CS523
Advanced Computer Architecture

Dr. A. Sahu
CSE IIT Guwahati

Advanced Computer Architecture

- Who needs high performance systems?
- How do you achieve high performance?
- How to analyze or evaluate performance?

- Power Performance Tradeoff : Green Computing
- Best architecture/design for a problem
- Parallel Architecture: Design and Programming

Who needs high performance systems?

- Who needs high performance systems?
- How do you achieve high performance?
- How to analyze or evaluate performance?

- Power Performance Tradeoff : Green Computing
- Best architecture/design for a problem
- Parallel Architecture: Design and Programming

Who needs high performance systems?

- Who needs high performance systems?
- How do you achieve high performance?
- How to analyze or evaluate performance?

- Power Performance Tradeoff : Green Computing
- Best architecture/design for a problem
- Parallel Architecture: Design and Programming

Who needs high performance systems?

- Who needs high performance systems?
- How do you achieve high performance?
- How to analyze or evaluate performance?

- Power Performance Tradeoff : Green Computing
- Best architecture/design for a problem
- Parallel Architecture: Design and Programming

Course Structure

- Pipeline processor principles and design, Instruction set architecture; Memory addressing; ILP; Hazards: dynamic scheduling, branch prediction **Intel Pentium and Core Architecture**
- Memory hierarchy: Cache, Disk, Cache Policy, BUS **Special emphasis on Analytical Study and Mathematical Modeling**
- Benchmarking and Performance Evaluation: SPEC, Media, Graphics Benchmarks
- Multiprocessor introduction: Shared-memory architectures and their synchronization and consistency issues **Case Study: Locking hardware and algorithm Task Scheduling on Multicore System**

Course Structure

- Green Computing: IBM Cell Processor (Multicore), CICSO Network Processor (Multicore), Low Power Processor Architecture Case Study : **Intel Atom and Arm Cortex, Low Power Scheduling, Data Center and Ware house Processing Case Study: Google Data Center**
- Advanced multi-core topics; Transactional Memory; Interconnection networks and NOC Routing, Massive Cores Reconfigurable Mesh
- Multi-core Cache Data Placement, Prefetching and Bandwidth Scheduling Shared Prefetching, Token/Probabilistic Scheduling
- Massive Multicore Programming Model: **GPU (NVIDIA Cuda and MAC/ATI OpenCL), OpenMP, Cilk and MPI**
- Five Recent popular papers on Muti core Design (from ISCA, HPCA and PLDI)

A Sahu slide 7

Grading Policy & General

- Class timing & Venue
  - Monday 4-5PM, Tuesday 3-4PM, Wednesday 2-3PM
  - Venue : 2001
- Grading
  - Mid Semester 30%
  - End Semester 40%
  - Lecture Note Scribing 30% (Latex + Xfig)
    - It will make a Good Book after the semester Over
- Attendance
  - 75% mandatory
- No single Best book is available for this course
  - So lecture note scribing is essential

A Sahu slide 8

Course Website

- http://jatinga.iitg.ernet.in/~asahu/cs523/
- Course Contents
- Text and Reference Books
- All lecture slides
- Summery of each class with references
- Other information
  - Simulators, Benchmarks, Source Code,
  - Referred Papers, EBooks

A Sahu slide 9

RISC/CISC

- RISC
  - Simple set of Instruction (32 bit)
  - All arithmetic must be in from Register
  - Load/Store are separated from arithmetic
  - Simple addressing mode
  - Result: **Simple Compiler**
- CISC: Complex
- OISC : **One instruction set computer**
  - **SBN (Subtract and Branch if Negative)**
  - **MOVE**

```
subneg a, b, c
        // Mem[b] = Mem[b] - Mem[a]
        //if (Mem[b] < 0) goto c
```

- ISE : Instruction set extension for Application

A Sahu 10

RISC/CISC

RISC	CISC
• Simple set of Instruction (32 bit) •All arithmetic must be in from Register •Load/Store are separated from arithmetic •Simple addressing mode • **80+ instructions** •Result: Simple Compiler • Simpler hardware • Power Efficient	•Variable Length Instructions •Complex Addressing mode • Load/Store may be mixed with arithmetic instruction •Result: **Compiler is complicated** • **5000+ instructions** •Complex hardware •Very high performance if compiler is efficient • Power hungry

Intermediate: **Instruction Set Extension**
DSP, NetworkProcessor, CryptoProcessor
Add application/domain specific instruction to RISC set

11

OISC (One Instruction Set Computer)

- Minimalistic Approach (Hardware)
- Synthesize others instructions
- Example : SBN, MOVE
- SBN (Subtract and Branch if Negative)

  SBN *a, b, c* // Mem[*b*] = Mem[*b*] - Mem[*a*] , if (Mem[*b*] < 0) goto *c*
- MOVE : MOVE src, dst

```
MOVE  src, acc      // load acc //LDA src
MOVE  acc, dst      // store acc  // STA dst
MOVE  src, add      // add to acc  // ADD src
MOVE  src, sub      // subtract from acc  // SUB src
MOVE  dst, pc       // jump indirect  // JMP dst
```

A Sahu 12

Power-Performance Tradeoff

- JOBs per Time
- JOBs Per Watt
- ATOM, ARM
- Power Aware Scheduling

A Sahu 13

Multicore Scheduling

- Single Processor Scheduling
  - OS Book: FCFS, SJF, SRM, PS, RR, RR+PP,
- Real time scheduling
  - EDF, Periodic Task Scheduling, Mixed Scheduling
- Power Aware scheduling
- Multicore scheduling
  - List scheduling, FCFS scheduling, Constraints Scheduling
- Power aware multicore scheduling
- Distributed scheduling and load balancing
  - Work Stealing, Receiver Initiated, Sender Initiated,

A Sahu 14

Data Center: Task/Job Scheduling

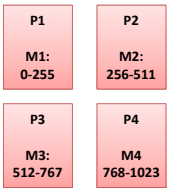
- MapReduce (Hadoop)
  - Mathematical Example :
- Parallel Machine
  - If N = 100000
  - Parallel Map (Square function for each X_i)
  - Reduction (Sum all square term to one sum S)
- Google/Yahoo: Map-reduce
  - MAP: **Count All words indivisibly in all document**
  - Reduce: **Index the sum properly** to access/search the document

$$S = \sum_{i=1}^{N} X_i^2$$

A Sahu 15

Data placement in Multicore: Algorithmic Treatment

- Suppose 4 processor: 1024 memory location
- All are share memory, any processor can access any location



Suppose in an application
- Pi access Mj, Aij times

Local access takes less time
Remote access takes more time

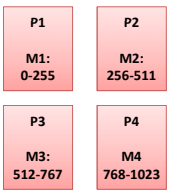
Optimize memory access time by address remapping to memory module

A Sahu 16

Data placement in Multicore: Algorithmic Treatment

Stable Matching Problem
Algorithm, Eva-Tardos Book Chapter 1

Processor accessing a memory more frequently should be nearer.
Memory accessed mostly by a processor should be nearer to that processor.
Every Processor have memory preference list and Every memory have Processor preference list



P1: M1, M3, M4, M2 M1 : P1, P3, P2, P4
P2: M1, M2, M3, M4 M2 : P2, P1, P3, P4
P3: M4, M1, M3, M2 M3 : P3, P1, P4, P2
P4: M2, M3, M1, M4 M4 : P2, P4, P1, P3

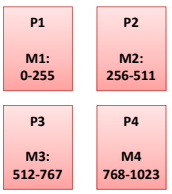
We have to assigned one processor to one memory and also one memory to one processor

Match should be stable (Preferred one should be allocated as far as possible)

A Sahu 17

Data placement in Multicore: Algorithmic Treatment

- Suppose 4 processor: 1024 memory location
- All are share memory, any processor can access any location



Variations of this problem

- Remote access time depends on remote location from source processor
- Each memory module can be of different size (Min to Max)
- Access pattern change time to time
- Caching and Copying

A Sahu 18

Lower Bound of Sorting

- Sequential Version O(NlogN)
- Parallel Version O(logN) using N Processor
  - Cole's Merge: Sorts N numbers in EREW
- Can we reduce if I give infinite processor ?
  - Any specific architectural feature assumption!
  - Lets assume
    - Given a Grid of Processor if one can broad cast message in a row/column in O(1) time independently
  - YES: one can sort in O(1) time using $N^{3/2}$ processor
  - Ref : Vaidyanatan book (Reconfigurable Arch)

A Sahu 19

Books : Text

- Patterson, D.A., and Hennessy, J.L. , "**Computer Architecture : A Quantitative Approach** ", Morgan Kaufmann Publishers, 5th Edition, Inc.2011
- Dezso Sima, Peter Kacsuk, Terence Fountain, " **Advanced Computer Architectures : A Design Space Approach**", Pearson Education India, 1997
- Michael J Flynn, "**Computer Architecture: Pipelined and Parallel Processor Design** ", Narosa Publishing India, 2003
- Some Recent Papers

A Sahu slide 20

Books: References

- Patterson, D.A., and Hennessy, J.L. , "**Computer Organization and Design: The Hardware/Software Interface**", Morgan Kaufmann Publishers, 4th Edition, Inc.2005, Ebook 3rd Ed
- Kai Hwang, **" Advanced Computer Architecture: Parallelism, Scalability, Programmability**", McGraw-Hill, first edition, 1992.
- Ramachandran Vaidtyanathan and J L Trahan, **" Dynamic Reconfiguration: Architectures and Algorithms** ", Kluwer Academic Publisher, New York, 2003
- David Kirk and Wen-mei Hwu " **Programming Massively Parallel Processors: A Hands-on Approach**", Morgan Kaufmann Publishers, 2010
- P Pacheco " **An Introduction to Parallel Programming**", Morgan Kaufmann Publishers, 2011
- David Culler, J.P. Singh and Anoop Gupta, "**Parallel Computer Architecture: A Hardware/Software Approach**", Morgan Kaufmann, first edition, 1998.
- Harvey G Cragon, " **Memory Systems and Pipelined Processors**", Narosa Book Distributors, India, 1998
- **Introduction to Parallel Programming**", Morgan Kaufmann Publishers, 2011

A Sahu slide 21