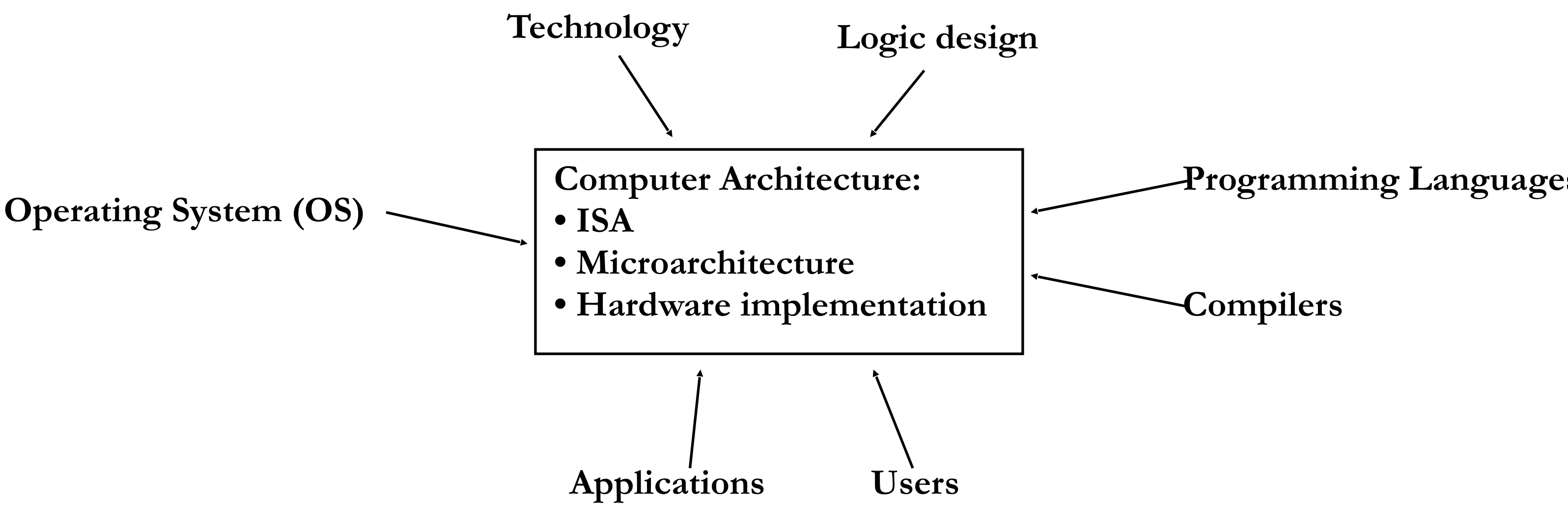
Summary: Computer Architecture

What is Computer Architecture?



Metrics of interest for computer architects

- Performance
- Cost
- Reliability
- Power
- Area

Performance of Computer Architectures

- Metrics:
  - Execution time, response time: overall time for a given computation (e.g. full program execution, one transaction)
  - Latency: time to complete a given task (e.g. memory latency, I/O latency, instruction latency)
  - Bandwidth, throughput: rate of completion of tasks (e.g. memory bandwidth, transactions per second, MIPS, FLOPS)
- MHz, MIPS, FLOPS: must use with caution
- Benchmarking:
  - toy benchmarks, synthetic benchmarks, kernels, real programs,
  - Input sets
  - E.g. SPECint, SPECfp, EEMBC Coremark

CPU Performance Terminology

A is n times faster than B means

$$\frac{\text{Execution time of B}}{\text{Execution time of A}} = n$$

A is m% faster than B means

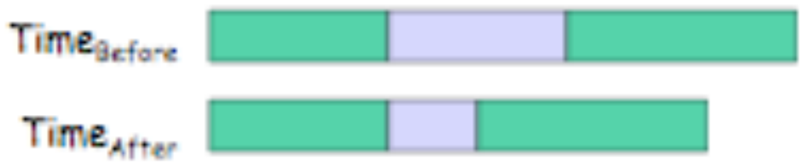
$$\frac{\text{Execution time of B} - \text{Execution time of A}}{\text{Execution time of A}} \times 100 = m$$

Improving Performance: Principles of CA Design

- Take advantage of parallelism
  - System level: multiple processors, multiple disks
  - Processor level: pipelining, superscalar issue
  - Circuit level: carry-lookahead ALU
- Principle of locality
  - Spatial and Temporal Locality
  - 90% of program executing in 10% of code
  - E.g. Caches
- Focus on the common case
  - Amdahl's law, CPU Performance equation
  - E.g. RISC design principle

Amdahl’s Law

Speedup due to an enhancement E:



$$\text{Speedup (E)} = \frac{\text{Execution time w/o E (Before)}}{\text{Execution time w E (After)}}$$

Suppose that enhancement E accelerates a fraction F of the task by a factor S, and the remainder of the task is unaffected, what is the *Execution time$_{after}$* and *Speedup(E)* ?

Amdahl's Law

$$\textit{Execution time}_{after} = \text{ExTime}_{before} \times \left[(1-F) + \frac{F}{S}\right]$$

$$\textit{Speedup(E)} = \frac{\text{ExTime}_{before}}{\text{ExTime}_{after}} = \frac{1}{\left[(1-F) + \frac{F}{S}\right]}$$

Amdahl's Law - Example

Q: Floating point instructions improved to run 2X; but only 10% of execution time are FP ops. What is the execution time and speedup after improvement?

Ans:

$F = 0.1, S = 2$

$$ExTime_{after} = ExTime_{before} \times [\ (1-0.1) + 0.1/2\ ] = 0.95\ ExTime_{before}$$

$$Speedup = \frac{ExTime_{before}}{ExTime_{after}} = \frac{1}{0.95} = 1.053$$

Factors that affect CPU performance

Instruction count (IC)

- Compiler, ISA

Cycles per instruction (CPI)

- ISA, microarchitecture

Clock time (1/*f*)

- Microarchitecture, technology

What determines these factors?

The CPU Performance Equation

CPU time = IC x CPI x Clock time

where: CPU time = execution time
IC = number of instructions executed (instruction count)
CPI = number of average clock cycles per instruction
Clock time = duration of processor clock

$$\textbf{CPU time} = \left(\sum_{i=1}^{n} \textbf{IC}_i \times \textbf{CPI}_i \right) \times \textbf{Clock time}$$

where: IC_i = IC for instruction (instruction group) i
CPI_i = CPI for instruction (instruction group) i

$$\textbf{CPI} = \frac{\left(\sum_{i=1}^{n} \textbf{IC}_i \times \textbf{CPI}_i \right)}{\textbf{IC}} = \sum_{i=1}^{n} \left(\textbf{CPI}_i \times \frac{\textbf{IC}_i}{\textbf{IC}} \right)$$

Examples

Example:

- Branch instructions take 2 cycles, all other instructions take 1 cycle
- CPU A uses extra compare instruction per branch
- Clock frequency of CPU A is 1.25 times faster than CPU B
- On A 20% of instructions are branches (thus other 20% are compare instructions)

Find the CPU time

$$CPI_A = CPI_{branch} \times \frac{IC_{branch}}{IC} + CPI_{others} \times \frac{IC_{others}}{IC} = 2 \times 0.2 + 1 \times 0.8 = 1.2$$

$$CPI_B = CPI_{branch} \times \frac{IC_{branch}}{IC} + CPI_{others} \times \frac{IC_{others}}{IC} = 2 \times 0.25 + 1 \times 0.75 = 1.25$$

$$\text{CPU time}_B = IC_B \times CPI_B \times \text{Clock time}_B = 0.8 \times IC_A \times 1.25 \times (1.25 \times \text{Clock time}_A)$$

$$= 1.25 \times IC_A \times \text{Clock time}_A$$

$$\text{CPU time}_A = IC_A \times CPI_A \times \text{Clock time}_A = IC_A \times 1.2 \times \text{Clock time}_A$$

CPU A is faster!

Improving CPU Performance

IC:

- Compiler optimizations (constant folding, constant propagation)
- ISA (More complex instructions)

CPI:

- Microarchitecture (Pipelining, Out-of-order execution, branch prediction)
- Compiler (Instruction scheduling)
- ISA (Simpler instructions)

Clock period:

- Hardware (Smaller transistors – Moore's law)
- ISA (Simple instructions that can be easily decoded)
- Microarchitecture (Simple architecture)