Midterm 2 Study Guide

- **Multipliers/Dividers**
  - Iterative improvement purposes (what was being reduced)
    - Reasons why registers could be removed/combined, ALUs could be reduced
  - Know the basic progression of each version, but no details. I would remind you in the question of any details you needed to know
  - Work through an example for multiple/divide version 3
    - Show register values for each iteration
  - Booth’s algorithm for multiply
    - What is the purpose of this algorithm
    - Basic idea of how it works
    - Do not need to work through an example

- **Floating point**
  - Convert a decimal number to binary single-precision floating point notation
  - Bias – what is it? What does it facilitate?
  - Single-precision vs. double-precision
    - Ranges
    - Register layout
  - Exceptions
    - Underflow, overflow
    - Infinity, NaN
  - Decimal representation operations
    - Work through addition/subtract multiple/divide with and without round and guard bits
  - Purpose of special bits
    - Round, guard, sticky
  - Why aren’t some FP operations associative? Give example
  - Challenges wrt to FP operations (e.g., precision, accumulated errors)

- **Performance**
  - Calculate CPI based on instruction mix
  - Calculate CPI speedup based on architectural changes
  - Compare CPIs of processors based on instruction mixes
  - Chart on slide 16

- **Pipelining**
  - Basic concept and goal
  - Key points on Lec10-slide 7
  - What makes pipelining hard?
  - Pipeline registers: purpose, overhead incurred, etc
  - Latency vs. bandwidth
    - How does pipelining affect these wrt to a single instruction?
  - Structural, control and data hazards
    - What are they?
    - Do they exist in the MIPS 5-stage pipeline as you have implemented? Why/why not?
    - If a hazard exists (independent of the architecture), what steps or measures can be taken to remove the hazard?
    - How do hazards affect the flow of instructions?
  - Why is it beneficial for all MIPS instructions to take all 5 stages, even if some stages are not used by all instructions?
  - Data dependencies vs. data hazard
  - Data forwarding
    - What is it?
    - How does it work?
    - What are the benefits?
    - How does it ensure correct data flow?
    - What data hazards cannot be removed by forwarding? What must be done?
  - Code evaluation:
• Given some assembly code, identify the data dependencies and data hazards
  o Software scheduling:
    • What is it?
    • Where is it done?
    • Why does it still ensure correct data flow?
    • Given code with load-use hazards, reorder code to remove hazards—See Lec11-slide 19-21
  o Branch hazards
    • Why are branches so bad for performance?
    • Describe the optimization that moves the branch determination from the 4th cycle to the 2nd cycle in your MIPS 5-stage pipeline.
    • What is a branch delay slot? How does it improve performance?
    • Branch prediction:
      • What is it?
      • How does it improve performance?
  o Other data hazards: RAW, WAW, and WAR
    • Identify them in assembly code or give assembly code that exhibits these hazards
  o Calculate speedup wrt pipelining
    • What is the ideal speedup?
    • What hinders achieving the ideal speedup?
    • Calculate pipeline speedup (see Lec11-slide 45) and compare different systems
• Memory hierarchies
  o Benefits of a hierarchical memory approach wrt to performance
  o Spatial vs temporal locality and how memory hierarchies exploit both
  o Terminology Lec 12-slide 8
  o Given an address and a cache configuration, determine the number of bits required for the block offset, index, and tag
  o Tradeoffs (small vs. large):
    • Total size
    • Line (block) size
    • Associativity
  o Calculations
    • Average memory access time given hit/miss rates and penalties
    • Compare different systems
  o Miss penalty components (bandwidth vs. latency)
  o 3 C’s for cache misses (conflict, capacity, cold)
    • Mechanisms to reduce each type
  o What happens on a miss:
    • How is the new location determined wrt to associativity?
    • Block replacement policies
      • Random, LRU, pseudo-random
  o Write policies: Write back vs write through
    • How do they operate?
    • Tradoffs and implications
    • What is a write buffer? What does it improve? How does it work?
  o Write allocate vs. write no-allocate
    • How do they operate?
    • Tradoffs and implications
  o Purpose of dirty and valid bits
• I/O
  o Magnetic disks
    • Basic layout (sectors, tracks, head, arm, etc)
    • Process to access a bit of data
    • Disk access time components
  o Queuing theory
    • Producer server model
    • Throughput vs response time
• How do you maximize/minimize?
• Why do they compete?
• What are the assumptions we have made to make it simpler
• Terminology
  • Arrival rate
  • Time in system
  • Time in queue
  • Time in server
  • Service rate
  • Total system latency = time in queue + time in server
  • Server utilization
• Calculations
  • Server utilization (Lec 13 slide 26)
  • Time in queue (Lec 13 slide 28)
  • Compare time in system (Lec 13 slides 29-30)
• Reliability vs availability
  • Define and compare
  • How can both be improved?
  • MTTF
  • MTTR
  • Calculate availability wrt to MTTF and MTTR
  • Calculate system reliability based on component reliability
  • How does redundancy help?
• Disk arrays
  • Basic principle in the beginning, why did they fall out of usage, and why are they back now
  • Effects on reliability and availability
  • RAID
  • What is the concept of RAID? Why is it important? Why is it useful?
  • Give any possible advantages/disadvantages to using RAID X. If I were to ask you this question, I would say what RAID X does to remind you
  • How do different RAID methods perform for little and big writes?
  • Know the differences between the following RAID models. The table on page 363 might be helpful
    • RAID 1 — mirrored
    • RAID 4 — parity-based with one parity disk
    • RAID 5 — parity-based with the parity spread across all disks
    • RAID 6 — row and diagonal parity
  • How can RAID 6 recover from multiple disk failures? Work through a recovery problem like in the slides
• Error vs fault vs failure
  • What are they, why are they different?
  • How can you prevent one from becoming another.
• Virtual Memory (VM)
  • What benefits does VM provide?
  • How is VM similar to caching? What similarities do they share?
  • How does VM abstract the main vs. secondary memory structure?
  • Page tables
    • What do they store? (know all fields’ purposes)
    • How are located?
    • Where are they stored?
    • How are they indexed?
  • Given an address and a page size, determine the page offset bits and the virtual tag
  • Page faults
    • What is it?
    • What handles page faults and why?
- Page replacement policies
- Optimal page sizes
  - Access rights
    - What are they?
    - Purpose?
    - Protection violation
  - TLBs
    - Purpose
    - Organization
    - Methods to reduce translation time (overlap with cache access, virtually indexed, physically tagged caches)
    - Problems
  - Caches and virtual addresses
    - Aliasing problem with virtually indexed, virtually tagged caches
- Most important questions in the 2006 sample test is question 1 and question 4