ECE 452: Computer Organization and Architecture

IN

Microprocessors

- Understand basic structure of a microprocessor

Addressing Modes

- Understand different addressing modes for operands in instructions

Assembly and C Language

- Program in assembly and C language

Memory and I/O

- Understand basic microprocessor memory and I/O system organization

Pre-requisites

- ECE 251with a minimum grade of C

Concepts:

- Overview of computer organization
- Hardware technology: CMOS, clock, timing
- Microarchitecture design for supporting instructions
- Instruction set architecture (datapath, control path)
- Pipelined processor design
  - Basic solutions for handling data and control hazards
  - Basic branch prediction techniques
  - Basic exception and interrupt handling techniques
  - Modern superscalar and VLIW processors
- Memory system
  - Memory hierarchy
  - Cache memory design and performance
  - Secondary memory (magnetic disks, Flash/SSD)
  - Virtual memory: paging, swapping
- I/O systems
  - I/O organization
  - On-chip bus and network-on-chip architectures
  - Off-chip bus architectures (e.g., PCI-E)
- Multicore, multiprocessor, and cluster systems
  - Basic concepts of multithreading
  - Basic concepts of GPUs
  - Basic concepts of grid and cluster computing
  - Basic concepts of machine learning accelerators
- Ethics in computing

Applications:

- Case studies of processor core architectures, including multicore CPUs and GPUs, and machine learning accelerators

Tools:

- Assembler and runtime simulator for a contemporary instruction set architecture
- Architectural simulator for processor-memory system design and exploration

OUT

Logic-level Hardware

- Evaluate logic level hardware characteristics that would affect system performance

Datapaths and Control

- Design simple datapaths for processors
- Design control paths using hardwired logic or microprogramming

Instruction Parallelism and Pipelining

- Exploit instruction level parallelism with pipelining to accelerate execution

Data Parallelism

- Analyze data parallel GPU/CPU systems
- Exploit data parallel programming and vector instructions to improve performance

Multicore/Multithreaded Parallelism

- Analyze and exploit core and node (thread) level parallelism, to improve performance, fault-tolerance, and energy efficiency

Memory and Storage

- Determine effective memory access latencies under a hierarchical memory storage system
- Architect and analyze cache hierarchies

Network-on-Chip, I/O Systems

- Analyze on-chip networks and I/O system performance, energy, and reliability
- Design for latency and bandwidth constraints

Reviewed 1/2023