Embedded Systems Concepts

Made by Ahmed Yasser

The main Topics

- ✓ Computer System.
- ✓ Embedded System.
- ✓ Microcontroller.
- ✓ Microcontroller Vs microprocessor.
- ✓ CPU.
- ✓ Instruction Set Architecture.
- ✓ Memory.
- ✓ Microcontroller Programming workflow.
- ✓ Embedded SW Development.

Computer System

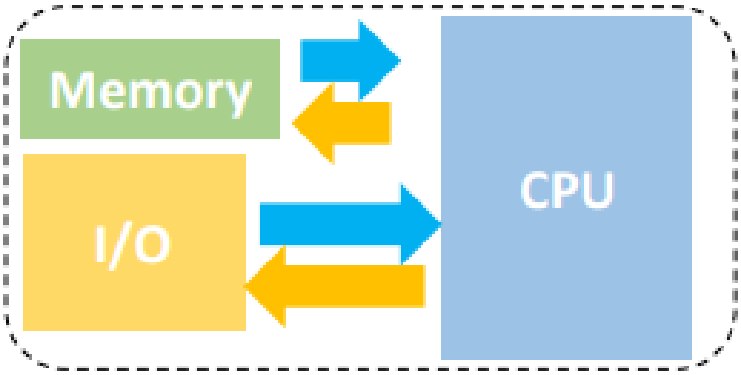
Computer System

The computer system aka Computing System is a basic, complete and functional computer, including all the hardware and software required to make it functional for a user and capable to perform mathmatical and logical operations.

It should have the ability to receive user input, process data, and with the processed data, create information for storage and/or output.

Computer System consists of

- ✓ **CPU.**
- ✓ **Memory.**
- ✓ **I/O.**



Computer System

Embedded system

Embedded system

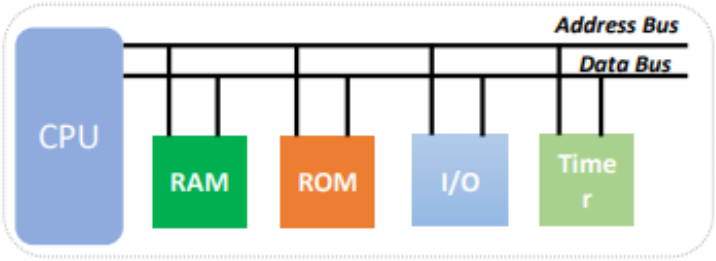
is a computing system with limited resources to perform specific tasks or has specific functions it is deferent to the computer system that it is not a general purpose.

Personal computer also has more powerful resources.

Embedded System

Microcontroller

It is a small computer which can control any thing around the world
it contains RAM, ROM, I/O, etc.
At times it is also termed as a mini computer or a computer on a single chip.

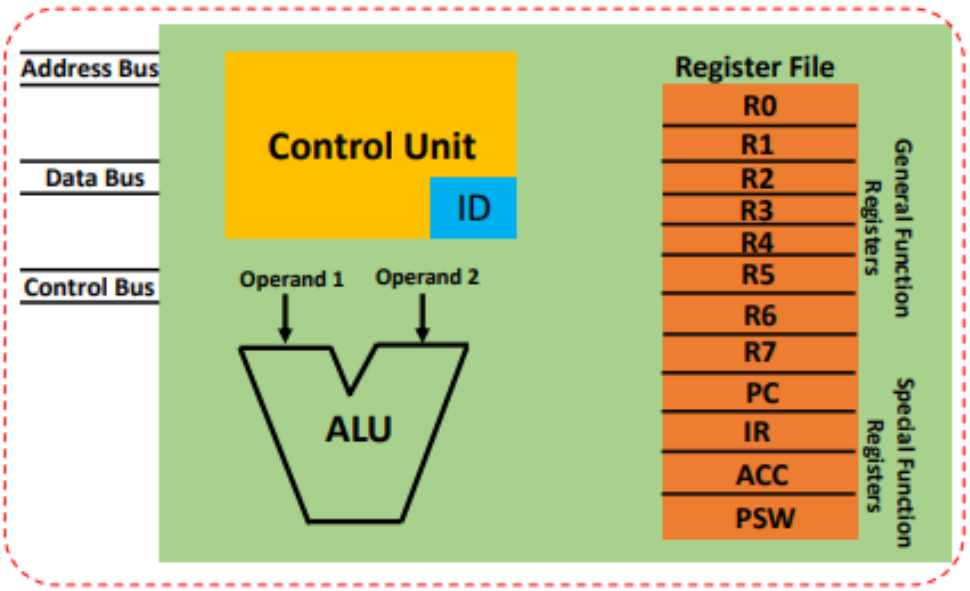


Microcontroller Vs Microprocessor

Microprocessor is an IC which has only the CPU inside them.
These microprocessors don't have RAM, ROM, and other peripheral on the chip.
A system designer has to add them externally to make them functional.

CPU

It is considered the brain of the microcontrollers.
It is responsible for executing arithmetic and logic operations.
CPU Executes Each instruction in three stages (Fetch – Decode - Execute).



IC and DSP

Integrated Circuits (ICs): Miniaturized electronic circuits combining semiconductor devices and passive components onto a single chip. Used in a wide range of electronics for various functions like digital logic, memory storage, and interfacing.

Digital Signal Processors (DSPs): Specialized microprocessors designed for real-time digital signal processing tasks such as filtering, modulation, and compression. Widely used in telecommunications, audio/video processing, radar systems, and more.

Control Unit

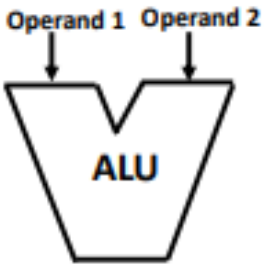
The control unit manages the execution of instructions by fetching them from memory, decoding them, and coordinating the necessary data transfers and operations within the CPU.

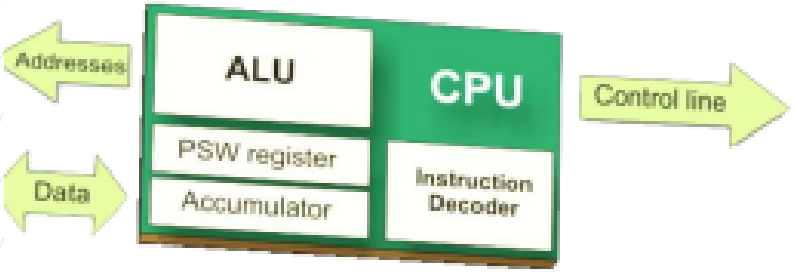
Instruction Decoder(ID)

It's a sub unit of control unit dedicated for decoding/interpreting the code instructions.

Arithmetic Logic Unit

It performs all mathematical (addition, subtraction, multiplication, divide) and logical operations (OR, AND, ...) upon data.





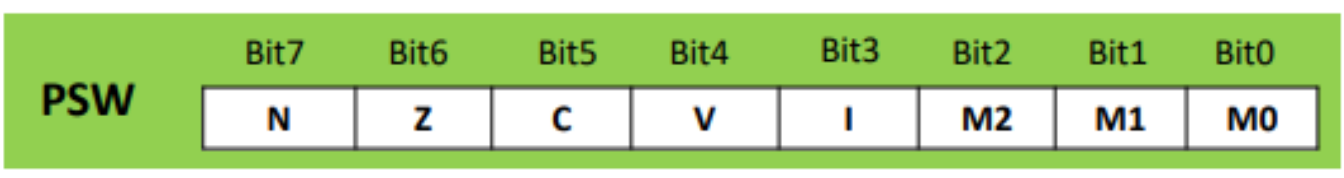
Register File

• It consists of several registers (memory) used by CPU operations.
There are two categories of these registers:
•General Function/Purpose Registers:
• It's used as a temporary memory for holding the instruction data.
• Special Function/Purpose Registers:
• Every register has it's own special function.
√ PC (program counter)
: • It always holds the next instruction address.
• It increments with each instruction fetching.
√ IR (Instruction Register) • An instruction register holds a machine instruction that is currently being executed.
√ ACC (Accumulator) • It's a kind of working desk used for storing all data upon which some operation should be.

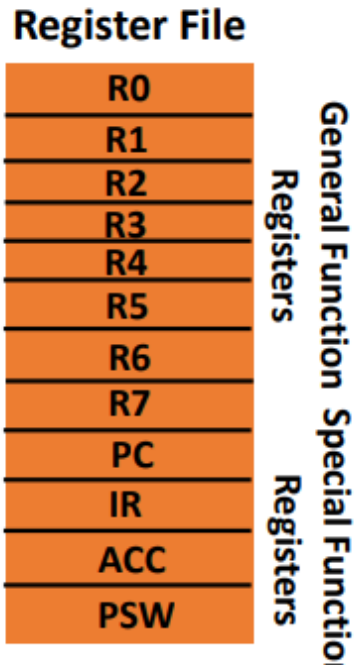
PSW (Program Status Word)

• It holds information about the most recently performed ALU operation.
• It controls the enabling and disabling of interrupts.
• Set the processor operating mode.

• N(Negative Flag) Z (Zero Flag) C (Carry Flag) V (Overflow Flag)
I (Interrupt Enable)
M0:M2 (Processor Mode)



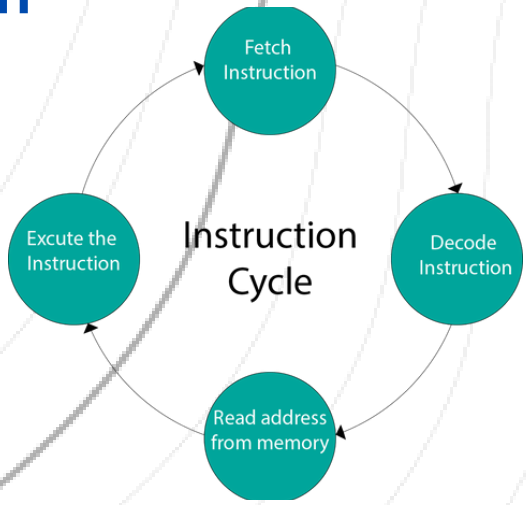
Register File



Instruction Execution

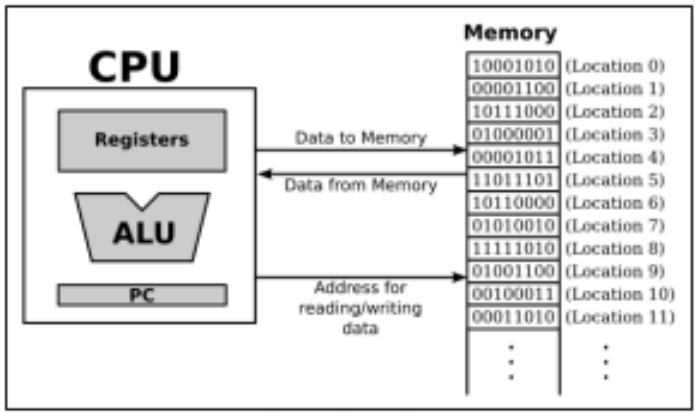
Fetch

• In this stage CPU (Control Unit) fetches the instruction from ROM.



Decode

•After fetching the instruction from memory, the control unit interprets its opcode (operation code) to determine the operation to be performed. The control unit also analyzes any additional fields within the instruction to extract operands, memory addresses, or other parameters required for the operation.



"CPU fetch"

Execute

• Now, the command will be executed.

• Then the result shall be stored back to memory.

Instruction Set Architecture

All instructions understandable to the microcontroller are called together the Instruction Set.

• When you write a program in assembly language, you actually specify instructions in such an order they should be executed.

• The main restriction here is a number of available instructions.

RISC

```
...     ...
...     ...
movlw   0x3F
movwf   TEMP1
btfsc   MAX3,7
goto    check
btfsc   MAX3,6
goto    opening
btfsc   MAX3,5
goto    closure
...     ...
...     ...
```

CISC

RISC

In this case, the microcontroller recognizes and executes only basic operations (addition, subtraction, copying etc.).

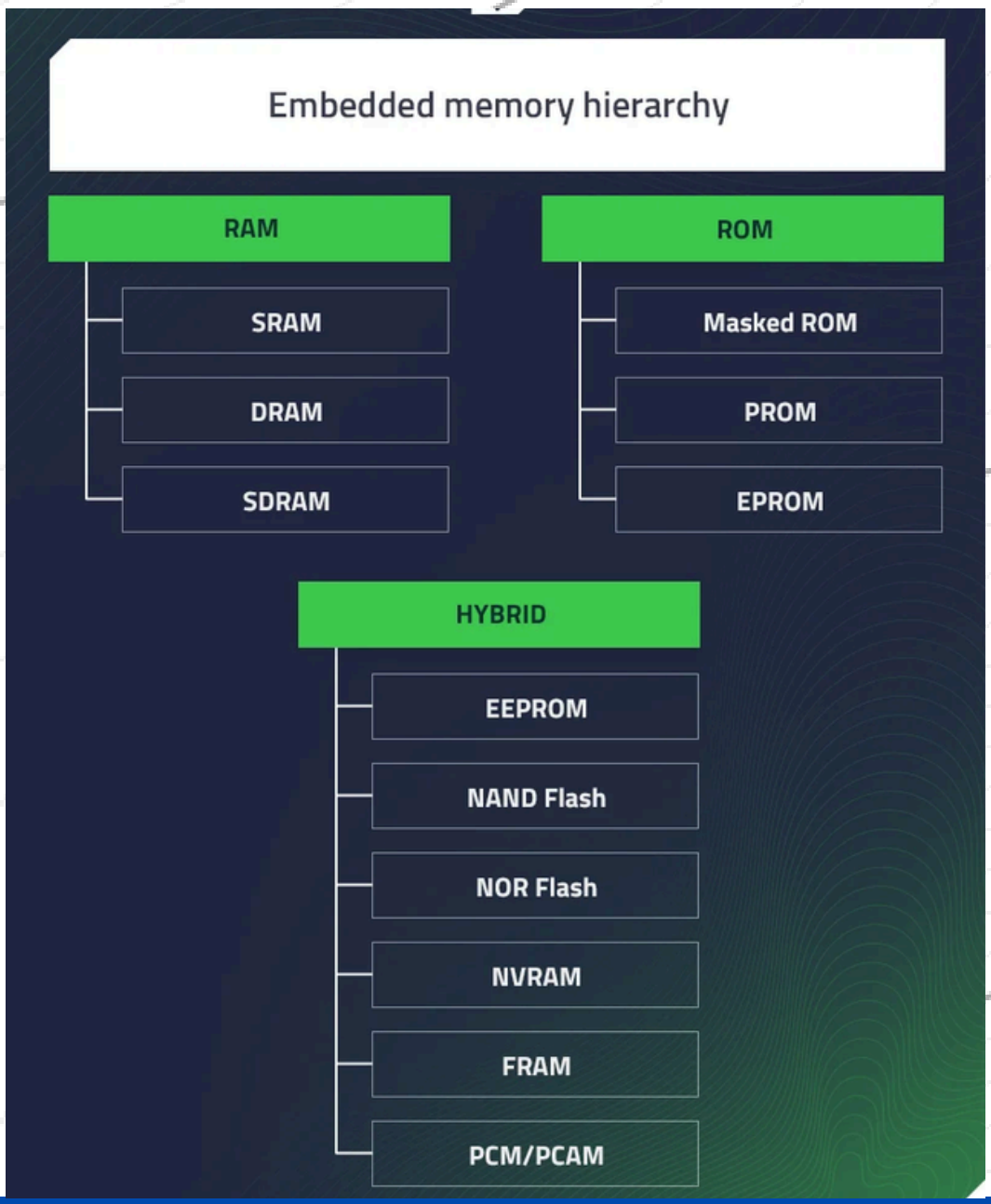
• Other, more complicated operations are performed by combining them. • For example, multiplication is performed by performing successive addition.

CISC

• CISC is the opposite to RISC! Microcontrollers designed to recognize more than 200 different instructions can do a lot of things at high speed.

• However, one needs to understand how to take all that such a rich language offers, which is not at all easy.

Memory.



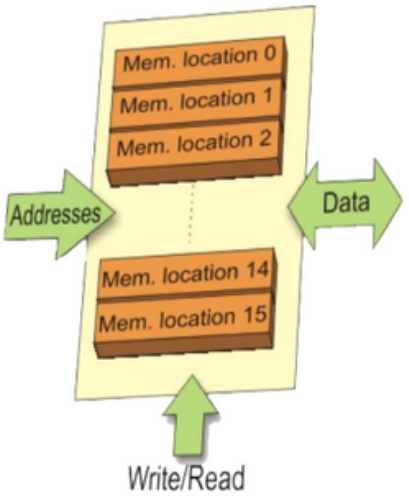
Memory.

Volatile

non-Volatile

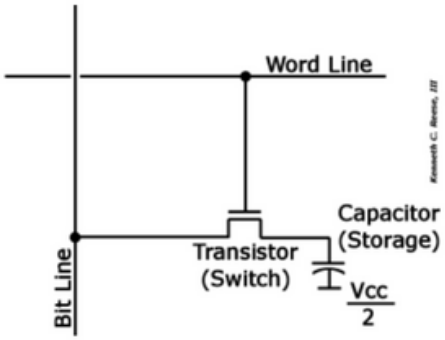
Volatile Memory Types.

• Volatile memory is the memory that lose its content with power loss. • We have two types of volatile memory:



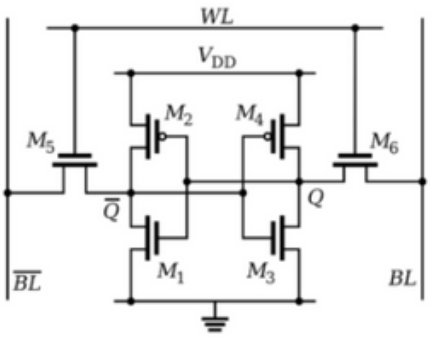
SRAM (Static Random Access Memory)

• It's memory cells (each memory cell represents a bit) construction is based on transistors.
• DRAM stores data by writing a charge to the capacitor by way of an access transistor.
• RAM uses capacitors that lose charge over time due to leakage.
• DRAM must be supplied with a voltage to retain memory (and is thus volatile)



DRAM (Static Random Access Memory)

• It's memory cells (each memory cell represents a bit) construction is based on transistors.
• SRAM does not use capacitors.
SRAM uses several transistors in a cross□coupled flip-flop configuration and does not have the leakage issue and does not need to be refreshed



So, SRAM is most suitable for Embedded Systems

Non-Volatile Memory Types.

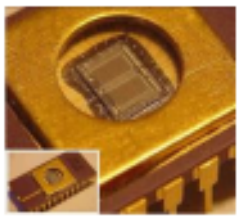
• Non-Volatile memory is the memory that doesn't lose its content with power loss. • We have five types of volatile memory

Masked ROM

• It's a type of memory that it's content is predefined by the manufacturer.
• It's most suitable for mass production.
• Reprogramming doesn't valid.

OTP ROM (One Time Programming)

• It's a type of memory that could be programed by user but for only one time.
• It's used for security reasons.
• Reprogramming doesn't valid.



"EPROM UV hole"

EPROM Erasable Programmable ROM

• It's a type of memory that could be programed by user multiple times.
• It could be erased in a special way using the UV.
• Reprogramming is valid.

EEPROM Electrical Erasable Programmable ROM

• It's a type of memory that could be programed by user multiple times.
• It could be erased electrically
. • Reprogramming is valid.

FLASH

• It's a type of memory that could be programed by user multiple times.
• It could be erased electrically
. • Reprogramming is valid.

We could notice that the EEPROM and FLASH have the same specs but actually there are many differences that we would study later.

THANK YOU