Lecture 2: Microprocessor Systems

Seyed-Hosein Attarzadeh-Niaki

Some slides due to Hongzi Zhu

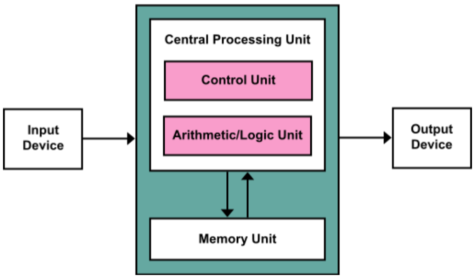
Review

- Early computers
- Mechanical age
- Electrical age
- Electronic age
  - Transistors
  - Integrated circuits
- Evolution of x86 and ARM processors

Outline

- Microcomputer systems
- Types of processors
- Components of microcomputer systems

Von Neumann Architecture



1.Five components partitioning

- Input
- Output
- Memory
- ALU
- Control unit

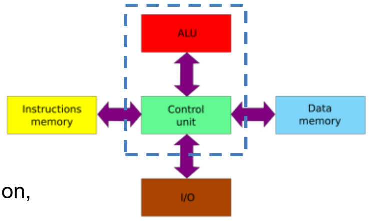
I/O devices

Memory

CPU

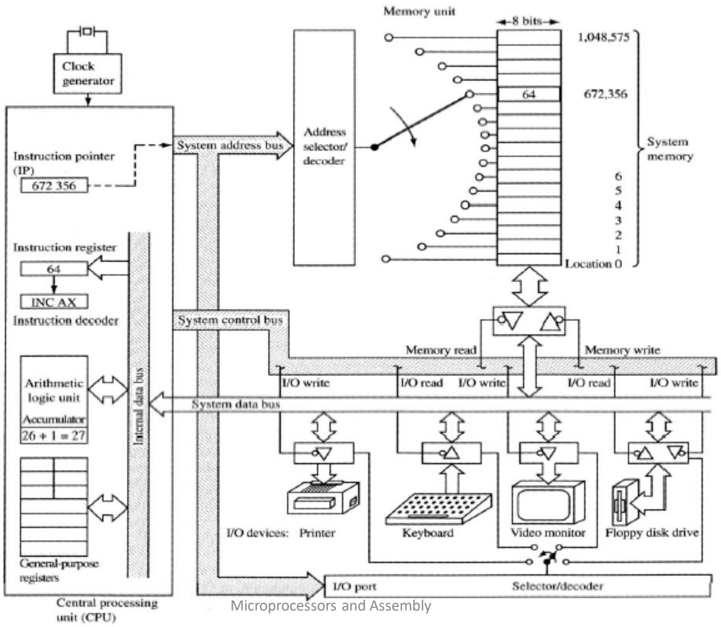
2. Three key concepts:

- Both instructions and data are *stored* in a *single* read-write memory
- The contents of memory are *addressable* by location, without regard to the type of data
- Execution occurs in a *sequential* fashion



Harvard architecture

Stored Program Concept

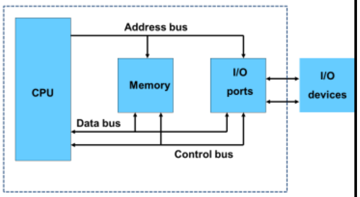


Microprocessors

- The CPU circuitry can been reduced to *IC* (Integrated Circuit) *s*cale, consisting of ***ALU***, ***CU*** and ***registers***
- Contains **no** RAM, ROM, or I/O ports on the chip itself
- E.g., Intel's x86 family (8088, 8086, 80386, 80386, 80486, Pentium); Motorola's 680x0 family (68000, 68010, 68020, etc)

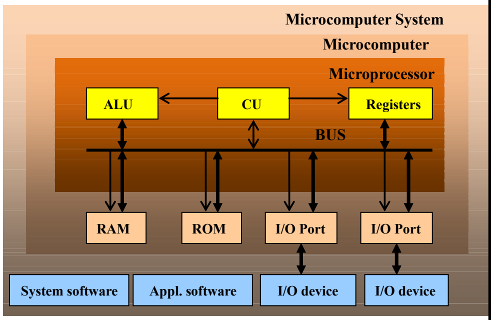
Microcomputer

- CPU: *processes information stored in the memory*
  - *Microprocessor*
- *Memory: stores both instructions and data*
  - ROM, RAM
- *Input/Output ports: provide a means of communicating with the CPU*
  - Connecting I/O devices, e.g., keyboard, monitor, tape, disk, printer and etc.
- *BUS: interconnecting all parts together*
  - Address bus
  - Data bus
  - Control bus



Microcomputer System

- Microcomputer
- Peripheral I/O devices
- Software
  - System software
    - e.g., OS, compilers, drivers
  - Application software
    - e.g. Word, MatLab, Media player, Latex…



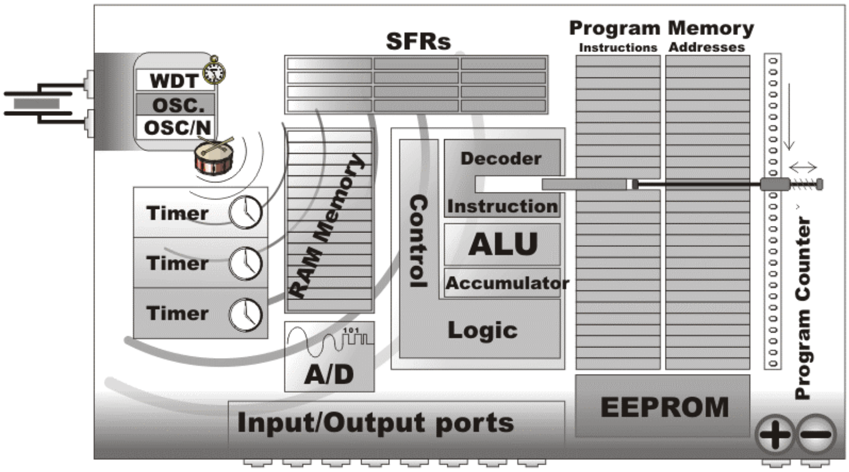
Microcomputer System Structure

- Microcomputer system (μCS)
  - Microcomputer （ μC ）
    - Microprocessor （μP）
      - ALU
      - CU
      - Registers
    - Memory
    - I/O ports
  - I/O devices
  - System software
  - Application software

Microcomputer / Microcontroller

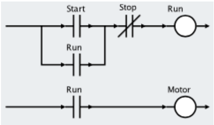
- A *microcontroller* is a *computer-on-a-chip*
  - A type of microprocessor emphasizing self-sufficiency and cost-effectiveness, in contrast to a general-purpose microprocessor
- Main difference between a microcontroller and a microprocessor:
  - Microprocessor is a CPU with ALU, control unit and registers
  - Microcontroller has a simple CPU with additional elements like ROM, RAM, peripherals (timer, I/O ports, etc).
- The boundary can be blurred
  - Intel Atom, AMD Geode

Components of a Microcontroller



Programmable Logic Controller (PLC)

- A specialized form of a microcontroller for industrial automation
- Ruggedized and adapted for the control of manufacturing processes
- Originated as replacements for control circuits using electrical relays to control machinery
- Classically programmed using ladder logic
  - A graphical notation originally used to specify logic constructed with relays and switches
- Today PLCs: Microcontrollers with suitable IO interfaces



Digital Signal Processor (DSP)

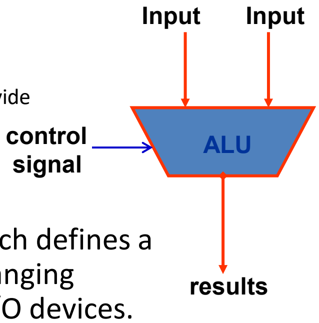
- Processors designed specifically to support numerically intensive signal processing applications
- Applications: Audio and video processing, radar, sonar, medical electronics, etc.
  - Large amounts of data
  - Perform sophisticated mathematical operations on the data
    - System identification, frequency analysis, machine learning, and feature extraction
- Characteristics
  - Harvard architecture
  - Addressing modes supporting auto increment, circular buffers, and bit-reversed addressing (for FFT)
  - Support fixed-point data precisions of 16-24 bits
- **Difficult to program**
  - Assembly or specialized (assembly-coded) C libraries

Graphics Processor Unit (GPU)

- A specialized many-core processor designed especially to perform the calculations required in graphics rendering
  - Render text and graphics, combine multiple graphic patterns, draw objects, 3D graphics, shading, and digital video
- GPUs have evolved towards more general programming models (CUDA, OpenCL)
- Typically quite power hungry

Hardware: CPU

- Arithmetic Logic Unit (ALU)
  - Arithmetic functions: add, subtract, multiply and divide
  - Logic functions: AND, OR, and NOT



- Control Unit works under *instructions*
- An instruction is a pre-defined code which defines a specific operation, processing and exchanging information among CPU, memory and I/O devices.
- CU contains an *instructor decoder*
  - decodes an instruction and generates all control signals, coordinating all activities within the computer
- CU contains a *program counter*
  - points to the address of the next instruction to be executed

Registers

- Registers are *specialized locations* that the *CPU uses* to read or write a binary number
- Registers are used to *hold data and address values*
- The *number and types* of registers depends on the CPU design
- Registers are used both by the CPU and the I/O subsystem

Instruction Set

Arithmetic and Logic Operations

Operation	Comment
Addition	
Subtraction	
Multiplication	
Division	
AND	Logical Multiplication
OR	Logical Addition
NOT	Logical Inversion
NEG	Arithmetic Inversion
Shift	
Rotate	

Decisions

Decision	Comment
Zero	Test a number for zero or not zero
Sign	Test a number for positive or negative
Carry	Test for a carry after addition or a borrow after subtraction
Parity	Test a number for an even or an odd number of ones
Overflow	Test for an overflow that indicates an invalid signed result after addition or subtraction

Reflection

- What if we need hardware support for more complex operations?
  - Square root
  - Complex filters
  - …

Instruction Sets

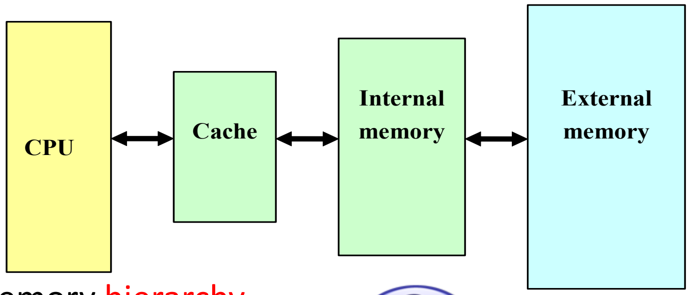
CISC (Complex Instruction Set Computers)

- Variable instruction length (1 word- *n* words)
- Variable execution time of different format instructions
- More instruction formats
- Upwardly compatible (new instruction set contains earlier generation's instructions)
- e.g., 80x86 family has more than 3000 instructions

RISC (Reduced Instruction Set Computers)

- Fixed size (1 word)
- Fixed time for all instructions
- Easy to pipeline the RISC instructions (fast)
- Fewer formats (simple hardware, shorter design cycle)
- e.g., PowerPC, MIPS, ARM, PIC's MCU

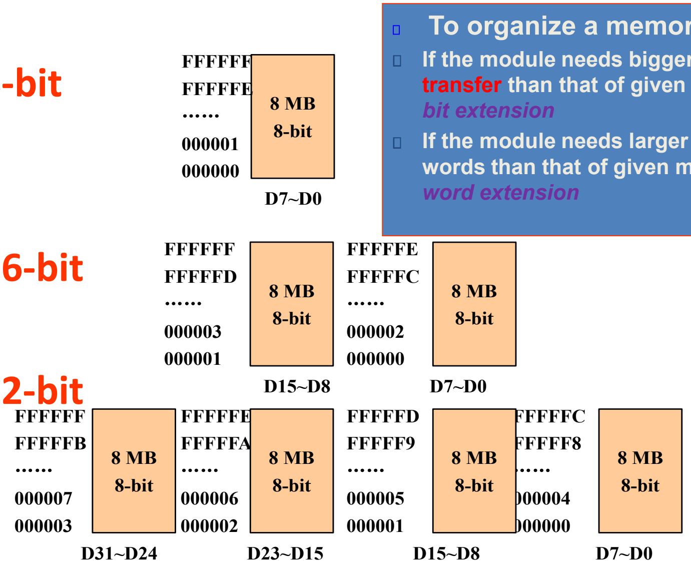
Hardware: Memory



- Memory <span>hierarchy</span>
  - *Cache*
  - *Primary memory: ROM, RAM*
  - *Secondary memory: magnetic disk, optical memory, tape, …*

Discrete Memory Module Organization

- 8-bit
- 16-bit
- 32-bit



- To organize a memory module:
  - If the module needs bigger unit of transfer than that of given memory chips, *bit extension*
  - If the module needs larger number of words than that of given memory chips, *word extension*

Memory Characteristics I

Location

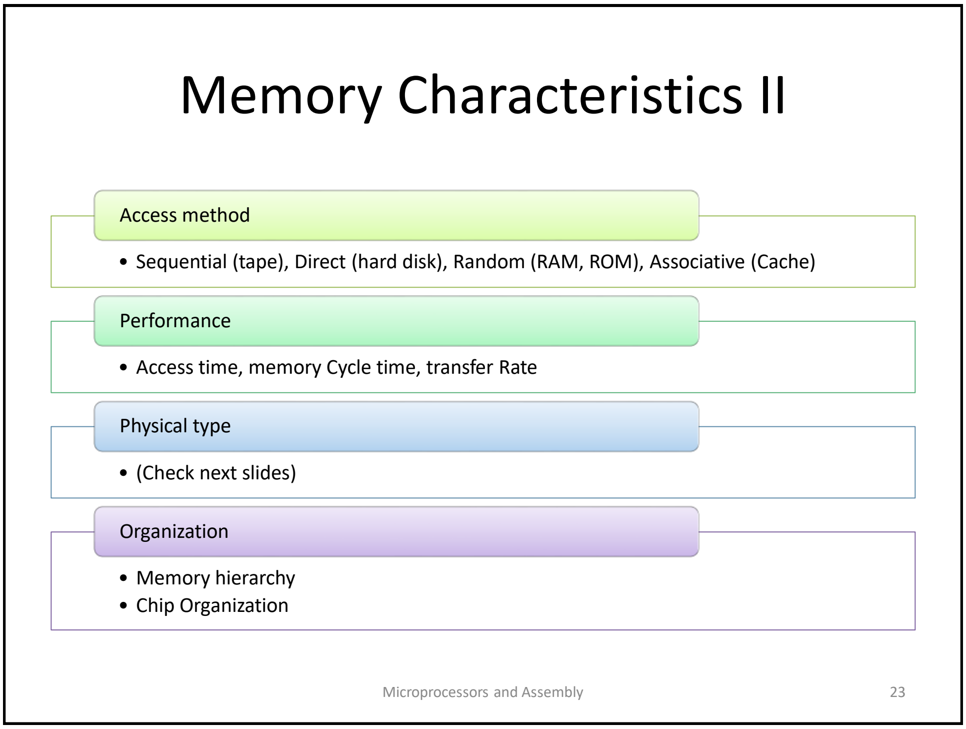
- CPU (registers), Internal (Cache, main memory), External (Disk, tape, DVD)

Capacity

- Word size, Number of words

Unit of transfer

- Internal: Word, External: Usually a block
- Addressable unit: Normally a Byte for internal memory, Cluster on disks



Memory Characteristics II

Access method

- Sequential (tape), Direct (hard disk), Random (RAM, ROM), Associative (Cache)

Performance

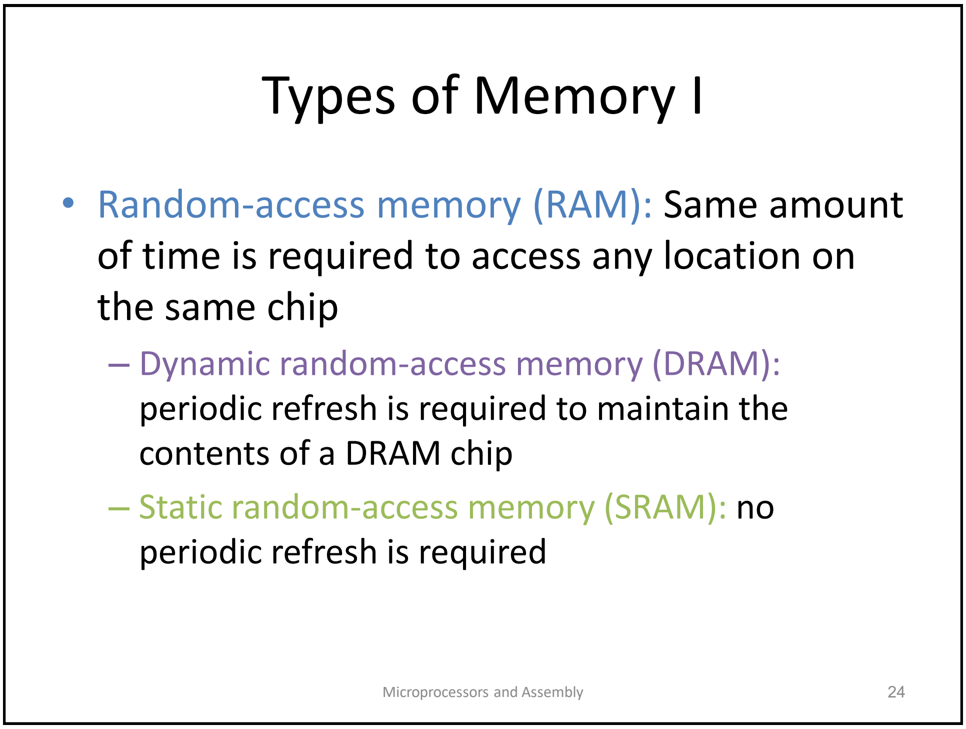
- Access time, memory Cycle time, transfer Rate

Physical type

- (Check next slides)

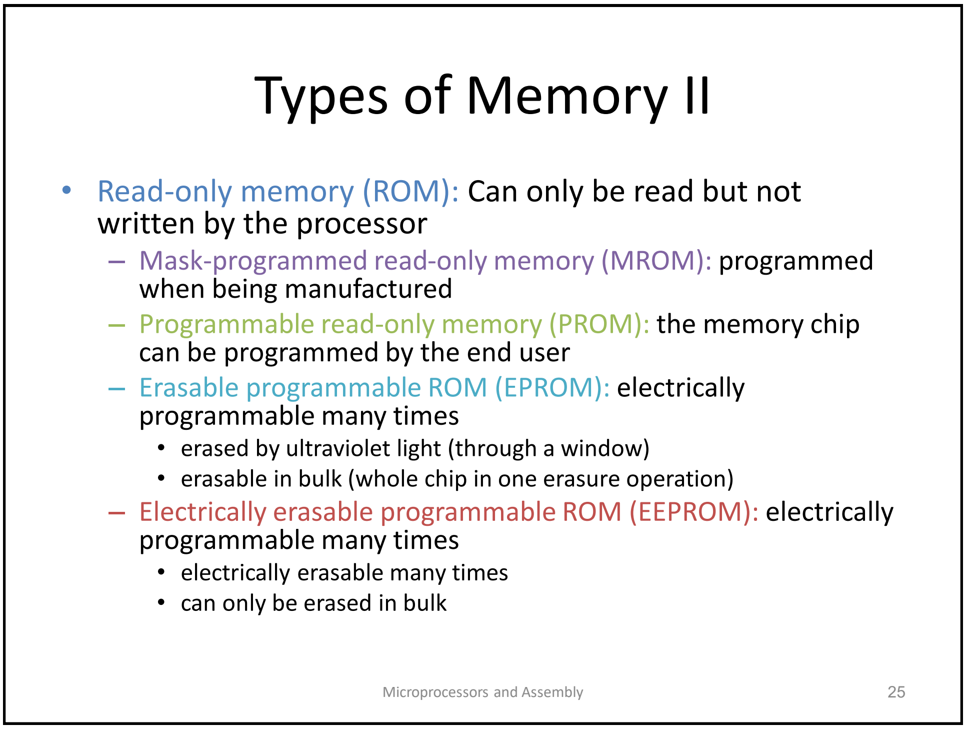
Organization

- Memory hierarchy
- Chip Organization



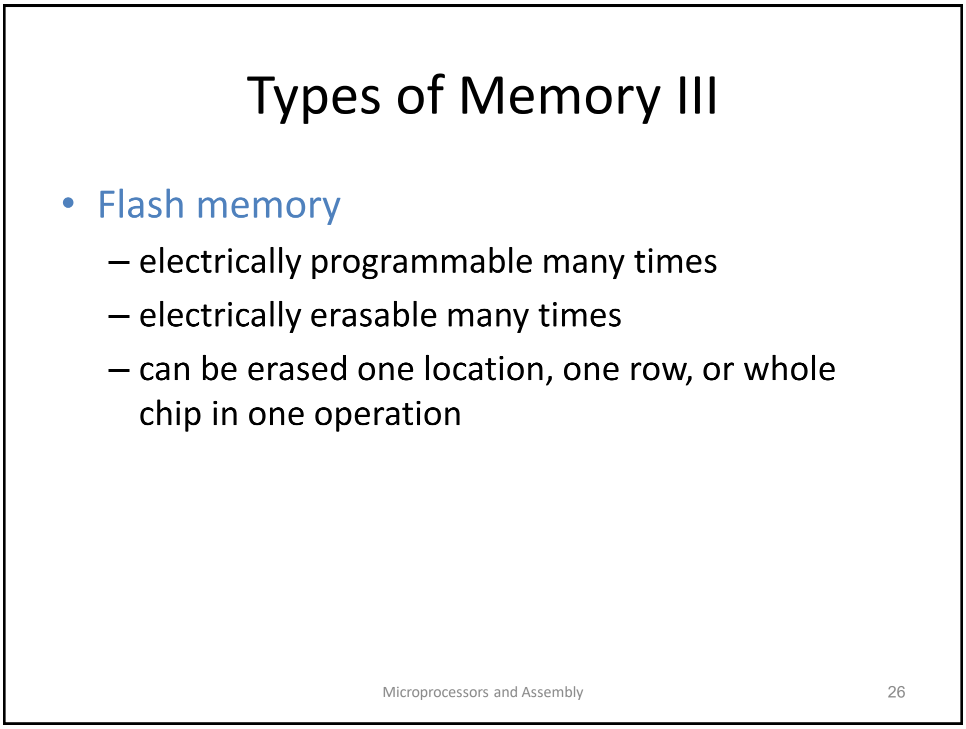
Types of Memory I

- Random-access memory (RAM): Same amount of time is required to access any location on the same chip
  - Dynamic random-access memory (DRAM): periodic refresh is required to maintain the contents of a DRAM chip
  - Static random-access memory (SRAM): no periodic refresh is required



Types of Memory II

- Read-only memory (ROM): Can only be read but not written by the processor
  - Mask-programmed read-only memory (MROM): programmed when being manufactured
  - Programmable read-only memory (PROM): the memory chip can be programmed by the end user
  - Erasable programmable ROM (EPROM): electrically programmable many times
    - erased by ultraviolet light (through a window)
    - erasable in bulk (whole chip in one erasure operation)
  - Electrically erasable programmable ROM (EEPROM): electrically programmable many times
    - electrically erasable many times
    - can only be erased in bulk



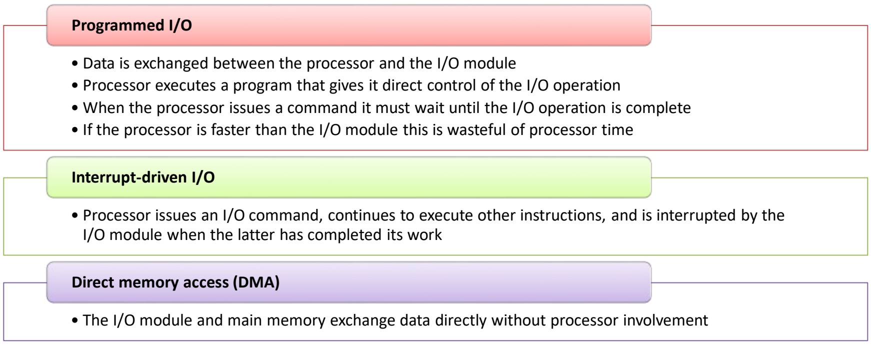
Types of Memory III

- Flash memory
  - electrically programmable many times
  - electrically erasable many times
  - can be erased one location, one row, or whole chip in one operation

I/O Devices

- Input devices such as *switches* and *keyboards* provide binary information to the microprocessor
- Output devices such as *LEDs*, *video screens*, and *printers* receive information from the microprocessor

I/O Operation Techniques



Programmed I/O

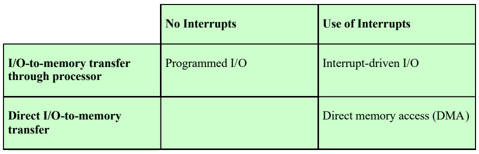
- Data is exchanged between the processor and the I/O module
- Processor executes a program that gives it direct control of the I/O operation
- When the processor issues a command it must wait until the I/O operation is complete
- If the processor is faster than the I/O module this is wasteful of processor time

Interrupt-driven I/O

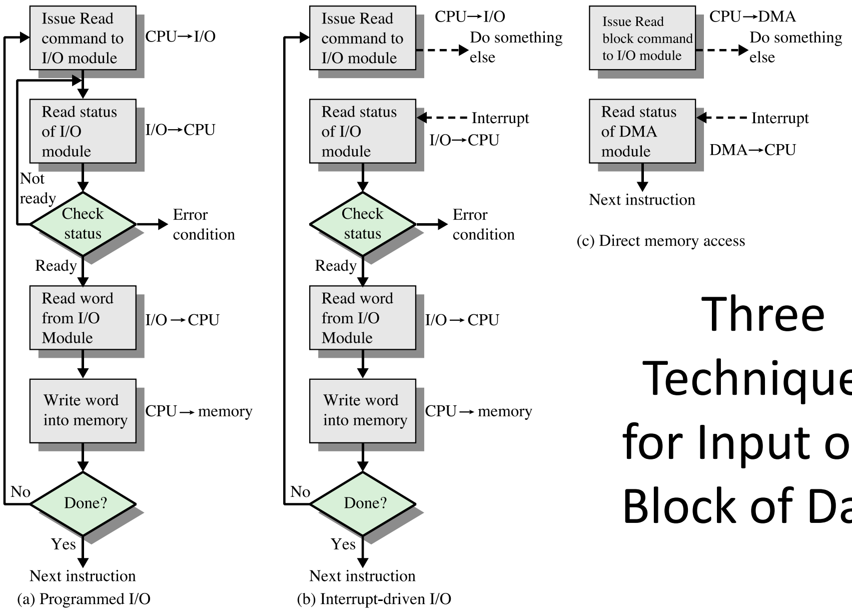
- Processor issues an I/O command, continues to execute other instructions, and is interrupted by the I/O module when the latter has completed its work

Direct memory access (DMA)

- The I/O module and main memory exchange data directly without processor involvement



	No Interrupts	**Use of Interrupts**
I/O-to-memory transfer through processor	Programmed I/O	Interrupt-driven I/O
Direct I/O-to-memory transfer		Direct memory access (DMA)



Issue Read command to I/O module

CPU→I/O

Read status of I/O module

I/O→CPU

Not ready

Check status

Error condition

Ready

Read word from I/O Module

I/O→CPU

Write word into memory

CPU→ memory

No

Done?

Yes

Next instruction

(a) Programmed I/O

Issue Read command to I/O module

CPU→I/O

Do something else

Read status of I/O module

Interrupt

I/O→CPU

Check status

Error condition

Ready

Read word from I/O Module

I/O→CPU

Write word into memory

CPU→ memory

No

Done?

Yes

Next instruction

(b) Interrupt-driven I/O

Issue Read block command to I/O module

CPU→DMA

Do something else

Read status of DMA module

Interrupt

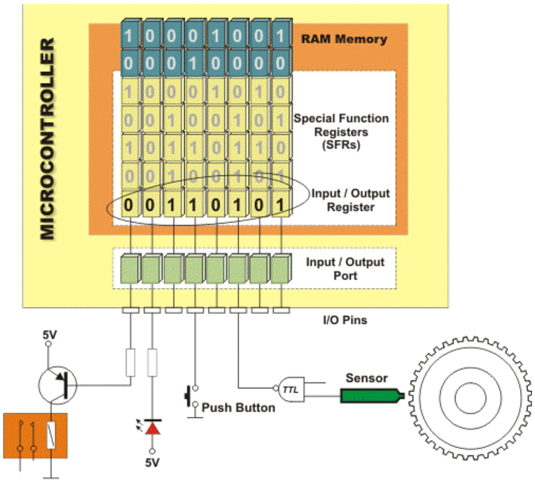
DMA→CPU

Next instruction

(c) Direct memory access

Three Techniques for Input of a Block of Data

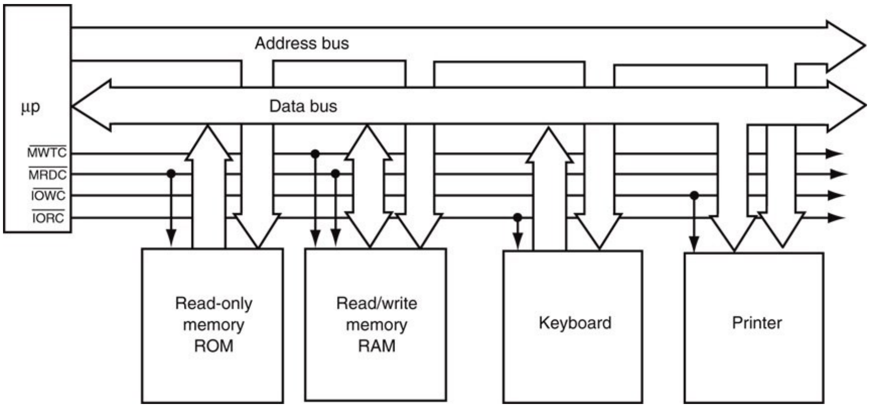
IO Devices in a Microcontroller

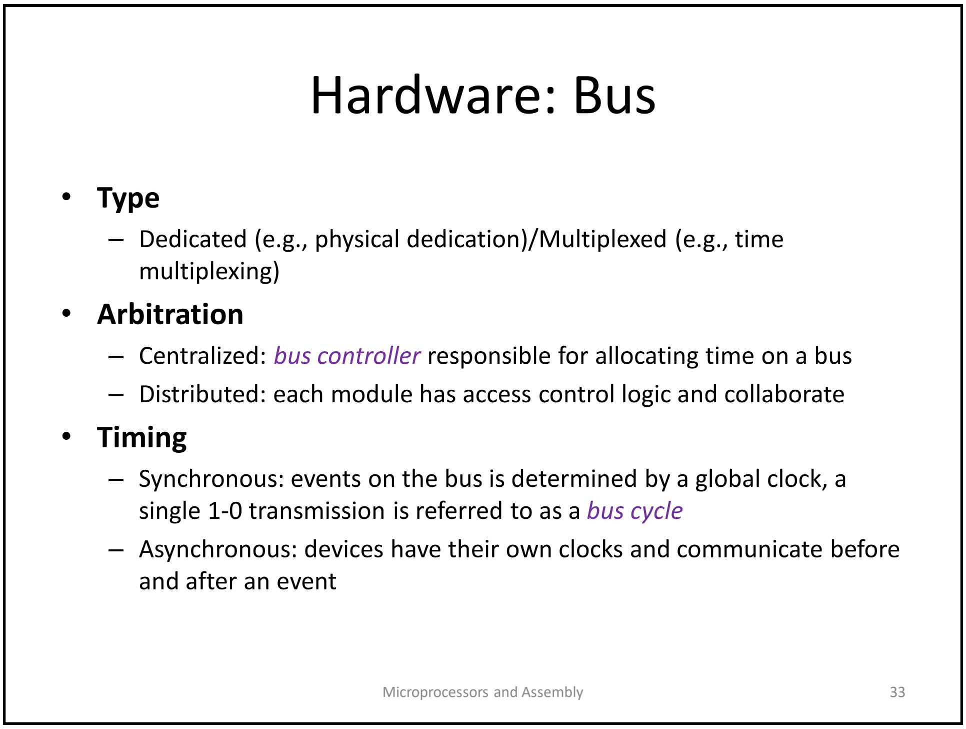


Hardware: Bus

- **A bus is a group of wires acting as a communication pathway connecting two or more devices**
- A shared transmission medium: one device at a time
- Transfer *address*, *data*, and *control information* between microprocessor, memory and I/O
- Arbitration:
  - Distributed protocols
    - e.g., CSMA/CD
  - Centralized scheme:
    - Master/Slave
      - Master activates a bus
      - Slave passively waits for command

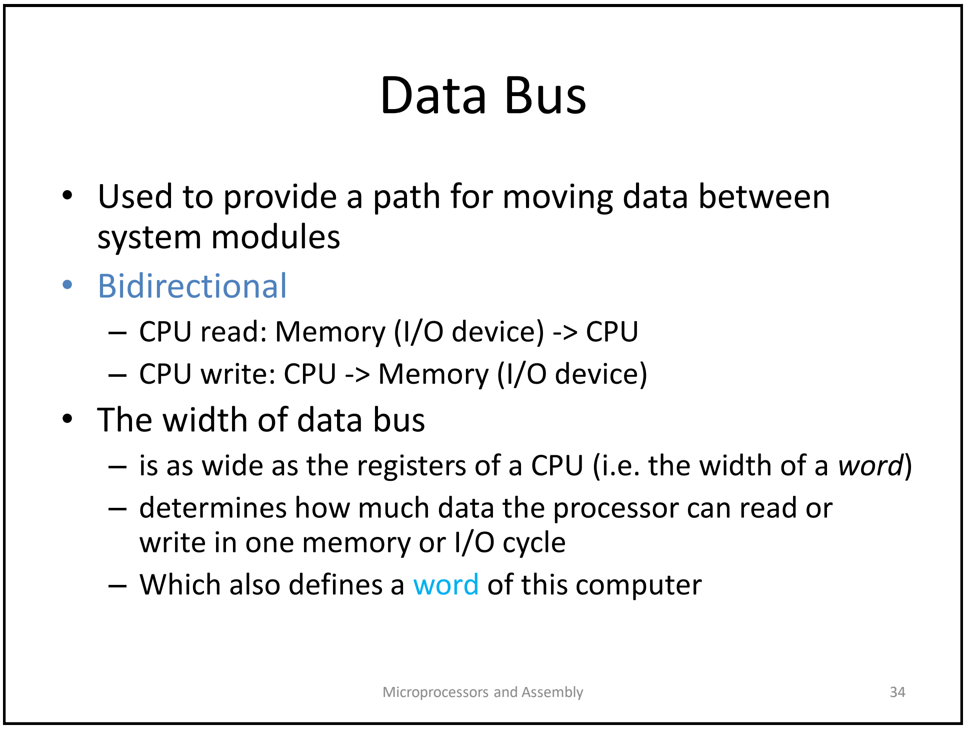
Bus Structure Illustration





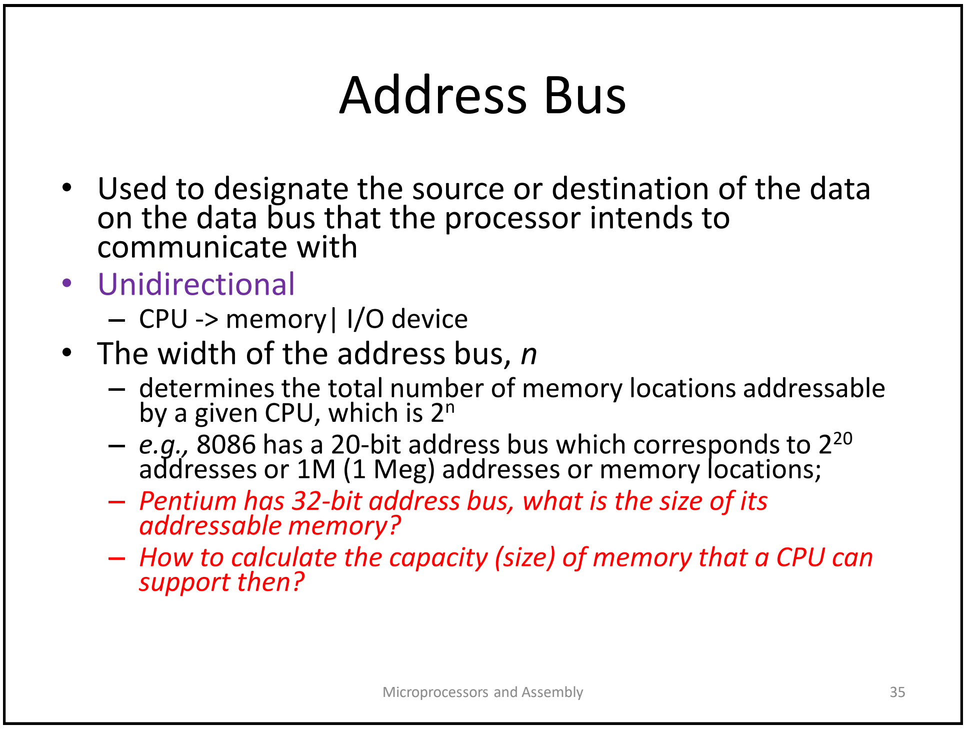
Hardware: Bus

- **Type**
  - Dedicated (e.g., physical dedication)/Multiplexed (e.g., time multiplexing)
- **Arbitration**
  - Centralized: *bus controller* responsible for allocating time on a bus
  - Distributed: each module has access control logic and collaborate
- **Timing**
  - Synchronous: events on the bus is determined by a global clock, a single 1-0 transmission is referred to as a *bus cycle*
  - Asynchronous: devices have their own clocks and communicate before and after an event



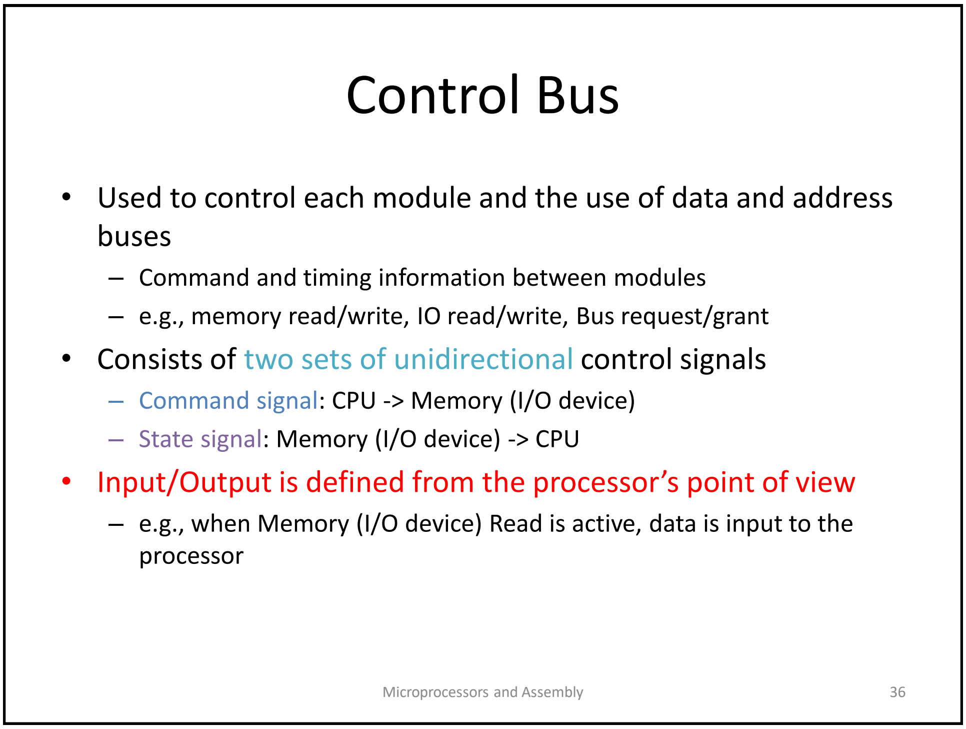
Data Bus

- Used to provide a path for moving data between system modules
- Bidirectional
  - CPU read: Memory (I/O device) -> CPU
  - CPU write: CPU -> Memory (I/O device)
- The width of data bus
  - is as wide as the registers of a CPU (i.e. the width of a *word*)
  - determines how much data the processor can read or write in one memory or I/O cycle
  - Which also defines a word of this computer



Address Bus

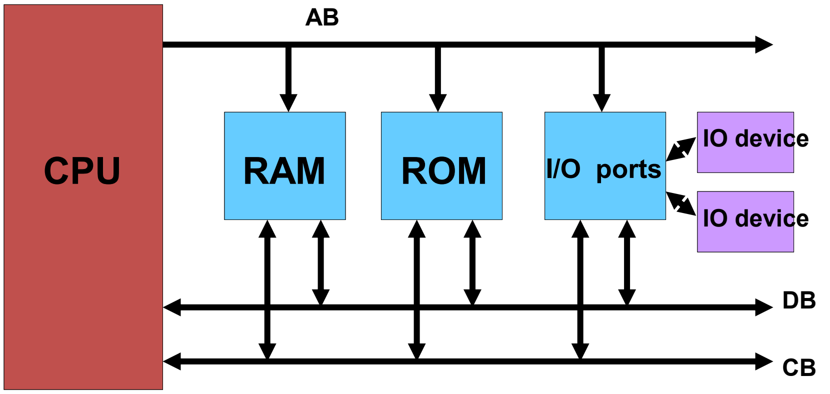
- Used to designate the source or destination of the data on the data bus that the processor intends to communicate with
- Unidirectional
  - CPU -> memory| I/O device
- The width of the address bus, *n*
  - determines the total number of memory locations addressable by a given CPU, which is 2^n
  - *e.g.*, 8086 has a 20-bit address bus which corresponds to 2^{20} addresses or 1M (1 Meg) addresses or memory locations;
  - *Pentium has 32-bit address bus, what is the size of its addressable memory?*
  - *How to calculate the capacity (size) of memory that a CPU can support then?*



Control Bus

- Used to control each module and the use of data and address buses
  - Command and timing information between modules
  - e.g., memory read/write, IO read/write, Bus request/grant
- Consists of two sets of unidirectional control signals
  - Command signal: CPU -> Memory (I/O device)
  - State signal: Memory (I/O device) -> CPU
- Input/Output is defined from the processor's point of view
  - e.g., when Memory (I/O device) Read is active, data is input to the processor

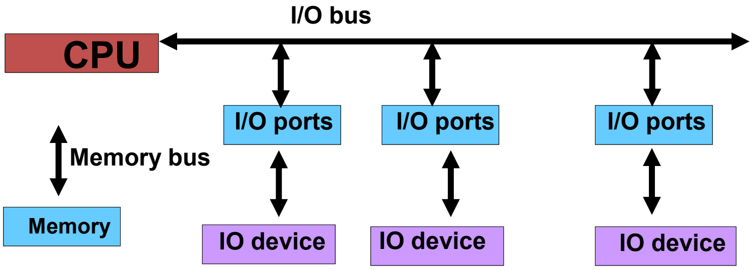
Single-bus Structure



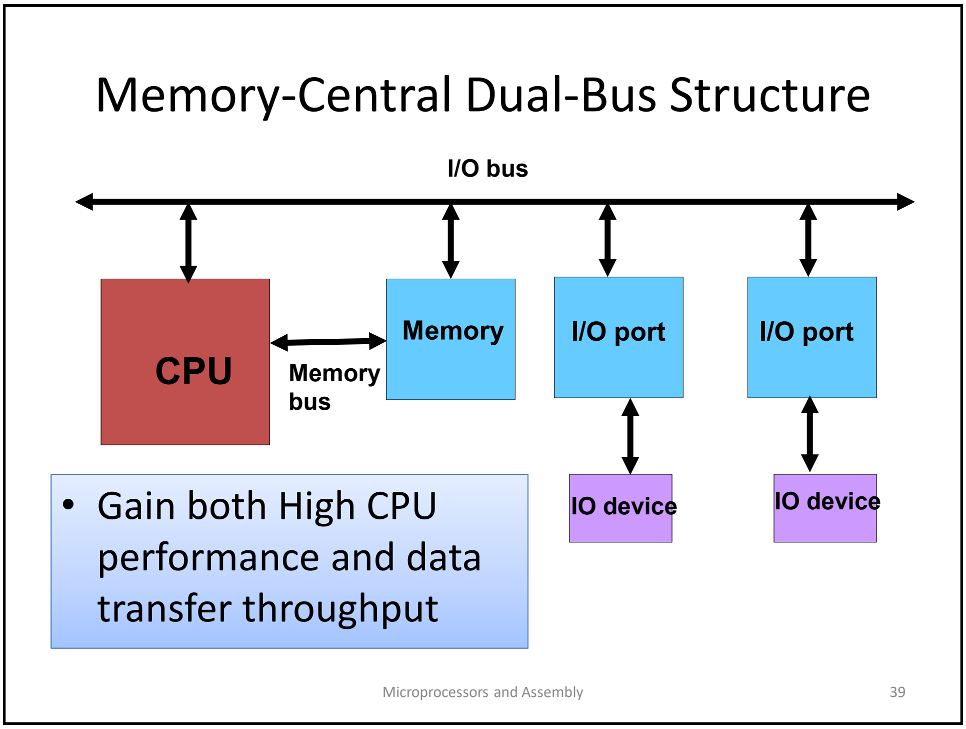
A bus connects all modules

- **Pro**: simple
- **Con**: poor performance in terms of throughput

CPU-Central Dual-Bus Structure

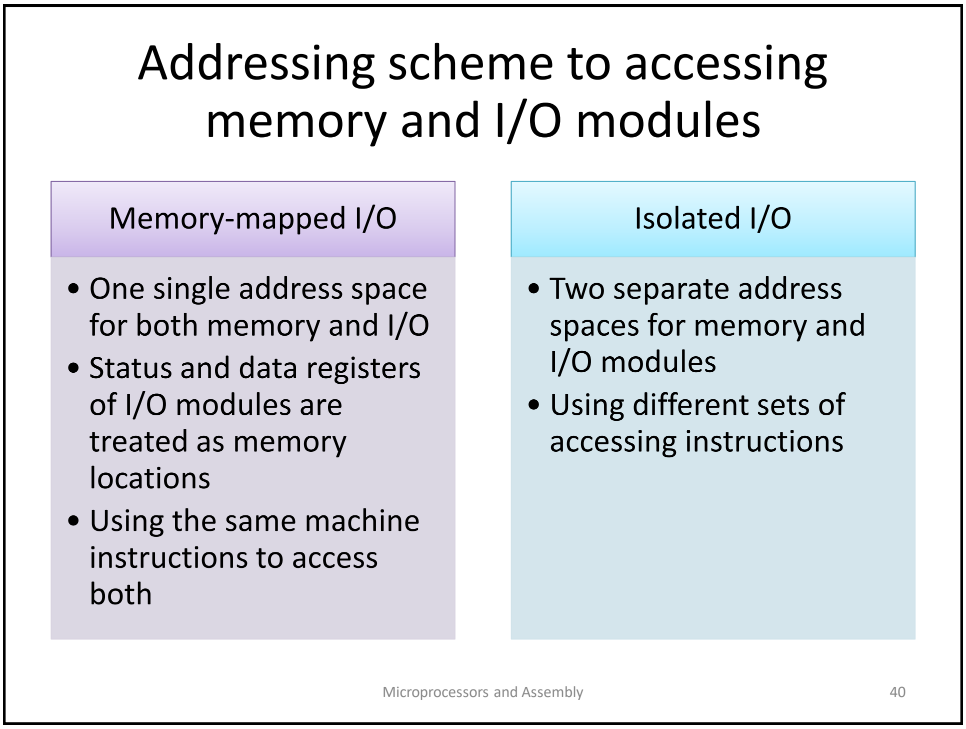


- A dedicated bus between CPU and memory, and a dedicated bus between CPU and I/O devices
  - **Pro**: efficient in terms of data transfer
  - **Con**: information between memory and I/O devices has to go through CPU. Therefore, poor CPU performance



Memory-Central Dual-Bus Structure

- Gain both High CPU performance and data transfer throughput



Addressing scheme to accessing memory and I/O modules

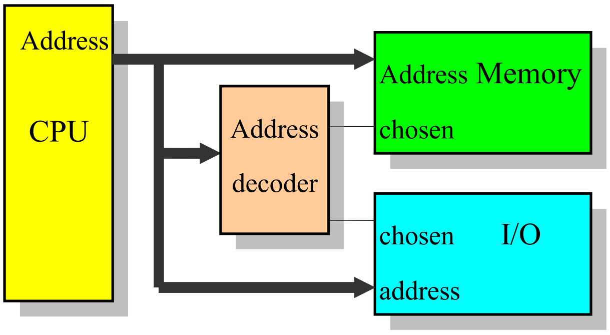
Memory-mapped I/O

- One single address space for both memory and I/O
- Status and data registers of I/O modules are treated as memory locations
- Using the same machine instructions to access both

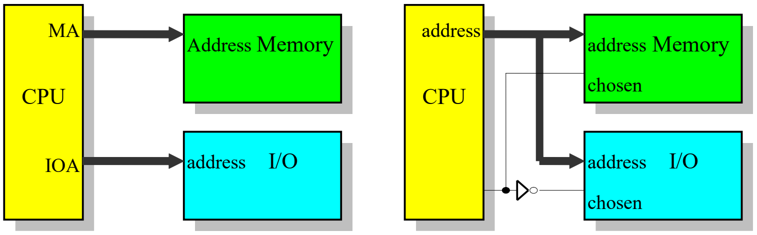
Isolated I/O

- Two separate address spaces for memory and I/O modules
- Using different sets of accessing instructions

Memory-mapped I/O



Isolated I/O

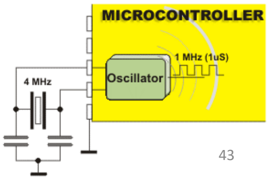


Dedicated address lines **Multiplexing address lines**

What is the essential difference between the memory-mapped and isolated I/O addressing schemes?

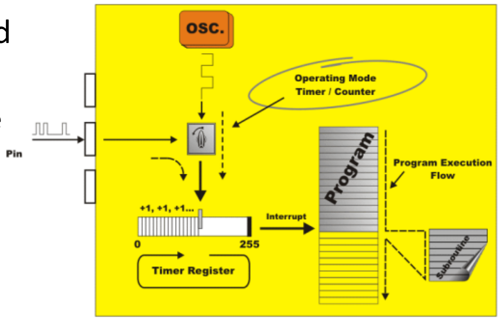
System Clock

- A particular type of signal that *oscillates* between a high and a low state
- *Coordinates* actions of circuits
- Circuits using the clock signal for synchronization may become active at either the *rising edge*, *falling edge*, or, in the case of double data rate, *both in the rising and in the falling edges* of the clock cycle
- The system clock determines the *speed of the microprocessor*
  - If the clock is sped up by a small amount, the microprocessor will probably still work, at the faster rate
  - If the clock is sped up by a larger amount, the microprocessor will start to make errors. Instructions will not be executed correctly.



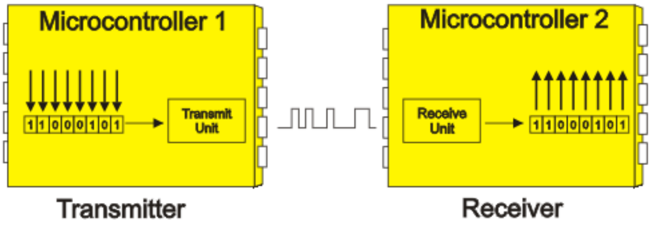
Timer/Counter

- A special register which its content is automatically incremented/decremented on each incoming pulse
  - *Timer*: the pulse is generate by an oscillator
  - *Counter:* the pulse is an external signal
- An interrupt signal can be generated upon overflow/zero



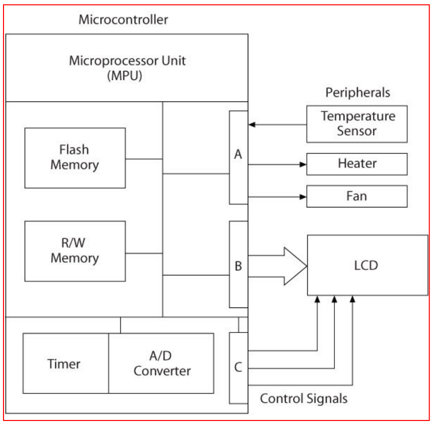
Serial Communication

- Suitable for long distance communications
- Different types of serial communication
  - How many devices communicate?
  - How fast they communicate?
  - What is the distance between devices?
  - Is it necessary to send and receive data simultaneously?

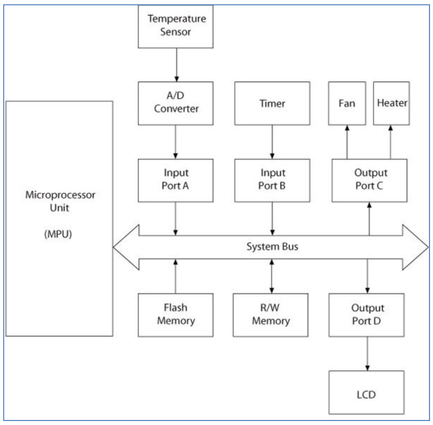


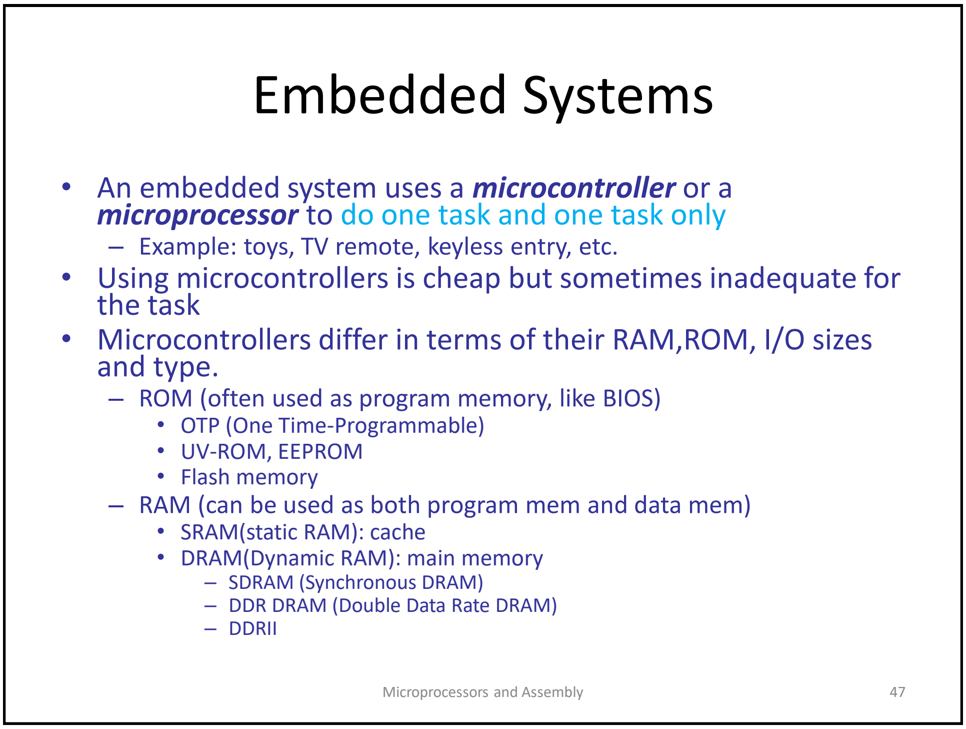
A time and temperature system

Microcontroller based design



Microprocessor based design





Embedded Systems

- An embedded system uses a ***microcontroller*** or a ***microprocessor*** to do one task and one task only
  - Example: toys, TV remote, keyless entry, etc.
- Using microcontrollers is cheap but sometimes inadequate for the task
- Microcontrollers differ in terms of their RAM,ROM, I/O sizes and type.
  - ROM (often used as program memory, like BIOS)
    - OTP (One Time-Programmable)
    - UV-ROM, EEPROM
    - Flash memory
  - RAM (can be used as both program mem and data mem)
    - SRAM(static RAM): cache
    - DRAM(Dynamic RAM): main memory
      - SDRAM (Synchronous DRAM)
      - DDR DRAM (Double Data Rate DRAM)
      - DDRII