INTRODUCTION

A.R. Hurson
Department of Computer Science
Missouri University of Science & Technology

Background Module1

- **Introduction**
  - Computer: A device that stores and retrieves data and sequentially (mainly) executes a stored program without human intervention.
  - A computer, like any other system, is a collection of entities (components) interconnected in order to perform a well defined function. This function is determined by the functions performed by its components and by the manner in which they are interconnected.
  - The function of the computer is a mapping of the input data to the output data:

    $$F: A \rightarrow B$$

  - In case of a digital system *A* and *B* are digital or discrete quantities.

Background Module1

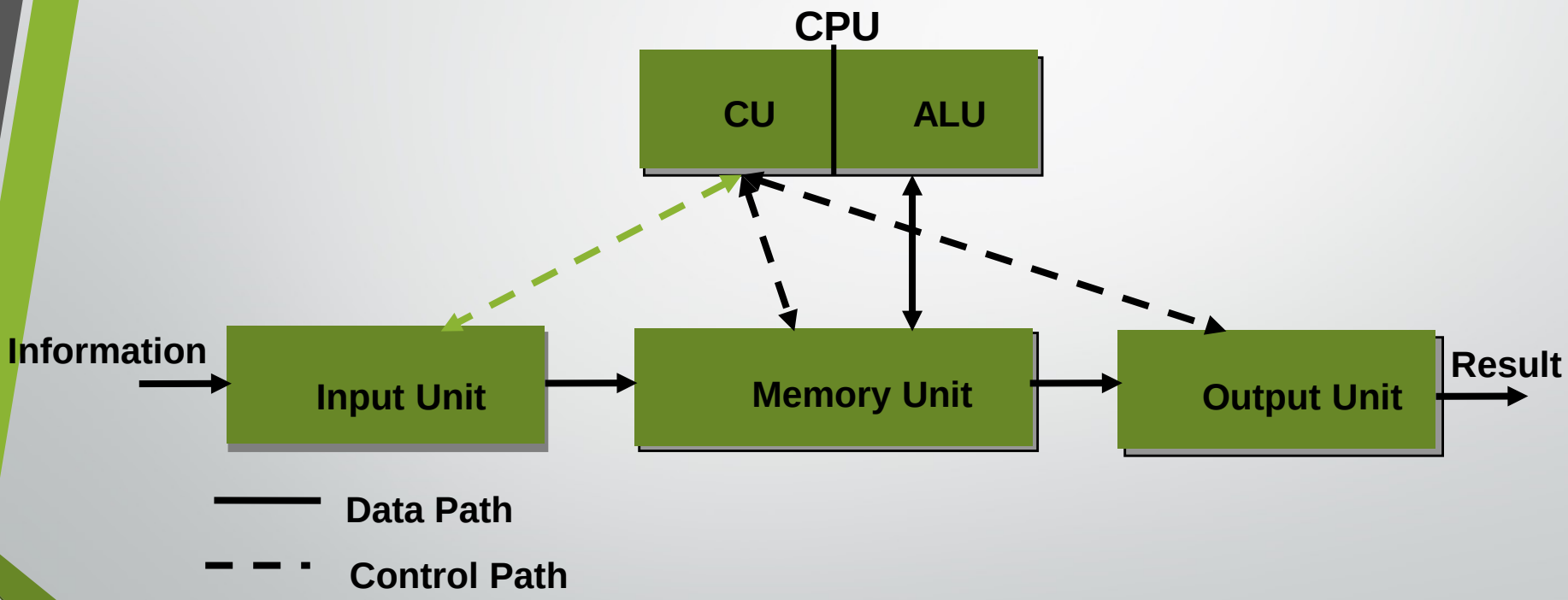
- **Introduction**
  - The **study of computers** is the study of its components, their interactions and their parallel activities and co-operations.
  - In this module a computer is viewed as a collection of five interconnected components:
    - Input Unit
    - Output Unit
    - Memory Unit
    - Central Processing Unit (CPU):
      - Control Unit (CU)
      - Arithmetic Logic Unit (ALU)

Background Module1

- **Introduction**
  - **Input Unit:** is an interface between the outside world and the internal parts. It performs two tasks: **Transmission** and **Translation** of information.
  - **Output Unit:** is an interface between the internal parts and the outside world. Its functions are the same as the Input Unit.
  - **Memory Unit:** acts as storage. It stores the instructions, data, intermediate, and final results.
  - **Central Processor Unit:** is used to:
    - Interpret the instructions and initiate their executions.
    - Perform arithmetic and logical operations on the data.

Background Module1

- **Introduction**



Background Module1

- **Introduction**
  - In general, a computer can be studied at four different levels: Electronic, Logic, Programming, and System. Though it is hard to generalize, usually:
    - Electronic level is the subject of physics, mathematics, and electrical engineering
    - Logic level is the subject of electrical engineering and computer engineering, and
    - Programming and System levels are the subjects of computer science and engineering.

Background Module1

Level	Components
Electronic	Active: transistor, voltage sources Passive: resistor, capacitor
Logic combinational sequential register	 gates, AND, OR, ... Flip Flops, ... • register, data, operators, ...
Programming compiler interpreter machine oriented assembly machine micro • • •	
System	control, processor, ...

Background Module1

- **Introduction**
  - A digital system can be viewed as a combinational and/or sequential device. Therefore, it can be evaluated at the combinational and/or sequential sublevels.
  - Similarly, a computer system can be studied in terms of the functions it could handle. This is the basis for **Logic Transfer Level**.
  - At Logic Transfer Level, one requires a set of notations (language) to carry out such evaluation. This method is called **Register Transfer Logic** and the language is called **Register Transfer Language**.

Background Module1

- **Introduction**
  - We take a **top-down approach** to study a complex object.
    - A complex object is recursively broken down into its components.
    - At each level, we study the components and the inter-actions among them.
  - In Computer science and engineering a computer will be studied at Logic, Programming, and System levels.
  - We will try to define a set of notations and rules in order to be able to study and analyze a computer.

Background Module1

- **Introduction**
  - **Computer**: A device capable of solving problems (data manipulation) by accepting data, performing prescribed operations on the data, and supplying the results.
  - **Microprocessor**: a computer that is contained in a single integrated circuit (all peripherals are off the CPU chip).
  - **Microcontroller**: A microprocessor with a number of integrated peripherals, typically used in control-oriented applications (everything is on one chip).
  - **Central processing unit**: The component of a computer system with the capability to control the interpretation and execution of the instructions. The CPU includes the arithmetic-logic unit and the control unit.

Background Module1

- **Introduction**
  - **Primary Memory**: Also called Main memory. It is a volatile/nonvolatile memory that holds the program and intermediate data during the course of a program execution.
  - **Secondary Memory**: A nonvolatile memory that is used to hold the program and data between the runs.

Background Module1

- **Introduction**
  - **Assembly Language**: A symbolic representation of machine language.
  - **Assembler**: A compiler that translates assembly language to machine language.
  - **Translation**: To change information from one form of representation to another without changing the meaning.
  - **μ operation**: A basic operation that is executed during a clock cycle.

Background Module1

- **Introduction**
  - **Transmission**: The act of sending information from one location and the receiving of the same information in another location.
  - **Logic Transfer Level**: A method in which a computer is studied based on its functionality.
  - **Register Transfer Language**: A tool which allows to represent, study, and analyze a computer at the Logic Transfer Level.

Background Module1

- Some Facts
  - Processor
    - Logic capacity: increases about 30% per year
    - Performance: 2x every 1.5 years
  - Memory
    - DRAM capacity: 4x every 3 years
    - Memory speed: 1.5x every 10 years
    - Cost per bit: decreases about 25% per year
  - Disk
    - Capacity: increases about 60% per year

Background Module1

- **Questions**
  - Why are we interested in computer?
  - what is the distinction between program and data?
  - What is a machine dependent language?
  - What is the difference between machine dependent and machine independent languages?
  - What is the difference between assembly instruction and machine instruction?
  - What is the difference between the assembly instruction and the micro instruction?

Background Module1

- **Questions**
  - Define the term “Little Endian”.
  - Define the term “Big Endian”.
  - Define the term “Instruction Format”.

Background Module1

- Performance Measures
  - We will make an attempt to introduce several performance metrics to evaluate the behavior of a computer.
  - We are also interested to study the suitability of these performance metrics.

Background Module1

- Performance Measures
  - Response Time **(**Execution time, Latency**)** — The time elapse between the start and the completion of an event.
  - Throughput **(**Bandwidth**)** — The amount of work done in a given time.
  - Performance — Number of events occurring per unit of time.

Background Module1

- Performance Measures
  - Note execution time is the reciprocal of performance — lower execution time implies higher performance.
  - Note Response time, Throughput, and Performance are all closely related to each other.

Background Module1

- Performance Measures
  - A system (*X*) is faster than (*Y*), if for a given task, the response time on *X* is lower than on *Y*.

$$n = \frac{\text{Execution time}_Y}{\text{Execution time}_X} = \frac{\frac{1}{\text{Performance}_Y}}{\frac{1}{\text{Performance}_X}} = \frac{\text{Performance}_X}{\text{Performance}_Y}$$

Background Module1

- Performance Measures — Example
  - Machine *A* runs a program in 10 seconds and machine *B* runs the same program in 15 seconds. Therefore:

$$n = \frac{\text{Execution time}_B}{\text{Execution time}_A} = \left(\frac{15}{10}\right) = 1.5$$

Background Module1

- Performance Measures
  - Response Time (Elapse time) — The latency to complete a task, including disk accesses, memory accesses, I/O activities, operating system overhead, ...

Background Module1

- Performance Measures
  - CPU time — The time the *CPU* is computing. It is further divided into:
    - User CPU time — The CPU time spent in the program,
    - System CPU time — The CPU time spent in operating system performing tasks requested by the program.

Background Module1

- Performance Measures
  - Average Execution time — Equal probability of running programs in the workload

$$\frac{1}{n}\sum_{i=1}^{n} Time_i$$

Where $Time_i$ is the execution time of the i^{th} program
And n is the number of the program in the workload.

Background Module1

- Performance Measures
  - Consequently we can define Harmonic Mean as

$$\frac{n}{\sum_{i=1}^{n} \frac{1}{Rate_i}}$$

where $Rate_i$ is proportional to $\frac{1}{Time_i}$

Background Module1

- Performance Measures
  - Weighted Execution time — unequal mix of programs in the workload

$$\sum_{i=1}^{n} Weight_i \times Time_i$$

where $weight_i$ is the frequency of the i^{th} program in the workload.

Note $\sum_{i=1}^{n} Weight_i = 1$

Background Module1

- Performance Measures
  - Similarly, weighted harmonic mean is defined as:

$$\frac{1}{\sum_{i=1}^{n} \frac{weight_i}{Rate_i}}$$

Background Module1

- Performance Measures
  - Million Instructions Per Second — MIPS is another performance measure to be used to evaluate computers.
  - MIPS (meaningless Indication of Processor Speed)

$$MIPS = \frac{I_C}{\text{Execution time} * 10^6}$$

Background Module1

- Performance Measures
  - Million Floating Point Operations Per Second — MFLOPS is another performance measure to be used to evaluate computers.

$$\text{MFLOPS} = \frac{\text{Number of floating point operations in a program}}{\text{Execution time} * 10^6}$$

Background Module1

- Performance Measures
  - Justify the following:
    - MIPS depends on the instruction set. Thus, it is hard to compare computers with different instruction sets.
    - MIPS depends on the instruction mix in a program.
    - MIPS can vary inversely to performance.