

Advanced Computer Systems Architecture

Chip-Multiprocessors: Applications and Architectures

CSE 526M

Prof. Patrick Crowley

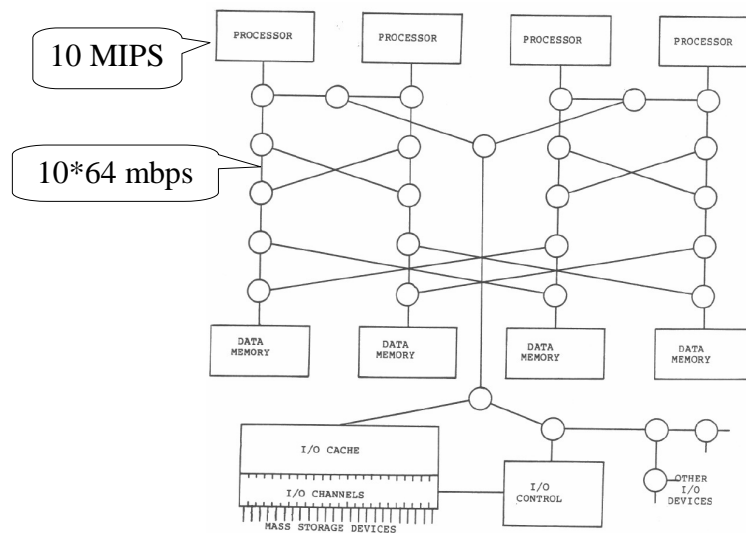
Plan for Today

- Announcements
 - Sandy Fraser talk tomorrow, 11am
- Questions
- Today's discussion

HEP

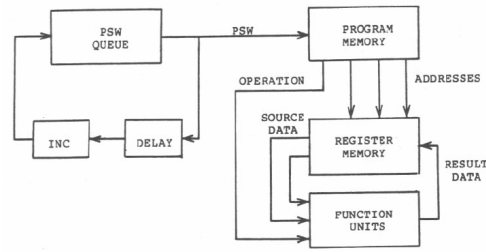
- Heterogeneous Element Processor
- Large, parallel computer built for scientific applications
- Processors are pipelined, each stage holds an instruction from a different thread
- System consists of processors, switches, links, memories and I/O units

System Organization



Processor Organization

- Process status word contains 20-bit PC
- 2K, 64-bit general purpose registers
- There is a minimum 8 cycle latency between instructions from a given thread
 - Need 8 threads for full utilization



Loads and Stores

- The scheduler function unit (SFU) moves data between memory and registers
- On load/store, the SFU
 - Sends a packet with the address, return address for processor and process ID, and 64-bits of data (for a store)
 - Removes the requesting PSW from the control queue, adds it to its own stalled queue
 - Handles the response packet

Control Flow

- Conditional branches can be implemented by storing, loading and modifying the current PSW
- Instructions also exist to create and terminate threads (i.e., processes)

Managing Shared Data

- Each data memory location is either “empty” or “full”
- Each register is “empty”, “full” or “reserved”
- A location is emptied when it is read, and filled when it is written
 - A load, e.g., can wait for a location to be full before consuming the data

Processes, Tasks and Jobs

- A set of processes with the same protection domain is a *task*
- A *job* consists of one or more tasks
 - Tasks within a job have disjoint registers and program memory, but share data memory
- Each job must indicate its max number of active tasks, the OS only loads the job when sufficient resources are available

Interconnection Network

- Each switch has 3 bidirectional ports
- Each port has its own routing table, which maps destination addresses to output ports (these are populated during system configuration)
- When two or three packets need to use the same output port, the highest priority packet is sent, and the others are routed elsewhere (no buffering)
- Packet priority increases with each re-direction
- Switch nodes check packet parity, errors are reported to a diagnostic subsystem

I/O

- User processes make I/O requests via supervisor calls (which implement file systems, and page mappings, etc.)
- The high speed I/O systems consists of
 - *I/O cache modules*, which attach to the switch to up to 32
 - *I/O channels*, which carry data at 1.2MB/s, and are controlled by
 - *I/O control processors*, which accept requests from processes (via memory locations) and handles completed I/O requests from channels

Files

- When a file is opened
 - An explicit number of frames are allocated in the I/O cache
 - A sequential direction is indicated (for future reads/writes); most I/O instructions have their effect in this direction
 - A random I/O operation is available to modify the current file position explicitly
- The supervisor process manages cache contents explicitly
 - Determines the page containing the requested data, moves those pages into the cache
 - Schedules read-ahead, write-behind with I/O control processor
 - A reference count is held for each cached page; when count is zero, the page can be purged (needed when files are shared between processes)

HEP Programming

- Fortran specific
 - CREATE statement spins a subroutine call off into its own thread
 - Normal subroutines can become their own thread with the RESUME statement
 - A special syntax (\$) prefix that allows access to full/empty variables for data sharing
 - Registers and local vars are allocated dynamically, so any routine can be called in any thread, at any time
- Synchronization can be achieved via empty/full states

Intellectual Digestion

- How does this flavor of multithreading compare to that of the MEs?
- Compare the HEP interconnection network to the interconnection approach used in the IXP. What are their relative strengths?
- Assuming similar I/O resources and learning curve, how would your project implementation have gone with the HEP? Faster/Slower? Harder/Easier? Better/Worse?

Multithreading

- HEP
 - Fine-grained (“context-switch” each cycle)
 - Dynamic number of threads, under program control

Interconnection

- IXP
 - On one chip, links are metal layers
 - Uses several buses
 - Good match for VLSI
- HEP
 - Links are cables
 - Switched network
 - Field-extendable

Assignment

- Tuesday (4/27):
 - **Commentary:** Reflections in a Pool of Processors/ An Experience Report on C.mmp/Hydra
 - 15 minute perspective: Bloom Filter
- Thursday (4/29)
 - **Commentary:** The Cosmic Cube
 - 15 minute perspective: DDOS Shield/Firewall