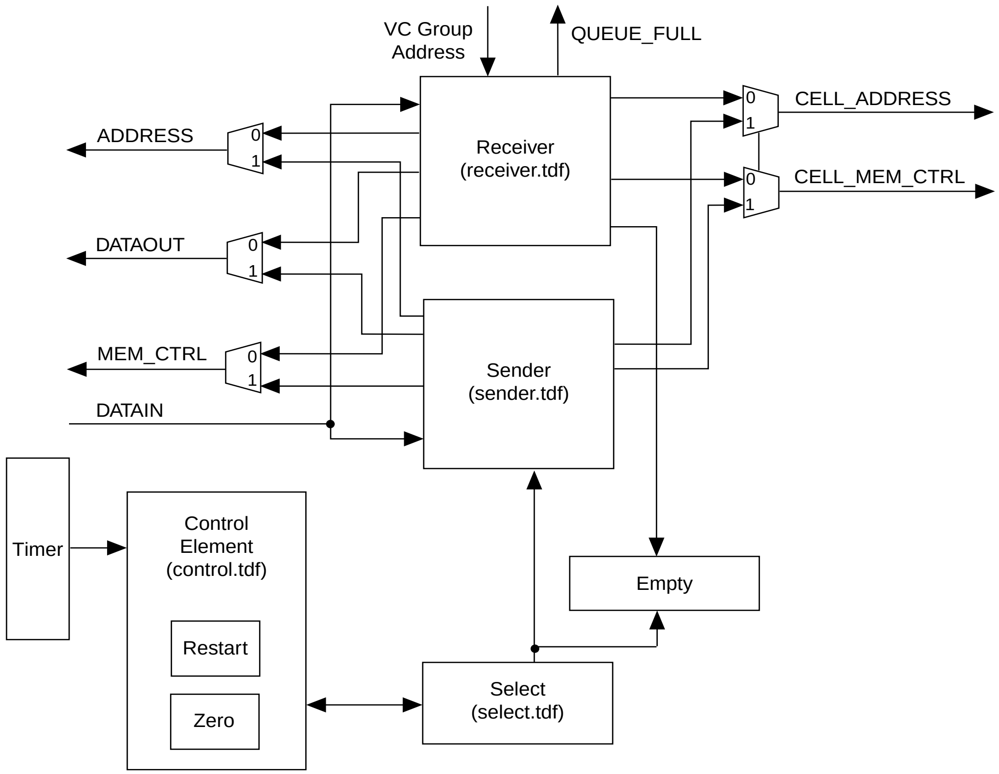
The control memory also has two ports that allow the switch controller to modify the bandwidth allocations of individual VC groups when new connections are set up or when existing connections are closed. The MODE signal generated by the control memory indicates whether the scheduler or the global switch controller has accessed the control memory. When the MODE signal is 1, the scheduler cannot access the control memory.

A cell cycle is the period between the entering and the scheduling of cells. Because the output link operates at SONET OC-3 speed (approximately 155 Mbps), the time required for all operations during the cell cycle—including entering cells into memory, assigning them an address, decoding the address, and scheduling them for transmission—must be performed in less than 2.5 ms. For example, when the cell memory is 64 bits wide with an access time of less than 25 ns, the controller logic operates with a Clock cycle of 10 MHz. A simple Clock divider from the 20-MHz system Clock is used to produce the 10-MHz Clock for the controller. One ATM cell can be easily stored in memory within 8 memory cycles. Two memory cycles can be completed in one controller cycle. Therefore, the load/store operation of a cell in the cell memory requires 4 controller cycles.

Scheduler Design

Figure 4 shows a block diagram of the scheduler controller.

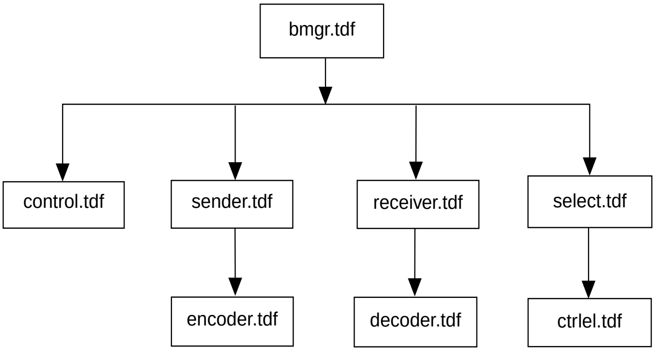
Figure 4. Scheduler Controller Block Diagram



In this design, the control unit is a state machine that coordinates the actions of all the functions. The receiver (**receiver.tdf**) receives and decodes the VC group address of the cell being added to the cell memory and produces the corresponding addresses for the control and cell memory. The selection unit works in parallel with the receiver function and chooses the queue from which cells are sent. Once a packet of cells is selected, the sender function drives the cell memory to route the packet to the transmitter.

Figure 5 shows the design hierarchy for the scheduler controller.

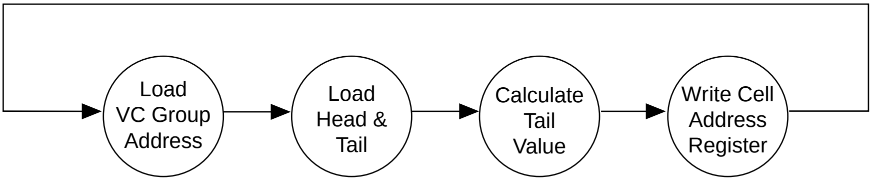
Figure 5. Scheduler Controller Design Hierarchy



Receiver Function (receiver.tdf)

Figure 6 illustrates the control flow of the receiver logic contained in **receiver.tdf**, which calls the decoder (**decoder.tdf**). Each time a queue receives a packet of cells, the corresponding bit in the `EMPTY` register is written. All queues that receive a packet have a "not-empty" status at the end of the cycle. The `EMPTY` register must be reset at the beginning of the cell cycle to indicate a not-empty queue.

Figure 6. State Machine Diagram (receiver.tdf)



A cell load is completed in the following 4 cycles, requiring 16 cycles to load 4 cells from the switch fabric. The `CONTROL[4..1]` signals indicate the functions to be performed in each of the cycles. This procedure is important because if the controls are not active, the registers are not updated.

1. In the first cycle, the VC group address, `VC_GROUP`, is loaded and decoded by **decoder.tdf**. The `PKT_STROBE` signal indicates that a valid packet is waiting to be processed.

2. In the second cycle, the head and tail pointers are retrieved from the control memory.

3. In the third cycle, the next state of the LFSR register that represents the tail is calculated.

4. In the fourth cycle, a signal is produced that indicates whether or not the queue is full. During this cycle, the cell-writing process is initiated and the address register of the cell memory is written.

The cell-writing process requires four controller cycles and is a completely pipelined process: as the memory is written, the header of the next packet can be processed. During all cycles, the receiver function uses multiplexers to control the memory signals.

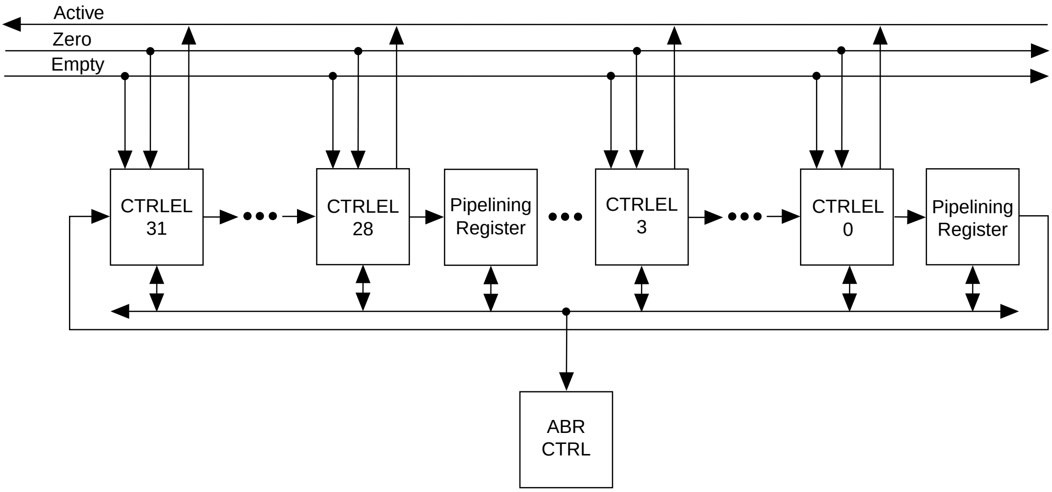
Select Function (select.tdf)

Figure 7 illustrates the weighted round-robin algorithm implemented by the select function, which calls control elements defined by **ctrlel.tdf**. One control element is assigned to each real-time queue; one register bit in each **ctrlel.tdf** file corresponds to a queue in each register. The logic associated with the control elements uses the values of the `EMPTY` and `ZERO` registers to select the queue from which the next packet will be scheduled for transmission.

The `EMPTY` register stores one bit per queue to indicate whether the queue is empty. The `ZERO` register stores one bit per queue to indicate whether any credits are available for the selected queue. The register is reset at the end of every frame to indicate the total number of credits available in the queues.

The `RESTART` register stores one bit per queue to indicate whether the credit counter was reloaded at the beginning of a new round. You can use this register to store the counters in the control memory and avoid reloading all of the counters. `TIMER` is a simple down-counter that keeps track of the frame time.

Figure 7. Block Diagram of select.tdf



The CTRLEL function that sent a packet during the previous cell cycle initiates the selection process by activating its CARRY_OUT line, which propagates a token signal through the chain. The next CTRLEL in the chain checks its EMPTY registers to see if the queue is empty, and its ZERO register to see if all available credits have expired, and then sets the bits of the EMPTY and ZERO registers accordingly. If both conditions are true, the CTRLEL function makes a transition to the next state by setting the FIRST_ROUND register, and propagates a carry-out of zero to the next control element. Otherwise, a carry-out of 1 is propagated to the next control element, selecting it for transmission.

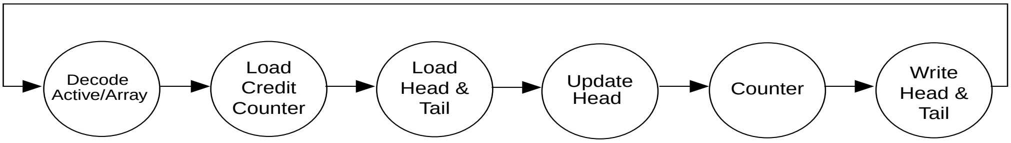
The selection process is analogous to a simple carry-chain adder. However, the logic involved in each state is more complex; therefore, the carry-out propagation through the entire chain of 32 control elements cannot be completed in one cycle. The control elements are conveniently organized in groups of four, as in a carry look-ahead adder. All control elements belonging to the same group complete their operation in the same cycle. At the end of each cycle, the carry-out information from the last control element in the group is stored in a pipelining register. Subsequent groups can complete their operation during subsequent cycles. Even if a control element initiates the process, it is not necessarily the first element to complete the decision cycle because an entire decision cycle requires eight rounds to complete. If a control element is given priority to send a cell during the first round, its FIRST_ROUND register becomes active and the carry propagation stops.

If the carry-out propagates back to the initial control element, no other queue was selected for transmission. The second round of the selection process is then initiated because none of the CTRLEL functions were selected in the first round. The ABR queue is checked for available packets and the carry-out signal is set to zero if the ABR queue has packets to send. Otherwise, the control element selects the first queue with a packet without checking the ZERO register, first setting the ACTIVE register and then clearing the carry-out signal to zero. The entire selection process is completed in a maximum of 16 cycles, which is the time required to load 4 new cells into the queues.

Sender Function (sender.tdf)

Figure 8 shows a state machine for the sender function (**sender.tdf**). The ACTIVE_ARRAY[31..0] signals indicate which queue was selected to send a packet. The sender logic that encodes the VC group address is contained in **encoder.tdf**.

Figure 8. State Machine Diagram of sender.tdf



During the first cycle, a 32-to-5 encoder generates the corresponding address of the control memory. The ACTIVE_ARRAY register allows only one bit of the register to be set at a time. Therefore, the multiple don't-care states make the implementation of the encoder relatively simple.

During the second cycle, the credit counter and the maximum number of credits for the queue are loaded into the COUNTER and RELOAD_VALUE, respectively, from the memory. By using SRAM elements to store these values, the same data path is used for access, thus simplifying the design. The design can be expanded to include more VCs if you add memory and address bits. If you use dedicated counters, you incur extra overhead. When you set up VCs, the maximum number of credits is stored as a value in the control memory and is used to reload the counter at the beginning of every frame.

In the third cycle, the counter is decremented, and both the head and tail pointers are loaded from the control memory. The head pointer is updated during the next cycle. It is not necessary to check for an empty queue before updating the head pointer, since the selection logic does not select the queue if it is empty. When the last cell in a queue is reached, the corresponding bit of the EMPTY register is set through the EMPTY_OUT bus. If all credits of any queue have been used, the corresponding bits of the ZERO register must be updated through the ZERO_OUT bus.

At the beginning of a frame, the RESTART register is preset. If a restart bit is set before a queue is selected, the queue's credit counter must be reloaded to its maximum value by loading the COUNTER with the content of the RELOAD_VALUE register. If the RESTART register is used, the credit counters do not need to be reloaded at the beginning of the frame. Instead, they can be updated with their maximum value after the frame has begun, when the corresponding queue is accessed by the sender function.

During the subsequent two cycles, the new values of the counter and the head and tail pointers are written back to the control memory. At the same time, the process of reading out the cell is initiated by the CTRLEL function. In both the sender and the receiver function, cells are read out when the local bus of the cell memory is idle. During the first 4 cell cycles of cell time, the cell memory is not used by the sender function; therefore, cells selected during the previous cycle can be sent.

Control Unit

The control unit, which is implemented in **control.tdf**, consists of a simple state machine that iterates through 25 states. With one-hot encoding, the state machine is implemented with a simple 25-bit shift register and logic to set the first register. At the beginning of the operation, a high value is shifted to the first register. When the MODE signal is low, the machine works as a circular shift-register. When the MODE signal is high, the scheduler's operation is suspended to update the control memory and the registers are cleared by the RESET signal.

Hardware Implementation

The design presented in this application note fits into an Altera EPF81188A device using approximately 950 logic cells and 73 pins. If you plan to expand your design in the future—for example, by adding virtual channels or priority information—Altera recommends using the EPF81500A device with 1,296 logic cells.

Glossary

The following definitions are from the "Glossary of ATM Terms," provided by 3Com in conjunction with Technology Transfer Institute, Business Communications Review, and McQuillan Consulting.

ABR (*Available Bit Rate*) One of five ATM Forum-defined service types. Supports variable bit rate data traffic with flow control, a minimum guaranteed data transmission rate, and specified performance parameters.

BISDN (*Broadband Integrated Services Digital Network*) ITU-TSS name for the application of ISDN concepts to the high-speed (above 1.544 Mbps) broadband area. Primarily identifies SONET as the transmission subsystem and ATM as the transport protocol of choice.

cell Unit of transmission in ATM. A fixed-size frame consisting of a 5-octet header and a 48-octet payload.

concentrator A network device used in FDDI and token ring networks, which provides connections for multiple stations, so they can communicate with other stations on the network.

frame A bundle of data, usually in binary form, organized in a specific way for transmission. Three principal elements are included in the frame: control information (destination, origin, length of frame); the data to be transmitted; and the error detection and correction bits. The data and control elements and error-control information are arranged in a specified format. A frame is the basic data transmission unit employed in bit-oriented protocols.

FR (*Frame Relay*) A network technology based on virtual circuits and supporting variable-length frame transmission between end users.

port The entrance or physical access point to a repeater, computer, multiplexer, device, or network where signals may be supplied, extracted, or observed.

routing (1) The assignment of a path by which a message will reach its destination. (2) The forwarding of a message unit along a particular path as determined by the parameters carried in the message. (3) Routing may also include the translation of messages between LAN segments that use different LLC protocols.

SONET (*Synchronous Optical Network*) ANSI standard for transmission over optical fiber. Used in the U.S. and Canada. A variation of the SDH international standard.

token In token-passing networks, the token gives the "holding" queue the right to transmit on the shared medium. The token circulates sequentially through the stations on the ring. FDDI specifically uses two classes of tokens: restricted and non-restricted.

VC (*Virtual Channel*) Each physical circuit in an ATM network is pre-configured with some number of virtual paths. Each virtual path may support many virtual channels. VCs are not assigned any dedicated bandwidth. Bandwidth is allocated on demand by the network as users have traffic to transmit.

VCI (*Virtual Channel Identifier*) The 16-bit number in an ATM cell header identifying the specific virtual channel on which the cell is traversing on the current physical circuit.

2610 Orchard Parkway
San Jose, CA 95134-2020
(408) 894-7000
Applications Hotline:
(800) 800-EPLD
Product Marketing:
(408) 894-7104
Literature Services:
(408) 894-7144

Altera, MAX+PLUS, MAX, and FLEX are registered trademarks of Altera Corporation. The following are trademarks of Altera Corporation: FLEX 8000, EPF81188A, and EPF81500A. Altera products marketed under trademarks are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

U.S. and European patents pending.

nsai

I.S. EN ISO 9001