Ch.0 & Ch.1
Introduction to Microcontroller

Yongjun Park
Hanyang University

Outline

- **What is Microcontroller? (Ch. 1.1)**
- 8051 Microcontroller (Ch. 1.2)
- Numbering and coding:
  binary, decimal, hexa-decimal, ASCII (Ch. 0.1)
- Basic digital logics (Ch. 0.2)
- Operations of a computer (Ch. 0.3)

Hanyang University
Division of Computer Science & Engineering

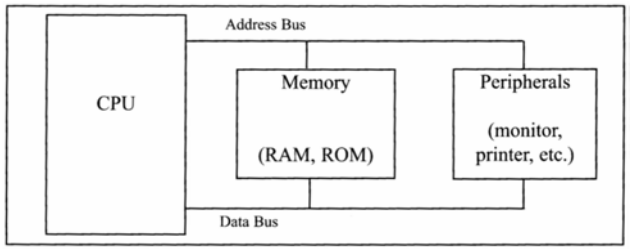
What is Microcontroller?

- **What are inside a computer?**
  - Central Processing Unit (CPU)
    - Execute(Process) information stored in memory
  - Memory
    - Store information
    - Random Access Memory (RAM)
    - Read-Only Memory (ROM)
  - I/O(Input/Output) devices
    - Also called peripherals
    - Monitor, Keyboard, Hard Drive, CD-ROM, Video Card, …
  - Bus
    - Strip of wires connecting CPU, memory, and I/O devices

Hanyang University
Division of Computer Science & Engineering

What is Microcontroller?

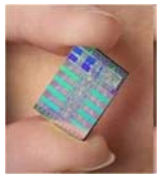
- **What are inside a computer?**
  - Bus
    - Strip of wires connecting CPU, memory, and I/O devices



Hanyang University
Division of Computer Science & Engineering

What is Microcontroller?

- **Microprocessor vs. Microcontroller**
  - Microprocessor: A CPU on a single integrated chip(IC)
    - A special type of CPU
    - The brain of computer
    - E.g.:
      - 8086, 80286, 80386, 80486, Pentium, Core 2 Duo, Core 2 Quad, …
      - K5, K6, Atholon 64, Opteron, Phenon, …
      - PowerPC G4, PowerPC G5, Xenon, Broadway, Cell
    - Contains no RAM, no ROM, no I/O ports


What is Microcontroller?

- **Microprocessor vs. Microcontroller**
  - Microcontroller:
    A microprocessor, and RAM, ROM, I/O ports, and timer on a single chip (Also called MCU)
    - "Computer on a chip"
    - Also called MCU (Micro-Controller Unit)
    - Usually not as powerful as general purpose microprocessor
    - Low power consumption, small in size, low cost.
    - A lot of MCUs are application specific
      (as against the general purpose microprocessor)

Microcontroller: Applications

- **Applications of Micro-Controller**
  - Home
    - TV, MP3, Camera, DVD/CD player, Cell phone, Alarm clock, …
  - Office
    - Scanner, Printer, Fax machine, Copier, Wireless router, …
  - Industry
    - Machinery, Equipment, Instrumentation, Rocket, …
- **Microcontroller is everywhere!**
  - Most of the applications requires the MCU to be
    - Small in size: the final product is small
    - Low cost: lower the price of the end product
    - Low power consumption: longer battery life
    - Simple (as long as it can have the job done)

Microcontroller: Embedded Systems

- **Embedded system**
  - A system with an embedded special-purpose computer designed to perform one of a few dedicated functions
    - The embedded computer is just part of a bigger system
    - The computer by itself cannot perform any meaningful functions
    - E.g. a MCU embedded in a washer
  - Contrast to: general purpose computer (Personal computer)
    - A PC itself is a computer system
    - You can use it to perform various tasks

Microcontroller: Embedded Systems

- **Embedded system**
  - Usually an MCU is embedded in a complete device including mechanical parts
    - E.g. Camera, Microwave, …
  - The operation software is embedded in hardware
    - E.g. the operation software is stored in the ROM on MCU
    - Doesn't have separate device(CD, Hard drive) to store programs
  - All the applications of MCU can be considered as embedded system

Microcontroller: Course Contents

- **What are we going to learn in this course?**
  - What are inside a microcontroller?
    - The basic structure of a microcontroller
  - How to program a microcontroller?
    - Assembly language
    - C language
  - How to build a system with a microcontroller?
    - I/O ports
    - Hardware connection

Outline

- What is Microcontroller? (Ch. 1.1)
- **8051 Microcontroller (Ch. 1.2)**
- Numbering and coding:
  binary, decimal, hexa-decimal, ASCII (Ch. 0.1)
- Basic digital logics (Ch. 0.2)
- Operations of a computer (Ch. 0.3)

8051

- **Four major 8-bit microcontrollers**
  - Freescale: 6811, Intel: 8051, Zilog: Z8, Microchip: PIC 16X
- **How do we decide which MCU to use?**
  - Speed
  - Power consumption
  - Amount of RAM and ROM on chip
  - Number of I/O pins
  - Cost per unit
  - Packaging
  - Availability
  - How easy is it to develop a product around it

8051: Overview

- **8051**
  - One of the most popular MCUs in the market
    - Several manufactures are building 8051
    - Wide availability, low cost
  - Clean structure: easy to learn, easy to use
- **8051 MCU family**
  - There are different variations of 8051 by different manufactures
    - Intel(8051, 8052), Dallas Semiconductor(DS89C4x0, x = 2, 3, 4, 5), Atmel(AT89C51), Philips, Texas Instruments, …
  - They differ in speed, ROM/RAM size, packaging, timer, I/O pins, timer, operation voltage, and other peripherals
    - E.g. some of them have built-in analog to digital converter(ADC)
  - They all support the same 8051 instruction set

8051

- **The 8051 chip that will be used in our course**
  - DS89C430 by Dallas Semiconductor
    - ROM: 16 KB
    - RAM: 256 Bytes
    - I/O pins: 32
    - Timers: 3
    - Interrupts: 6
    - Clocks per machine cycle: 1
    - Operation voltage: 5V

Outline

- What is Microcontroller? (Ch. 1.1)
- 8051 Microcontroller (Ch. 1.2)
- **Numbering and coding: binary, decimal, hexa-decimal, ASCII (Ch. 0.1)**
- Basic digital logics (Ch. 0.2)
- Operations of a computer (Ch. 0.3)

Numbering: Decimal and Binary

- **Decimal and binary number system**
  - Decimal: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
  - Binary: 0, 1 (used by computer)
  - Weight associated with each digit
    - Decimal: $256_{10} = 2 \times 10^2 + 5 \times 10^1 + 6 \times 10^0$
    - Binary: $101_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
  - Convert binary to decimal
    - Method 1: use the weights of digits
    - Method 2: divide the decimal by 2 repeatedly until the quotient becomes 0, and keep track of the remainder
    - Example
      - Convert 11011_2 to decimal number

Numbering: Hexadecimal System

- **Hexadecimal system: base-16 system**
  - Hexadecimal: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
  - Mainly used as a convenient representation of binary number
    - 4 binary digits → 1 hex digit
  - Convert binary to hexadecimal
    - Example: Convert 111101_2
  - Convert hexadecimal to binary
    - Example: $29BH_{16}$

Number Systems

Decimal	Binary	Hex
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Numbering: Hexadecimal System

- **Hexadecimal system (Cont'd)**
  - Convert from hexadecimal to decimal
    - Use weights of digits (… 16^3, 16^2, 16^1, 16^0)
    - Example: Convert $6FEH_{16}$ to decimal
  - Convert from decimal to hexadecimal
    - Method 1: use weights of digits
    - Method 2: keep divide by 16 and keep track of quotient
    - Example: Convert 45_{10} to hexadecimal

Numbering: Hexadecimal System

- **Hexadecimal system (Cont'd)**
  - Counting
    - Decimal:

      0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18 ,19, 20, …

    - Hexadecimal:

      0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, ?

      0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, 10, 11, 12, 13, 14, …

Numbering: Arithmetic

- **Binary addition**
  - Example: Find the sum of 1101_2 and 1001_2

- **Hexadecimal addition**
  - Example: Find the sum of $23D9H_{16}$ and $94BEH_{16}$

Numbering: 2's Complement

- **2's Complement**
  - To get 2's complement of a binary number
    - 1. Invert all bits (1 → 0, 0 → 1): 1's complement
    - 2. Add 1 to the result
  - Example
    - Find the 2's complement of 01100011_2
  - Usually used to represent negative numbers, and to calculate the subtraction of binary numbers
  - We will discuss its application in Ch.6 Arithmetic

Numbering: ASCII

- **ASCII**
  - American Standard Code for Information Interchange
  - Use binary patterns to represent numbers and English alphabet
  - Standard ASCII code
    - Each character is represented by 7-bit
    - Totally there are 128 characters in the 7-bit ASCII table
  - Extended ASCII code
    - Each character is represented by 8-bit
    - Totally there are 256 characters in the 8-bit ASCII table
  - The complete ASCII table can be found at Appendix F of Mazidi's book

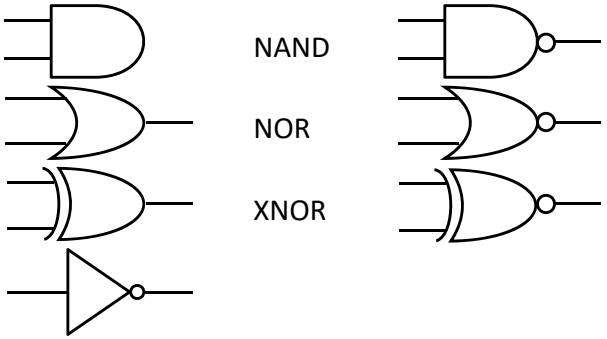
Hex	*Symbol*	*Hex*	*Symbol*
41	A	61	a
42	B	62	b
43	C	63	c
44	D	64	d
...	...	...	...
59	Y	79	y
5A	Z	7A	z

Outline

- What is Microcontroller? (Ch. 1.1)
- 8051 Microcontroller (Ch. 1.2)
- Numbering and coding: binary, decimal, hexa-decimal, ASCII (Ch. 0.1)
- **Basic digital logics (Ch. 0.2)**
- Operations of a computer (Ch. 0.3)

Logic

- **Binary logic**
  - Use two voltages to represent 0 and 1
    - E.g. 0V → '0', 5V → '1'
- **Logic gates**
  - AND
  - OR
  - XOR
  - NOT
  - NAND
  - NOR
  - XNOR



Logic

- **Logic gates**

Summary of 2-input logic							
Inputs		Outputs of each gate					
A	B	AND	NAND	OR	NOR	EX-OR	EX-NOR
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

Outline

- What is Microcontroller? (Ch. 1.1)
- 8051 Microcontroller (Ch. 1.2)
- Numbering and coding:
  binary, decimal, hexa-decimal, ASCII (Ch. 0.1)
- Basic digital logics (Ch. 0.2)
- **Operations of a computer (Ch. 0.3)**

Computer

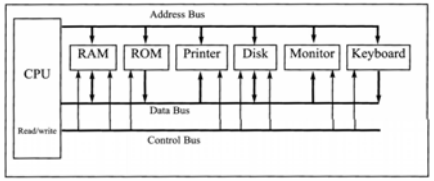
- **Terminology**
  - Bit(b): 1 bit — 0 or 1
  - Nibble: 4 bits — 0000, 0001, 0010, …, 1111
  - Byte: 8 bits — 0000 0000, 0000 0001, …, 1111 1111
  - Word: 16 bits — 0000 0000 0000 0000, …, 1111 1111 1111 1111
- **Prefix**
  - Kilo-: $2^{10} = 1{,}024 \approx 10^3$
    - E.g. 1 Kilobyte: 1KB = 1,024 bytes = 1,024 x 8 bits, 1 Kilobit: 1Kb = 1,024 bits
  - Mega-: $2^{20} = 1{,}024 \times 1{,}024 = 1{,}048{,}576 \approx 10^6$
  - Giga-: $2^{30} = 1{,}024 \times 1{,}024 \times 1{,}024 \approx 10^9$
  - Tera-, Peta-, Exa-, …

Computer: Structure

- **Structure**
  - CPU: Process information in memory
  - Memory
    - RAM(Random Access Memory):
      temporary storage of programs that it is running
      - The data is lost if computer is turned off (Volatile memory)
    - ROM(Read-Only Memory):
      contains programs and information essential for computer operation
      - E.g. When a computer is powered on, it will first execute a program stored in ROM to perform initialization before loading the operating system
      - Permanent and usually cannot be changed by the user (Non-volatile memory)
  - Peripherals
    - Serial ports, Parallel port, Keyboard, Monitor, …

Computer: Structure

- **Bus**
  - Strip of wires used to connect CPU with memory and peripherals
  - Data bus
    - The data lines used to carry information in and out of CPU
    - The more data lines, the better the CPU
      - Analogy: highway with more lanes
    - Typical values: 8-bit, 16-bit, 32-bit, 64-bit
      - A 32-bit bus can send out 4-byte of data at one time
    - Bidirectional
      - Data can get in or out of CPU



Computer: Structure

- **Bus (Cont'd)**
  - Address bus
    - Many devices are connected to a single data bus, how does the CPU know which device the data is from or to? → Address bus!
    - Address bus is used to identify device and memory connected to CPU
      - Each byte in memory has its unique address
    - If there are *n* address lines, then the total address range is 2^n bytes
      - E.g. 8-bit address line: $2^8 = 256$ bytes address range
      - E.g. 32-bit address line: $2^{32} = 4GB$ address range
        - » If the system has a total of 32-bit address lines, the maximum supported memory is 4GB

Computer: Structure

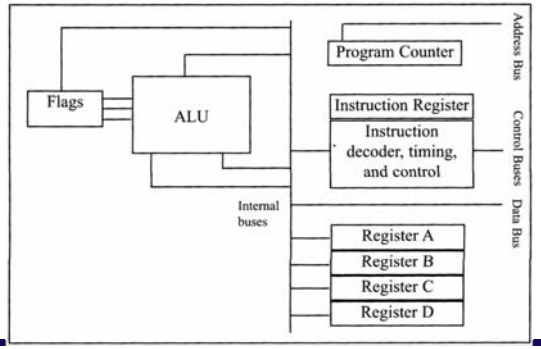
- **Bus (Cont'd)**
  - Address bus
    - Each device is assigned a range of addresses
      - E.g. 8-bit address line with 64-byte RAM, 32-byte ROM, 16 I/O ports
        - » RAM: address 0 – 63
        - » ROM: address 64 – 95
        - » I/O ports: address 96 – 112
        - » Printer: 113 – 114
        - » …
    - Unidirectional
      - Its value can only be changed by CPU

Computer: Structure

- **Bus (Cont'd)**
  - Control bus
    - CPU sends control information to devices to control their operations
    - E.g. CPU sends read or write control information to the devices to indicate it wants to read from the device, or write data to the device
  - The operation of a computer relies on the combination of the three different buses
    - E.g. CPU wants to read a data byte from memory location 32
      - 1. CPU set the value of the address bus to 32
      - 2. CPU use the control bus to put the memory in read mode
      - 3. CPU read the data byte on the data bus

Computer: CPU

- **Inside CPU**
  - ALU (Arithmetic Logic Unit)
    - Arithmetic functions (add, subtract, …)
    - Logic functions (and, or, not, …)



Computer: CPU

- **CPU**
  - Registers
    - Temporarily store information
    - Data read from memory or device will first be stored in registers, then CPU will process data in register
    - Calculation results will be store in register, then send out to memory
    - E.g. 3 + 5
      - 1. 3 will be first loaded to register A, 5 will be first loaded to register B
      - 2. CPU calculates 3+5, the result 8, will be stored in register A
    - Typical size: 8-bit, 16-bit, 32-bit (most popular nowadays), 64-bit

Computer: CPU

- **CPU (Cont'd)**
  - Instruction register, Instruction decoder, Program Counter
    - Instruction:
      a special binary pattern corresponds to a certain operation by CPU
      - E.g. 1011 0000 (B0H): move data to register A
      - E.g. 0000 0100 (04H): add a value to register A
    - Program is a sequence of instructions, and it is stored in memory
    - CPU reads the program from the memory, one instruction at a time, and the current instruction is temporarily stored in instruction register
    - The instruction decoder interprets the meaning of the instruction, so CPU can execute according to the instruction
    - Program counter:
      points to the memory address of the next instruction to be executed

Computer: CPU

- **Example**
  - A program stored in the memory address range 1400 – 1406

Address	Contents of memory
1400	(B0) code for moving a value to register A
1401	(21) value to be moved
1402	(04) code for adding a value to register A
1403	(42) value to be added
1404	(04) code for adding a value to register A
1405	(12) value to be added
1406	(F4) code for halt

PC					
IR					
RA					

  - PC: Program Counter, IR: Instruction Register, RA: Register A