Advanced Computer Systems Architecture

Chip-Multiprocessors: Applications and Architectures

CSE 526M
Prof. Patrick Crowley

Plan for Today

• Questions
• Today’s discussion
Multiprocessors, Multicomputers and the Intel IXP

- Parallelism: Implicit vs explicit
- Control Structure of Parallel Systems
- Communication Mechanisms
- Examples of Physical Organizations
- CMPs and the Intel IXP

Implicit vs Explicit Parallelism

- Implicit: a programmer writes a serial program and it gets executed in a parallel fashion
- Explicit: a programmer indicates how the parallelization should happen
- Parallelism can be expressed/exploited at various levels of granularity, from individual instructions up to completely independent programs
Parallelism

• In what order should these instructions execute? Can they overlap?

Add R1, R2, R3  #R1 holds result
Add R4, R5, R6  #R4 holds result
Add R7, R8, R9  #R7 holds result

Dependences

• What about these instructions?

Add R1, R2, R3  #R1 holds result
Sub R4, R1, R5  #use of R1
And R6, R1, R7  #use of R1
Or R8, R1, R9   #use of R1
Xor R10, R1, R11 #use of R1
Limits on Implicit Parallelism

- Dependences
  - Data hazards
- Control flow
  - Branches introduce uncertainty as to which instructions need to be executed (i.e., control hazards)
- These apply to both pipelining and superscalar issue, and, generally, to any serial or implicitly parallel microprocessor

Opportunities for Explicit Parallelism

- What about this code?
  
  ```c
  for (i=0; i<1000; i++)
      c[i] = a[i] + b[i];
  ```
Multiprocessors

- Have been secondary to uniprocessors recently (the last 20 years or so)
- Have strong appeal if you can build them with uniprocessors and associated technologies
  - If you build a parallel system out of individual computers, it is traditionally called a multicomputer
- As always, seem to be the path to future performance as uniprocessor returns diminish
  - Newer ILP-based techniques are becoming complex while providing modest performance improvements
  - Chip-multiprocessors are on the (mainstream) horizon

Programmer’s Perspective

- Two important distinctions
  - Control structure of the parallel platform
    - How are parallel tasks expressed?
  - Communication mechanism
    - How are interactions between parallel tasks carried out?
Flynn’s Taxonomy (1966)

- Single Instruction stream, Single Data stream (SISD)
  - Conventional uniprocessor
- Single Instruction stream, Multiple Data stream (SIMD)
  - One global control unit
  - Examples
    - Vector processors
    - Intel’s MMX and SSE (Streaming SIMD Extensions)
- Multiple Instruction stream, Single Data stream (MISD)
  - Commercial examples?
  - Some prototype stream processors might match (e.g., Imagine)
- Multiple Instruction stream, Multiple Data stream (MIMD)
  - The most general, most common

(MIMD) Shared-memory vs Message Passing

- These are both communication mechanisms
- SM: single shared-address space (extension of uniprocessor)
  - SMP or UMA: Symmetric shared-memory MultiProcessor, Uniform Memory Access (centralized, shared bus)
  - NUMA: non-UMA (decentralized “distributed shared memory”, interconnection network)
    - NUMA-CG: cache coherent (directory-based protocol)
    - NUMA w/out cache coherence
    - Chastens
    - COMA: cache-only memory architecture
- MP: processors communicate by messages (send, receive)
  - Multicomputers (message-passing paradigm)
  - Synchronous (e.g., RPC) vs. asynchronous (sender doesn’t wait for reply)
Shared-Memory Multiprocessor

Distributed-Memory Multiprocessor

© 2003 Elsevier Science (USA). All rights reserved.
Clusters: Google Floorplan

Rack Study
Shared-Memory vs. Message Passing

- An old debate that is not that important today
  - Many systems are built to support a mixture of both
  - Shared virtual memory (e.g., network of workstations; coherence at the page level)
- Shared-memory pros
  - Ease of programming
  - Good for communication of small items
  - Less overhead on O.S.
  - Hardware-based cache coherence
- Message-passing pros
  - Simpler hardware (more scalable)
  - Explicit communication (both good and bad)
  - Easier for long messages

Parallel Processing Caveat

- Multiprocessors are used to
  - Speed computations
  - Solve larger problems
- Recall Amdahl’s law
  - If $x\%$ of your program is sequential, speedup is bounded by $1/x$
- At best linear speedup (if no sequential code sections)
  - Superlinear speedup occurs due to larger or more numerous caches and memories
- Speedup is limited by the communication/computation ratio and synchronization
SMPs

- Most common form of multiprocessor today
- Based on off-the-shelf unprocessors
- Presence of caches:
  - Reduces contention on memory bus
  - Exacerbates the coherence problem, which we’ll talk about later

Pentium II Xeon Example
(440GX AGPSet)
Multiple Processors on One Chip

- Why?
  - Technology (i.e., transistor densities) permits it
  - High compute throughput
  - Low power requirements due to simpler circuits
  - Some applications match nicely
- Some Examples
  - Sun’s MAJC (4 processors)
  - IBM’s Power4 (2 processors)
  - Broadcom’s SiByte (2 processors)
  - Intel’s IXP 2800 (17 processors)
- Seems (to me) to be the wave of the future
- Note: all parallel programming issues still apply to CMPs!

Intel IXP2800

- Application: networking
- Novel aspects
  - Chip-multiprocessor
    - ARM-compatible XScale core
    - 16 embedded RISC-like cores
      - Each supports 8 thread contexts in hardware
      - Each has its own local instruction and data memory
  - Multiple, heterogeneous memory interfaces
  - Multiple, heterogenous I/O interfaces
  - Great programmer flexibility and responsibility
**IXP2800 Logical Organization**

**Future Discussions**

- Once we’ve completed a simple project on the IXP, we will re-vist some of these topics, along with a few others
- **Topics**
  - Communication costs
  - Interconnection networks
  - Process-Processor Mapping & Techniques
  - Algorithm design and performance modeling
Assignment

- Post the MPOC commentary if you have not done so
- Readings:
  - Commentary: The Next Generation of Intel IXP Network Processors