Arch: Introductions

Alex Chi

Update: March 19, 2020

Contents

1 Introduction

1. Evolution of Processors
   - 8086: x86 Architecture
   - Pentium
   - Core
2. x86 Manufacturers
   - Intel
   - AMD
   - VIA
   - Others are acquired or discontinue
3. ISA
   - x86 is CISC (complex instruction set computer)
     - x86 has varying length instructions
   - RISC (reduced instruction set computer)
4. Die

   Wafer consists of many Dies.
5. First Microprocessor
   - Intel 4004
   - Max CPU clock rate: 108 kHz
6. General
   - Transistors doubles every 1.5-2.0 yrs
   - Process speed doubles every 1.5-2.0 yrs
   - DRAM size doubles every 1.5-2.0 yrs
7. Single Processor Performance
   - 2003: from uni-processor to multi-processor
   - Pipeline (ILP) reaches its limit
   - Power limit
   - Move to multi-processor
   - Clock rate 1% growth
8. Effects
   - enhanced capability
   - lead to new class of computers
   - dominance of microprocessor-based computers
   - software development shift focus on productivity
9. Current Trends
   - DLP

- TLP
- RLP

2 Classes of Computers

- Personal Mobile Device
- Desktop Computing
- Servers
- Clusters / Warehouse Scale Computers
- Supercomputers (require faster network than WSCs)
- Embedded Computers
- refer to textbooks for details

2.1 Flynn's Taxonomy

- Single instruction stream, single data stream (SISD)
- SIMD (GPU, AVX extensions, vector arch)
- MISD (no commercial implementation)
- MIMD (tightly-coupled / loosely-coupled)

2.2 Two Kinds of Parallelism in Applications

- Data-Level Parallelism (operate on many data at the same time)
- Task-Level Parallelism (tasks can operate independently)
- example: web crawler
  - crawl web pages
  - parse HTML **data** (data-level)
  - run parse HTML **task** (task-level)

2.3 Four Major Ways for Exploiting Parallelism

- ILP (pipelining, data-level)
- Vector arch / GPU (data-level)
- Thread-Level Parallelism (multi-core, data-level or task-level)
- Request-Level Parallelism (clusters, data-level or task-level)

3 Define Computer Architecture

- old view: ISA
- ISA is same as programming model
- abstracts hardware and software interface

4 Contents of ISA

- register (size, number)
- class of ISA (register-memory, complex or load/store)
- memory addressing (byte addressing, little/big-endian)
- addressing modes
- instruction operands (e.g. RISC-V 3 operands, x86 only 2)
- available operations (e.g. RV32I doesn't support hardware mul and div)
- control flow instructions (e.g. ret or jalr in RISC-V reads return address from ra, but x86 reads from stack and pop)
- instruction encoding (fixed length / variable length)
- etc.

4.1 Addressing Mode

- register, r4 ← r4 + r3
- immediate, r4 ← r4 + 3
- displacement, r4 ← r4 + M[100+r1]
- register deferred, r4 ← r4 + M[r1]
- etc. refer to textbook

4.2 MIPS ISA

- `| op 6 | rs 5 | rt 5 | rd 5 | shamt 5 | funct 6 |` (32 bits)
- op: operation
- rs: register first source
- rt: register second source
- rd: register destination
- shamt: shift amount

- funct: function code

4.2.1 Addressing Mode Examples

- register addressing (no shamt)
- immediate addressing (rd-funct as operand)
- base (displacement) addressing (rd-funct as offset + base register)
- pc-relative addressing (rd-funct as offset + PC)

4.3 Micro-architecture

- micro-arch, also called computer organization, is the way a given ISA is implemented on a processor.
- a given ISA can be implemented with different micro-archs

4.3.1 Concepts

- pipelining
- hierarchical memory organization
- cache
- cache coherence
- branch prediction
- super-scalar
- out-of-order execution
- register renaming
- multi-processing and multi-threading

4.4 Computer Architecture in General

- working in constraints
- market target?
- cost/performance?
- tradeoff in material and process
- Computer Architecture is about designing the organization and hardware to meet goals and functional requirements

5 Trends in Technology

5.1 Five Critical Implementation Techs

- IC technology
- semiconductor DRAM
- semiconductor flash
- magnetic disk technology
- network technology

5.2 Example: ENIAC

- refer to slides

5.3 IC technology

- vacuum tube → transistor → semiconductor (gates, memory cells, interconnections)

5.3.1 Quantify

- bandwidth or throughput: total work done in a given time
- latency or response time: time between start and completion of an event
- example: in a pipelined processor, it's running a series of `add` instruction. There might be multiple `add` running at the same time.
  - throughput: how many `add` can be issued in a given time
  - latency: for one `add` instruction, how long does it take from issue to complete
- bandwidth outpaced latency

5.3.2 Feature size

- transistor size in x or y dimension
- transistor density increases
- wire latency do not increase that fast

6 Trends in Power and Energy in ICs

6.1 Power and Energy

- power is unit time energy

6.1.1 Thermal Design Power (TDP)

- characterize sustained power consumption
- use as target for power supply and cooling
- lower than peak power, higher than average power consumption

6.1.2 Clock Rate

- can be reduced dynamically to reduce power consumption

6.2 Dynamic Energy and Power

- dynamic energy per transistor
  - used for a transistor switching from 0 to 1 / 1 to 0
  - $\frac{1}{2}$ capacitive load $\times$ voltage^2
  - related to number of transistor and technology
- dynamic power per transistor
  - $\frac{1}{2}$ capacitive load $\times$ voltage^2 $\times$ frequency switched
- voltage is the key
  - voltage of processor has become lower

6.3 Power

- 130W, maximum for air cooling
- we use energy of a specific task to compare CPUs

6.4 Techniques

- turn off clock
- dynamic voltage-frequency scaling

- low power state for DRAM, disks
- overclocking, turning off cores

6.5 Static Power

- current × voltage
- scales with number of transistors
- power gating: turning off the power supply

7 Trends in Cost

- refer to slides and textbook
- I don't want to type exactly the same thing in slides
- From my perspective, quantifying these things is the core of the book CAAQA. However, this is not our course orients to.

8 Dependability

- MTTF ~ Exp(n)
- MTTF Components ~ Exp($n_1 + n_2 + \dots$)
- $E(Exp(x)) = \frac{1}{x}$
- Serial MTTF

$$\frac{1}{\frac{1}{MTTF} + \frac{1}{MTTF}}$$

- Redundant MTTF

$$\frac{MTTF^2}{2 \times MTTR}$$

9 Measuring Performance

9.1 Typical Performance Metrics

- latency (response time)
- throughput (bandwidth)

9.2 Benchmarks

- a common program for testing the execution times of computers
- e.g. kernels (matrix multiply), top programs (e.g. sorting), synthetic benchmarks
- we use benchmark suite (SPEC)
- SPEC uses performance ratio
- and Geometric Mean

10 Quantitative Principles

10.1 Principles for Computer Design

- take advantage of parallelism
- principle of locality
- focus on common case

10.2 Amdahl's Law

- performance improvement of using a new feature is limited by the fraction of the time the new feature can be used

10.3 CPI

- CPI = $\frac{\text{CPU clock cycles for a program}}{\text{Instruction count}}$
- CPU Time = Instruction count $\times$ Cycles per instruction $\times$ clock cycle time (Unit: sec/program)

10.4 Different Instruction Type requires Different CPI

- use $IC_o \times CPI_i$