# Final Exam Review

Format of exam similar to midterm: short questions, multiple choice, and quantitative questions

Topics will focus on material after midterm

## Compiler optimization

- Components of backend of compiler
  - Key components of register allocation and scheduling
- Register allocation using graph coloring
- Instruction scheduling
  - List scheduler
- Compiling for ILP processors
  - Need larger regions to schedule
  - Region based scheduling
    - Trace scheduling, superblock, hyperblock etc.
    - Construct traces of more frequently used regions
    - Schedulers may use speculation, predication etc.

## Design of Memory Hierarchies

### Key Principles
- Locality – most programs do not access code or data uniformly

### Cache Design Goal
- Cache is level closest to processor
- Small size, fast access time

### Cache Design Questions
- Placement Organization: block size, direct, associative, set associative
- Replacement policy: LRU, FIFO etc.
- Addressing
- Write strategies

## Cache Performance Models and Challenges

- Model in terms of cache miss rate/hit rates, and miss penalty
  - Actual depends on instruction mix and number of accesses to memory (inst and data)
- Improving cache performance implies developing techniques to
  - Improve miss penalty
  - Improve miss rate
  - Improve time to access cache
- Multiprocessor cache coherence problem
  - Need cache coherence protocols
    - Snoopy bus based protocols are easy to implement, typically use write invalidate protocol.
Multiprocessor Architectures

- Connect multiple processors to form a high performance computer architecture
  - Use shared memory design
  - Communicate through shared variables
  - Or distributed memory
  - Communicate using interconnection networks - network topology and send/receive protocols needed
- Problems of synchronization and coordination
  - Synchronization constructs include barrier, send/receive, locks, semaphores
  - Synchronization instructions constitute an overhead and can slow down the process
- SIMD and MIMD parallel machines
  - SIMD simpler to build but not flexible, MIMD more difficult to manage

Parallel Algorithms and Networks

- Network topology affects performance
  - Latency in network plays role in communication time
  - Can have static topology or dynamic topology
- Computational Models for parallel algorithms
  - Idealized model is the PRAM model
  - Can have CREW, EREW, CRCW PRAM models
  - Develop parallel algorithms using a PRAM model
  - Performance measured using speedup and efficiency
  - Efficiency can drop with increase in synchronization and communication time
  - Scalability of parallel algorithms is an issue

Embedded and Reconfigurable Architectures

- Challenges in embedded systems
  - Adds new challenges due to unique constraints
  - Constraints include power, time deadlines, size etc.
  - New compiler optimization criterion
  - Some of the old techniques may no longer apply
- Reconfigurable architectures
  - The datapath and control path can be reconfigured
  - Concept of a “soft core” ISA; change ISA using same chip but different setting of hardware
  - FPGAs are an example of reconfigurable architectures