Assumption on what you Know

- Basic Computer Architeture
- Basic Computer Components
- Logic Design

Reference

patterson & Hennessy's
Computer Org. and Design
[MIPS Edition]

main focus: 4 & 5 chaptens

Grading

- Assignments (building a simple CPU weekly)
- mid-class quizes
- Discussion participaten

Understanding Performance

Algorithm

Determines the number of **operations** → work to execute the code

Language, Compiler, Instruction Set

Determines the number of **instructions** → tied to processor design to execute the code.

Processor Architecture

Determines **how fast** the instructions can be executed.

I/O system and OS

Determine **how fast** memory and I/O are accessed.

7 Great ideas in Computer Architeture

1- Use abstration to simplify

helps reduce complexity of high level view.

In architecture we abstract details bus, caches, men.

2- Make the common case faster

Raises question of what is the common case?

3- perfomance via parallelism

Executing things at the same time.

4- perfomance via pipelining

Overlaping portions of execution.

5 - performance via prediction

looking at past behavior in the workload for how future execution will likely happen.

6 - Hierarchy of memories

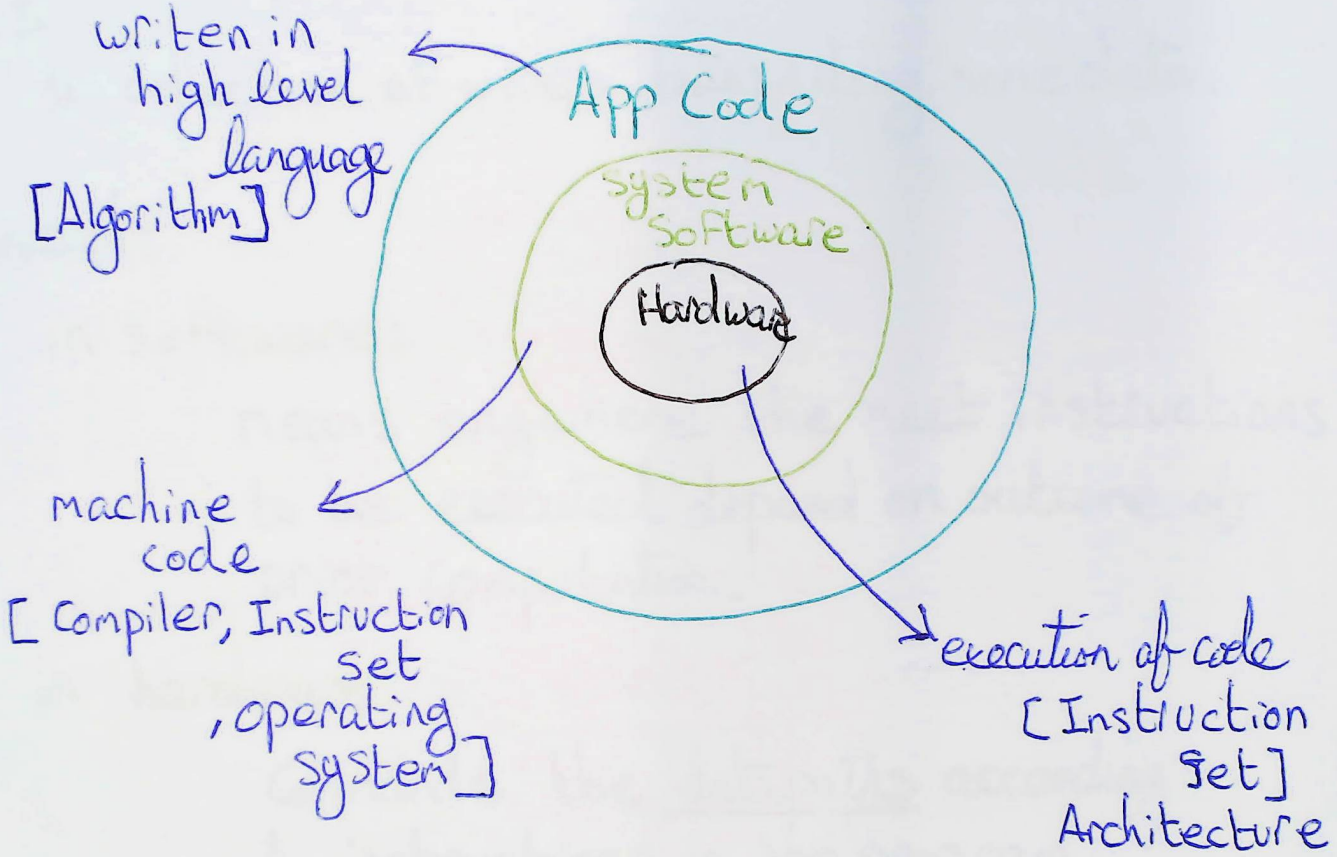
Small memories are fast to acces

Larger memories are slow to access

7 - Depedability via Redundancy

Do redundant work and cross-validation.

Below your Code



Common Terminology

Bus

a collection of wires transmiting some data.

Control:

in software:

means anywhere the next instructions to be executed depend on outcome of prior computation.

in hardware:

Commands the datapaths according to instructions in the program.

datapaths: parts of circuit that do arithmatic/logic.

Caches / Rams

a place for storing data
given an address, and can get data
or write data to the address.

* caches: can have data present or not

Instruction set (ISA)

The set of available instructions in the hardware, and how the are encoded.

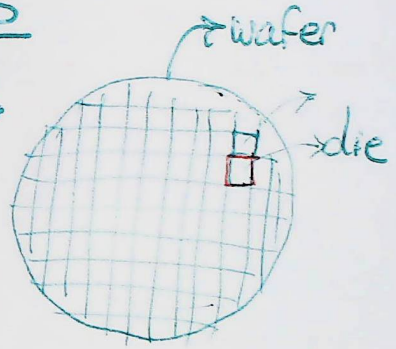
(ex. Mips, ARM, x86)

Can have very different "ISA" for processor the execute special compiled machine code.

Technology Factors in CPU Design (which impact Architecture Design)

VLSI

- How wide transistors are.
- Capacitance. Capacity observed on wires. How fast you can clock.
- Time to propagate data.
- Yield: proportion of working dies per wafer.



* Cost per die ↑ $= \frac{\text{Cost per wafer}}{\#\text{working dies per wafer}}$

$= \frac{\text{Cost per wafer}^{\uparrow}}{\downarrow \text{number of dies per wafer} \times \text{Yield}^{\downarrow}}$

Latency vs throughput

- **Latency**: How long it takes, From start to Finish for something to complete.
  - ex: Response time of Ran.
- **Throughput**: How much of something can be done in a given amount of time.
  - (Bandwidth)
  - ex. number of cores in a processor.

How to compare performance

$$\text{Performance} \simeq \frac{1}{\text{Time Execute}}$$

- CPU_X is (N) times Faster that CPU_Y

$$\text{Ratio } N = \frac{\text{Performance X}}{\text{Performance Y}}$$

$$\%\text{speedup} = \frac{\text{Perf}_x - \text{Perf}_y}{\text{Perf}_y}$$

Trade off between #clocks and clockperiod

$$\text{cpu time} = \#\text{cpu clock} \cdot \text{clock period}$$

#cpu clock: Algorithm, Compiler, ISA, O/S

clock period $\simeq \frac{1}{\text{clock Freq}}$: Technology, Architecture

$$= \frac{\#\text{cpu clock}}{\text{clock Freq}}$$

clocks per Instruction

cpI = # clocks to execute an instruction

$$\boxed{\text{total num. clock} = \#\text{inst} \times \text{cpI}}^{*}$$

Different Inst Types have different CPI

$$\text{total clocks} = \sum_{i}^{N} (\#\text{inst}_i \times \text{CPI}_i)$$

Example

class	A	B	C
CPI	1	2	3
seq1	2	1	2
seq2	4	1	1

which seq of inst. runs faster, at same clock freq?

seq1 = 2x1 + 1x2 + 2x3 = 10.

seq2 = 4x1 + 1x2 + 1x3 = 9.