Fundamentals of Microprocessor and Microcontroller

Dr. Farid Farahmand

Updated: 2/23/19

First Let's Review a Few BASIC Things…..

Unsigned

Signed

Data Format (8-bit) (1 of 4)

- Unsigned Integers: All eight bits (Bit0 to Bit7) represent the magnitude of a number
  - Range 0 to FF in Hex and 0 to 255 in decimal

Unsigned

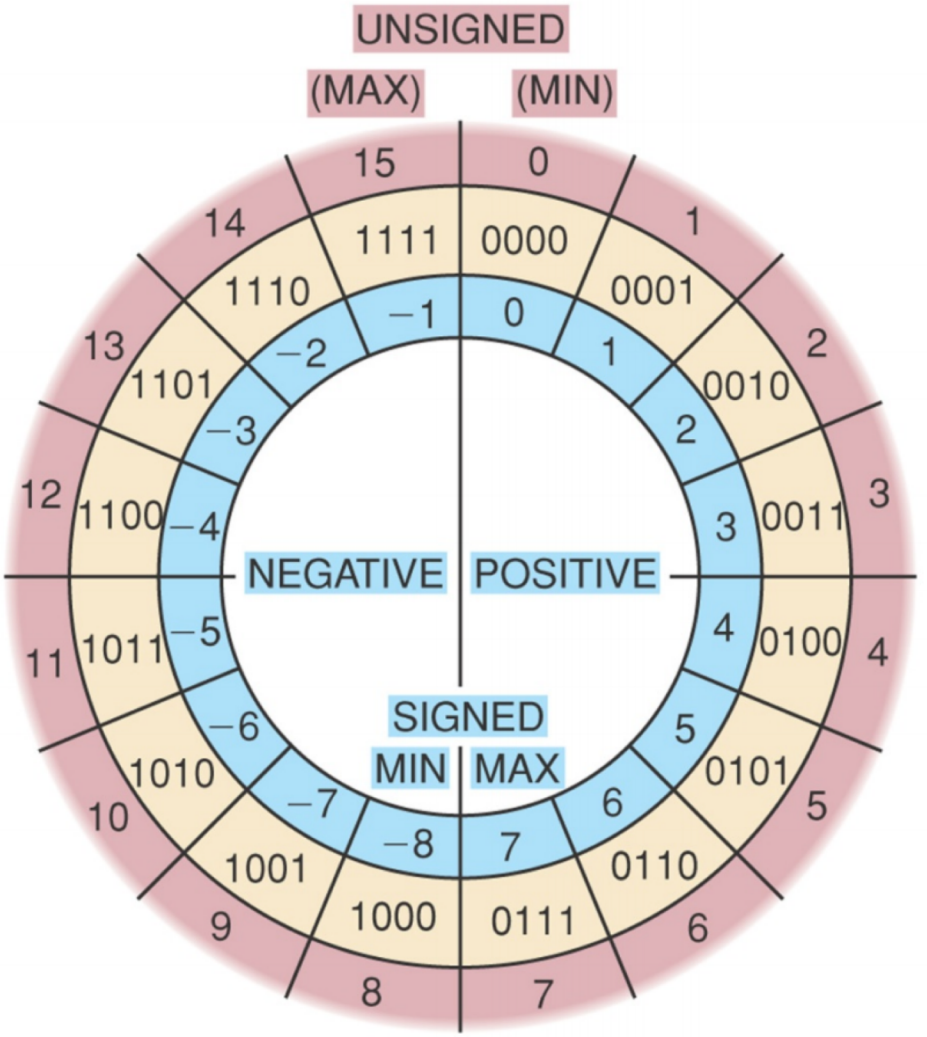
Signed

Data Format (8-bit) (2 of 4)

- Signed Integers: Seven bits (Bit0 to Bit6) represent the magnitude of a number.
  - The 8th bit (Bit7) represents the sign of a number. The number is positive when Bit7 is zero and negative when Bit7 is one.
  - Positive numbers: 0 to 7F (0 to 127)
  - Negative numbers: 80 to FF (-1 to -128)
  - All negative numbers are represented in 2's complement

Representing NEG values using 2's complement

Integer Signed	2's Complement
7	0111
6	0110
5	0101
4	0100
3	0011
2	0010
1	0001
0	0000
-1	1111
-2	1110
-3	1101
-4	1100
-5	1011
-6	1010
-7	1001
-8	1000



Data Format (8-bit) (3 of 4)

- Binary Coded Decimal Numbers (BCD)
  - 8 bits of a number divided into groups of four, and each group represents a decimal digit from 0 to 9
  - Four-bit combinations from A through F in Hex are invalid in BCD numbers
    - Example: 0010 0101 represents the binary coding of the decimal number 25d which is different in value from 25H.

Data Format (8-bit) (4 of 4)

- American Standard Code for Information Interchange (ASCII)
  - Seven-bit alphanumeric code with 128 combinations (00 to 7F)
  - Represents English alphabet, decimal digits from 0 to 9, symbols, and commands

See this for more information:
http://web.sonoma.edu/users/f/farahman/sonoma/courses/es310/resources/Digital%20Arithmetic.pdf

Evolution of Integrated Devices

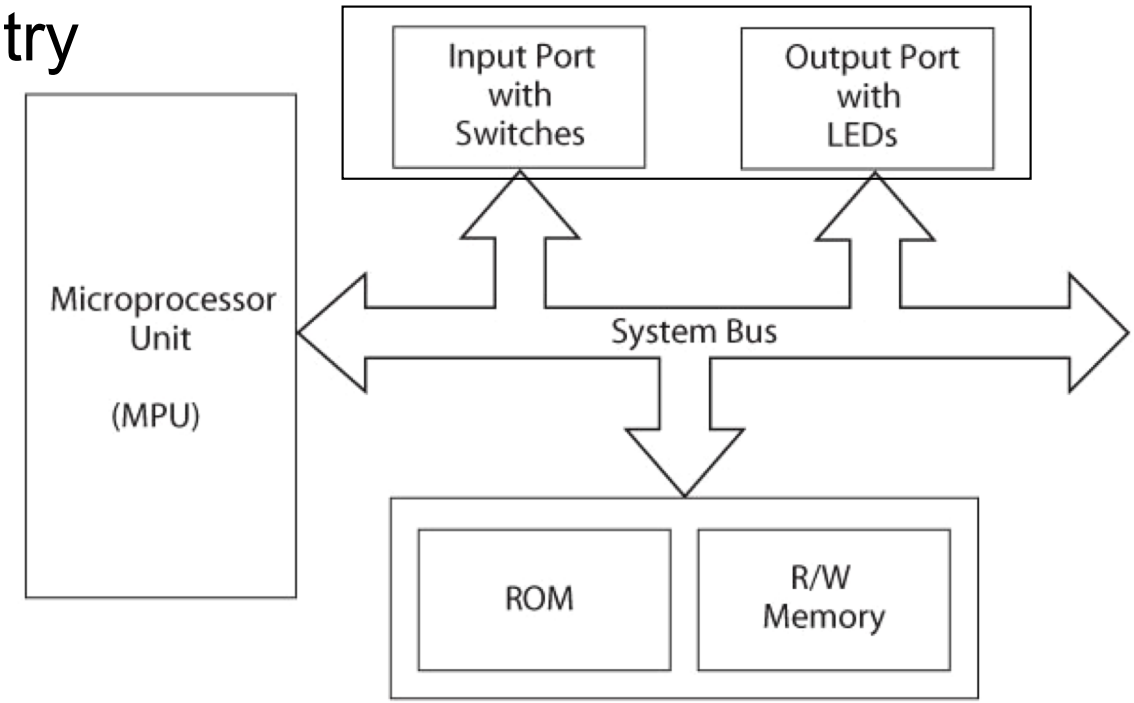
- First came transistors
- Integrated circuits
  - SSI (Small-Scale Integration) to ULSI
  - Very Large Scale Integration circuits (VLSI)
- 1- Microprocessors (MPU)
  - Microcomputers (with CPU being a microprocessor)
  - Components: Memory, CPU, Peripherals (I/O)
  - Example: Personal computers
- 2- Microcontroller (MCU)
  - Microcomputers (with CPU being a microprocessor)
  - Many special function peripheral are integrated on a single circuit
  - Types: General Purpose or Embedded System (with special functionalities)

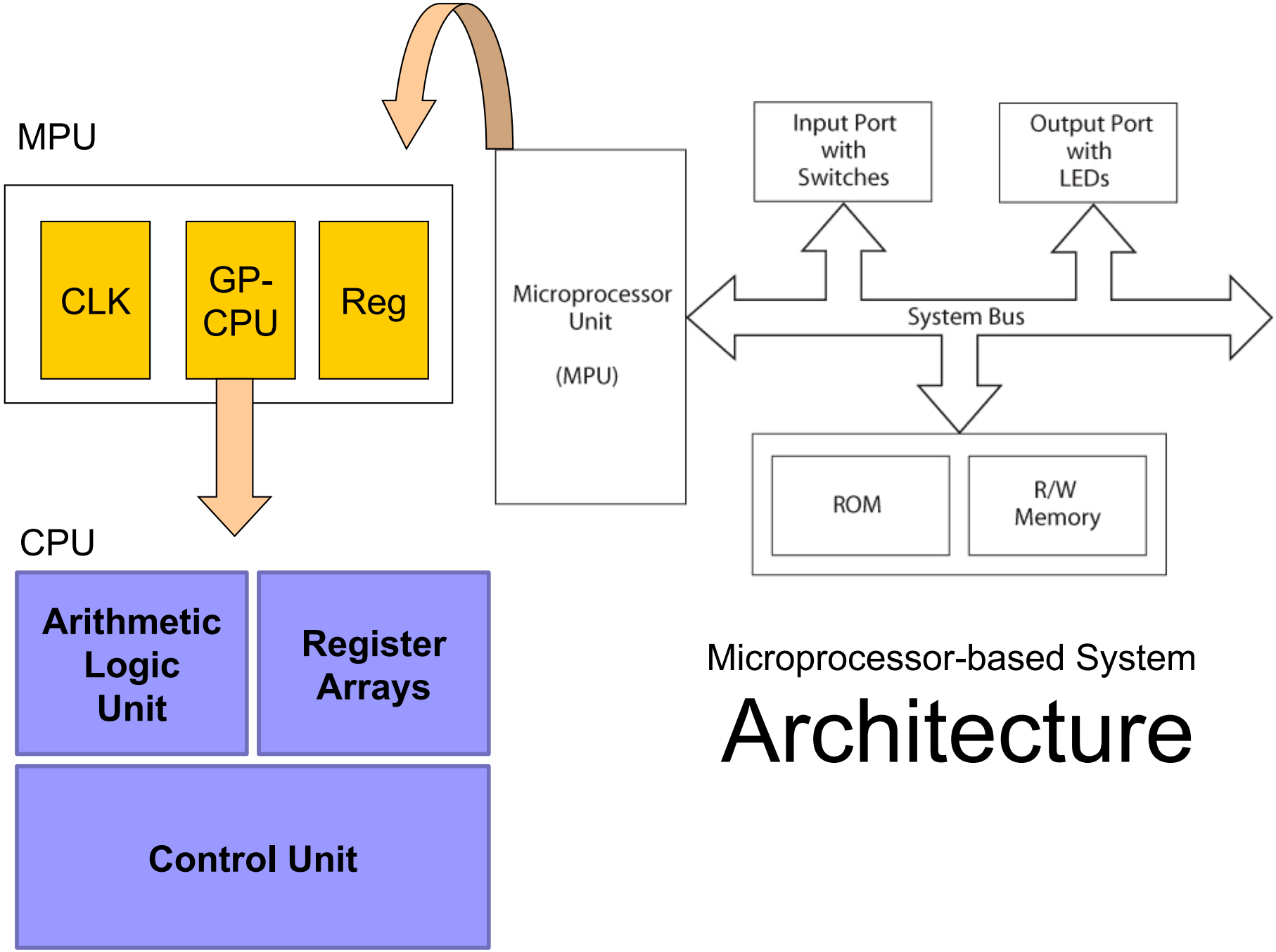
Microcontrollers – Embedded Systems and Integrated Devices

- An embedded system is a special-purpose computer system designed to perform one or more dedicated functions often in real-time
- Embedded systems often have dedicated system software
  - System software: A group of programs that monitors the functions of the entire system
- Embedded systems are generally microprocessor-based

Microprocessor-Based Embedded Systems

Microprocessor
- Memory
- Input/Output (I/O) circuitry
- Buses
  - Address bus
  - Data bus
  - Control bus

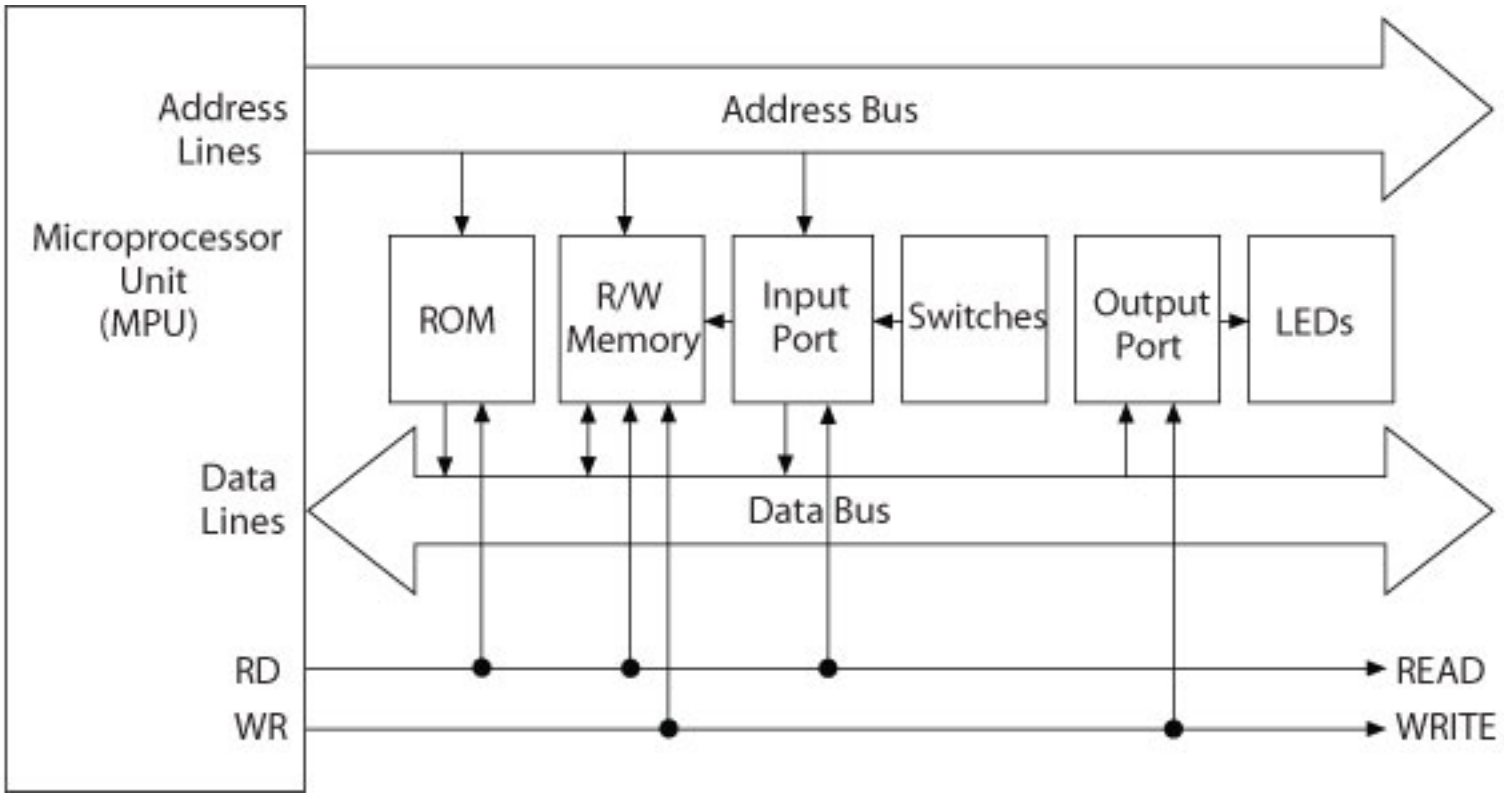




Microprocessor-based System

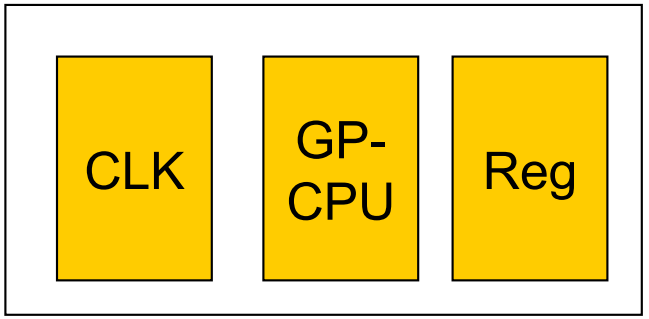
Architecture

Microprocessor-Based System with Buses: Address, Data, and Control



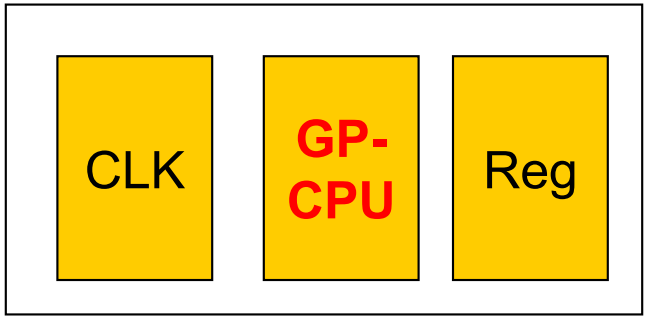
Microprocessor-based Systems:
Microprocessor (1)

- The microprocessor (MPU) is a computing and logic device that executes binary instructions in a sequence stored in memory.
- Characteristics:
  - General purpose central processor unit (CPU)
  - Binary-based
  - Register-based
  - Clock-driven
  - Programmable

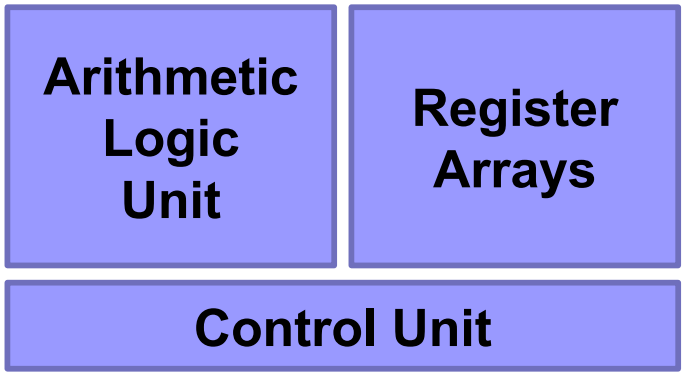


Microprocessor-based Systems

GP CPU



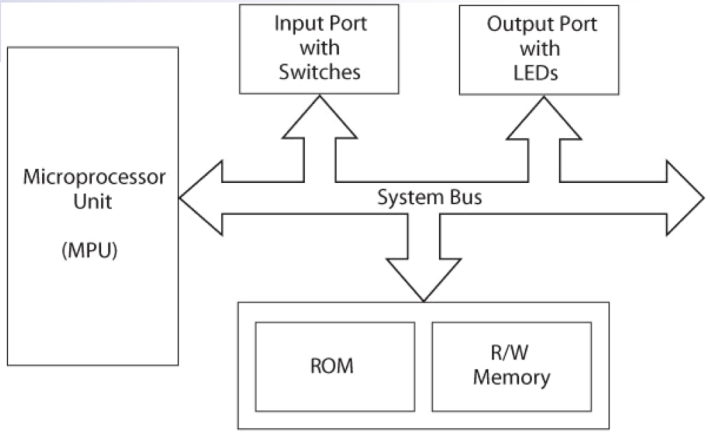
- the "brain" of the computer
  - **its job is to fetch instructions, decode them, and then execute them**
  - **8/16/32/etc –bit (how it moves the data**
- contains:



ALU performs computing tasks – manipulates the data/ performs numerical and logical computations
Registers are used for temp. storage
Control unit is used for timing and other controlling functions – contains a program counter (next instruction's address and status register)

System software: A group of programs that monitors the functions of the entire system

Microprocessor-based Systems - Memory



- Memory is a group of registers
  - 16 register – address: 0-15 – in binary: 0-1111; Address lines: A0-A3
- Serves two major purposes
  - storing the binary codes for the sequence of instructions specified by programs (program)
  - storing binary data that the computer needs to execute instructions (data)

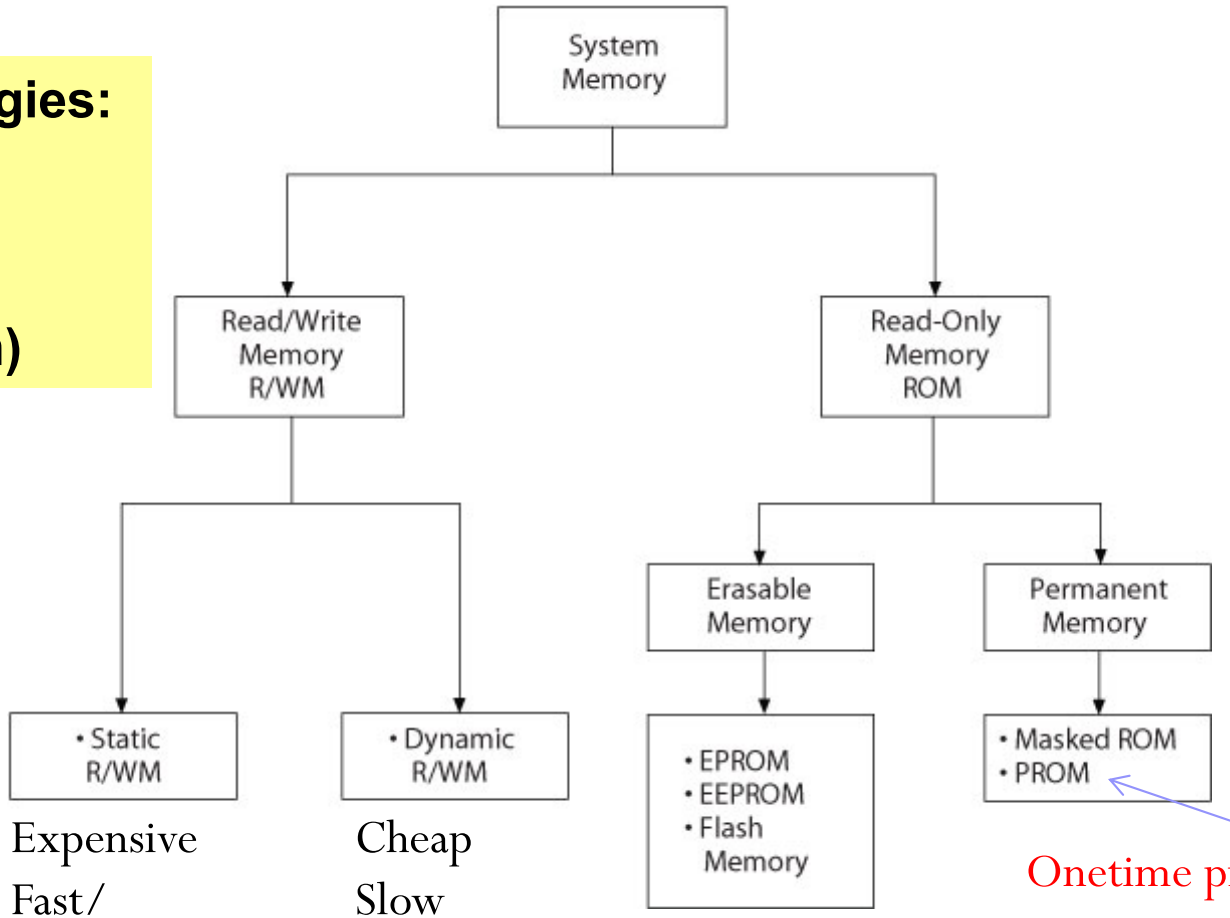
Microprocessor-based Systems

Memory Types

- **R/W or RAM: Read/Write Memory**
  - It is volatile (losses information as power is removed)
  - Write means the processor can store information
  - Read means the processor can receive information from the memory
  - Acts like a Blackboard!
- **ROM: Read-Only memory;**
  - It is typically non-volatile (permanent) – can be erasable
  - It is similar to a Page from your textbook

Microprocessor-based Systems
Memory Classification

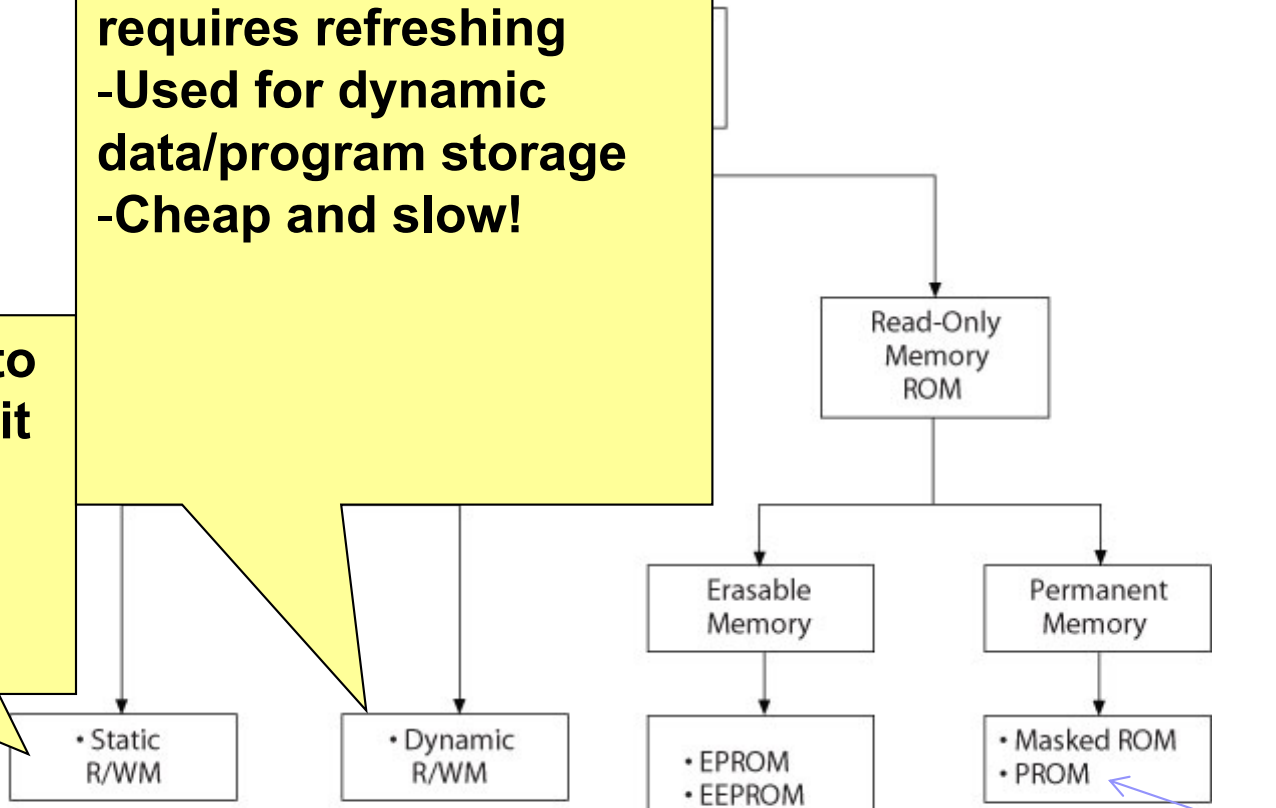
Basic Technologies:
Semiconductor
Magnetic
Optical
(or combination)



Microprocessor-based Systems
Memory C

-one transistor and one capacitor to store a bit
-Leakage problem, thus requires refreshing
-Used for dynamic data/program storage
-Cheap and slow!

-4/6 transistor to save a single bit
- Volatile
- Fast but expensive



Expensive
Fast/

Cheap
Slow

Electronically Erasable PROM

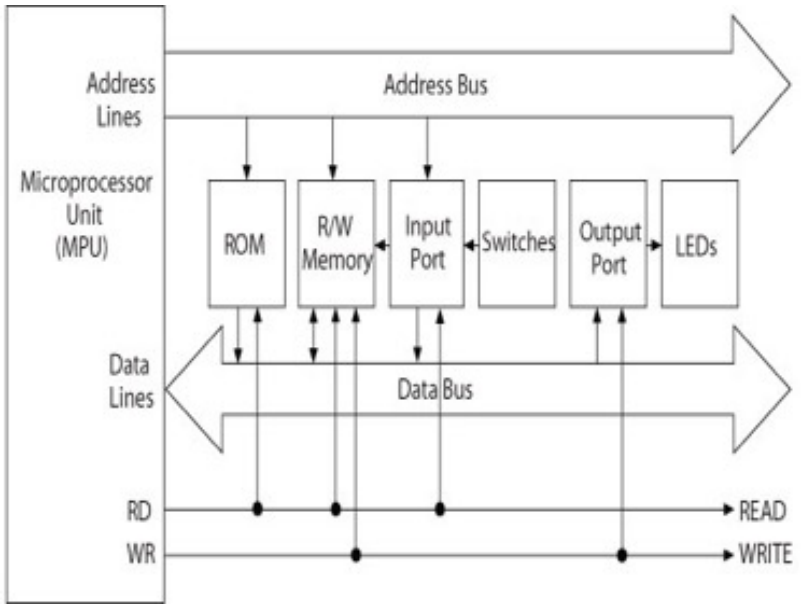
Onetime programmable

Erasable ROMs

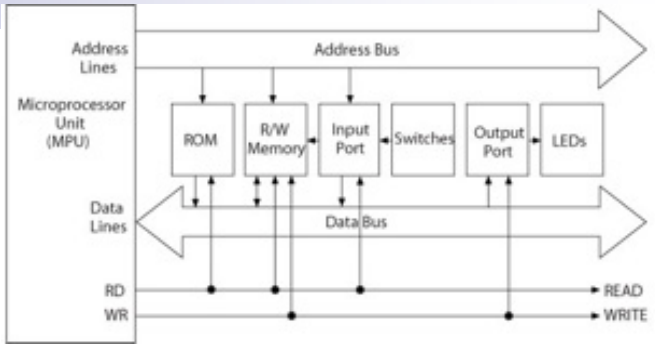
- Marked Programmed ROM
  - Programmed by the manufacturer
- Programmable ROM (PROM)
  - Can be programmed in the field via the programmer
- Erasable Programmable ROM (EPROM)
  - Uses ultraviolet light to erase (through a quartz window)
  - OTP refers to one-time programmable
- Electrically Erasable Programmable ROM (EEPROM)
  - Each program location can be individually erased
  - Expensive
  - Requires programmer
- FLASH
  - Can be programmed in-circuit (in-system)
  - Easy to erase (no programmer)
  - Only one section can be erased/written at a time (typically 64 bytes at a time)

Microprocessor-based Systems
I/O Ports

- The way the computer communicates with the outside world devices
- I/O ports are connected to Peripherals
  - Peripherals are I/O devices
    - **Input devices**
    - **Output devices**
  - Examples
    - **Printers and modems,**
    - **keyboard and mouse**
    - **scanner**
    - **Universal Serial Bus (USB)**



Microprocessor-based Systems - BUS



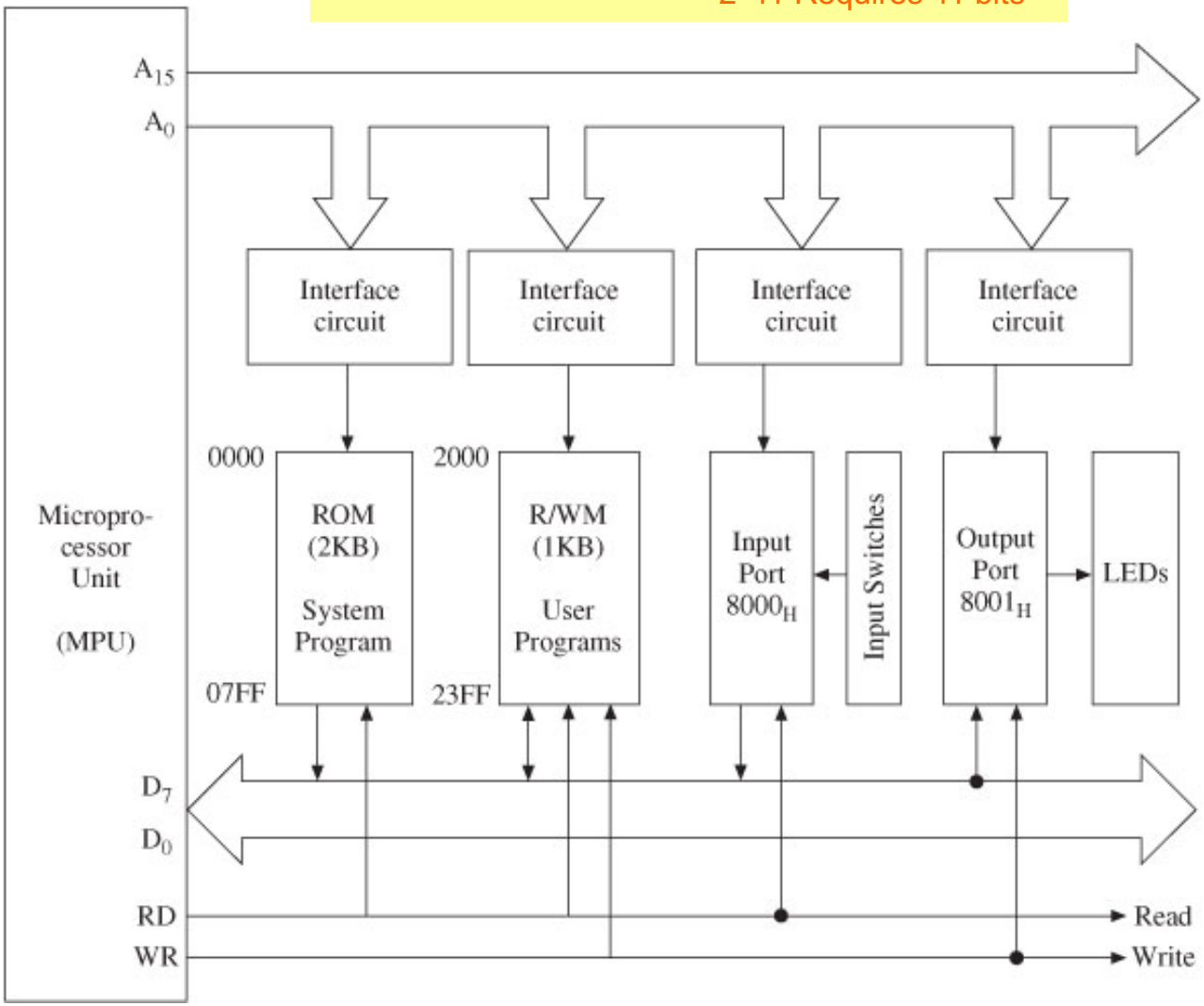
- **The three components – MPU, memory, and I/O – are connected by a group of wires called the BUS**

- **Address bus**
  - **consists of 16, 20, 24, or 32 parallel signal lines (wires) - unidirectional**
  - **these lines contain the address of the memory location to read or written**
- **Control bus**
  - **consists of 4 to 10 (or more) parallel signal lines**
  - **CPU sends signals along these lines to memory and to I/O ports**
    - **examples: Memory Read, Memory Write, I/O Read, I/O Write**
- **Data bus**
  - **consists of 8,16, or 32 parallel signal lines**
  - **bi-directional**
  - **only one device at a time can have its outputs enabled,**
  - **this requires the devices to have three-state output**

Microprocessor-Based System - Expanded System

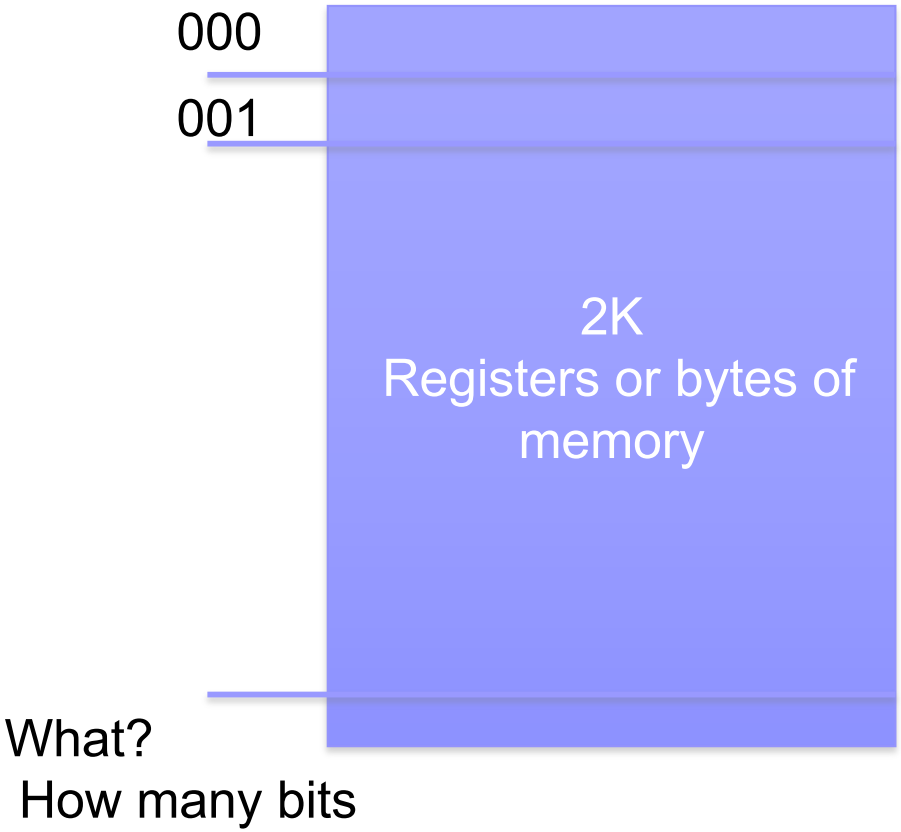
Remember: 111 1111 1111 = 7FF = 2^11-1 = 2047
2^11=2K=2048
2^11 Requires 11 bits

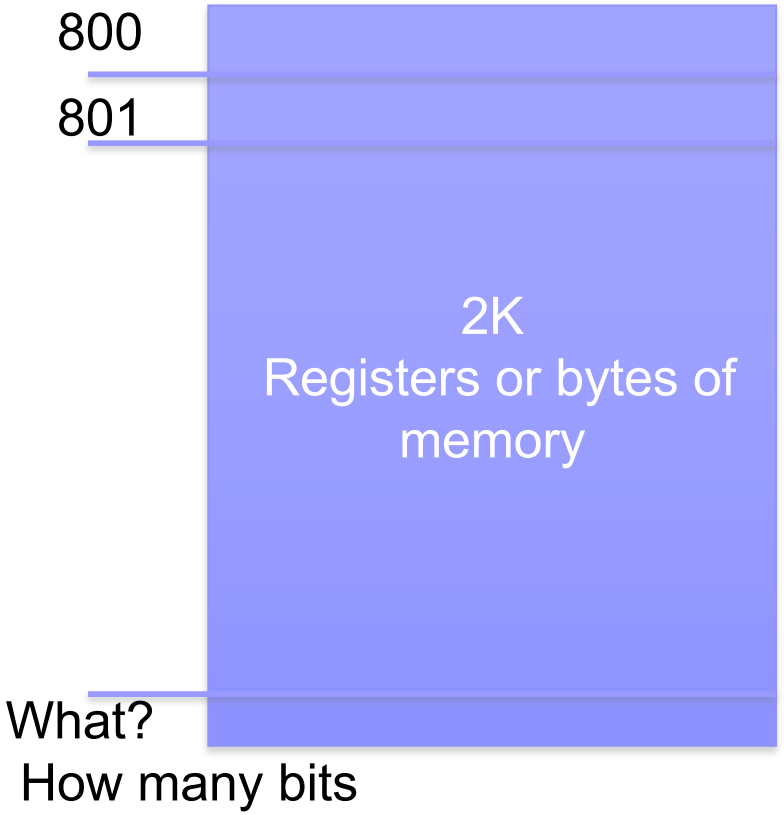
1. **Note the directions of busses**
2. **What is the width of the address bus?**
3. **What is the value of the Address but to access the first register of the R/WM?**



You must know how to draw it!

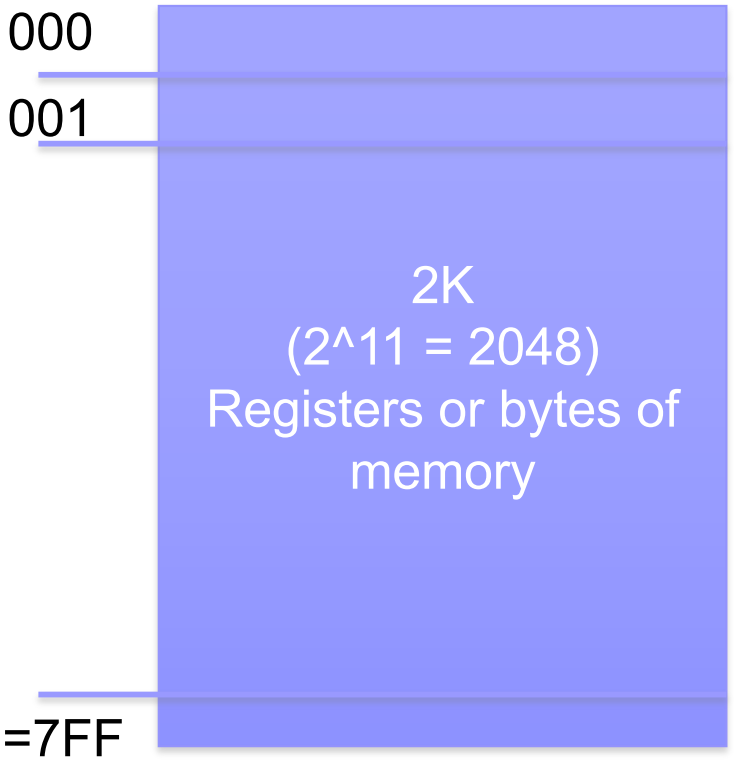
Example



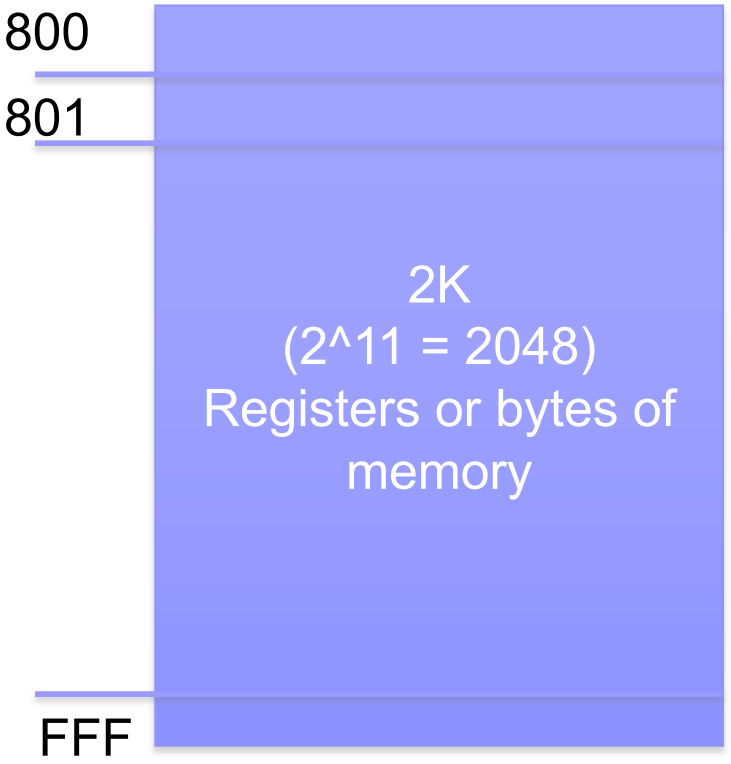


How much memory do we have?

Example



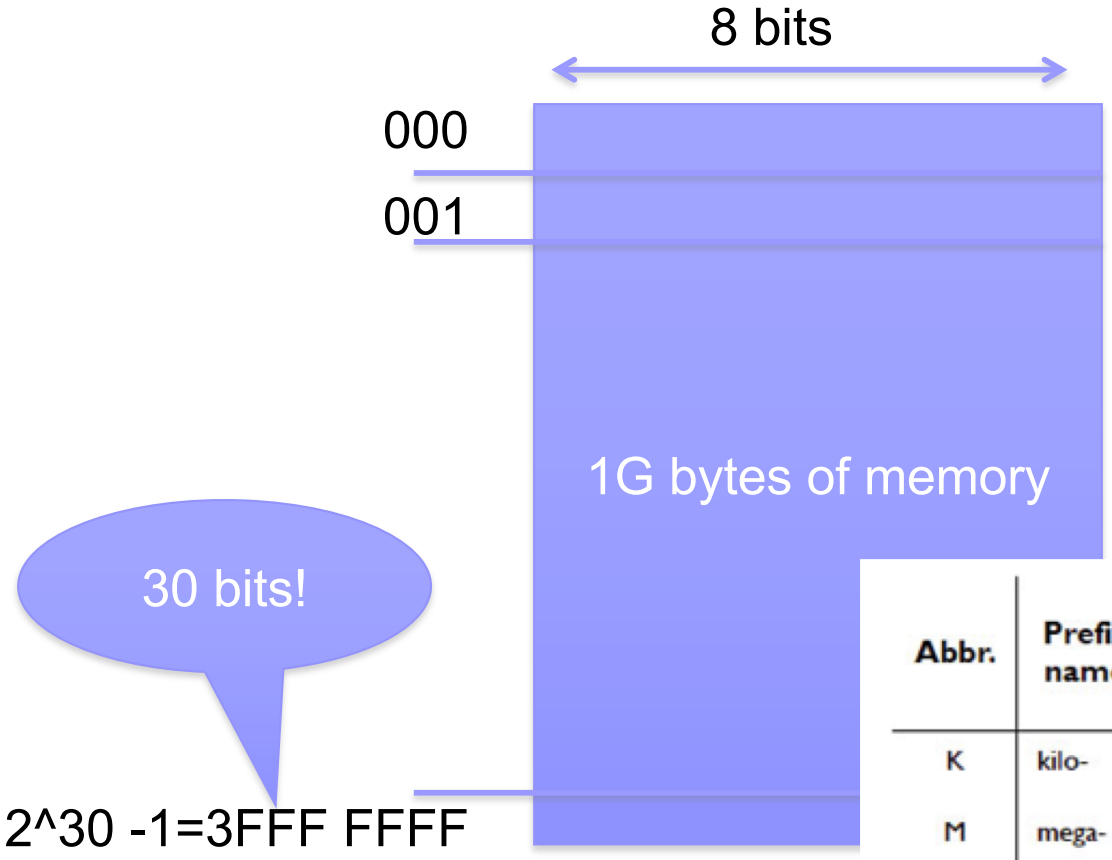
2^11 -1=7FF
Requires 11 bits



Requires 12 bits
For a total of 4095 bytes

Total of 4K bytes of memory: 2^12 (FFF) → 12 bits ; last values 2^12-1 = 4096-1

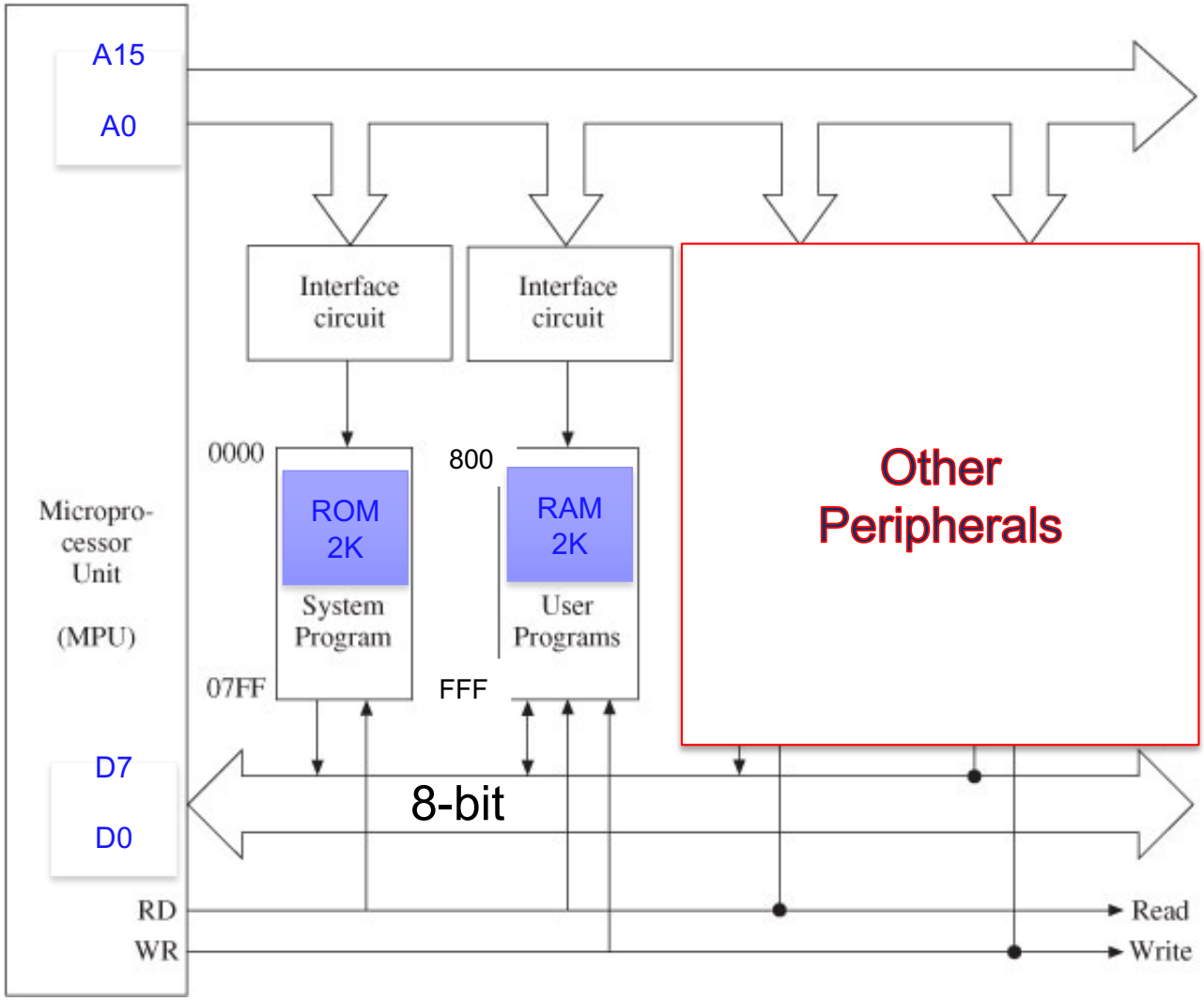
Example



Next number: 4000 0000 (in Hex)

Abbr.	Prefix name	Decimal size	Size in thousands	Binary approximation	Address variable size
K	kilo-	10^3	$1{,}000$	$1{,}024 = 2^{10}$	10
M	mega-	10^6	$1{,}000^2$	$1{,}024^2 = 2^{20}$	20
G	giga-	10^9	$1{,}000^3$	$1{,}024^3 = 2^{30}$	30
T	tera-	10^{12}	$1{,}000^4$	$1{,}024^4 = 2^{40}$	40
P	peta-	10^{15}	$1{,}000^5$	$1{,}024^5 = 2^{50}$	50
E	exa-	10^{18}	$1{,}000^6$	$1{,}024^6 = 2^{60}$	60

Example of an 8-bit MPU



So what are microcontrollers?

What is a Microcontroller?

- A microcontroller is a small computer on a single integrated circuit containing
  - processor core,
  - memory,
  - programmable input/output peripherals
- Used for specific (embedded) applications

Embedded controllers

- Used to control smart machines
- Examples: printers, auto braking systems
- Also called microcontrollers or microcontroller units (MCU)

Embedded controllers

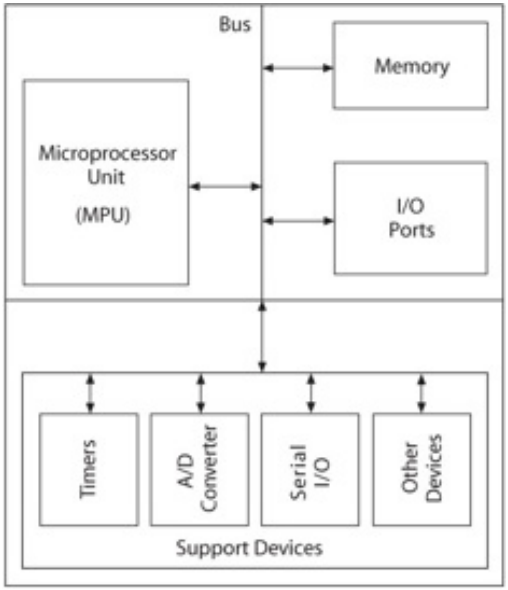
Software Characteristics

- No operating systems
- Execute a single program, tailored exactly to the controller hardware
- Assembly language (vs. High-level language)
  - Not transportable, machine specific
  - Programmer need to know CPU architecture
  - Speed
  - Program size
  - Uniqueness

Microcontroller Unit (MCU)

Block Diagram

- An integrated electronic computing and logic device that includes three major components on a single chip (System-on-Chip (SOC))
  - Microprocessor
  - Memory
  - I/O ports
- Includes support devices
  - Timers
  - A/D converter
  - Serial I/O
  - Parallel Slave Port
- All components connected by common communication lines called the system bus.



First Microcontrollers

- IBM started using Intel processors in its PC
  - Intel started its 8042 and 8048 (8-bit microcontroller) – using in printers
- Apple Macintosh used Motorola
- 1980 Intel abandoned microcontroller business
- By 1989 Microchip was a major player in designing microcontrollers
  - PIC: Peripheral Interface Controller

Different Microcontrollers (MCU)

What is the difference?
8/16/24/32 bits
Architecture
Package
Capability
Memory
Software (IDE)/cloud
ADC (10-12 bit)

- ARM core processors (from many vendors)
- Atmel AVR (8-bit), AVR32 (32-bit), and AT91SAM (32-bit)
- Cypress Semiconductor PSoC (Programmable System-on-Chip)
- Freescale ColdFire (32-bit) and S08 (8-bit)
- Freescale 68HC11 (8-bit)
- Intel 8051
- Infineon: 8, 16, 32 Bit microcontrollers[9]
- MIPS
- Microchip Technology PIC, (8-bit PIC16, PIC18, 16-bit dsPIC33 / PIC24), (32-bit PIC32)
- NXP Semiconductors LPC1000, LPC2000, LPC3000, LPC4000 (32-bit), LPC900, LPC700 (8-bit)
- Parallax Propeller
- PowerPC ISE
- Rabbit 2000 (8-bit)
- Renesas RX, V850, Hitachi H8, Hitachi SuperH (32-bit), M16C (16-bit), RL78, R8C, 78K0/78K0R (8-bit)
- Silicon Laboratories Pipelined 8051 Microcontrollers
- STMicroelectronics ST8 (8-bit), ST10 (16-bit) and STM32 (32-bit)
- Texas Instruments TI MSP430 (16-bit)
- Toshiba TLCS-870 (8-bit/16-bit).

http://en.wikipedia.org/wiki/Microcontroller A more complete list is here: http://en.wikipedia.org/wiki/Category:Microcontrollers

MCU Architecture

- RISC
  - Reduced instruction set computer
  - Simple operations
  - Simple addressing modes
  - Longer compiled program but faster to execute
  - Uses pipelining
  - Most embedded system
- CISC
  - Complex instruction set computer
  - More complex instructions (closer to high-level language support)
  - x86 standard (Intel, AMD, etc.), but even in the mainframe territory CISC is dominant via the IBM/390 chip

Bench marks: How to compare MCUs together
MIPS: Million Instructions / second (Useful when the compilers are the same)

CISC vs RISC

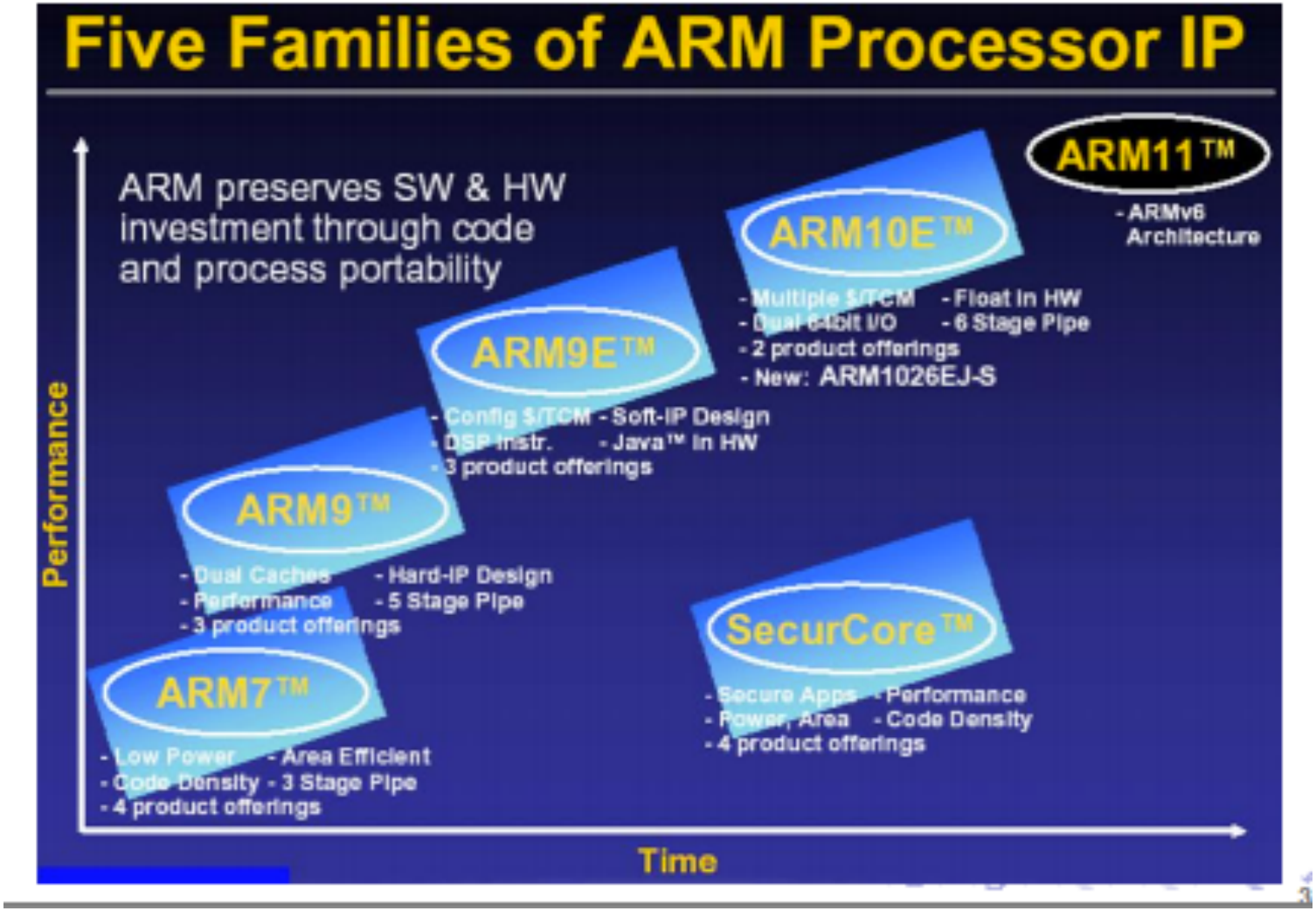
CISC **Pentium/x86 are CISC-based**	**RISC** **ARM-based** **Most mobile-phones**
Complex instructions require multiple cycles	Reduced instructions take 1 cycle
Many instructions can reference memory	Only Load and Store instructions can reference memory
Instructions are executed one at a time	Uses pipelining to execute instructions
Few general registers	Many general registers

RISC and CISC architectures are becoming more and more alike.
Read the LINK on the web site!

A Bit About ARMs Architecture (Advanced RISC Machine)

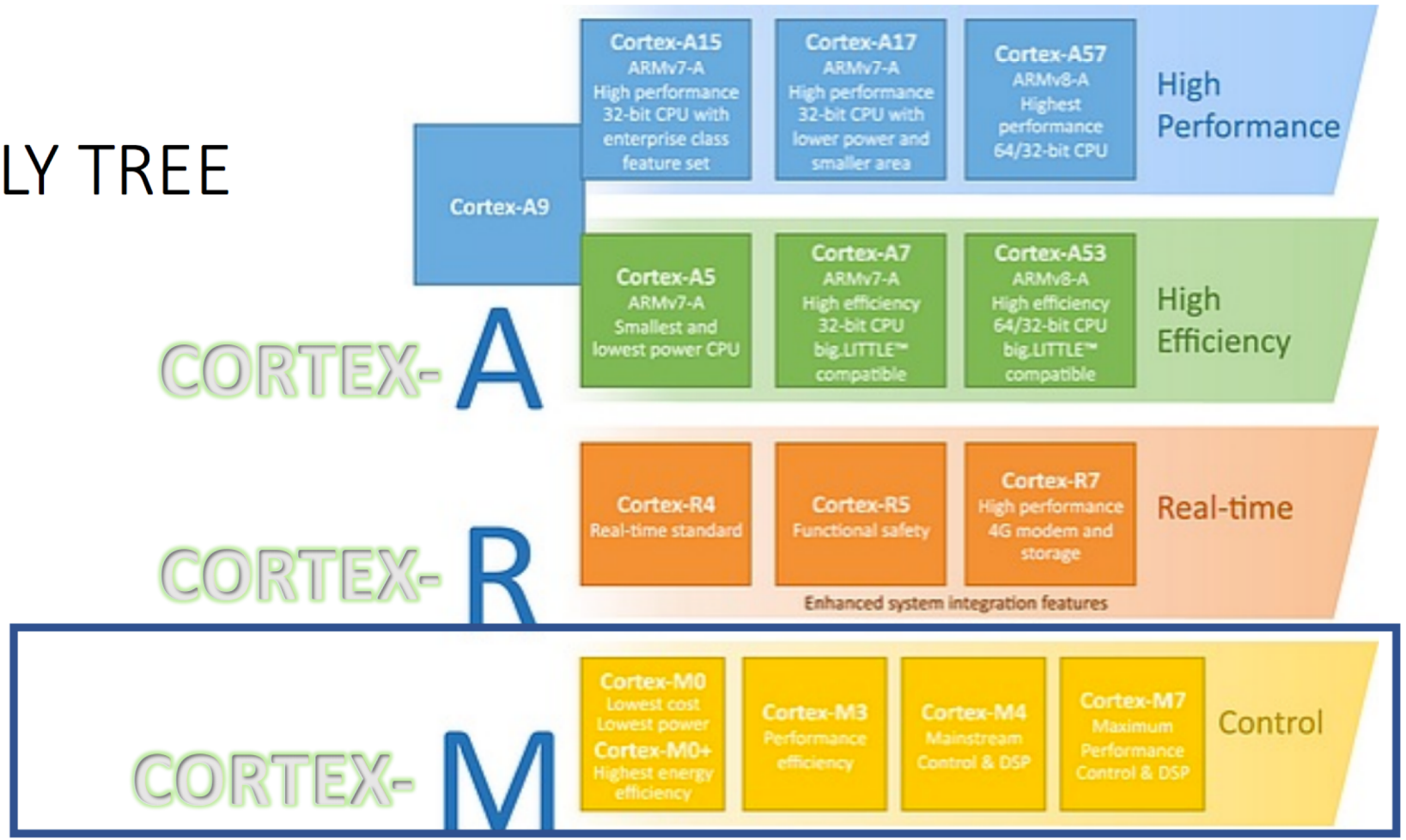
- ARM design takes the RISC based computer design approach – Linux –like architecture
- ARM is a British semiconductor (and software) design company that designs and licenses ARM processor cores to semiconductor manufacturers
  - They just sell the ARM *core*
  - Other manufacturers license the core from them and then design microcontrollers around that core by adding in peripherals and memory to suit their design goals
- There are different cores for different applications
  - Cortex-M0/M0+, Cortex-M3, or Cortex-M4.

ARM Processor IP



ARM Processor IP

ARM
FAMILY TREE

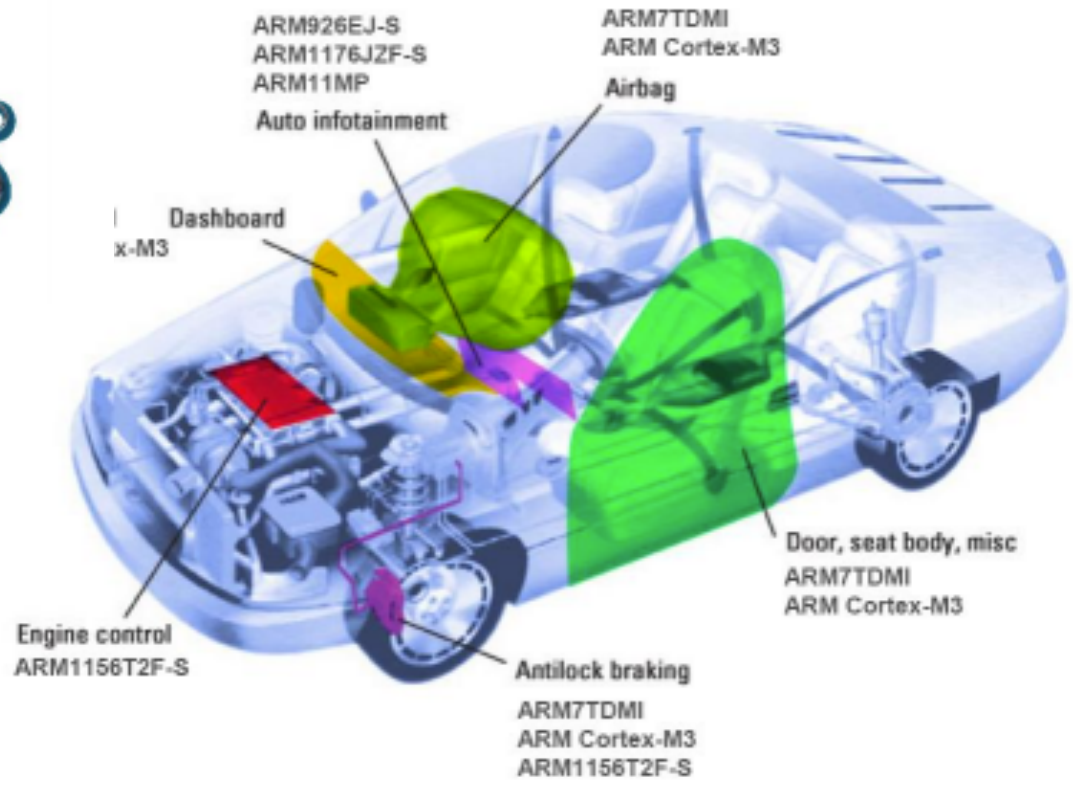


Applications of ARM-Based Microcontrollers

Most Cellphones!

Who is using ARM? Check this out!

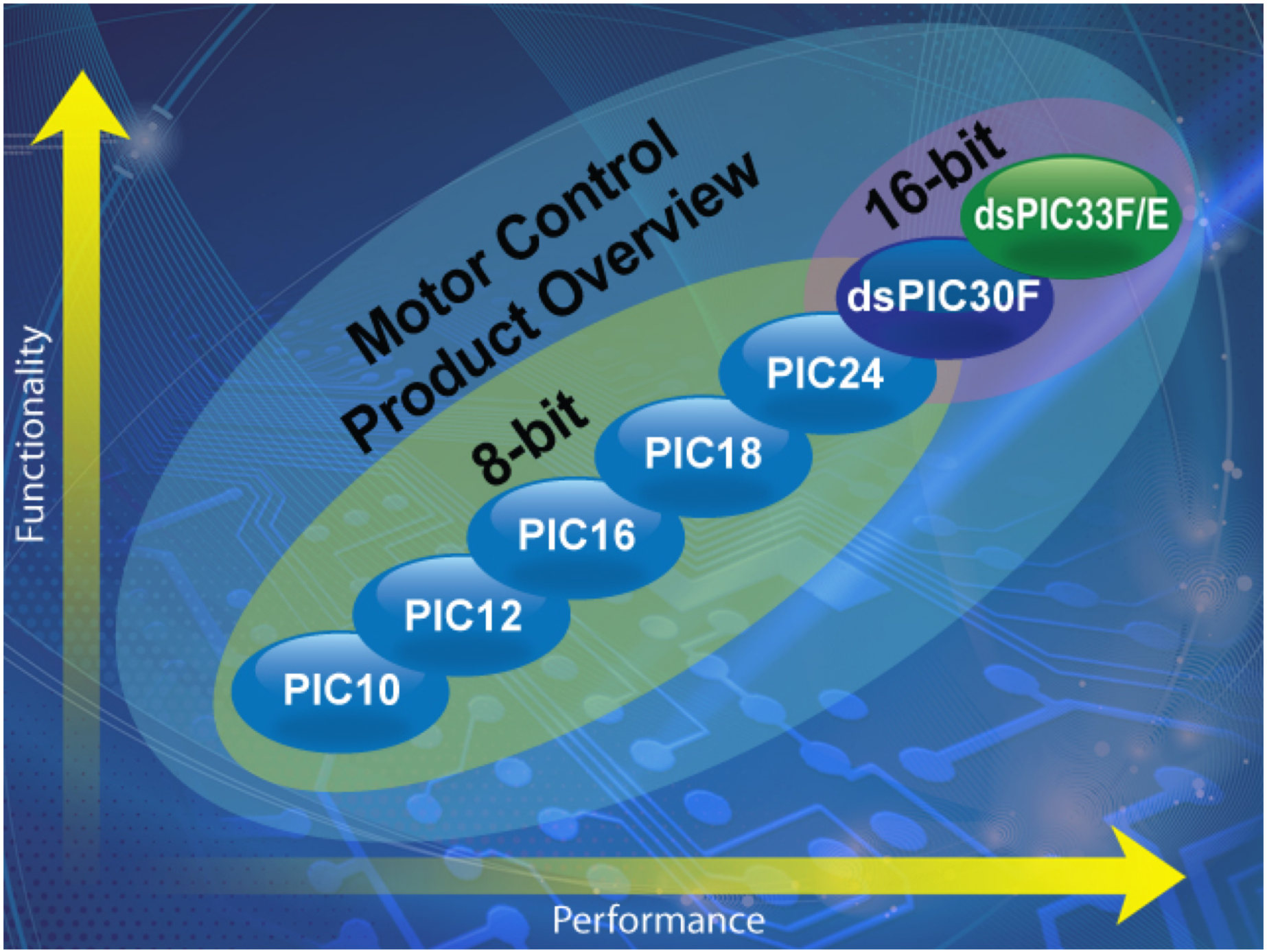
http://en.wikipedia.org/wiki/List_of_applications_of_ARM_cores



Back to Our 8-bit Controllers… (Main Players)

- Microchip
  - RISC architecture (reduced instruction set computer)
  - Has sold over 2 billion as of 2002
  - Cost effective and rich in peripherals
- Motorola
  - CISC architecture
  - Has hundreds of instructions
  - Examples: 68HC05, 68HC08, 68HC11
- Intel
  - CISC architecture
  - Has hundreds of instructions
  - Examples: 8051, 8052
  - Many difference manufacturers: Philips, Dallas/MAXIM Semiconductor, etc.
- Atmel
  - RISC architecture (reduced instruction set computer) – with CISC instruction set!
  - Cost effective and rich in peripherals
  - Claims to be very code efficient – less memory for the same code!
  - AVR (Advanced Virtual RISC): TunyAVR, MegAVR, XmegaAVR
- Freescale
- Ziglog (Z8)

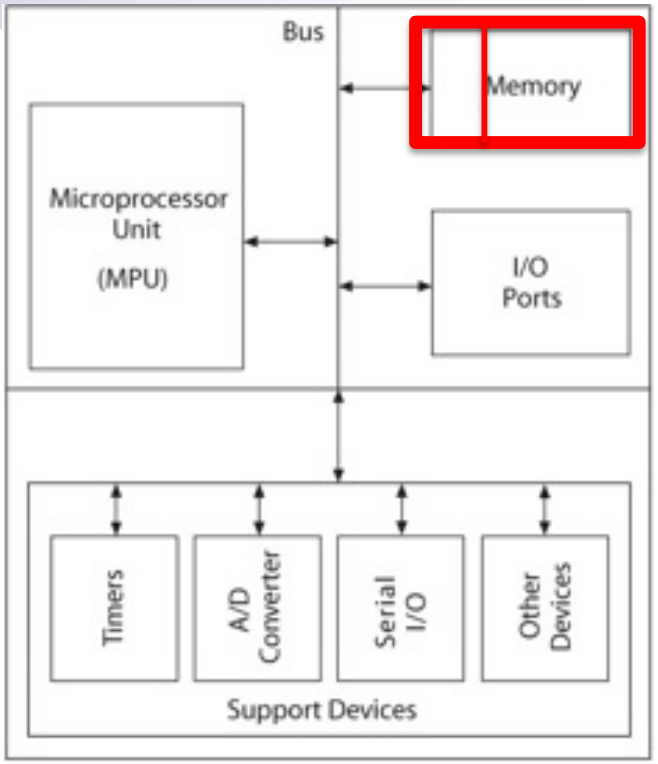
What is the difference?
Speed
Package
Power
RAM/ROM
IO Pins
Software (IDE)/cloud


Motor Control
Product Overview
16-bit
dsPIC33F/E
dsPIC30F
PIC24
8-bit
PIC18
PIC16
PIC12
PIC10
Functionality
Performance

What you Need to Use Microcontrollers

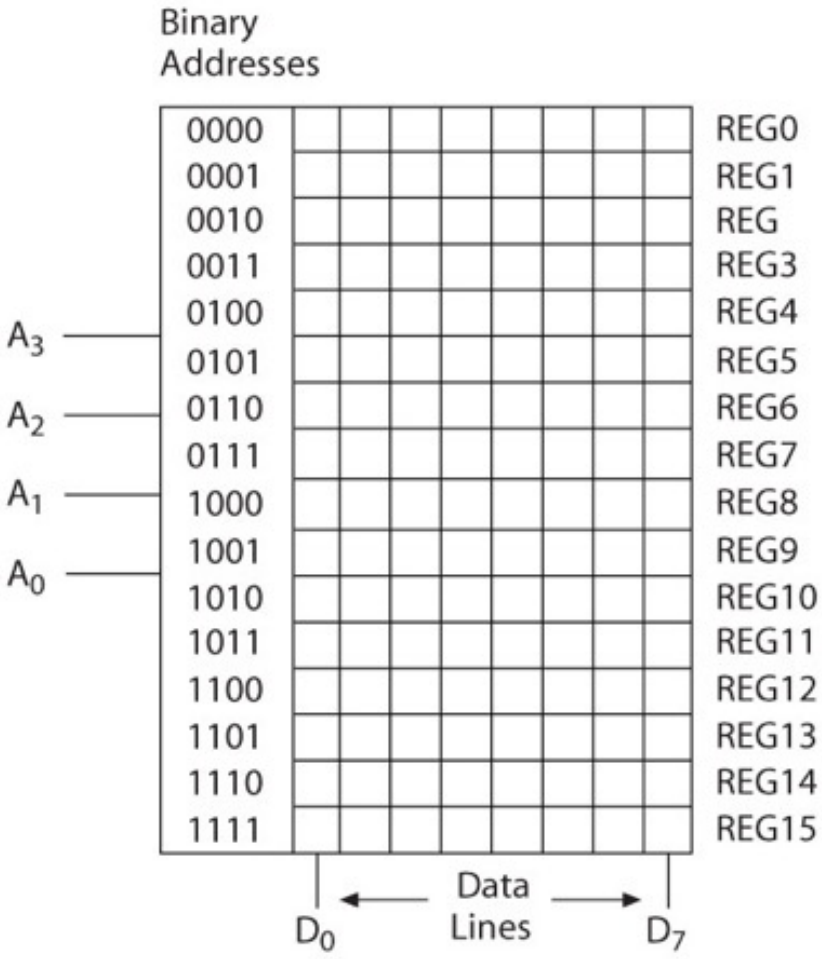
- A *target* - the actual microcontroller
- A *toolchain* — this is the software you use to write your code
  - Most developers use an *IDE* — integrated development environment — which contains a text editor, plus functionality for compiling and downloading your programs to the target
  - The toolchain can be locally installed or on **cloud!**
- A *Programmer/debugger* — this is the device that connects the computer to the microcontroller to download code to it
  - Your PICKIT3!
  - Allows real-time debugging of the program

Programming MCUs..



- Memory devices can store two types of information:
  - Data (RAM)
  - Programs (a series of instructions that tell the MPU in the microcontroller what to do!) - ROM

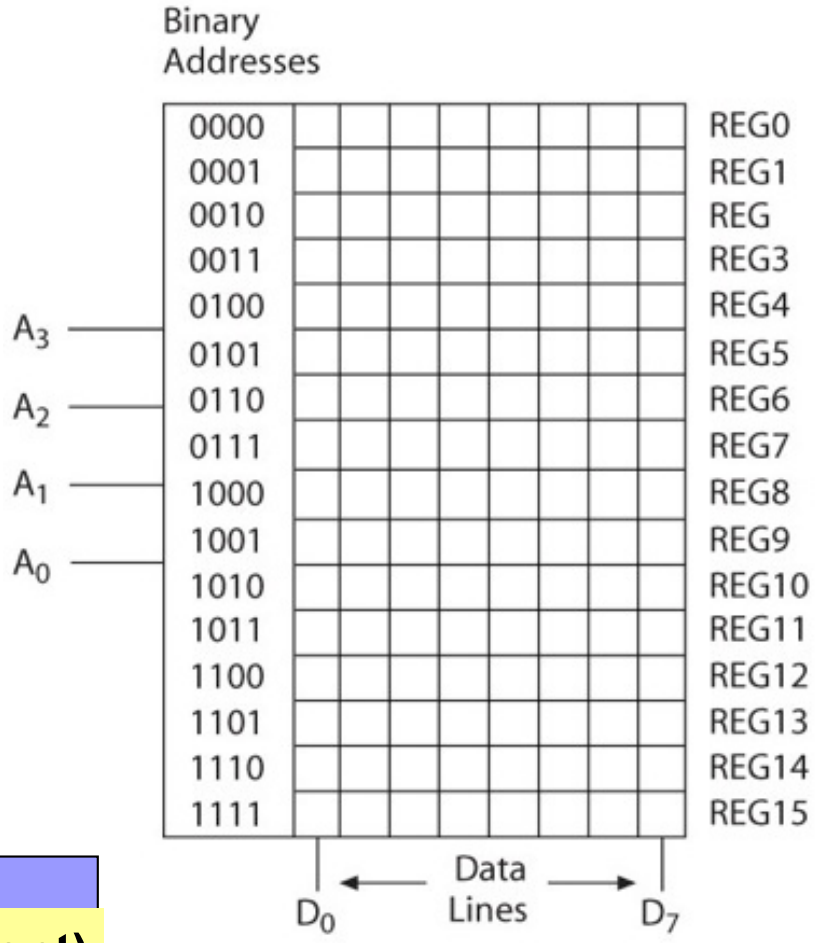
Memory



- A semiconductor storage device consisting of registers that store binary bits
- Two major categories
  - Read/Write Memory (R/WM)
  - Read-only-Memory (ROM)

Storing Bits in Memory

- We can store in different memory types
  - EEPROM, FLASH, RAM, etc.
- In an 8-bit RAM
  - Each byte is stored in a single memory register
  - Each word is stored in two memory locations (registers)
  - DATA 0x1234
    - 0x12→REG11 (High-order byte)
      - 0001 0010
    - 0x34→REG10 (Low-order byte)
      - 0011 0100

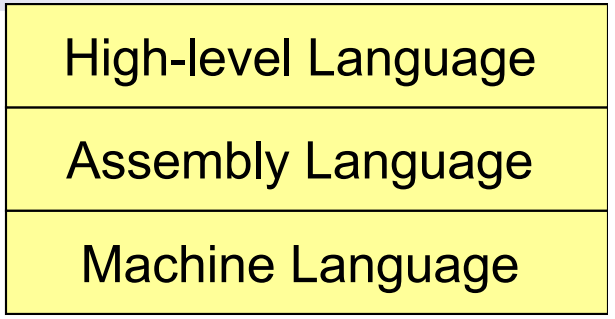


Remember -8→111 1000 (in two's complement)

So, How Do We Write the Instructions and Tell the MPU What to Do?

… We use a Software Language

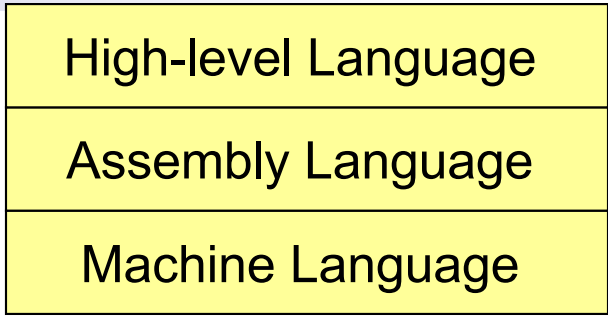
Software: From Machine to High-Level Languages (1 of 3)



- Machine Language: binary instructions
  - All programs are converted into the machine language of a processor for execution
  - Example:

Machine Instruction	Machine Operation
00000000	Stop Program
00000001	**Turn** bulb fully on
00000010	**Turn** bulb fully off
00000100	Dim bulb by 10%

Software: From Machine to High-Level Languages (2 of 3)

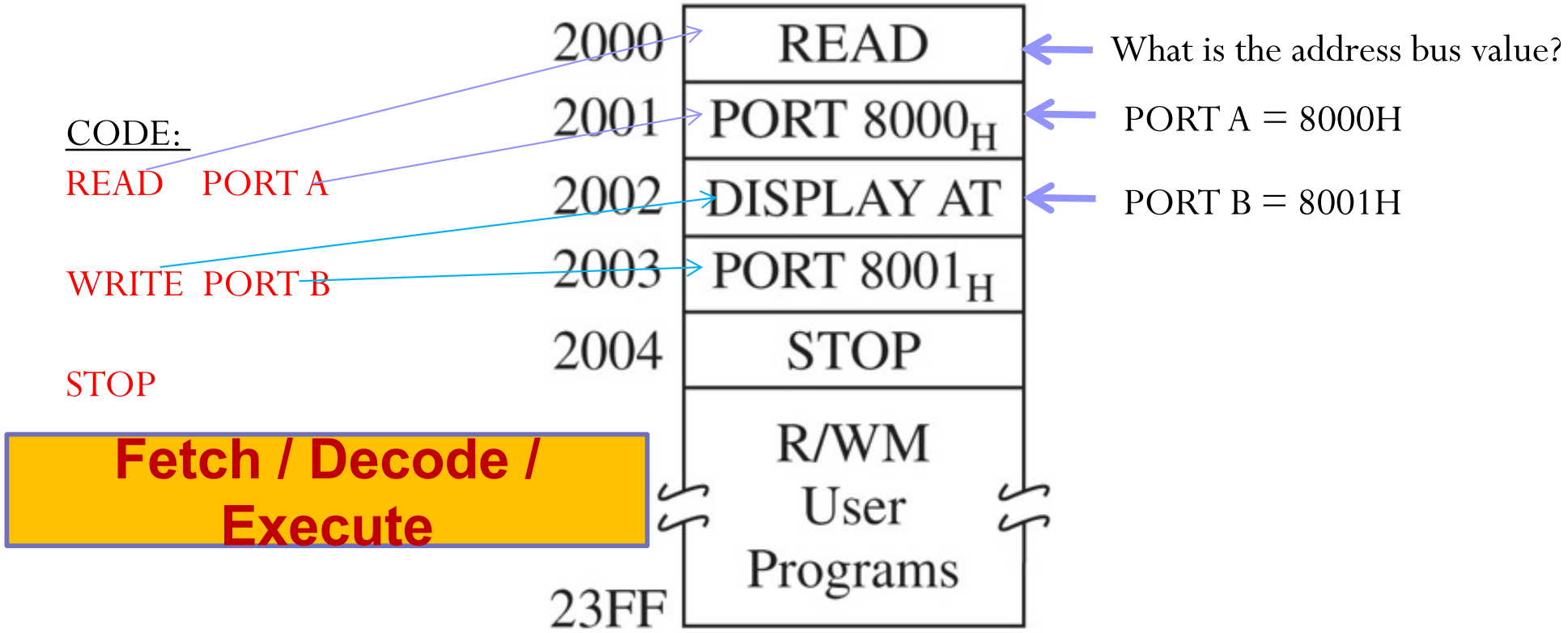


- Assembly Language: machine instructions represented in mnemonics
  - Has one-to-one correspondence with machine instructions
  - Efficient in execution and use of memory; machine-specific and not easy to troubleshoot
  - See next slide….

Assembly Language Example:

Symbolic Representation of
Program Memory Contents

- Addresses Registers



Software: From Machine to High-Level Languages (3 of 3)

High-level Language
Assembly Language
Machine Language

- High-Level Languages (such as BASIC, C, and C++)
  - Written in statements of spoken languages (such as English)
    - machine independent
    - easy to write and troubleshoot
    - requires large memory and less efficient in execution

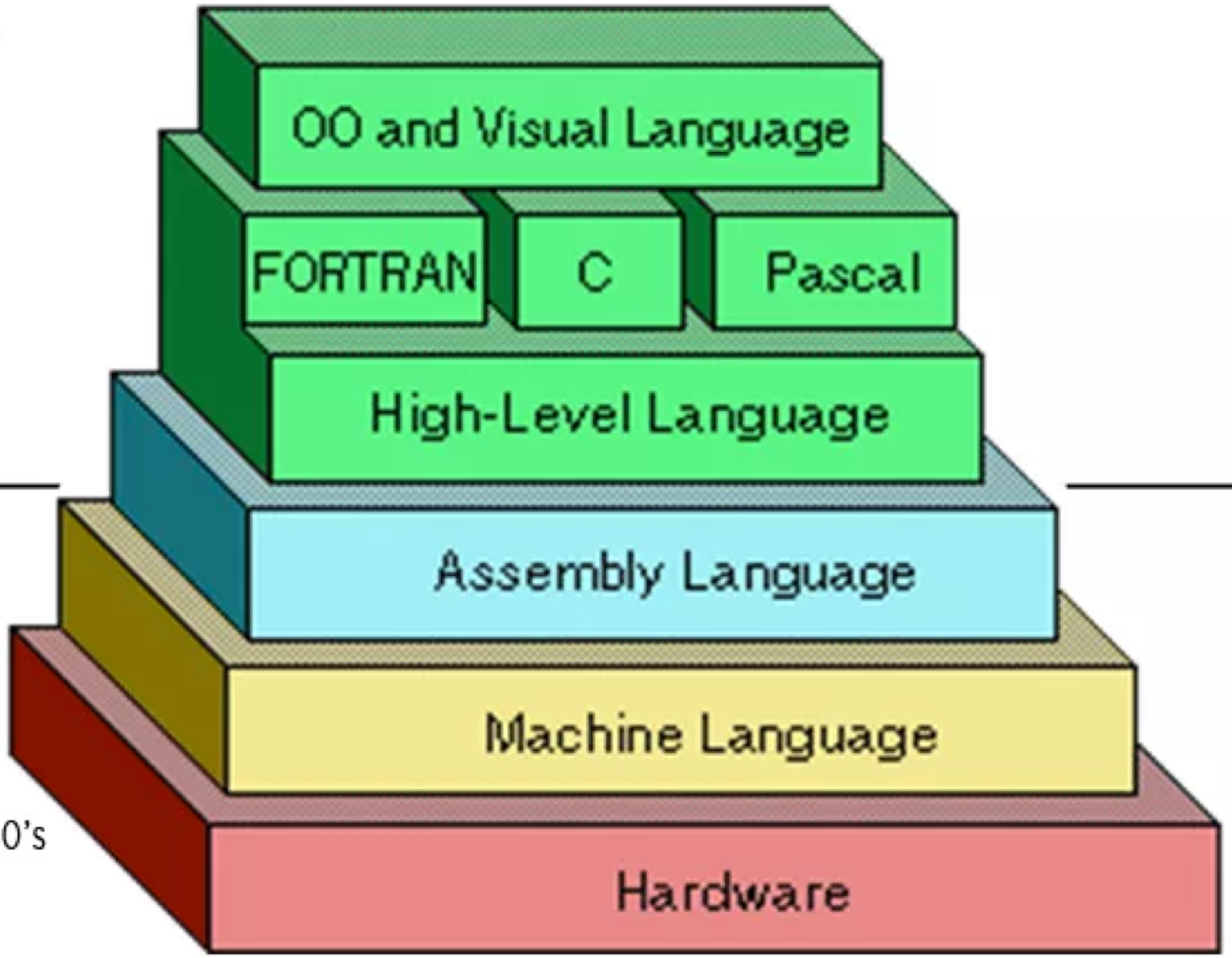
```
2   #include<conio.h>
3   void main()
4   {
5   int num,i;
6   printf("Enter a number below 100\n");
7   scanf("%d",&num);
8   for(i=1;i<100;i++)
9   {
10      printf("%d\n",i);
11      if(i==num)
12      break;
13  }
14  getch();
15  }
```

High Level Language

- Easy for Programmers to understand
- Contains Engilish Words

Low Level Langugae

- The computer's own Language
- Binary numbers, in 1's and 0's

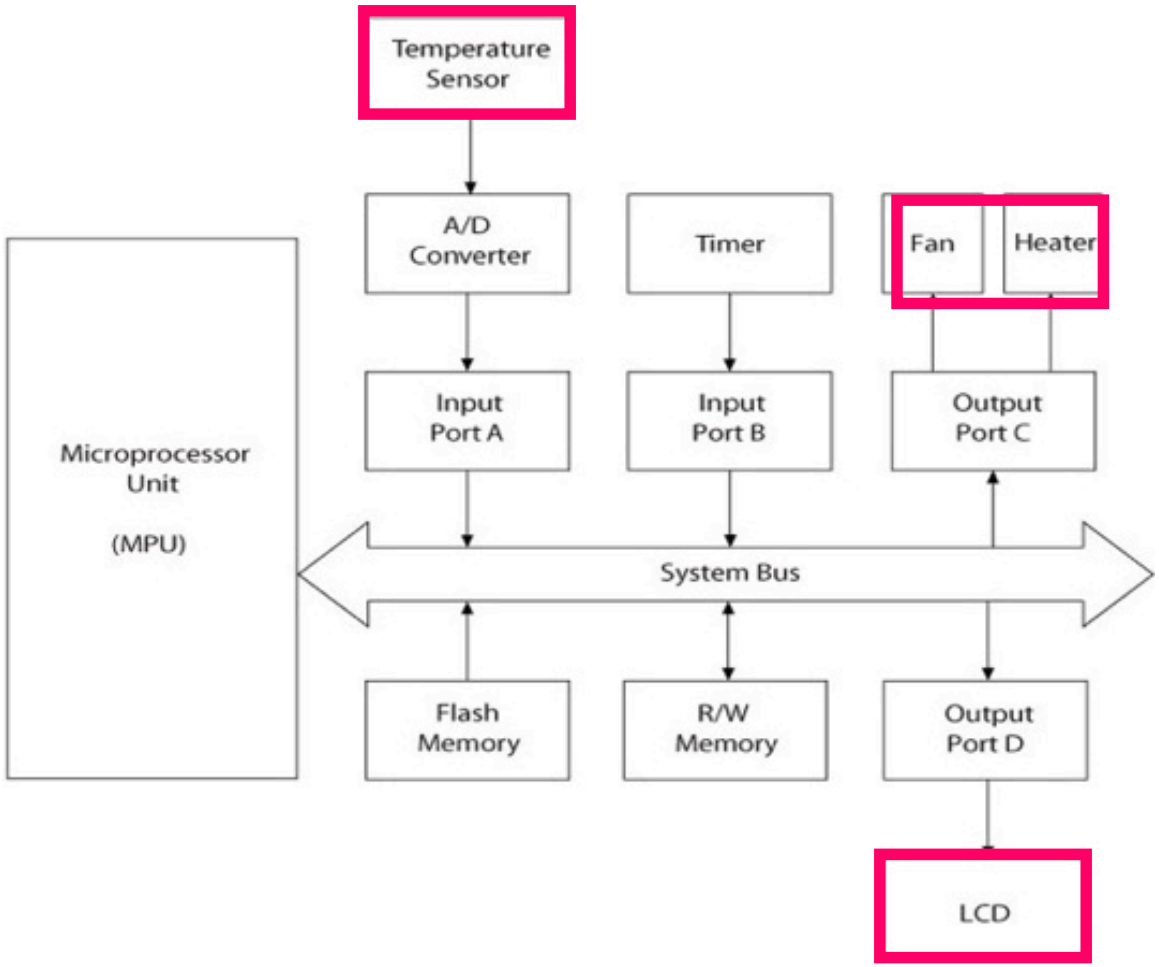


justcode.me

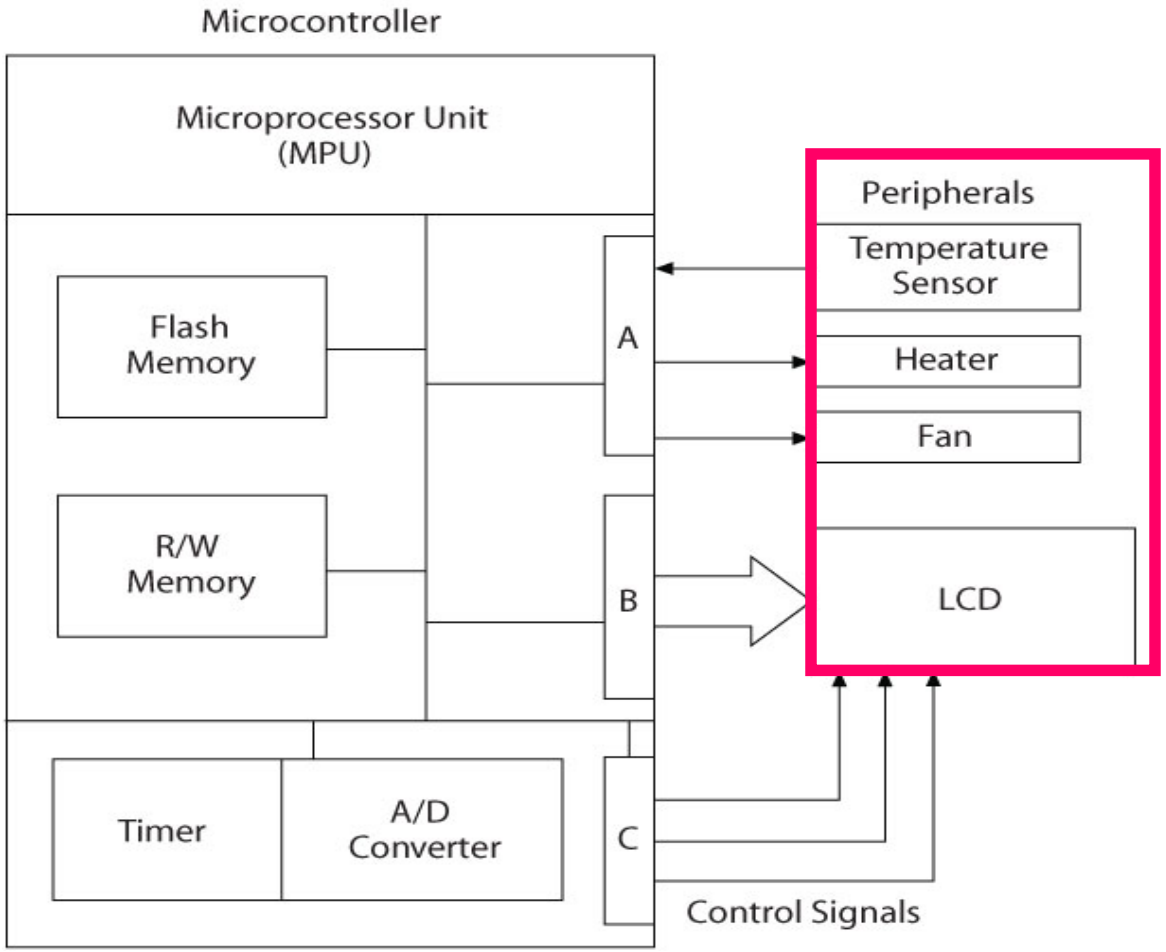
Design Examples …..

Microcontrollers vs. Microprocessors

MPU-Based Time and Temperature System



MCU-Based Time and Temperature System



References

- Computer History Museum: http://www.computerhistory.org/
- Read about microcontrollers: http://www.mikroe.com/en/books/picbook/2_01chapter.htm
- Lots of good information exist on Wikipedia about microcontrollers http://en.wikipedia.org/wiki/
- History of transistors: http://inventors.about.com/library/weekly/aa061698.htm
- Nice transistor timeline by Intel: http://www.intel.com/technology/timeline.pdf
- I used a few slides from here: http://www.ceng.metu.edu.tr/courses/ceng336/_documents/introduction.pdf
- ARM related references:
  - http://mc2.unl.edu/2013/10/03/getting-started-with-arm-microcontrollers/
  - http://www4.cs.fau.de/Lehre/SS06/HS_AKES/slides/ARM.pdf - Very good reference !

References - RISC

- http://cse.stanford.edu/class/sophomore-college/projects-00/risc/
- http://en.wikipedia.org/wiki/Complex_instruction_set_computer
- http://en.wikipedia.org/wiki/RISC
- http://arstechnica.com/articles/paedia/cpu/pipelining-1.ars/4