Architecture and Organization

The computer lies at the heart of computing. Without it most of the computing disciplines today would be a branch of theoretical mathematics. A professional in any field of computing should not regard the computer as just a black box that executes programs by magic. All students of computing should acquire some understanding and appreciation of a computer system's functional components, their characteristics, their performance, and their interactions. Students need to understand computer architecture in order to make best use of the software tools and computer languages they use to create programs. In this introduction the term architecture is taken to include instruction set architecture (the programmer's abstraction of a computer), organization or microarchitecture (the internal implementation of a computer at the register and functional unit level), and system architecture (the organization of the computer at the cache, and bus level). Students should also understand the complex tradeoffs between CPU clock speed, cache size, bus organization, number of core processors, and so on. Computer architecture also underpins other areas of the computing curriculum such as operating systems (input/output, memory technology) and high-level languages (pointers, parameter passing).

The learning objectives specified for these topics correspond primarily to the core and are intended to support programs that require only the minimum 36 hours of computer architecture. For programs that want to teach more than the minimum, the same topics (AR1-AR6) can be treated at a more advanced level by implementing a two-course sequence. For programs that want to cover the elective topics, those topics can be introduced within a two-course sequence and/or be treated in a more comprehensive way in a third course

AR. Architecture and Organization (36 core hours)

AR/DigitalLogicandDataRepresentation [core]
AR/ ComputerArchitectureandOrganization [core]
AR/ InterfacingandI/OStrategies [core]
AR/MemoryArchitecture [core]
AR/FunctionalOrganization [core]
AR/Multiprocessing [core]
AR/PerformanceEnhancements [elective]
AR/DistributedArchitectures [elective]
AR/Devices [elective]
AR/DirectionsInComputing [elective]

AR/DigitalLogicandDataRepresentation [core]

Minimum core coverage time: 7 hours
(was Digital logic and digital systems AR1 and Machine level representation of data AR2.)

Topics:

- Introduction to digital logic (logic gates, flip-flops, circuits)
- Logic expressions and Boolean functions
- Representation of numeric data
- Signed and unsigned arithmetic
- Range, precision, and errors in floating-point arithmetic
- Representation of text, audio, and images
- Data compression

Learning Objectives:

1. Design a simple circuit using fundamental building blocks.
2. Appreciate the effect of AND, OR, NOT and EOR operations on binary data
3. Understand how numbers, text, images, and sound can be represented in digital form and the limitations of such representations
4. Understand how errors due to rounding effects and their propagation affect the accuracy of chained calculations.
5. Appreciate how data can be compressed to reduce storage requirements including the concepts of lossless and lossy compression.

AR/ComputerArchitectureandOrganization [core]

Minimum core coverage time: 9 hours
(was Assembly level machine organization AR3)

Topics:

- Overview of the history of the digital computer
- Introduction to instruction set architecture, microarchitecture and system architecture
- Processor architecture – instruction types, register sets, addressing modes
- Processor structures – memory-to-register and load/store architectures
- Instruction sequencing, flow-of-control, subroutine call and return mechanisms
- Structure of machine-level programs
- Limitations of low-level architectures
- Low-level architectural support for high-level languages

Learning Objectives:

1. Describe the progression of computers from vacuum tubes to VLSI.
2. Appreciate the concept of an instruction set architecture, ISA, and the nature of a machine-level instruction in terms of its functionality and use of resources (registers and memory).
3. To understand the relationship between instruction set architecture, microarchitecture, and system architecture and their roles in the development of the computer.
4. Be aware of the various classes of instruction: data movement, arithmetic, logical, and flow control.
5. Appreciate the difference between register-to-memory ISAs and load/store ISAs.
6. Appreciate how conditional operations are implemented at the machine level.
7. Understand the way in which subroutines are called and returns made.
8. Appreciate how a lack of resources in ISPs has an impact on high-level languages and the design of compilers.
9. Understand how, at the assembly language level, how parameters are passed to subroutines and how local workplace is created and accessed.

AR/InterfacingandI/OStrategies [core]

Minimum core coverage time: 3 hours
(was Interfacing and communication AR5)

Topics:

- I/O fundamentals: handshaking and buffering
- Interrupt mechanisms: vectored and prioritized, interrupt acknowledgment
- Buses: protocols, arbitration, direct-memory access (DMA)
- Examples of modern buses: e.g., PCIe, USB, Hypertransport

Learning Objectives:

1. Appreciate the need of open- and closed-loop communications and the use of buffers to control dataflow.
2. Explain how interrupts are used to implement I/O control and data transfers.
3. Identify various types of buses in a computer system and understand how devices compete for a bus and are granted access to the bus.
4. Be aware of the progress in bus technology and understand the features and performance of a range of modern buses (both serial and parallel).

AR/MemoryArchitecture [core]

Minimum core coverage time: 5 hours
(was Memory system organization AR4)

Topics:

- Storage systems and their technology (semiconductor, magnetic)
- Storage standards (CD-ROM, DVD)
- Memory hierarchy, latency and throughput
- Cache memories - operating principles, replacement policies, multilevel cache, cache coherency

Learning Objectives:

1. Identify the memory technologies found in a computer and be aware of the way in which memory technology is changing.

2. Appreciate the need for storage standards for complex data storage mechanisms such as DVD.
3. Understand why a memory hierarchy is necessary to reduce the effective memory latency.
4. Appreciate that most data on the memory bus is cache refill traffic
5. Describe the various ways of organizing cache memory and appreciate the cost-performance tradeoffs for each arrangement.
6. Appreciate the need for cache coherency in multiprocessor systems

AR/FunctionalOrganization [core]

Minimum core coverage time: 6 hours
(was Functional organization AR6)

Topics:

- Review of register transfer language to describe internal operations in a computer
- Microarchitectures - hardwired and microprogrammed realizations
- Instruction pipelining and instruction-level parallelism (ILP)
- Overview of superscalar architectures
- Processor and system performance
- Performance – their meeasures and their limitations
- The significance of power dissipation and its effects on computing structures

Learning Objectives:

1. Review of the use of register transfer language to describe internal operations in a computer
2. Understand how a CPU’s control unit interprets a machine-level instruction – either directly or as a microprogram.
3. Appreciate how processor performance can be improved by overlapping the execution of instruction by pipelining.
4. Understand the difference between processor performance and system performance (i.e., the effects of memory systems, buses and software on overall performance).
5. Appreciate how superscalar architectures use multiple arithmetic units to execute more than one instruction per clock cycle.
6. Understand how computer performance is measured by measurements such as MIPS or SPECmarks and the limitations of such measurements.
7. Appreciate the relationship between power dissipation and computer performance and the need to minimize power consumption in mobile applications.

AR/Multiprocessing [core]

Minimum core coverage time: 6 hours
(was Multiprocessing and alternative architectures AR7)

Topics:

- Amdahl’s law
- Short vector processing (multimedia operations)
- Multicore and multithreaded processors
- Flynn’s taxonomy: Multiprocessor structures and architectures
- Programming multiprocessor systems
- GPU and special-purpose graphics processors
- Introduction to reconfigurable logic and special-purpose processors

Learning Objectives:

1. Discuss the concept of parallel processing and the relationship between parallelism and performance.
2. Appreciate that multimedia values (e.g., 8-/16-bit audio and visual data) can be operated on in parallel in 64-bit registers to enhance performance.
3. Understand how performance can be increased by incorporating multiple processors on a single chip.
4. Appreciate the need to express algorithms in a form suitable for execution on parallel processors.
5. Understand how special-purpose graphics processors, GPUs, can accelerate performance in graphics applications.
6. Understand the organization of computer structures that can be electronically configured and reconfigured

AR/PerformanceEnhancements [elective]

(was performance enhancements AR8)

Topics:

- Branch prediction
- Speculative execution
- Superscalar architecture
- Out-of-order execution
- Multithreading
- Scalability
- Introduction to VLIW and EPIC architectures
- Memory access ordering

Learning Objectives:

1. Explain the concept of branch prediction its use in enhancing the performance of pipelined machines.
2. Understand how speculative execution can improve performance.
3. Provide a detailed description of superscalar architectures and the need to ensure program correctness when executing instructions out-of-order.
4. Explain speculative execution and identify the conditions that justify it.
5. Discuss the performance advantages that multithreading can offer along with the factors that make it difficult to derive maximum benefits from this approach.
6. Appreciate the nature of VLIW and EPIC architectures and the difference between them (and between superscalar processors)
7. Understand how a processor re-orders memory loads and stores to increase performance

AR/DistributedArchitectures [elective]

(was Architecture for networks and distributed systems AR9)

Topics:

- Introduction to LANs and WANs and the history of networking and the Internet
- Layered protocol design, network standards and standardization bodies
- Network computing and distributed multimedia
- Mobile and wireless computing
- Streams and datagrams
- Physical layer networking concepts
- Data link layer concepts (framing, error control, flow control, protocols)
- Internetworking and routing (routing algorithms, internetworking, congestion control)
- Transport layer services (connection establishment, performance issues)

Learning Objectives:

1. Explain the basic components of network systems and distinguish between LANs and WANs.
2. Discuss the architectural issues involved in the design of a layered network protocol.
3. Explain how architectures differ in network and distributed systems.
4. Appreciate the special requirements of wireless computing.
5. Understand the difference between the roles of the physical layer and data link layer and appreciate how imperfections in the physical layer are handled by the data link layer.
6. Describe emerging technologies in the net-centric computing area and assess their current capabilities, limitations, and near-term potential.
7. Understand how the network layer can detect and correct errors.

AR/Devices [elective]

(new material)

Topics:

- Representing analog values digitally – quantization and sampling
- Sound and audio, image and graphics, animation and video
- Multimedia standards (audio, music, graphics, image, telephony, video, TV)
- Input transducers (temperature, pressure, position, movement)
- Input devices: mice, keyboards (text and musical), scanners, touch-screen, voice

- Output devices: displays, printers
- Encoding and decoding of multimedia systems including compression and decompression
- Example of computer-based systems: GPS, MP3 players, digital cameras

Learning Objectives:

1. Understand how analog quantities such as pressure can be represented in digital form and how the use of a finite representation leads to quantization errors.
2. Appreciate the need for multimedia standards and be able to explain in non-technical language what the standard calls for.
3. Understand how multimedia signals usually have to be compressed to conserve bandwidths using lossless or lossy encoding.
4. Discuss the design, construction, and operating principles of transducers such as Hall-effect devices and strain gauges
5. Appreciate how typical input devices operate.
6. Understand the principles of operation and performance of various display devices.
7. Study the operation of high-performance computer-based devices such as digital cameras

AR/Directions in Computing [elective]

7 hours (New topic)

Topics:

- Semiconductor technology and Moore's law
- Limitations to semiconductor technology
- Quantum computing
- Optical computing
- Molecular (biological) computing
- New memory technologies

Learning Objectives:

1. To appreciate the underlying physical basic of modern computing.
2. Understand how the physical properties of matter impose limitations on computer technology
3. Appreciate how the quantum nature of matter can be exploited to permit massive parallelism
4. Appreciate how light can be used to perform certain types of computation
5. Understand how the properties of complex molecules can be exploited by organic computers
6. To get an insight into trends in memory design such as ovonic memory and ferromagnetic memories