Chapter 1: Fundamentals of Computer Design (Part 1)

What is computer architecture?
Why study computer architecture?
Common principles
What is Computer Architecture?
What is Computer Architecture?

Instruction set architecture
- Interface between hardware and software
- Instructions visible to programmer
  - e.g., Intel IA32 vs. IA64; ARM v7 vs. ARM v8

Organization or Microarchitecture
- High-level aspects of the system
  - e.g., how many functional units, pipeline organization, cache hierarchy, cores, interconnect, …
  - e.g., AMD Athlon X4 vs. Intel Core i7;
    ARM Cortex-A53 vs. Cortex-A72

Implementation or hardware
- Logic design, packaging, …
  - e.g., AMD Athlon X4 880K vs. 860K (4 vs. 3.7 GHz)
Previously, Computer Architecture ~ ISA

Instruction set architectures

Most ISAs today are general-purpose register based

Operands may be registers or memory locations

Register-memory vs. load-store

Addressing modes

Register, immediate, displacement, ...

Operand sizes

8 bits, 16 bits, 32 bits, 64 bits, SP and DP FP

Operations: Arithmetic, memory, control flow, floating point

Encoding: fixed vs. variable length

Action no longer in ISA

But not always the case: CISC vs. RISC – what happened?

Our main focus: organization
Goals of the Computer Architect
Goals of the Computer Architect

Depends on type of computer
  Desktop
  Personal mobile device
  Server
  Embedded
  Cluster/warehouse-scale
  Supercomputer
Goals of the Computer Architect

Functional goals
- Meet application area demands
- Compatibility with previous systems
- Standards (e.g., IEEE floating point)
- Last through trends

Performance: Latency, throughput, real-time constraints, scalability

Cost
- Power, Energy, Temperature, …

Dependability
- Maintainability, Verifiability, …’ity…

Need to be familiar with design alternatives and criteria for selecting among them
Why Study Computer Architecture? - Historical Trends

[Graph showing historical trends in computer performance, with labels for specific processors and years.]
Why Study Computer Architecture? - Historical Trends

Figure 1.1 Growth in processor performance since the late 1970s. This chart plots performance relative to the VAX 11/780 as measured by the SPEC benchmarks (see Section 1.8). Prior to the mid-1980s, processor performance growth was largely technology driven and averaged about 25% per year. The increase in growth to about 52% since then is attributable to more advanced architectural and organizational ideas. By 2003, this growth led to a difference in performance of about a factor of 25 versus if we had continued at the 25% rate. Performance for floating-point-oriented calculations has increased even faster. Since 2003, the limits of power and available instruction-level parallelism have slowed uniprocessor performance, to no more than 22% per year, or about 5 times slower than had we continued at 52% per year. (The fastest SPEC performance since 2007 has had automatic parallelization turned on with increasing number of cores per chip each year, so uniprocessor speed is harder to gauge. These results are limited to single-socket systems to reduce the impact of automatic parallelization.) Figure 1.11 on page 24 shows the improvement in clock rates for these same three eras. Since SPEC has changed over the years, performance of newer machines is estimated by a scaling factor that relates the performance for two different versions of SPEC (e.g., SPEC89, SPEC92, SPEC95, SPEC2000, and SPEC2006).
Why Study Computer Architecture?
Why Study Computer Arch? Technology Trends**

Technology changes fast and on different curves

Capacity
- Transistors/chip: 1.5X/year
- DRAM: 1.4X/year
- Disk: 1.3X to 2X/year
- Flash: 1.5X/year

Performance (~ 20 years):
- CMOS scaling trends
  - Transistor vs. wire speed
  - Voltage curve flatter
- Power steeper
- Reliability worse

Major recent implications
- Memory wall, ILP wall, power wall
- Reliability/verifiability/maintainability/… walls???
- Led to multicore in spite of limited software base
- Recent: heterogeneity
Figure 1.9 Log–log plot of bandwidth and latency milestones from Figure 1.10 relative to the first milestone. Note that latency improved 6X to 80X while bandwidth improved about 300X to 25,000X. Updated from Patterson [2004].
Technology trends
Applications change
  Scientific, business, personal computing, cloud, internet of things
  Databases, graphics, multimedia, communications, next killer app?
New languages
  E.g., shift from assembly to high-level languages
  E.g., shift from C/C++ to Java/Python/Ruby
Compiler / hardware boundary shifts
Relationship to Prerequisites

Prerequisite
  How to design a uniprocessor?

This course
  How to design a uniprocessor WELL?
  Emphasis on Quantitative vs. Qualitative
  How to design a multiprocessor?

Be sure to check the handout for details on the prerequisites