CENG 351: Computer Architecture I - Final Exam Study Sheet
(Only topics after Multi-Cycle Processor)

Foundational Concepts (From Previous Sections)
These will not be directly tested but are important to understand topics on the final.

MIPS Instruction Format (recap)
- R-type: opcode (6) | rs (5) | rt (5) | rd (5) | shamt (5) | funct (6)
- I-type: opcode (6) | rs (5) | rt (5) | immediate (16)
- J-type: opcode (6) | address (26)

Instruction Types
- add, sub, lw, sw, beq, j, slt, etc.
- Byte-addressable memory: each memory address refers to 1 byte

ALU Basics
- Performs arithmetic & logic operations
- Controlled by ALU control signals (e.g., Add = 0010, Sub = 0110)

Datapath Components
- Registers: Hold intermediate values
- MUX: Chooses between inputs
- Memory: Instruction + Data memory
- PC (Program Counter): Holds address of current instruction

Control Signals (from single-cycle architecture)
- RegDst, ALUSrc, MemtoReg, RegWrite, MemRead, MemWrite, Branch, Jump, ALUOp

---

Diagrams

1. 5-Stage MIPS Pipeline
Cycle IF | ID | EX | MEM | WB
Instr1: IF | ID | EX | MEM | WB

Instr2: IF | ID | EX | MEM | WB
Instr3: IF | ID | EX | MEM | WB

Stages:
- IF: Instruction Fetch
- ID: Instruction Decode/Register Fetch
- EX: Execute/ALU
- MEM: Memory Access
- WB: Write Back

2. Cache Mapping (Direct-Mapped Example)
Address: [Tag | Index | Offset]
Bits: [20 | 6 | 6] for 32-bit addr, 64 sets, 64B block

- Block Size: 2^6 = 64B
- # Sets: 2^6 = 64
- Index bits = log2(# sets), Offset = log2(block size)

3. Virtual Memory Address Translation
Virtual Address (32-bit): [VPN | Offset]
[20b | 12b] 4KB page

Page Table Lookup:
VPN used to find PPN
Physical Address = [PPN | Offset]

- Page Offset = log2(Page Size) = 12 bits (for 4KB)
- VPN = upper 20 bits of virtual address

---

1. Pipelined Processor

Key Concepts
- A pipeline splits instruction execution into stages to improve throughput.
- Each stage performs a part of the instruction: IF, ID, EX, MEM, WB.
- Pipeline registers store intermediate values between stages.

Hazards

- Data Hazard: Instruction depends on result of previous instruction.
  - Example: add $t0, $t1, $t2 followed by sub $t3, $t0, $t4
  - Solutions:
    - Forwarding (bypass values from later stages)
    - Stalling (insert a bubble)
- Control Hazard: Caused by branches or jumps.
  - Solution: Branch prediction or delay slots, or flush on mispredict.
- Structural Hazard: Two instructions need same hardware in same cycle.
  - Solution: Add more hardware or schedule instructions.

Performance

- Speedup: Ideally = number of pipeline stages.
- CPI (Cycles per Instruction):
  - Ideal CPI = 1 (if no hazards or stalls)
  - Real CPI > 1 due to stalls

Instruction Reordering

- Instructions may be issued out of program order but retire in order.
- Improves performance in deep pipelines.

---

2. Advanced Microarchitecture

Deep Pipelining

- More stages = higher clock speed, but more hazards and complexity.

Multicore Processors

- Homogeneous: All cores are the same (e.g., 4 identical CPUs).
- Heterogeneous: Mixed cores (e.g., high-efficiency + high-performance cores).

Superscalar

- Multiple instructions issued per cycle (e.g., 2-issue, 4-issue).
- Requires dependency checks and instruction window.

Multithreading

- Executes multiple threads on the same core.

- Types:
  - Coarse-grained: switch on cache miss
  - Fine-grained: switch every cycle
  - Simultaneous multithreading (SMT): multiple threads issued per cycle

Out-of-Order Execution
- Dynamic scheduling and reordering of instructions.
- Reorder buffer ensures in-order commit.
- Solves data hazards by renaming registers and tracking dependencies.

SIMD (Single Instruction, Multiple Data)
- One instruction operates on multiple data elements.
- Common in GPUs and vector processors.

---

3. Cache Memory

Memory Hierarchy
- Registers > L1 Cache > L2 Cache > L3 Cache > Main Memory > Disk

Key Definitions
- Hit Rate: Fraction of accesses found in the cache
- Miss Rate: Fraction of accesses not found (1 - Hit Rate)
- Hit Time: Time to access data in the cache
- Miss Penalty: Time to fetch data from lower memory

Formula
- AMAT (Average Memory Access Time):
  AMAT = Hit time + (Miss rate Miss penalty)

Cache Parameters
- Block Size (B): Size of each cache block (in bytes)
- Cache Size (C): Total size of cache
- Number of Blocks (N): C / B
- Associativity (A):
  - Direct-Mapped: A = 1
  - Fully Associative: A = N

  - Set-Associative: A = number of blocks per set

Address Breakdown
- Tag | Index | Offset
  - Offset = log2(block size)
  - Index = log2(# sets)
  - Tag = remaining bits

Cache Organizations
- Direct-Mapped: One block per set
- Set-Associative: N-way set associative
- Fully Associative: Any block can go anywhere

---

4. Virtual Memory

Purpose
- Illusion of large memory per process
- Protection and isolation between processes

Key Terms
- Page: Fixed-size block of virtual memory
- Page Size: Typically 4KB = 2^12 bytes
- Page Offset: log2(page size)
- Virtual Page Number (VPN) and Physical Page Number (PPN)
- Page Table: Maps VPN to PPN
- Page Fault: Page not in memory; requires loading from disk

Address Translation
- Virtual Address = VPN | Page Offset
- Physical Address = PPN | Page Offset

Translation Steps
1. Extract VPN and Offset from virtual address
2. Use VPN to index into page table
3. If valid bit = 1, get PPN
4. Combine PPN with offset Physical Address

5. If valid bit = 0 Page fault Load from disk Update page table

Page Table Organization

- One entry per virtual page
- Valid bit, PPN, Access rights, etc.

Swapping

- If physical memory is full, an existing page is replaced (e.g., LRU)

Quick Reference Formulas

AMAT = Hit time + Miss rate Miss penalty

Offset = log2(Block size)

Index = log2(Number of sets)

Tag = Remaining address bits

VPN = Virtual Address[31:12] (if 4KB page)

Offset = Virtual Address[11:0]