Introduction

- Eight great ideas:
  - Design for Moore's Law.
  - Use abstraction to simplify design.
  - Make common case faster.
  - Parallelism.
  - Pipelining.
  - Prediction.
  - Hierarchy of memories.
  - Dependability via redundancy.

Chapter 1

- Clock Rate (CR) = 1 / Clock Cycle (CC).
- Effective Clock Cycles per Instruction (CPI) = $\sum_{I=1}^{n} = CPI_i \times IC_i$.
- Instruction Count (IC).
- CPU time = IC x CPI x CC or IC x CPI / CR.

Chapter 3

- Locality
  - Temporal Locality: if a memory location is referenced it will most likely be referenced again shortly.
  - Spatial Locality: if a memory location is referenced its neighbouring memory location will most likely be referenced shortly.
- Miss types
  - Compulsory/Cold
  - Conflict miss: replacement algorithm keeps mapping the new data to the same block.
    - Ex. 0, 8, 0, 8, 0, 8 if use mod 4 to map blocks then it'll miss every time.
  - Capacity miss: working set is larger than the cache block.
- Average Memory Access Time (AMAT) = Hit time + Miss Rate x Miss Penalty.
- CPI_{stall} = CPI_{ideal} + Memory-stall cycles.
- Memory-stall cycles = number of access / instruction x miss rate x miss penalty.
- Cache size (C) = #Blocks (B) x #Lines (N) x #Sets (R)
- Block bits (b) = $\log_2 B$
- Set bits (s) = $\log_2 R$
- tag bits = number of available bits - b - s
- Types of Cache organization
  - Direct-mapped: One line per set, one block per line. resolves to one block per set.
  - Fully associative: One set with multiple lines, where each line is a block.
  - n-way Set Associative: Multiple sets, with multiple lines, and multiple blocks per line.

- Middle-Order bit indexing is good for spatial locality.
- Write hits
    - Cache-Memory consistent: write to the buffer, only stall when the buffer is full.
    - Cache-Memory inconsistent: write the cache block to the memory when its evicted, write-back. Need a dirty bit to determine if the block needs to be written to memory or not. Can use write buffer to buffer write-backs.

Chapter 4

- Design Principles:
    - Simplicity favours regularity.
    - Smaller is faster.
    - Make common case faster.
    - Good design demands good compromises.
- MIPS registers
    - $t0 - $t9 [8 - 15, 24 - 25] for temporary values.
    - $s0 - $s7 [16 - 23] for saved variables.
    - $v0 - $v3 [2 - 3] for return variables.
    - $a0 - $a3 [4 - 7] for argument variables.
- Words are 4 bytes each.
- MIPS is Big Endian: most-significant byte at the least address of a word.
- Instruction formats
    - R-format: [OP | rs | rt | rd | shft | funct] [6, 5, 5, 5, 5, 6]
    - I-format: [OP | rs | rt | immediate] [6, 5, 5, 16]
    - J-format: [OP | jump target] [6, 26]
- Operands addressing modes:
    - Immediate.
    - Register addressing.
    - Base addressing: register relative, pseudo-direct.
- Instruction addressing modes:
    - PC-relative: PC + offset, used for branching.
    - Pseudo-direct: jump address, used for jumping.
- Synchronization in MIPS
    - Load Linked: ll rt, offset(rs)
    - Store conditional: sc rt, offset(rs).
    - Succeeds if location not changed since ll. Sets rt to 1.
    - Fails otherwise. Sets rt to 0.

Chapter 5

- Five stages of the Datapath
  - Instruction Fetch (IF)
  - Instruction Decode (Dec)
  - Arithmetic-Logic Unit (Exec)
  - Memory Access (Mem)
  - Register Write (WB)
- Controllers
  - RegDst = add + sub
  - ALUSrc = ori + lw + sw
  - MemtoReg = lw
  - RegWrite = add + sub + ori + lw
  - MemWrite = sw
  - nPCsel= beq
  - ExtOp = lw + sw
  - ALUctr = add/sub, or

Chapter 6

- Hazards
  - Structural: attempt to use the same hardware.
  - Data: attempt to use data before its read.
  - Control: attempt to make decision before a condition is evaluated.
- Hazard solutions
  - Forwarding.
  - Delays.
  - Speculation.
- Very Long Instruction Word (VLIW)
- Scheduling/Reordering.
- Unrolling.
- Superscalar
  - Instruction-fetch and issue: fetch, decode and issue instruction to await execution.
  - Instruction-execution: calculate the result, once operands and instruction are ready
  - Instruction-commit: write-back once its safe to do so.
- Instruction and fetch -> functional units -> commit unit.

Chapter 7

- MESI snooping
  - Modified
  - Exclusive
  - Shared
  - Invalid

- Read Hit
    - No state change
- Read Miss
    - No valid copies
        - Change to E.
    - Another cache has an E
        - Change E to S.
        - Change the rest of the caches to S.
    - Several other caches have S
        - Change local cache to S.
    - One cache has M
        - Share value with caches and Memory.
        - Change all caches to S.
- Write Hit
    - M
        - No change, update local value.
    - E
        - Change E to M.
    - S
        - Change value to M, invalidate other caches.
- Write Miss
    - All caches are invalid
        - Change to M.
    - E , M or S
        - Change to M.
        - Invalidate other caches.
- False Sharing Miss: miss caused, even though the word was not changed.
- True Sharing Miss: miss caused, but the word was changed.
- Types of hardware multithreading
    - Fine-grain: switches threads every instruction.
    - Coarse-grain: switches on costly stalls.
    - Simultaneous Multithreading (SMT)