The Myrinet High-speed Network

LISHA/UFSC
Thiago Robert dos Santos
Prof. Dr. Antônio Augusto Fröhlich
{ robert | guto }@lisha.ufsc.br
http://www.lisha.ufsc.br/~{ robert | guto }

September 2003
Parallel Computing
10 Years Ago

- MPP
- Mainframe
- PC
- Vector Supercomputer
- Workstation
Parallel Computing Today
Clusters of Commodity Workstations

- Commodity off-the-shelf processors interconnected by high-speed off-the-shelf networks acting as a multicomputer
- Cluster performance is highly dependent on the interconnecting system
  - Low latency
  - High bandwidth
  - Scalable
- Cluster interconnect options
  - Myrinet (Myricom)
  - SCI (Dolphin)
  - QsNet (Quadrics)
  - InfiniBand (Intel, ...)

September 2003
http://www.lisha.ufsc.br
Myrinet

- Cost-effective, high-performance, high-availability packet switching technology
- ANSI/VITA 26-1998 standard
- HW and SW interfaces are open and published
  - Encourage researchers to explore many communication design options
- Composed of crossbar switches and network adaptors
  - Source routing (worm-hole)
Myrinet Crossbar Switches

- Source routing (worm-hole)
  - Cut-through switching method
  - Forwarding of a packet begins as soon as its destination is determined, even before the whole packet had arrived
- Arbitrary topology
- Scalable to thousands of nodes
Myrinet Network Adaptor

- Programmable communication device
- Architecture
  - LANai 32-bits RISC processor
  - SRAM memory
  - 3 DMA engines
    - send, receive, host
  - PCI interface
LANai

- RISC Processor
- Packet Interface
- DMA/Checksum Engine
- Memory Interface
- Myrinet-Link Interface

LBUS
SRAM, EEPROM

EBUS
PCI

Myrinet SAN

LANai Core
LANai Chip
Myrinet Control Program (MCP)

- Compiled by the LANai GNU cross-compiler (*lanai-gcc*)
- Loaded by the host into Myrinet SRAM
  - With an ordinary loader
  - Built-in in the driver
    - Image -> string
  - Entry point at 0x00000000
LANai Special Registers

- Used to configure and control LANai operations and report operations status
- Control interrupt generation
- Memory-mapped
- Can be accessed both by the LANai on-chip processor and from the EBUS
Data Communication

- Packets are injected into the network and consumed off of the network by the *packet interface*
- The DMA Engine is responsible for data transfers between the SRAM and the host system memory
- Three kinds of DMA
  - Receive DMA (packet interface)
  - Send DMA (packet interface)
  - EBUS-LBUS (PCI interface)
Implementation

- Development of a component that provides means to
  - access a Myrinet NIC
  - initialize it
  - send a buffer to the Myrinet SAN
  - receive a buffer from the Myrinet SAN
Accessing the LANai

- Linux char device interface - easy way to access the NIC
- PCIMAP (PCI Mapping tool) can be used to handle PCI related operations
  • probes for a specific device (Myrinet NIC)
  • provides a way to map the device memory to user virtual memory – mmap()
LANai 4.X Memory Layout

<table>
<thead>
<tr>
<th>Storage Type</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROM</td>
<td>0x00080000</td>
</tr>
<tr>
<td>Special Registers</td>
<td>0x000C0000</td>
</tr>
<tr>
<td>FPGA</td>
<td>0x000E0000</td>
</tr>
<tr>
<td>SRAM</td>
<td>0x00100000</td>
</tr>
</tbody>
</table>

The memory layout above could be mapped to a structure like this:

typedef struct {
    Myrinet_PROM prom;
    Myrinet_Registers regs;
    volatile unsigned int fpga[MYRINET_FPGA_SIZE / sizeof(int)];
    unsigned char sram[MYRINET_RAM_SIZE];
} Myrinet;
Initializing LANai 4.x

- The initialization procedure for LANai 4.x is described in the documentation provided by Myricom
- After initialization and before any network access the TIMEOUT, MYRINET and WINDOW special registers must be initialized
Special registers involved

- **CLOCK** – Clock Configuration
- **MP** – Memory Protection
- **IMR** – Interrupt Mask
- **EIMR** – External Interrupt Mask
- **MYRINET** – Myrinet-Link Configuration
- **WINDOW** – SAN-Link Sampling Window Selection
- **TIMEOUT** – Incoming Message Blocking Timeout Selection
Send/receive

- Easier way
  - Sender
    - copies the buffer to be sent to the NIC SRAM \( \text{memcpy}() \)
    - uses the send DMA engine
  - Receiver
    - uses the receive DMA engine
    - copies the received data from the SRAM to the user buffer
Special registers involved

- SB – Send Byte
- SH – Send Half-Word
- SW – Send Word
- SA – Send-Message Alignment
- SMP – Send-Message Pointer
- SML – Send-Message Limit
Special registers involved

- RB – Receive Byte
- RH – Receive Half-Word
- RW – Receive Word
- RMP – Receive-Message Pointer
- RML – Receive-Message Limit
EBUS-LBUS DMA

- More elaborate way:
  - use the EBUS-LBUS DMA engine to do the copying instead of \texttt{memcpy}()}
Special registers involved

- LAR – LBUS Address Register
- EAR – EBUS Address Register
- DMA_CTR – DMA Counter
- DMA_DIR – DMA Direction
ISR Special Register

- Interrupt Status Register
- Contains the LANai chip status information
- Each bit in ISR has a meaning:
  - `send_int (3)` – signals the completion of a send DMA
  - `recv_int (1)` – signals the completion of a receive DMA
- Used in all operations:
  - wait for a DMA completion
  - check the initialization status
  - interrupt generation
References

- LANai 4.x documentation provided by Myronic
  - initialization
  - DMA engines
  - send/receive examples

- Definitions for LANai 4.x (file myrinet.h)
  - structures corresponding to the LANai 4.x memory layout
  - bit mapping constants for special registers