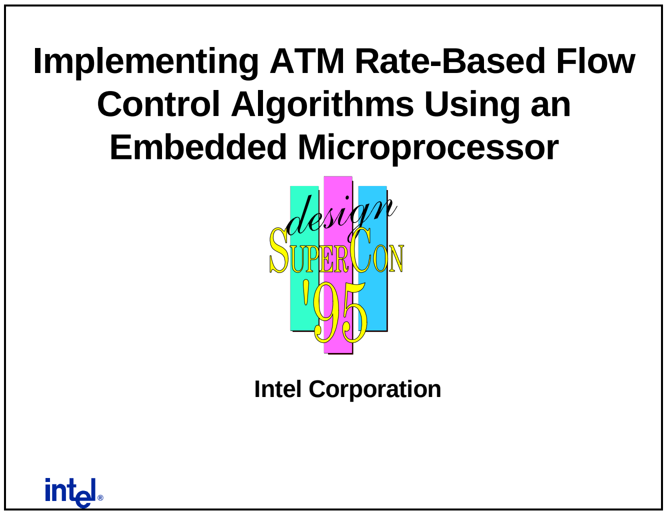
Implementing ATM Rate-Based Flow Control Algorithms Using an Embeddded Microprocessor

Slide #1

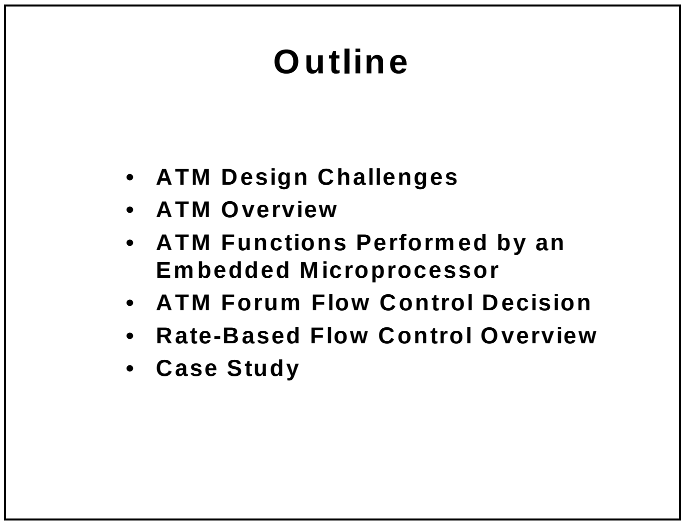


Implementing ATM Rate-Based Flow Control Algorithms Using an Embedded Microprocessor

design
SuperCon
'95

Intel Corporation

Slide #2



Outline

- ATM Design Challenges
- ATM Overview
- ATM Functions Performed by an Embedded Microprocessor
- ATM Forum Flow Control Decision
- Rate-Based Flow Control Overview
- Case Study

This presentation, discusses some of the current challenges of Asynchronous Transfer Mode (ATM) and how an embedded microprocessor can be used to address these challenges. Also discussed are the ATM functions most suitable for an embedded microprocessor and provide a detailed analysis of one of the latest topics in ATM, rate-based flow control.

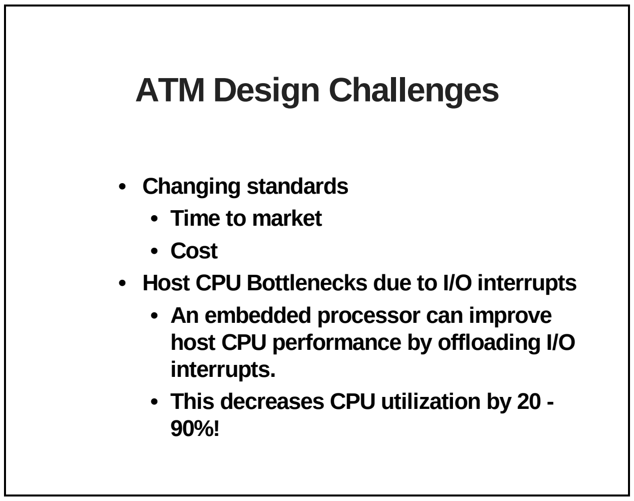
The rate-based flow control analysis includes an overview of rate-based flow control, a case study using an i960® microprocessor to implement the rate-based flow control algorithm, and performance results.

The ATM system designer faces many challenges in designing an ATM network interface card. The number of virtual circuits to support and which ATM chip sets to use are just two of the technical issues. These two issues must be weighed against cost and time to market. To compound the problem, many of the standards are still being refined.

The ATM market is in a state of flux: there are many different players and products. Time to market is critical. The system designer needs to design as quickly as possible, but risks producing a design that will be incompatible with evolving standards. A flexible design that can adapt quickly to changing standards is required.

An embedded microprocessor, like the i960® processor, provides that flexibility by implementing ATM functionality in software at speeds fast enough to support 155Mbps.

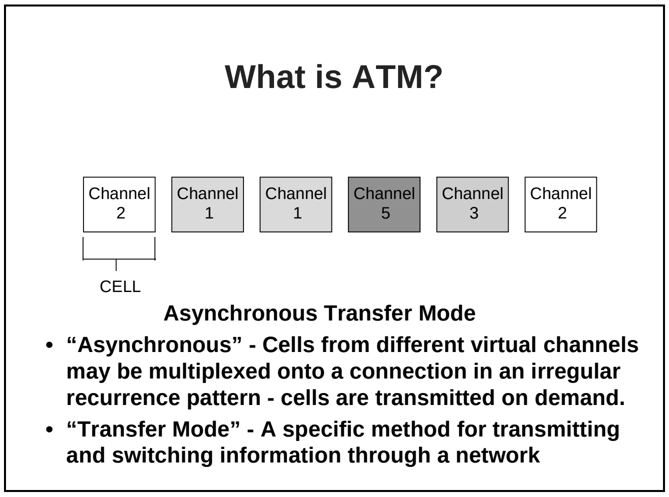
Slide #3



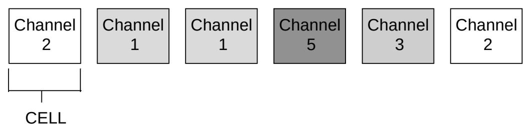
ATM Design Challenges

- **Changing standards**
  - **Time to market**
  - **Cost**
- **Host CPU Bottlenecks due to I/O interrupts**
  - **An embedded processor can improve host CPU performance by offloading I/O interrupts.**
  - **This decreases CPU utilization by 20 - 90%!**

Slide #4



What is ATM?



Asynchronous Transfer Mode

- **"Asynchronous" - Cells from different virtual channels may be multiplexed onto a connection in an irregular recurrence pattern - cells are transmitted on demand.**
- **"Transfer Mode" - A specific method for transmitting and switching information through a network**

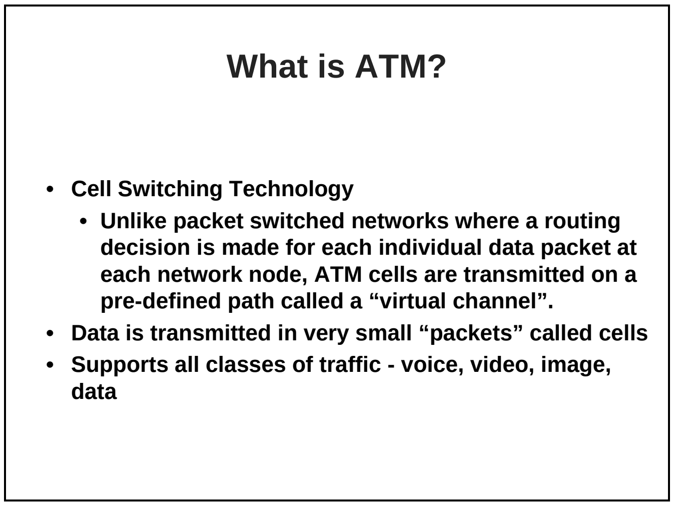
ATM presents many design challenges. The rapidly evolving standards make designing to the most current spec difficult. In addition, it's risky to commit an ASIC implementation to evolving standards. An embedded processor eliminates these problems by providing a flexible solution for updating to the latest standards. This prevents costly and time-consuming hardware revisions while keeping up with evolving standards.

An embedded microprocessor addresses another design issue that becomes critical as network bandwidth increases: host CPU loading. The host CPU, especially on a server, gets quickly overloaded with processing I/O interrupts. A comparison was done between two different 100Mbps fast ethernet Network Interface Cards (NIC) in a server. On one NIC an 80960 embedded processor handled I/O functions and I/O interrupts. The other NIC did not have an embedded processor on board. The NIC with the 80960 benchmarked host CPU utilization at up to 90% lower than the NIC without an 80960. This frees the host for increased application processing in client/server environments.

ATM stands for Asynchronous Transfer Mode. In ATM, data streams, divided up into 53-byte cells, are transmitted across a network on a pre-defined path called a virtual channel (VC). Virtual channels will be defined later.

Each data stream requires a separate virtual channel. The example above shows a snapshot of four data streams on four corresponding virtual channels. The data streams were broken up into cells and multiplexed onto one wire. Cells are transmitted on demand. For example, if data stream one requires a higher data rate than data stream three, its cells may be multiplexed onto the wire more frequently. This is in contrast to Time Division Multiplexing (TDM) in packet switching, which multiplexes packets onto a connection in a synchronous, ordered pattern. Using the previous example, if data streams one and three were multiplexed together by TDM in a packet switched network, stream one packets and stream three packets would travel in alternating time slots. If we apply the above example in which data stream one requires a higher data rate than data stream three, to TDM, much of data stream three's time slots would be empty, wasting bandwidth.

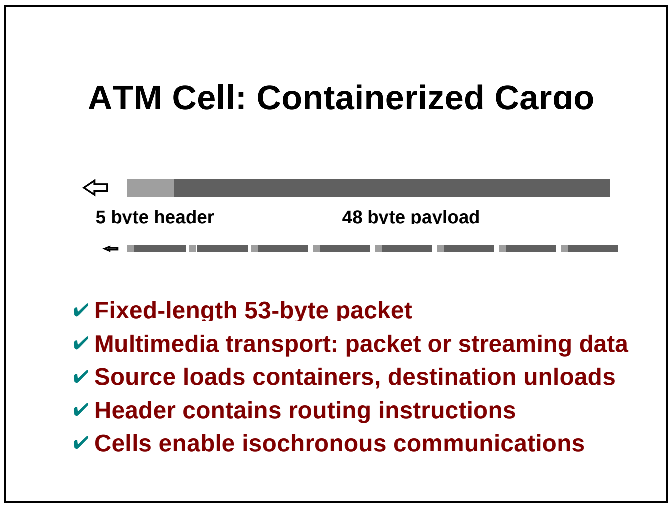
Slide #5



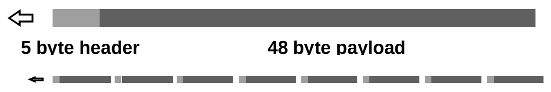
What is ATM?

- Cell Switching Technology
  - Unlike packet switched networks where a routing decision is made for each individual data packet at each network node, ATM cells are transmitted on a pre-defined path called a “virtual channel”.
- Data is transmitted in very small “packets” called cells
- Supports all classes of traffic - voice, video, image, data

Slide #6



ATM Cell: Containerized Cargo



- ✔ Fixed-length 53-byte packet
- ✔ Multimedia transport: packet or streaming data
- ✔ Source loads containers, destination unloads
- ✔ Header contains routing instructions
- ✔ Cells enable isochronous communications

ATM is a cell switching technology. It relies on pre-defined paths, called virtual channels, to move cells through ATM switches across a network

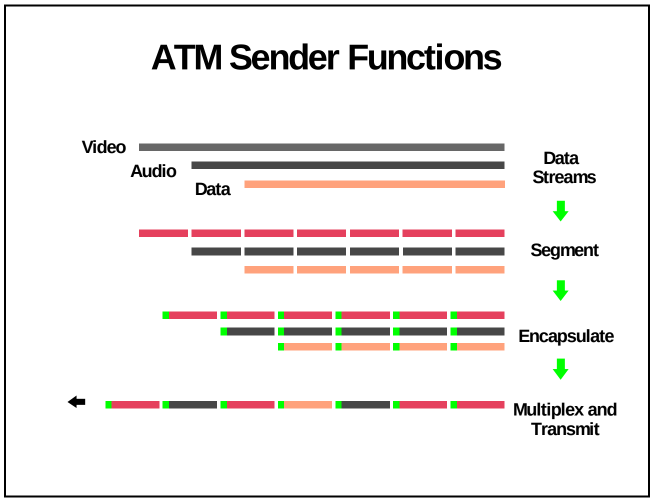
Data is packaged in small fixed-length packets called cells. The small size of the cell and its fixed size allow ATM to support multiple types of traffic: video, voice, data, and image.

An ATM cell can be thought of as a container for data. Compare this to a shipping company which pre-packages all cargo into identical boxes. Since all the boxes are identical in size and shape, it’s easier to move them from place to place and truck to truck.

ATM is the same way. The cells are a fixed 53 bytes in length, unlike packet switched data which varies in length. This makes the ATM cell switching more efficient.

The cell itself consists of two parts: a 5-byte header and a 48-byte data field called a payload. The header contains the virtual channel information needed to send the cell across the network.

Slide #7



ATM Sender Functions

The sender's job is to get data to the destination. Let's take an example of three data streams that need to be transferred from point A to point B. Each stream of data has its own agreed upon data transmission rate. For example:

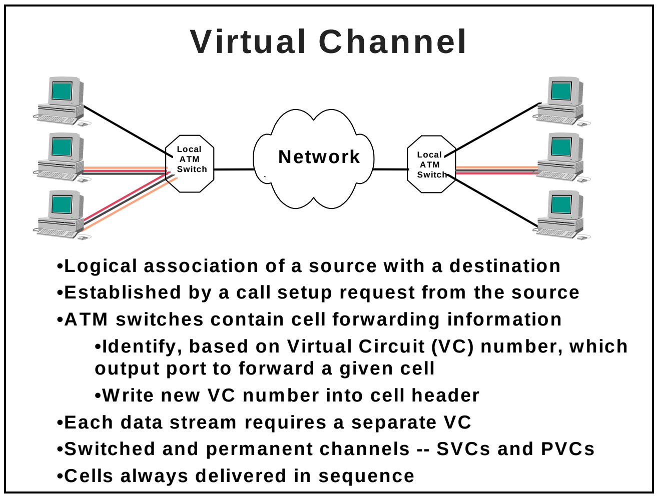
Video:	isochronous	10 Mbps
Audio:	isochronous	1 Mbps
Data:	asynchronous	1 Kbps

Frames are segmented into 48-byte cells, and headers are attached. A scheduling service then schedules the cells so that end-to-end bandwidth requirements will be met.

At the destination end station, this order is reversed. Cells are demultiplexed, decapsulated, and reassembled into streams and delivered to the application.

The cells are transported between end points through virtual channels. Each data stream has a separate and dedicated virtual channel. So, given this example, three virtual channels would be set up: one for data, one for audio, and one for video.

Slide #8



Virtual Channel

- Logical association of a source with a destination
- Established by a call setup request from the source
- ATM switches contain cell forwarding information
  - Identify, based on Virtual Circuit (VC) number, which output port to forward a given cell
  - Write new VC number into cell header
- Each data stream requires a separate VC
- Switched and permanent channels -- SVCs and PVCs
- Cells always delivered in sequence

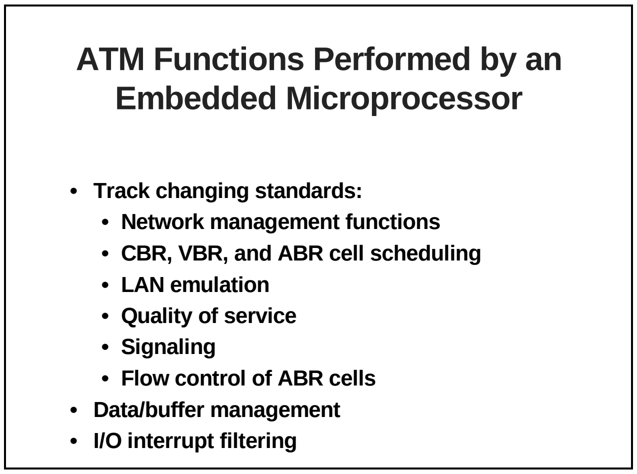
A virtual channel is the logical association between the source and destination. It's termed a virtual channel because unlike a circuit switched network, such as your telephone, where the physical connection is dedicated to your call, the virtual channel is a logical mapping from the source to the destination.

Before any data is transferred, a VC is established by a call setup request from the source, which sets up the connection from the source to the destination through ATM switches. An ATM switch is a device used to route cells to different places in the network. When a cell arrives at an ATM switch, it will be forwarded to an output port of the switch, based on the VC number in its header. Before leaving the switch, its VC number will be modified to forward it to the next switch.

The term "virtual channel" refers to the entire pre-defined path between source and destination. But the VC numbers are local to each point-to-point link; for example, the source to the local ATM switch or switch-to-switch.

Virtual channels can be set up in a fixed fashion (permanent virtual channel) in which no signaling is required. These channels are established at subscription time. Virtual channels can also be set up on an as-needed basis. These virtual channels are called switched virtual channels (SVC) and are established at call setup.

Slide #9



ATM Functions Performed by an Embedded Microprocessor

- Track changing standards:
  - Network management functions
  - CBR, VBR, and ABR cell scheduling
  - LAN emulation
  - Quality of service
  - Signaling
  - Flow control of ABR cells
- Data/buffer management
- I/O interrupt filtering

While many ATM functions can be performed by an embedded microprocessor, some are ideally suited to software implementation. Microprocessors can perform:

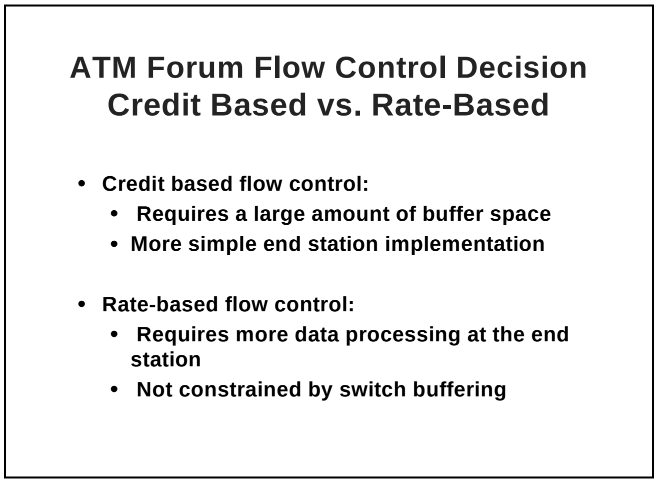
- Network management functions
- Constant Bit Rate(CBR), Variable Bit Rate (VBR), and Available Bit Rate (ABR) cell scheduling
- LAN emulation
- Quality of service
- Signaling
- Flow control of ABR cells

The ATM Forum standards for all of these functions are still being finalized. A designer can partition tasks between hardware and software and rely upon software flexibility to make standards changes or provide headroom for value-added networking functionality.

An embedded microprocessor can also perform data and buffer management between host memory and local memory. Another important embedded processor function, as mentioned earlier, is I/O interrupt filtering.

As an example of an ATM function that is evolving with maturing standards, this paper examines rate-based flow control and how it can be implemented with an embedded microprocessor.

Slide #10



ATM Forum Flow Control Decision Credit Based vs. Rate-Based

- Credit based flow control:
  - Requires a large amount of buffer space
  - More simple end station implementation
- Rate-based flow control:
  - Requires more data processing at the end station
  - Not constrained by switch buffering

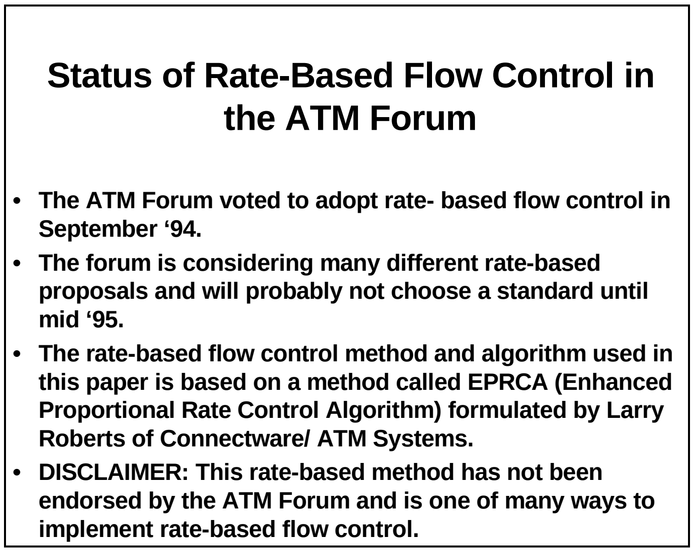
The ATM Forum recently decided between two methods of traffic flow control, credit based and rate-based, and voted to adopt rate-based flow control.

The older credit based method ensured sufficient bandwidth by preventing data transmission until the required bandwidth was established. Bandwidth was allocated by ‘credits’, and if more bandwidth was needed, additional credits had to be requested. This method was costly, requiring a large amount of buffer space in the ATM switch. However, this method was relatively simple for the end system to implement, merely requiring a register to hold the amount of available credit.

Rate-based flow control is an end-to-end method that uses network and destination congestion information to determine the amount of data that can be transmitted. This method is more economical than credit based because it doesn’t require the large amounts of buffer space needed for credit based. However, rate-base flow control has more data processing overhead. The source end system is required to calculate new rates based on congestion information received from the destination and to modify the outgoing cell rate for each virtual channel.

While the ATM Forum has voted to adopt rate-based flow control, different rate-based methods are being discussed, and a specific method has not yet been chosen.

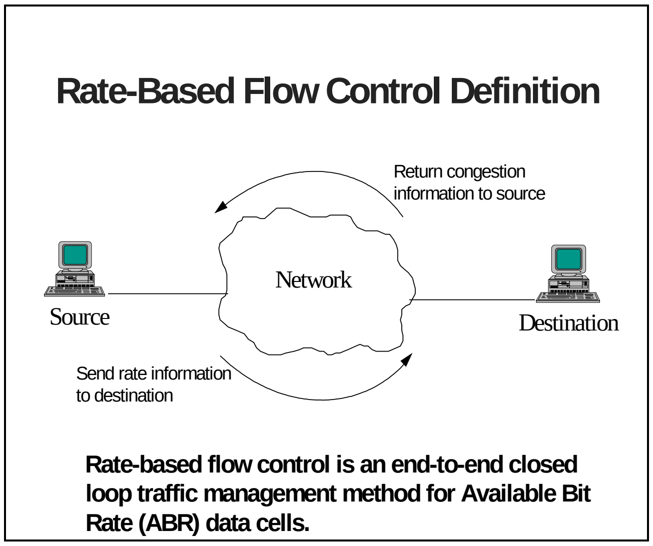
Slide #11



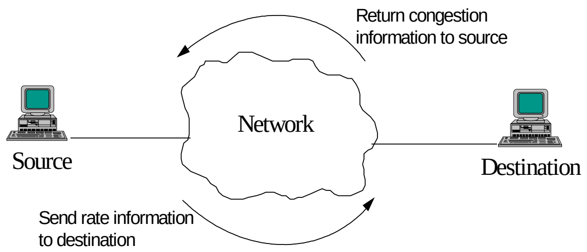
Status of Rate-Based Flow Control in the ATM Forum

- **The ATM Forum voted to adopt rate- based flow control in September ‘94.**
- **The forum is considering many different rate-based proposals and will probably not choose a standard until mid ‘95.**
- **The rate-based flow control method and algorithm used in this paper is based on a method called EPRCA (Enhanced Proportional Rate Control Algorithm) formulated by Larry Roberts of Connectware/ ATM Systems.**
- **DISCLAIMER: This rate-based method has not been endorsed by the ATM Forum and is one of many ways to implement rate-based flow control.**

Slide #12



Rate-Based Flow Control Definition


Return congestion information to source
Network
Source
Destination
Send rate information to destination

Rate-based flow control is an end-to-end closed loop traffic management method for Available Bit Rate (ABR) data cells.

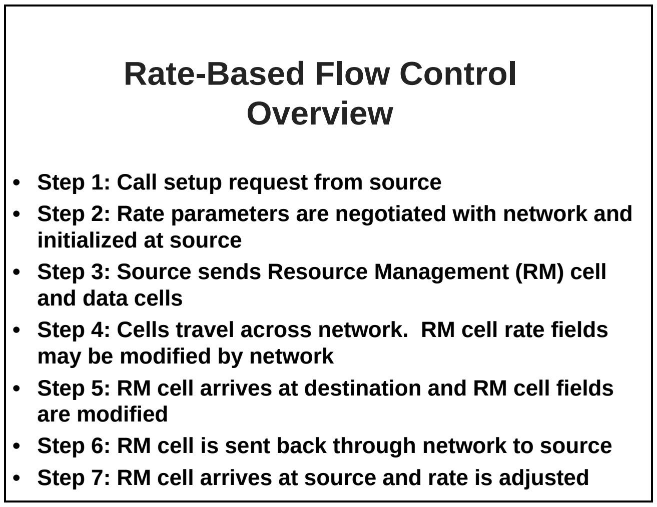
The ATM Forum’s decision to adopt rate-based flow control was just the first step toward a rate-based standard. Many different rate-based proposals have been submitted to the ATM Forum and are under discussion. The final standard is expected to be completed by mid ‘95.

Rate-based flow control is an end-to-end closed loop traffic management method for ABR data cells.

Throughout this presentation, I’ll use the notation “source end station” and “destination end station” to indicate a host/Network Interface Card (NIC) system.

The source end system interleaves a special flow control cell called a Resource Management (RM) cell into the data cell stream. As the RM cell travels through the network, ATM switches can modify RM cell parameters to indicate that congestion was experienced. At the destination, the RM cell direction is reversed, and congestion information is incorporated into various RM cell fields. The RM cell is then sent back to the source, which calculates a new rate and modifies the current source rate according to the congestion feedback carried in the RM cells.

Slide #13



Rate-Based Flow Control Overview

- **Step 1: Call setup request from source**
- **Step 2: Rate parameters are negotiated with network and initialized at source**
- **Step 3: Source sends Resource Management (RM) cell and data cells**
- **Step 4: Cells travel across network. RM cell rate fields may be modified by network**
- **Step 5: RM cell arrives at destination and RM cell fields are modified**
- **Step 6: RM cell is sent back through network to source**
- **Step 7: RM cell arrives at source and rate is adjusted**

The rate-based flow control method can be divided into seven major steps:

- Call setup request from source.
  - Source requests certain parameters from the network.
  - The network may adjust some of these values before the call setup is completed.

- Rate parameters are negotiated with the network and initialized at source.
  - Minimum and maximum data rates are negotiated.

- Source sends Resource Management (RM) cell and data cells.
  - An RM cell is sent out by the source, followed by Nrm data cells. Nrm is the number of cell times between RM cells. Nrm is initialized to a desired value (32 in EPRCA) before data transfer begins.
  - One RM cell will be sent out every Nrm (32) data cells.

- Cells travel across network. RM cell rate fields may be modified by network.
  - Network indicates congestion by modifying rate fields in the RM cell.
  - There are five RM cell fields: Direction, Congestion Indicator, Current Cell Rate, Minimum Cell Rate, and Explicit Rate.

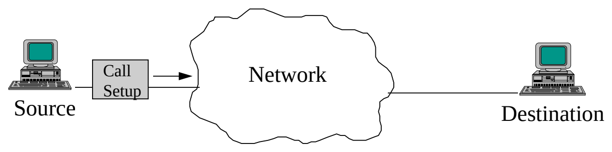
- RM cell arrives at destination.
  - RM cell fields are modified.
  - Direction bit is set to backwards and rate parameters are changed based on destination requirements.

- RM cell is sent back through the network.
  - Network indicates congestion by modifying a rate field in the RM cell.

- RM cell arrives at source.
  - Source calculates new data rate based on RM cell values.

Slide #14

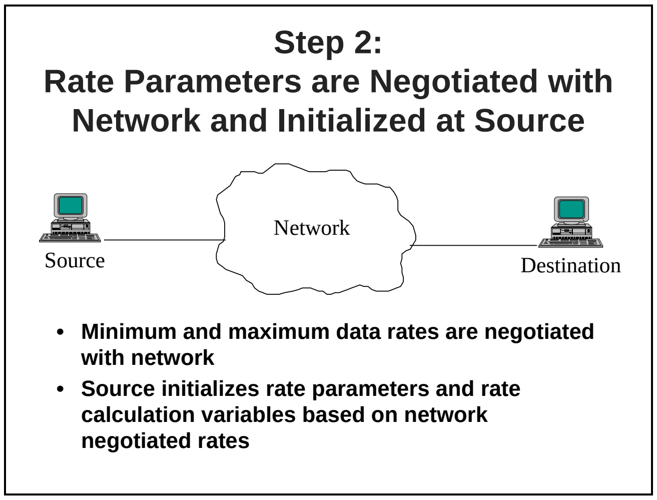
Step 1: Call Setup Request from Source



- **Source makes call setup and rate parameters request**
- **The network may alter these parameters before granting permission for the call**

A connection from the source end station to the destination end station is initiated by a call setup request from the source. During the call setup, rate (and other) parameters will be negotiated with the network. After the network has granted permission for the call, rate parameters such as the peak cell rate and the minimum cell rate are initialized to the values that the network has agreed to support.

Slide #15



Step 2:
Rate Parameters are Negotiated with Network and Initialized at Source

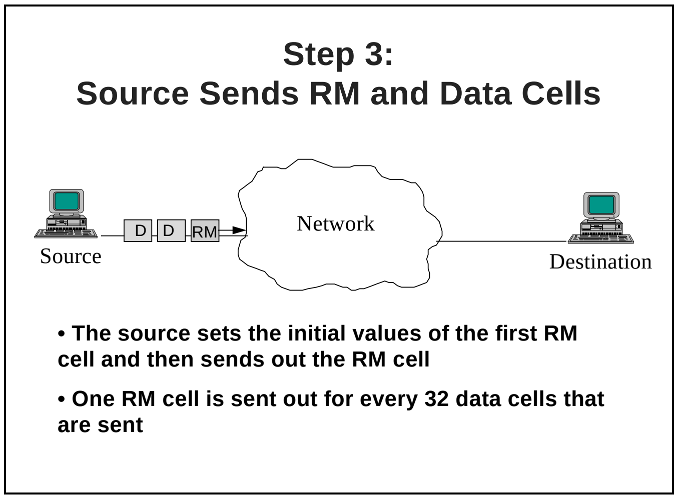
Network
Source
Destination

- Minimum and maximum data rates are negotiated with network
- Source initializes rate parameters and rate calculation variables based on network negotiated rates

Now that the source rate parameters have been set-up, the source must set the initial values of the rate calculation variables.

One of the most important rate calculation variables that the source must keep track of is the Allowed Cell Rate, ACR. The source sets the ACR to the initial cell rate (negotiated with the network) after being idle.

Slide #16



Step 3:
Source Sends RM and Data Cells

D
D
RM
Network
Source
Destination

- The source sets the initial values of the first RM cell and then sends out the RM cell
- One RM cell is sent out for every 32 data cells that are sent

The source must also set the initial values of the first RM cell:

- Current Cell Rate (CCR) = Allowed Cell Rate (ACR)
  The CCR and the ACR are the same value during RM cell generation. The CCR travels in the RM cell and the ACR is the value kept by the source.

- Explicit Rate (ER) = Peak Cell Rate (PCR)
  The ER is the value in the RM cell that will be used by the network and the destination to feed congestion information back to the source. The network and the destination can alter this value. Initially it is set to the maximum rate: PCR.

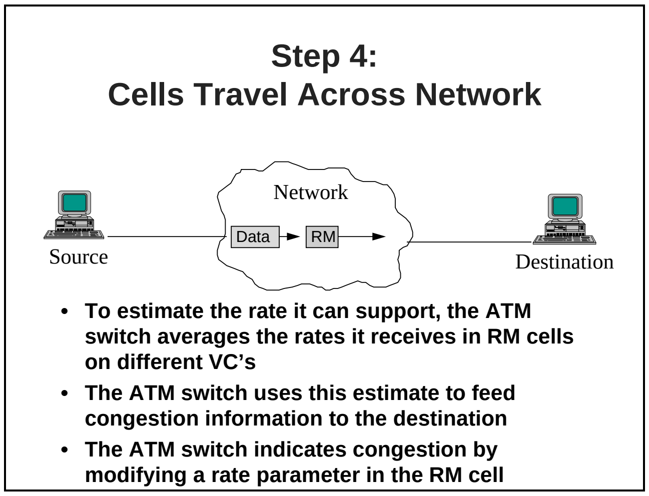
- Direction = forward

- Congestion Indication = none

- Minimum Cell Rate = 1

After the first RM cell is sent, 32 data cells are sent. This process continues for the duration of the connection: one RM cell is sent out for every 32 data cells.

Slide #17



Step 4: Cells Travel Across Network

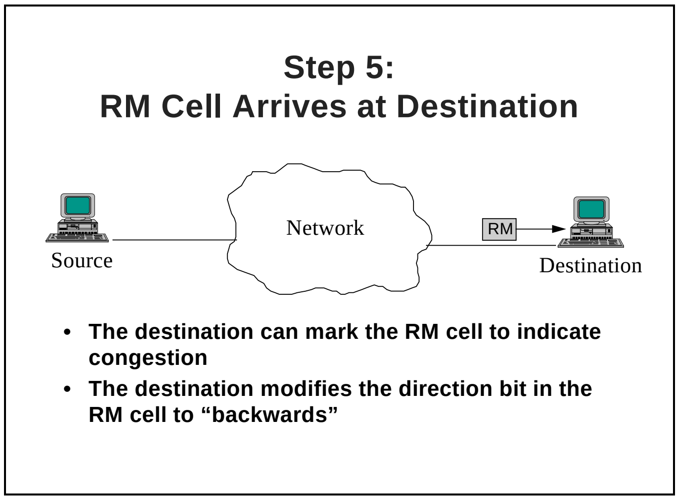
Network
Data
RM
Source
Destination

- To estimate the rate it can support, the ATM switch averages the rates it receives in RM cells on different VC's
- The ATM switch uses this estimate to feed congestion information to the destination
- The ATM switch indicates congestion by modifying a rate parameter in the RM cell

The network switches will mark the RM cell based on congestion encountered:

- The ATM switches and the destination end station may modify the Explicit Rate (ER) field of the RM cell to whatever rate they can support.
- Each switch develops an estimate of the rate it can support by averaging the VC rates in the RM cells.
- The ER field may not be increased. The switch indicates congestion by decreasing the ER but may not indicate surplus capacity by increasing this field. This prevents congestion information from being lost as the RM cell travels through the network. For example, the congestion information would be lost if one switch reduced the ER and the next switch was allowed to increase the ER. Therefore, rate increases are handled at the source.

Slide #18



Step 5: RM Cell Arrives at Destination

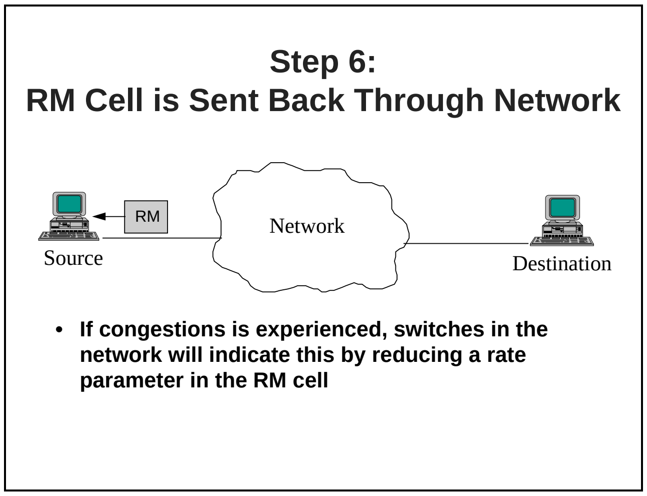
Network
RM
Source
Destination

- The destination can mark the RM cell to indicate congestion
- The destination modifies the direction bit in the RM cell to "backwards"

The destination will mark the RM cell based on congestion encountered:

- The destination end station may modify the ER to whatever rate it can support.
- The destination may also set the CI bit based on congestion to support older congestion feedback methods.
- The destination modifies the direction bit to "backwards" and sends the RM cell back to the source.

Slide #19



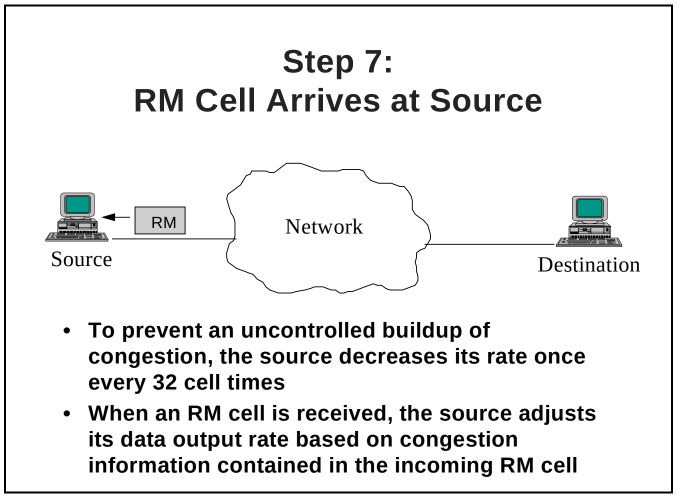
Step 6: RM Cell is Sent Back Through Network

RM
Network
Source
Destination

- If congestions is experienced, switches in the network will indicate this by reducing a rate parameter in the RM cell

As the RM cell travels back through the network to the source, each switch must examine the ER value in the RM cell and determine if it can support that rate. If not, it must reduce the ER to the rate it can support.

Slide #20



Step 7: RM Cell Arrives at Source

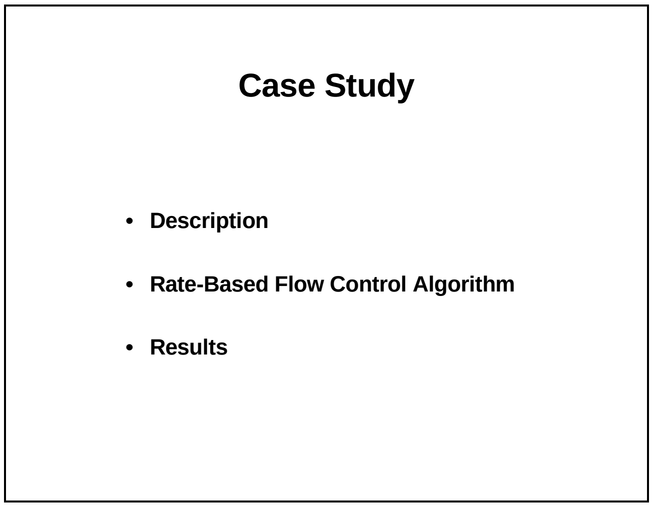
RM
Network
Source
Destination

- To prevent an uncontrolled buildup of congestion, the source decreases its rate once every 32 cell times
- When an RM cell is received, the source adjusts its data output rate based on congestion information contained in the incoming RM cell

When the RM cell arrives at the source, a new rate must be calculated. The source will use certain “rate calculation” variables in conjunction with information in the incoming RM cell to calculate a new rate, ACR. If no congestion was encountered, the source may increase the rate. If congestion is indicated in the ER field of the RM cell, the ACR is set to ER. However, the ACR must be kept within the minimum and maximum rates agreed upon at call setup (MCR and PCR).

Every 32 cell times, the source will decrease the rate. If an RM cell is returned to the source, and no congestion has been experienced, the source may restore the rate. In addition, it may also increase the rate. The rate is decreased every 32 cell times to help prevent an uncontrolled buildup of congestion in the network. If no RM cells are returned to the source due to congestion in the network, the source can help alleviate the congestion by continuously decreasing its rate until an RM cell notifies it that it may calculate a new rate.

Slide #21



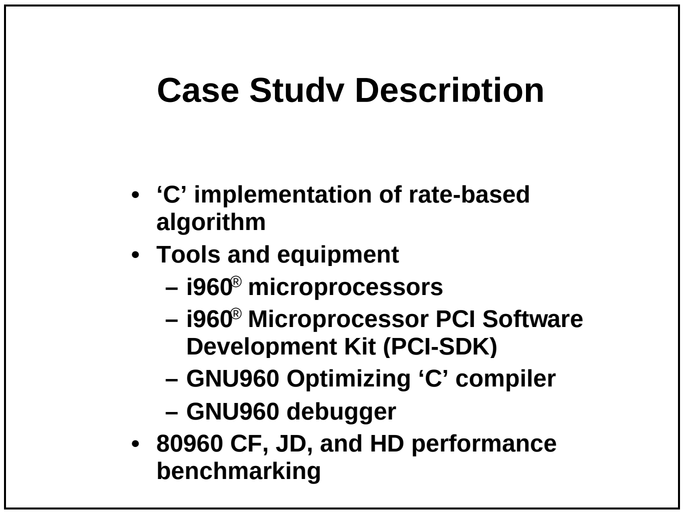
Case Study

- Description
- Rate-Based Flow Control Algorithm
- Results

The purpose of this case study is two-fold:

- To show an ATM function can be implemented in software using an i960® microprocessor
- To provide performance results demonstrating the horsepower of the 80960

Slide #22



Case Study Description

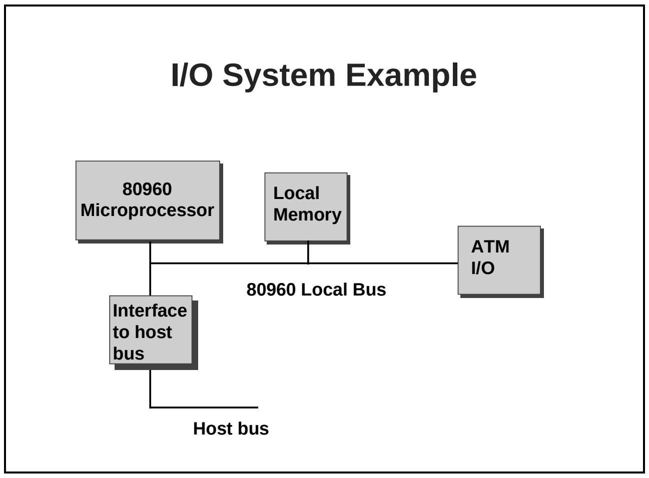
- ‘C’ implementation of rate-based algorithm
- Tools and equipment
  - i960® microprocessors
  - i960® Microprocessor PCI Software Development Kit (PCI-SDK)
  - GNU960 Optimizing ‘C’ compiler
  - GNU960 debugger
- 80960 CF, JD, and HD performance benchmarking

The EPRCA rate-based flow control method was implemented in C and compiled into 80960 processor assembly code. The EPRCA was formulated by Larry Roberts of Connectware/ATM Systems and is subject to change. The actual C code that I implemented is based on his ideas, but may be different from his intended implementation.

The 80960 is supported by a wide variety of compilers, evaluation platforms, debuggers and other tools. I chose to use the GNU960 optimizing compiler and debugger and the i960® microprocessor PCI-SDK board. This board has a PCI interface which may be connected to a host PCI bus and uses the PLX 9060 PCI to 960 bridge chip to provide an interface to the PCI bus from the 80960 processor bus.

The EPRCA algorithm was benchmarked on the 80960CF, 80960JD microprocessors and simulated on the 80960HD microprocessor. The PCI-SDK board has interchangeable CPU modules for the entire 80960 microprocessor family: the S-series, K-series, J-series, C-series, and H-series. Any of these modules can be swapped into the PCI-SDK board to easily determine which processor meets the performance requirements of a given application. The PCI-SDK also has an interface for interchangeable I/O interface target modules.

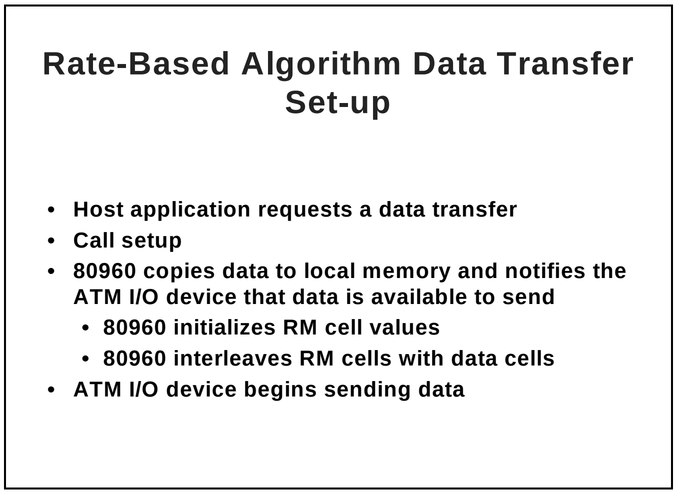
Slide #23



I/O System Example

The diagram above shows a simplified example of how the 80960 would look in an ATM system or on the PCI-SDK board. I've generalized the ATM peripherals into one box labeled ATM I/O - this would contain a SAR (segmentation and reassembly) chip, physical layer devices, etc. On a PCI-SDK this would be the interchangeable I/O target module. There is also a box labeled "interface to host bus." On the PCI-SDK, this is the PLX 9060 PCI to 960 bridge chip.

Slide #24



Rate-Based Algorithm Data Transfer Set-up

- **Host application requests a data transfer**
- **Call setup**
- **80960 copies data to local memory and notifies the ATM I/O device that data is available to send**
  - **80960 initializes RM cell values**
  - **80960 interleaves RM cells with data cells**
- **ATM I/O device begins sending data**

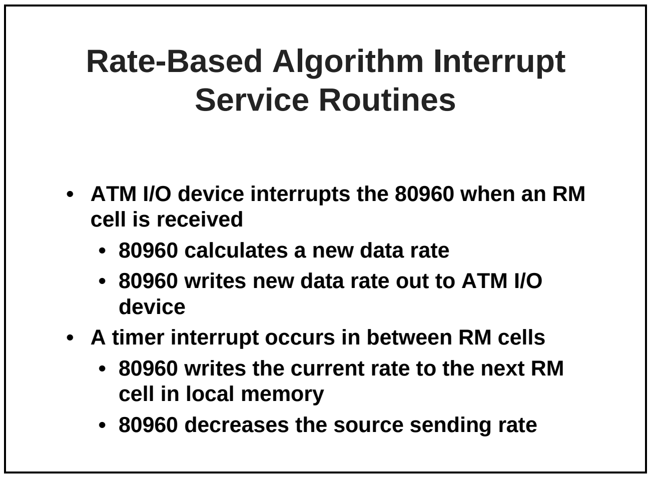
The rate-based flow control algorithm can be divided into two sections, data transfer set-up and RM cell manipulation.

This algorithm performs rate-based flow control from the source end station perspective. Of course every end station must act as a source and a destination. However, the source functions require a much greater amount of processing than the destination and thus were the target of this algorithm.

To begin a transaction, the host application requests a data transfer and the ATM call set-up phase is performed. The 80960 then takes over and transfers the data from the host to local memory. However, since RM cells must be interleaved with data cells, and the ATM I/O device will pull data from memory in a contiguous fashion, the easiest way to insert RM cells into the data stream is to actually interleave them with the data in local memory. As the 80960 copies the data into local memory, it will insert a "blank" RM cell in-between every 32 data cells. The RM cell is not exactly "blank"; four of its five parameters are initialized at call setup and do not change. But the fifth field, the CCR, will be initialized to ACR and updated on the fly. As the 80960 copies the data into local memory, it keeps track of the location of all the RM cells. This information is needed to update the CCR value on the fly as data is sent across the network.

The 80960 notifies the ATM I/O that data is available to send and the data transfer is initiated.

Slide #25



Rate-Based Algorithm Interrupt Service Routines

- ATM I/O device interrupts the 80960 when an RM cell is received
  - 80960 calculates a new data rate
  - 80960 writes new data rate out to ATM I/O device
- A timer interrupt occurs in between RM cells
  - 80960 writes the current rate to the next RM cell in local memory
  - 80960 decreases the source sending rate

Now that the RM cells have been interleaved with data cells, there are two tasks left for the processor:

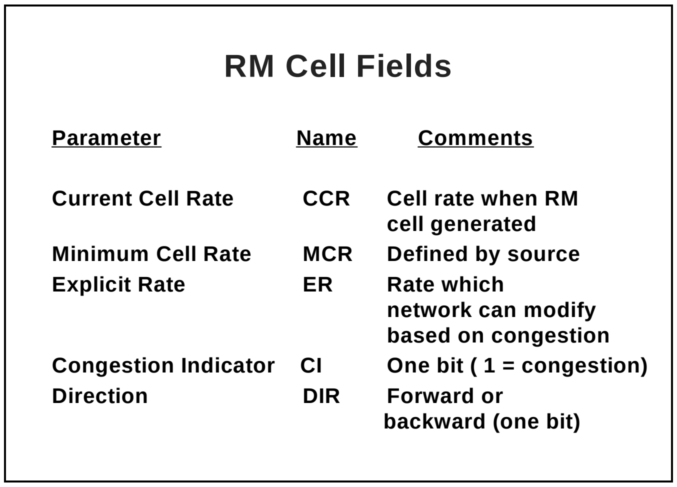
- Process an incoming RM cell
- Update the CCR field in the RM cells in local memory

To implement the function described above, two interrupt service routines (ISR) were developed. Using interrupts allows the processor to perform rate control only when necessary, freeing it to do other work, such as I/O processing.

Let's take the RM cell processing first. The ATM device detects that an RM cell has been received and interrupts the 80960. The 80960 then reads the RM cell values and calculates a new data rate. The 80960 then writes the new rate out to the ATM device.

The CCR is updated by using a timer to time out and cause an interrupt in-between the transmission of RM cells. The 80960 updates the CCR in the next RM cell in memory and decreases the rate.

Slide #26



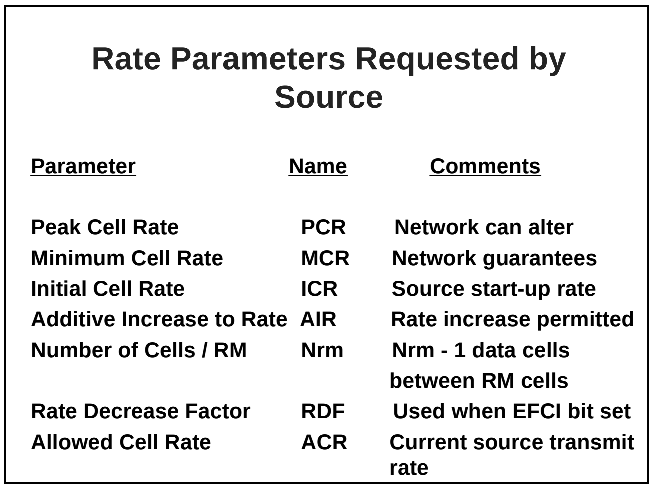
RM Cell Fields

Parameter	Name	Comments
Current Cell Rate	CCR	Cell rate when RM cell generated
Minimum Cell Rate	MCR	Defined by source
Explicit Rate	ER	Rate which network can modify based on congestion
Congestion Indicator	CI	One bit (1 = congestion)
Direction	DIR	Forward or backward (one bit)

The RM cell has five fields that are used by ATM switches, the destination end station, and the source end station. The switches and the destination end station read these fields to determine the current status of the source and feed rate modifications back to the source if they cannot support the current rates.

These fields are initialized based on rates negotiated between the source and the network at call setup.

Slide #27



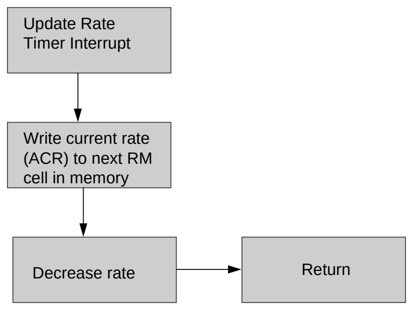
Rate Parameters Requested by Source

Parameter	Name	Comments
Peak Cell Rate	PCR	Network can alter
Minimum Cell Rate	MCR	Network guarantees
Initial Cell Rate	ICR	Source start-up rate
Additive Increase to Rate	AIR	Rate increase permitted
Number of Cells / RM	Nrm	Nrm - 1 data cells between RM cells
Rate Decrease Factor	RDF	Used when EFCI bit set
Allowed Cell Rate	ACR	Current source transmit rate

There are seven source / destination rate parameters that are used to calculate and specify data rates. These rates are initialized at call setup. The ACR is periodically recalculated by the source based on incoming RM cell information. The other six parameters are not changed for the duration of a call.

Slide #28

Rate-Based Algorithm Flow Chart Update Rate ISR


Update Rate Timer Interrupt
Write current rate (ACR) to next RM cell in memory
Decrease rate
Return

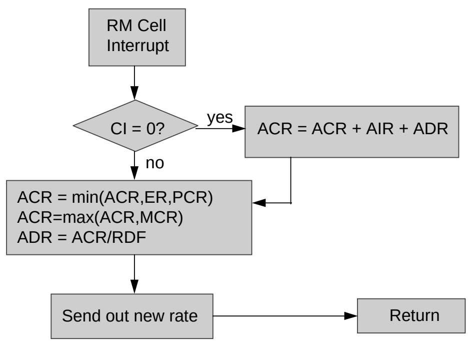
The Update Rate ISR is a bit tricky. The objective is to update the next RM cell that will be sent out by the ATM I/O device. The difficulty is in knowing which RM cell will be sent out next. However, since the 80960 created the RM cells in local memory, it has kept track of where they are located. RM cells are sent out every 32 data cells. This translates to one RM cell being transmitted at a minimum of 87us intervals for one virtual channel at 155Mbps. If two channels were active, two sets of RM cells and data cells must be managed, but must share the 155Mbps bandwidth.

Using the worst case scenario of an 87us interval, the 80960 sets a timer to interrupt in the middle of this interval. At this point the CCR update ISR is triggered. The 80960 knows which RM cell is to be sent next and its location. It updates the CCR with the current value of ACR and writes this to the next RM cell in memory. It also takes advantage of this ISR to decrease the source sending rate. This interrupt is repeated every 87us, thus occurring on approximately the 16th cell of the 32 cell data transfer. The 80960 adjusts the timer value as the data rate is modified.

This method is easily expanded to multiple virtual channels, with the 80960 keeping track of each VC's data set in memory and synchronizing timer interrupts.

Slide #29

Rate-Based Algorithm Flow Chart
Receive RM Cell ISR

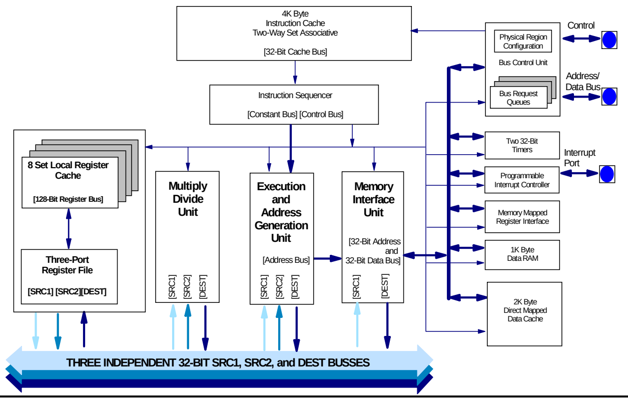


A hardware interrupt is generated when the source receives an RM cell. The RM cell conveys congestion information in the CI field and the ER field. Recall that the network switch and the destination may decrease the ER field based on congestion.

If CI=0, the ACR will be increased by Additive Increase to Rate (AIR) + Additive Decrease to Rate (ADR). ADR must be added to ACR to recover previous decreases in rate. If CI = 1, indicating congestion, the ACR increase is bypassed. The ACR is set to the minimum of ACR or ER but is not allowed to drop below MCR or rise above PCR. The Allowed Decrease in Rate is set to ACR / RDF. This means that rate decreases are not linear but are a function of the ACR.

Slide #30

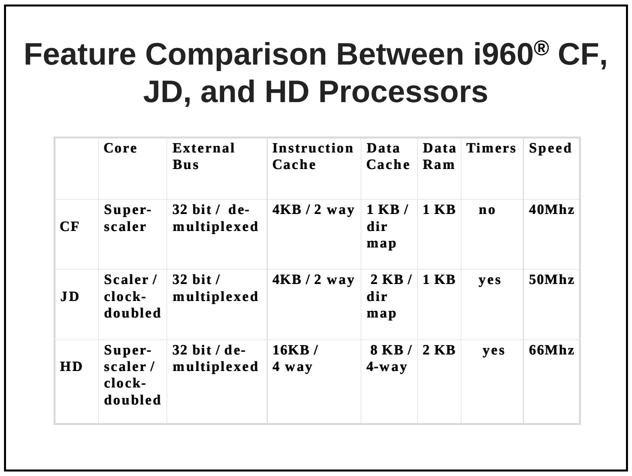
i960® JX Microprocessor Block Diagram


4K Byte Instruction Cache Two-Way Set Associative
[32-Bit Cache Bus]
Instruction Sequencer
[Constant Bus] [Control Bus]
8 Set Local Register Cache
[128-Bit Register Bus]
Three-Port Register File
[SRC1] [SRC2][DEST]
Multiply Divide Unit
[SRC1] [SRC2] [DEST]
Execution and Address Generation Unit
[Address Bus]
[SRC1] [SRC2] [DEST]
Memory Interface Unit
[32-Bit Address and 32-Bit Data Bus]
[SRC1] [DEST]
Physical Region Configuration
Bus Control Unit
Bus Request Queues
Control
Address/ Data Bus
Two 32-Bit Timers
Programmable Interrupt Controller
Interrupt Port
Memory Mapped Register Interface
1K Byte Data RAM
2K Byte Direct Mapped Data Cache
THREE INDEPENDENT 32-BIT SRC1, SRC2, and DEST BUSSES

The 80960 JX microprocessor has many features ideally suited to ATM applications. Here are some of the features useful in flow control:

- Two on-chip timers
    - These can be used in the Update Rate ISR
- Interrupt controller with up to 248 interrupt vectors
- Interrupt vector caching
- On-chip instruction cache
    - Critical routines such as the Update Rate ISR and the Receive RM Cell ISR can be locked into the I-cache to prevent fetches from external memory
- On-chip local register cache
    - 0-8 frames can be reserved for interrupts
- Superscalar (H series and C series)
- High bandwidth burst bus
- On-chip data RAM
- Nine Addressing modes
    - Useful for indexing in writing new CCR values to RM cells.

Slide #31

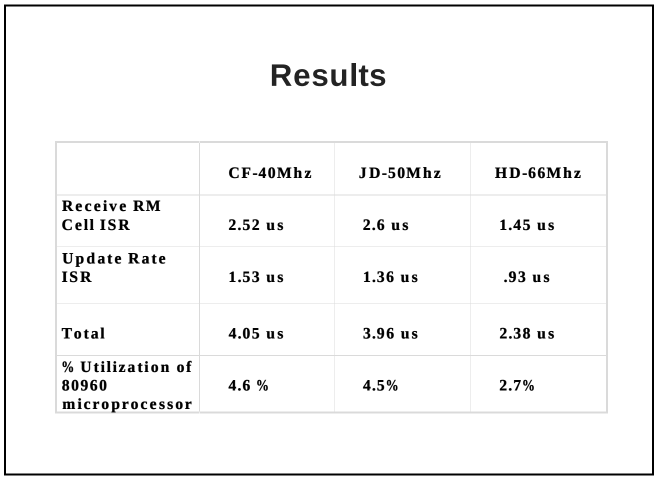


Feature Comparison Between i960® CF, JD, and HD Processors

	Core	External Bus	Instruction Cache	Data Cache	Data Ram	Timers	Speed
CF	Super-scaler	32 bit / de-multiplexed	4KB / 2 way	1 KB / dir map	1 KB	no	40Mhz
JD	Scaler / clock-doubled	32 bit / multiplexed	4KB / 2 way	2 KB / dir map	1 KB	yes	50Mhz
HD	Super-scaler / clock-doubled	32 bit / de-multiplexed	16KB / 4 way	8 KB / 4-way	2 KB	yes	66Mhz

The ATM system designer can choose between a variety of i960® microprocessors and choose the performance level and feature set that best meets his or her needs.

Slide #32



Results

	CF-40Mhz	JD-50Mhz	HD-66Mhz
Receive RM Cell ISR	2.52 us	2.6 us	1.45 us
Update Rate ISR	1.53 us	1.36 us	.93 us
Total	4.05 us	3.96 us	2.38 us
% Utilization of 80960 microprocessor	4.6 %	4.5%	2.7%

The performance results show that the 80960 processor family members can easily perform rate-based flow control with plenty of horsepower to spare.

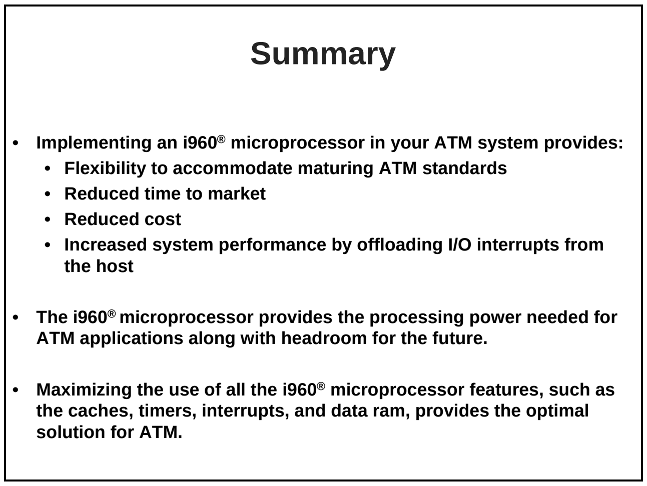
At 155 Mbps, one cell time is approx. 2.7 us:

$$2.7 \text{ us /cell} = \frac{53 \text{ bytes/cell} * 8 \text{ bits/byte}}{155 \times 10^6 \text{ bits/sec}}$$

One RM cell is sent every 32 cell times. At 155Mbps, the time between RM cells is 87us (87us = 32 cell times * 2.7 us/cell).

The i960® processor results, at under two cell times at 155 Mbps for processing an RM cell and updating the rate, show that more than 30 cell times are available for other processor tasks. This means that more than 95% of the processor's horsepower is available for other tasks, such as I/O processing and other ATM functions.

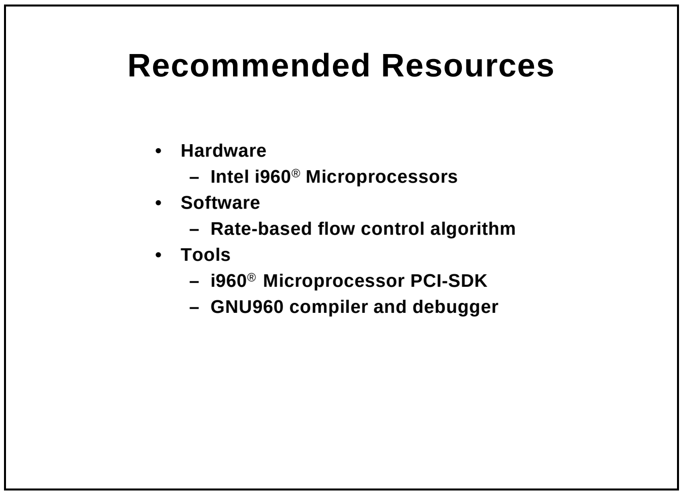
Slide #33



Summary

- Implementing an i960® microprocessor in your ATM system provides:
  - Flexibility to accommodate maturing ATM standards
  - Reduced time to market
  - Reduced cost
  - Increased system performance by offloading I/O interrupts from the host
- The i960® microprocessor provides the processing power needed for ATM applications along with headroom for the future.
- Maximizing the use of all the i960® microprocessor features, such as the caches, timers, interrupts, and data ram, provides the optimal solution for ATM.

Slide #34



Recommended Resources

- Hardware
  - Intel i960® Microprocessors
- Software
  - Rate-based flow control algorithm
- Tools
  - i960® Microprocessor PCI-SDK
  - GNU960 compiler and debugger

This paper outlined many of the design challenges of ATM and how an embedded microprocessor such as the 80960 processor can be used to solve them.

ATM systems employing an embedded microprocessor provide maximum flexibility for evolving standards, reduce time to market, eliminate costly hardware redesigns, and differentiate the design by offloading I/O interrupts from the host.

Implementing ATM Rate-Based Flow Control Algorithms Using an Embeddded Microprocessor

The Decrease ISR is triggered by a software interrupt from the Next Cell Time ISR or from a timer 2 time-out interrupt. Both the software interrupt and the timer 2 interrupt will cause the Decrease ISR to be entered once every Nrm (32) cell times. However, the software interrupt for the Next Cell time ISR causes a decrease during active mode and timer 2 is used only during inactive mode.

If activity mode equals inactive then timer 2 must be re-initialized to Nrm (32) cell times inside this ISR to cause a constant decrease in rate while inactive. The ACR is decreased by ADR but is not allowed to fall below MCR.

Rate-based flow control algorithm operation can be described in terms of two modes:

- Activity mode
    - Active
    - Inactive
- Cell Status mode
    - Cell to Send
    - No Cell to Send

When an end station is idle or "inactive" and a new connection is initiated, an RM cell must be immediately sent out. The ACR is decreased and the activity mode is set to active. In active mode, data and RM cells are sent out. An RM cell is sent out every Nrm (32) data cells and the ACR is decreased.

When there are no cells to send out, the activity mode is changed to inactive and the ACR is decreased every Nrm (32) cell times until ACR = ICR.

A hardware interrupt is used to notify the processor that a cell is ready to be sent. The cell status is tracked in a boolean variable called "Cell to Send" and is set equal to one or "true". The activity mode is checked, and if found inactive, a software interrupt is generated and program execution jumps to the Next Cell Time ISR.

If the activity mode equals active, execution returns to normal program flow. The active mode implies that the Next Cell Time timer, timer 1, is running. The function of this timer is explained on the following page. Since a cell can only be sent at the source's allowed rate, the cell to be sent out must wait until the Next Cell Time timer expires.

The Next Cell Time ISR flowchart details the data and RM cell event scheduling. This routine is triggered by an interrupt which indicates that the next cell time (NCT) has arrived.

If the source has a cell to send out, the activity mode is set to active and a timer is set up to interrupt after one cell time. This timer, the Next Cell Time timer, or timer 1, generates an interrupt when it times-out. This interrupt is set up to vector to the Next Cell Time ISR so that data and RM cells may be scheduled. The timer value corresponds to ACR controls the rate at which data and RM cells are sent out.

The next step is to check the value of count to determine whether a data cell or an RM cell should be sent. Count is a variable which is used to count the number of cell times that have elapsed and is decremented once every cell time. If count equals zero, an RM cell is sent out. The source then re-initializes count to 32, obtains all of the RM cell parameters, and sends out a new RM cell. Otherwise a data cell is sent.

If an RM cell was just sent then the rate is immediately decreased. However, if a data cell was sent out, the source will wait until the next RM cell is sent before decreasing the rate. This has the effect of decreasing the rate once every 32 cell times. An incoming RM cell can recover the rate decrease and also increase the rate if the RM cell parameters indicate that the network can support it.

If the Next Cell Time ISR is triggered but there is no cell to send out, then the activity mode is set to inactive, the Next Cell Timer, timer 1, is disabled, and a second timer, timer 2, is set to time-out every Nrm (32) cell times. Timer 2 is used in inactive mode to trigger rate decreases.