Computer Architecture Cheatsheet

A concise reference to key concepts in computer architecture, covering instruction sets, memory hierarchies, pipelining, and parallel processing techniques.

Instruction Set Architecture (ISA)

Instruction Formats

Zero-Address Instructions	Uses a stack architecture. Operands are implicitly taken from the stack. Example: `ADD` (pops two top elements, adds, pushes result).
One-Address Instructions	Uses an accumulator. One operand is implicit (accumulator), the other is explicit. Example: `LOAD X` (loads value at address X into the accumulator).
Two-Address Instructions	Two explicit operands, one serves as both source and destination. Example: `ADD A, B` (A = A + B).
Three-Address Instructions	Three explicit operands: two sources, one destination. Example: `ADD A, B, C` (A = B + C).
RISC vs CISC	RISC (Reduced Instruction Set Computing) uses simpler instructions, while CISC (Complex Instruction Set Computing) uses more complex instructions.

Addressing Modes

Immediate Addressing	Operand is a constant value. Example: `MOV R1, #5` (move the value 5 into register R1).
Direct Addressing	Operand is the memory address. Example: `LOAD R1, 1000` (load the value at memory address 1000 into R1).
Indirect Addressing	Operand is a register that contains the memory address. Example: `LOAD R1, (R2)` (load the value at the memory address stored in R2 into R1).
Register Addressing	Operand is a register. Example: `MOV R1, R2` (move the value in R2 into R1).
Register Indirect Addressing	The register contains the address of the operand. Example: `LOAD R1, (R2)`
Displacement Addressing	Effective address is the sum of a register and a constant. Example: `LOAD R1, 100(R2)`

Memory Hierarchy

Levels of Memory Hierarchy

Memory hierarchy is organized to provide a cost-effective balance between speed and size.
The levels are typically:

1. **Registers**: Fastest, smallest.
2. **Cache**: Fast, small.
3. **Main Memory (RAM)**: Moderate speed and size.
4. **Secondary Storage (Disk)**: Slowest, largest.

Cache Memory

Cache Hit	Data is found in the cache.
Cache Miss	Data is not found in the cache, requiring access to main memory.
Cache Line	Small block of data that is transferred between cache and main memory.
Cache Mapping Techniques	Direct Mapping, Associative Mapping, Set-Associative Mapping.

Cache Replacement Policies

LRU (Least Recently Used)	Replaces the least recently used cache line.
FIFO (First-In, First-Out)	Replaces the oldest cache line.
LFU (Least Frequently Used)	Replaces the least frequently used cache line.
Random Replacement	Randomly selects a cache line to replace.

Pipelining

Basic Pipeline Concepts

Pipelining improves throughput by overlapping the execution of multiple instructions.
Instructions are divided into stages (e.g., Fetch, Decode, Execute, Memory Access, Write Back).

Pipeline Hazards

Data Hazard	An instruction depends on the result of a previous instruction still in the pipeline. Solutions: Forwarding (bypassing), Stalling.
Control Hazard (Branch Hazard)	The pipeline doesn't know which instruction to fetch next because a branch instruction outcome is not yet known. Solutions: Branch Prediction, Delayed Branching.
Structural Hazard	Two instructions need the same hardware resource at the same time. Solution: Stall one of the instructions.

Pipeline Performance Metrics

Throughput	Number of instructions completed per unit of time.
Latency	Time taken to execute a single instruction.
Speedup	Ratio of execution time without pipelining to execution time with pipelining.

Parallel Processing

Parallelism Types

Instruction-Level Parallelism (ILP)	Executing multiple instructions simultaneously within a single processor. Techniques: Pipelining, Superscalar execution, Out-of-order execution.
Data-Level Parallelism (DLP)	Performing the same operation on multiple data elements simultaneously. Techniques: SIMD (Single Instruction, Multiple Data), Vector processors.
Thread-Level Parallelism (TLP)	Executing multiple threads simultaneously on multiple processors or cores. Techniques: Multithreading, Multicore processors.

Multiprocessor Architectures

Shared Memory Multiprocessors	Processors share a common memory space. Examples: SMP (Symmetric Multiprocessing), NUMA (Non-Uniform Memory Access).
Distributed Memory Multiprocessors	Processors have their own private memory and communicate via message passing. Examples: Clusters, Massively Parallel Processors (MPP).

Flynn's Taxonomy

SISD (Single Instruction, Single Data)	Traditional sequential computers.
SIMD (Single Instruction, Multiple Data)	Vector processors, GPUs.
MISD (Multiple Instruction, Single Data)	Rarely used.
MIMD (Multiple Instruction, Multiple Data)	Multiprocessors, multicomputers.