The Memory-Integrated Network Interface

Our zero-copy ATM Memory-Integrated Network Interface targets 1-Gbps bandwidth with 1.2-µs latency for applications that need to send or receive data at very frequent intervals. Applications can send or receive packets without operating system support, initiate packet transmission with one memory write, and determine packet arrival. At the same time the operating system can use MINI for communications and for supporting standard networking software. Simulations show MINI "bounces" a single ATM cell in a round-trip time of 3.9 µs at 10 Mbytes/s.

Ron Minnich

David Sarnoff Research Center

Dan Burns

Frank Hady

Supercomputing Research Center

The bandwidth and high latency of current networks limit the set of cluster computing applications to those that require little use of the network. Running a distributed computation on a network of 100 or more machines for five minutes allows only a few seconds of communications if speedup is to reach acceptable levels. Thus such applications must require only a byte of communication for every thousand to ten thousand floating-point operations. While these sorts of low communication rate applications exist, studies of other types of applications[1] show a much higher frequency of communication with large amounts of data, requiring application-to-application latency on the order of one microsecond.

New high-bandwidth networks should widen the scope of cluster computing applications to include those requiring substantial communications. Asynchronous transfer mode (ATM) in particular provides a clear long-term path to gigabit-per-second bandwidths. However, the commercial network interfaces designed for these new networks are not suitable for a large set of distributed and cluster computing applications. These interfaces require operating system interaction each time a message is sent or received, greatly increasing latency and decreasing throughput. We needed an interface that does not involve the operating system in sending or receiving messages but does support optimized versions of TCP/IP (Transmission Control Protocol/Internet Protocol) and NFS (Network File System).

Our Memory-Integrated Network Interface (MINI) architecture integrates the network interface into the memory hierarchy, not the input/output hierarchy, as shown in Figure 1, next page. Typical implementations of network interfaces use the I/O bus. They suffer from bandwidth and latency limitations imposed by DMA start-up times, low I/O cycle times (compared to the CPU cycle times), limited I/O address space, multiple data copying, and CPU bus contention. Placing the network interface on the other side of main memory avoids these problems.

MINI meets certain application-driven performance goals. That is, we based the architecture of the interface on our experiences with applications that have succeeded (and failed) in a cluster and a distributed programming environment. We specified the performance shown later for two point-to-point connected workstations. We did not consider a switched system in the goals, as the ATM switch market remains immature.

The goals are as follows:

1. 1-µs application-to-application latency for small, single ATM cell messages. (Measured from the initiation of a host send command until the data is loaded into the target node's memory, excluding fiber delay.)

2. 1-Gbps sustained application-to-application bandwidth for large (~4 Kbyte) messages.
3. A multiuser environment. (Measurements on 1 and 2 take place in this environment).
4. Interoperability with an existing network standard (ATM).
5. Support for high-performance (zero-copy) TCP/IP and NFS.
6. Direct network access to 256 Mbytes of main memory, 64 Mbytes mapped at a time.
7. Nonblocking polling of message receipt status.
8. Support for many virtual channels (1K to 4K).

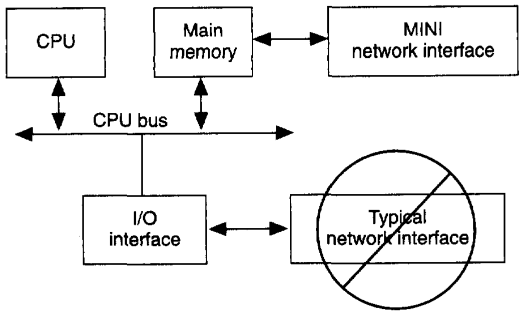

Figure 1. MINI network interface hierarchy.

Hardware architecture

Figure 2 shows the major components of the MINI architecture. MINI has several memory-addressable areas that allow communication between the host and network interface. The physical layer interface card (PIC) block controls host access to these memory-addressable areas, converting the DRAM accesses issued by the host. This block is the only part of the architecture that must be redesigned when porting the design to another host. Note that we currently use a 72-pin SIMM bus, the timing and pinout of which is standard on most workstations and personal computers.

Before using the interface, a process must reserve and initialize channels into the network. MINI allows user processes or the operating system to have one or more channels, each corresponding to an ATM virtual channel (VC), into the network. The VC/CRC (cyclic redundancy check) control table contains two control word entries for each VC, a send and a receive. Each control word contains a pointer to a double-page area in main memory, the VC status, and the ATM cell header information. The operating system initializes control words whenever a process requests a channel into the network. Also, part of the channel initialization process allocates to the requesting process the double page indicated by the control word. MINI supports a maximum of 4K VCs when page size is 4 Kbytes, 2K VCs for 8-Kbyte pages, and 1K VCs for 16-Kbyte pages.

Setting up the channel is relatively slow, since it involves creating an ATM circuit with a remote host; once complete, the channel operates with very little overhead. User processes or the operating system can initiate a send with a write to either the high- or low-priority FIFO buffer.

The MINI control logic uses the information in the high- or low-priority FIFO along with the proper entry of the VC control table to read the required data directly from main memory, segment the data into ATM Adaptation Layer 5 cells, and send it into the network. (AAL5 specifics ATM cells for computer-to-computer communications.) An update

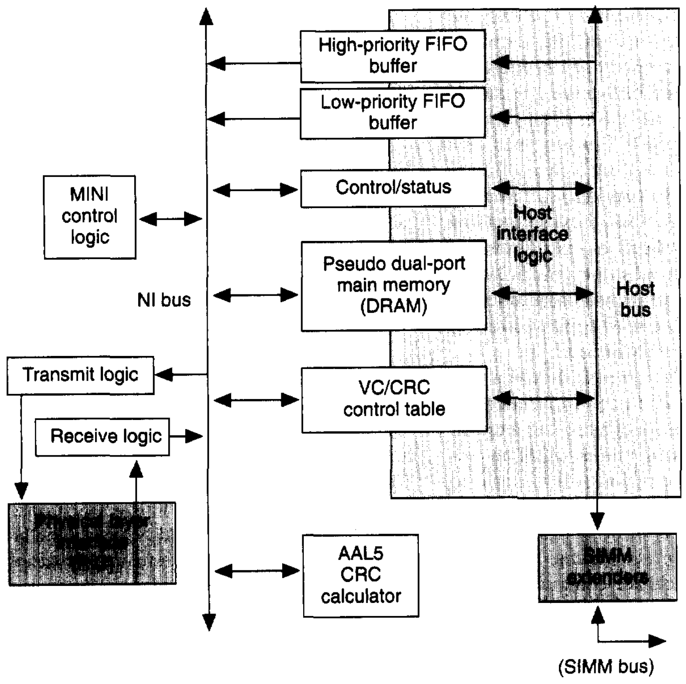

Figure 2. MINI architecture.

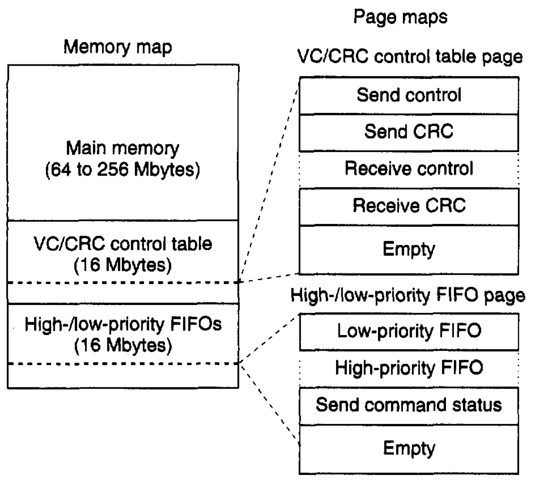
Memory map
Main memory (64 to 256 Mbytes)
VC/CRC control table (16 Mbytes)
High-/low-priority FIFOs (16 Mbytes)
Page maps
VC/CRC control table page
Send control
Send CRC
Receive control
Receive CRC
Empty
High-/low-priority FIFO page
Low-priority FIFO
High-priority FIFO
Send command status
Empty

Figure 3. Host memory and maps.

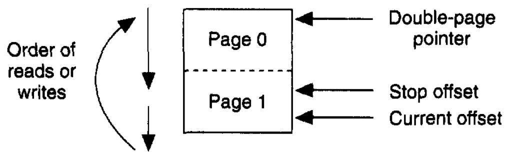

Figure 4. Double pages data handling.

of the VC control table send word notifies the sending process of a completed send. The control logic reassembles cells arriving from the network directly into main memory at a location stored in the VC control table. User processes or the operating system can check arrival status with VC control table reads at any time.

The main memory map of a machine featuring a MINI interface appears in Figure 3. The structure of the memory map is key to allowing a user process to send and receive data without operating system involvement in a multiuser environment. At channel setup time, the control logic allocates six pages to a channel. Two pages in main memory space hold the receive data and two more hold the send data. One page within the VC/CRC control table space is user read only and initialized by the operating system at setup time. This page contains the send and receive control words, and a send CRC and receive CRC. The final page allows a user process to write to the high- or low-priority FIFO. Using the address bus to indicate the channel number on which the send is to

Table 1. The VC table transmit control word.

Item	Bits
Upper VCI bits	4 to 6
GFC*	4
Stop offset	10 to 12
Double-page pointer	13 to 15
Current offset	10 to 12
Header HEC	8
Last cell header HEC**	8

*Generic flow control
**Header error correction

Table 2. The VC table receive control word.

Item	Bits
PDU count	8
Status	7
Dropped-cell count	10 (MSB is sticky)
Double-page pointer	13 to 15
Current offset	10 to 12
Stop offset	10 to 12

take place achieves multiuser protection.

VC/CRC control table. This dual-port SRAM table holds setup and status information for each channel. Information used to send messages into the network remains in the 64-bit transmit word, while information used upon receiving data stays in the 64-bit receive word. The transmit control word (see Table 1) includes a pointer to a double page allocated by the operating system as the channel's data space. MINI sends data from within the double page, as shown in Figure 4, and generates ATM headers using the data stored in the send word.

Sending starts at the current offset, which is updated after each ATM cell transmits. MINI stops sending after reaching the stop offset. When MINI reaches the end of a page of memory during a send or receive, it wraps to the beginning of the double page. It can place the data in page 1, with the header at the end of page 0 (right before the data) and the trailer at the start of page 0 (right after the data). This layout permits sending and receiving data in a page-aligned manner, even if a header and trailer are present. An example of its use appears in the NFS discussion later.

The receive control word (Table 2) includes a current offset, stop offset, and double-page pointer used as described for the transmit word. A protocol data unit (PDU) count increments each time a full AAL5 message is received, giving the process using the channel a convenient place to poll for message arrival. When the interface drops arriving cells, possibly due to contention for main memory, the drops are recorded in the dropped cell count field. This field features a "sticky" most significant bit so that no drops go unnoticed. Status bits denote whether the channel represented by the control word has been allocated and record possible errors. They also determine whether to place arriving cells on 6-word boundaries (packed) or on 8-word boundaries with AAL5 CRCs and real-time clock values placed in two extra words.

Tables 1 and 2 contain a number of fields specified as ranges. The size of the pages used by the host determines the size of the current and stop offsets held. A 512-word page (4 Kbytes) requires an offset of only 10 bits while a 2K-word (16 Kbytes) page requires a 12-bit offset. MINI provides address space for 256 Mbytes of network-mappable memory. This requires a 15-bit double-page pointer for 4-Kbyte pages and a 13-bit double-page pointer for 16-Kbyte pages. MINI allows 64 Mbytes of memory to be mapped into the network at any one time. This number follows from MINI's support for 4K channels, each referencing a pair of 8-Kbyte double pages.

The same dual-port SRAM used to hold the VC control words also holds a transmit and receive AAL5 CRC value for each VC. Each time a cell passes across the network interface bus, the AAL5 CRC calculator fetches the channel's CRC value. (This occurs in parallel with the network interface controller fetch of the VC control word). The AAL5 CRC calculator updates the channel's CRC to include the passing cell, and then stores the result in the CRC table (again in parallel with the VC control word store). This method of calculating the CRC allows for arbitrary interleaving of cells from different VCs.

High- and low-priority FIFOs. A user or operating system host write to either the high- or low-priority FIFO initiates message sends. During this write, MINI takes the start and stop offsets within the channel's double page from the host data bus. For user process-initiated sends, MINI latches the virtual channel identifier (VCI) on which the send is to take place from the address bus. Therefore a process can initiate a write on a given VC only if it gains write access to the proper address. A special case of VCI equal to zero allows the kernel to access all channels without having to gain access (and fetch TLB entries) to a separate page for each channel. When the host address bus indicates a request to send on channel zero, MINI latches the VC number from the data bus rather than the address bus. The kernel always occupies channel zero.

Two FIFOs assure that low-latency messages can be sent even when the interface is currently sending very large messages. A process performing bulk data transfer will send large messages using the low-priority FIFO. Another process can still send a small, low-latency message by writing to the high-priority FIFO. MINI will interrupt any low-priority sends between ATM cells, send the high-priority message, and then resume sending the low-priority message.

MINI does not stall unsuccessful writes to the command FIFOs. To determine if a send command write was successful, the process can read the send command status word associated with each channel.

Pseudo dual-port memory. This section of the host's main memory is directly accessible to the network interface. We decided to make this space very large, possibly encompassing all of the host's memory. Due to its size, storage for this block must be constructed from dynamic RAM.

Since dual-port DRAMs do not exist, we will initially construct this block with off-the-shelf SIMMs and multiplexers, and a pseudo dual-port main memory. The network interface will take control of the memory during unused processor-memory cycles. When the processor accesses main memory, MINI will relinquish its use of the SIMMs in time for the processor to identify a normal access.

Arbitration for use of main memory is much less complex when the host has a memory busy line back to the CPU. When the CPU detects a main memory access, MINI could assert the memory busy line, causing the CPU to wait until MINI completes its memory access.

The best method for sharing main memory between the host and network interface would be to implement a true dual-port DRAM. Such a DRAM would simultaneously fetch two rows and decode two column addresses. Collisions for the same row would cause the second requester, either the host or network interface, to wait. Unfortunately, such chips do not yet exist.

MINI control logic. The state machine within the MINI control logic moves data between the transmit and receive FIFOs and main memory, updating VC/CRC table entries, processing transmit commands, and segmenting or reassembling ATM cells. The MINI control logic operates on one ATM cell at a time and services the received cells first. This minimizes the probability of receive logic FIFO overflow. When no received cells are present, MINI moves cells from main memory to the transmit logic in response to send commands from the host.

Cell receipt involves the following steps:

1. Using the VCI in the received cell as an index into the VC/CRC control table, MINI fetches the channel's receive control word and CRC.
2. The receive control word forms a main memory address, and MINI initiates a store and updates the CRC.
3. When the pseudo dual-port main memory begins the requested write, MINI increments the offset counter, requests the next write, and updates the CRC.
4. Step 3 repeats until six words have been written to main memory.
5. MINI stores the updated receive control word and CRC in the VC/CRC control table.

When the final cell of an AAL5 PDU is received, MINI stores two extra words (the locally calculated AAL5 CRC and the real-time clock value) at main memory addresses succeeding the final data words. Also on a final cell, MINI initializes the channel's CRC control table entry. If the flag called Pack Arriving Cells has not been set, MINI stores the real-time clock value and the current AAL5 CRC value after each cell. Note that if the stop offset is reached while storing incoming cells, no further writes to the main memory occur, though VC/CRC control table updates still take place.

Cell transmission involves the following steps:

1. The VCI held in the high- or low-priority FIFO reads the receive word and CRC from the VC/CRC control table.
2. MINI sends the ATM cell's header, formed using information from the VC control table receive word, to the transmit FIFO while a main memory read is requested.
3. When granted use of main memory, the network interface initiates a read from the address specified by the FIFO and VC control table contents.
4. MINI sends the word read from main memory to both the transmit FIFO and the CRC logic, increments the offset counter, and initiates a read from the next address location.
5. Step 4 repeats until six words have been read from main memory.
6. MINI stores the updated VC control send word and CRC in the VC/CRC control table, and the VCI within the network interface controller for use on future cell sends.

Control/status, transmit, and receive logic. The control/status block contains a 100-ns-resolution, real-time clock readable by both the host and network interface, along with board configuration and test logic. The transmit and receive logic blocks contain FIFOs that perform asynchronous transfer of ATM cells between the physical layer interface and the network interface.

Packaging. Implementation of the MINI architecture requires the six different boards shown in Figure 5. Two of the boards, the Finisar transmit and receive modules, are off-the-shelf parts required to implement the physical link into the network. These modules have a 16-bit interface that runs at 75 MHz. The PIC board is the network interface to the Finisar conversion board that translates 64-bit network interface words into Finisar 16-bit words. The network interface board contains all the hardware necessary to implement the MINI architecture. The main memory is off-the-shelf, 16-Mbyte SIMMs residing on the network interface board. SIMM extenders plug into the host's SIMM slots, extending the memory bus to the network interface board.

The separation of the PIC functions from the network interface board eases the migration of the design to other hosts or physical media. For instance, to use the design over different

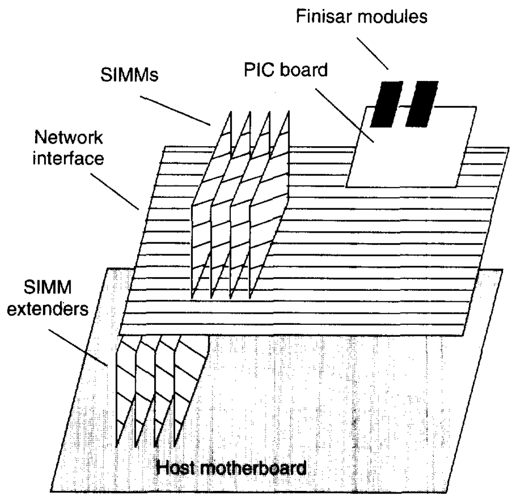

Figure 5. MINI boards.

physical media, designers would replace the PIC board and Finisar modules. To change from an Indy host to a Sun, DEC, or HP system, designers would redesign host-dependent portions of the network interface board, using the same 64-bit interface to the PIC. This eliminates the need to redesign the physical layer.

Flow control. MINI provides hardware support for the low-level form of flow control developed by Seitz to keep the network from dropping cells.[2] MINI sends a stop message from a receiving node to the transmitting node (or switch port) when the receiving FIFO fills past some predetermined high-water mark. Once the receiving FIFO is less full, the receiving node restarts transmission by sending a start message.

The MINI architecture also supports software-implemented flow control. In the event that it becomes necessary to drop incoming cells, the MINI hardware keeps a count of the number of cells dropped on a per-VC basis within the VC control table. Real-time clock values automatically passed within messages may also prove useful.

Performance. A detailed VHDL simulation of the initial MINI design gathered some performance statistics.

Throughput. ATM cells are packaged and transmitted at a rate of one cell every 400 ns (20 MINI control logic state machine states), or 0.96 Gbps. Note that this is the real data transfer rate. The total bit transfer rate, including ATM header information is 1.12 Gbps. The PIC board removes the cells 64-bits at a time from a transmit FIFO at a rate of 1.2 Gbps and sends them 16 bits at a time (at 75 MHz) to a Finisar

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase® Smarter legal research.