Embedded systems

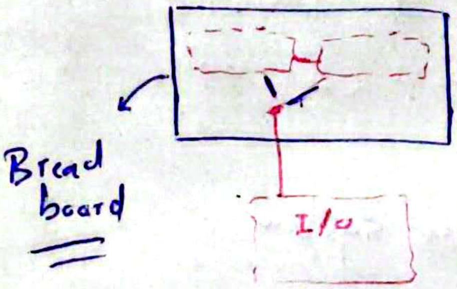
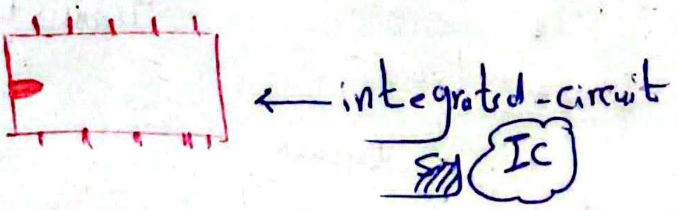
Computer
- → processor
- → memory : Ram/Rom
- → I/o Peripherals : Keyboard / printer/ --

General Purpose
Ex: Labtops

Special Purpose
Ex: Washing machine ✓ Not a technical

→ Embedded system
- Power
- Cost
- Performance & size

How To make Es

System board	system on chip
	
* Large size	* Small size (✓)
* Higher Cost	* Less cost (✓)
* Higher Power consumption	* Lower Power – (✓)
*	
Same	Performance
→ Configurability (✓)	×

Can be modified ↵

IC : integrated-circuit → single circuit do a specific task according to its connection

Ex: Timer 555
op-amp

ULSI : Million of transistors on the same chip-size

Very Large Scale integration

chip-size → size, functionality

*Moore's Law : No of transistors on a single chip-size doubles each two years

---

MPU : micro processor unit

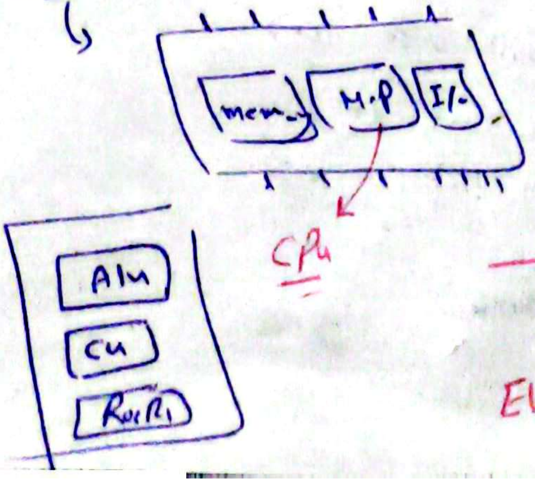
- Processor → Vacuum tubes → Bit: 0 : Vacuum, 1 : charged
- MicroProcessor → Transistors
- CPU → Central Processing unit

Master → Primary

GPU → secondary

MP بيكون ، الأساسي في القطاع)

* **MCU** : micro Controller unit → Computer system



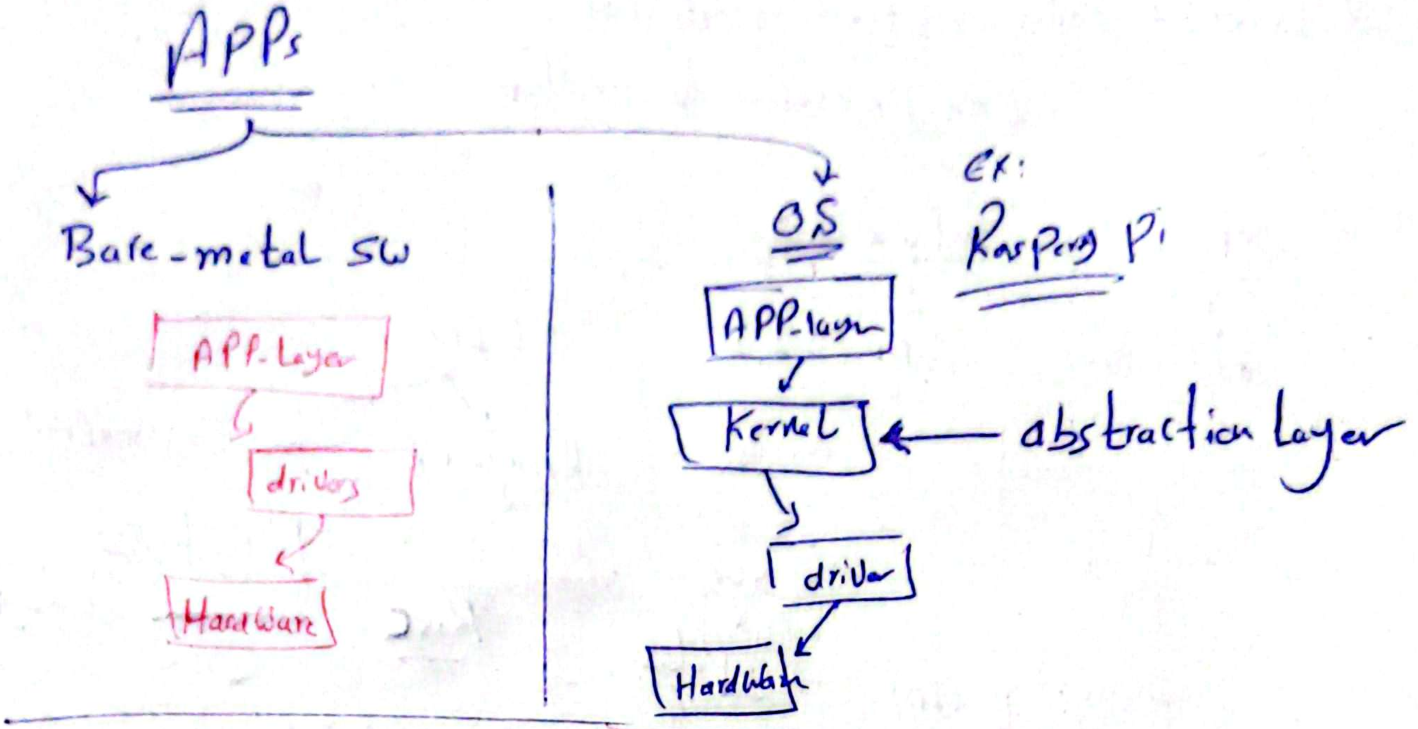
Ardunio kit + sensors & actuators

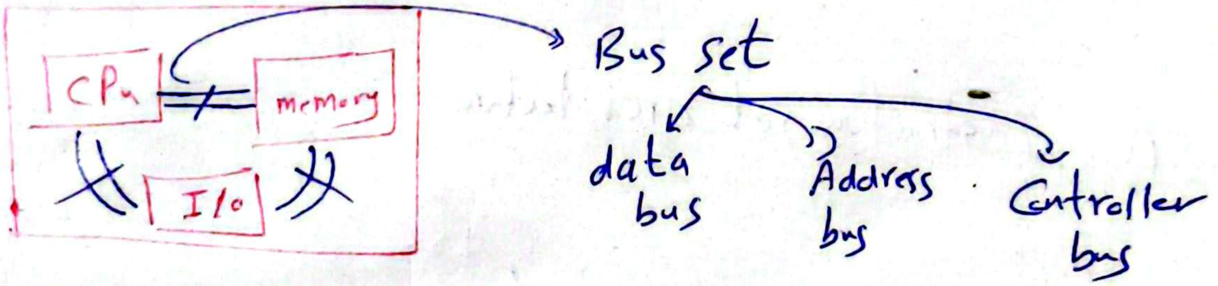
ECU
Electrical Control units

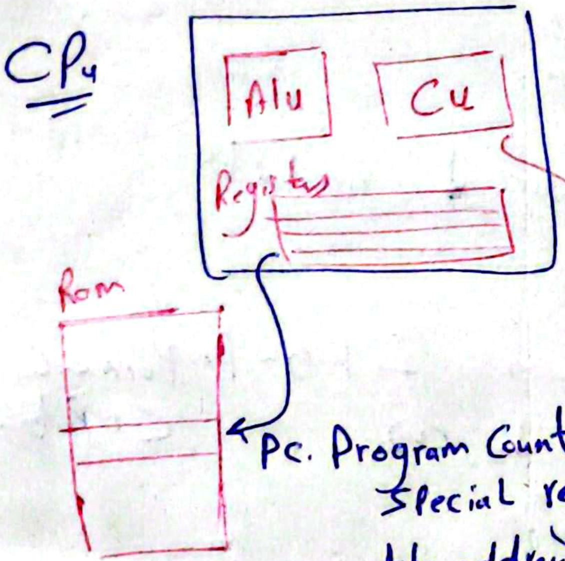
*GPU كارت شاشة → Graphics

*DSP → Digital signal Processing

Voice / temp.







Instruction life cycle:
(Fetch _decode - execute)

Fetch instruction From Rom to a Register → IR instruction register

Decode instruction by instruction Decoder (ID) < set: OP-Code dict
Format Ex: 8-bit form

000	111	01
op-code = add	operand 1	operand 2

Execute by Alu → General-Register

Pc. Program Counter: Special register Keeps the address of next instruction → اللى عليها الدور فى التنفيذ

*Assembly codes, instructions differ one Processor to another

(+ -) mPu 1 → add ...

mPu 2 (+ - * ÷) → ADD 111

Instruction needs

HW → Instruction decoder (ID)

SW → Compiler

$x * 5 \rightleftharpoons x+x+x+x+x$

(ID)

→ Hard wired → Logic gates

Risc

→ memory mapped

→ set

Search in memory

Cisc

ISA : Instruction set architecture

Risc → Reduced

inst. ↓

* Software ↑

→ يترجم ركلبات الى ميفهماش

Cisc → Complex

↑

* Software ↓ , needn't

Risc | Cisc → High Performance Computer

Size ↑ Hardware ~ Size ↓ (v)

Alu↓ ~ Alu ↑

cost HW↓ SW↑ ~ SW ↓ HW ↑

* → + Performance [illegible] ~ Performance [illegible] (Search)

Power Alu↓ ID↑ ~ Alu↑ ID↓

Registers

General Purpose

data store
(TemPorary)

* register int x;
Stored in CPU
Not Ram

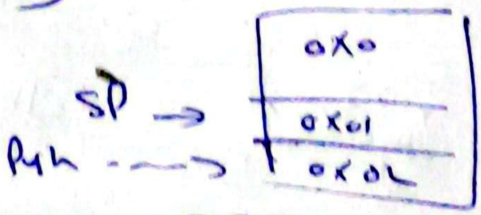
&x → Error
can't access address

Access by name:
r0, r1, r2, —

Special Function

① PC : store address of next instruction
Program Counter

② SP → stack Pointer
في مكان ال Memory



③ ACCumalor → Old

④ IR → instruction register
store instruction after Fetching

⑤ PSW → Process status Word
0 → +ve
1 → -ve
← sign flag, overflow
Zero
flags →
يغيروا آخر Ins. دخلت مع ال Alu

8 bit microProcessor
G register: 8 bit size

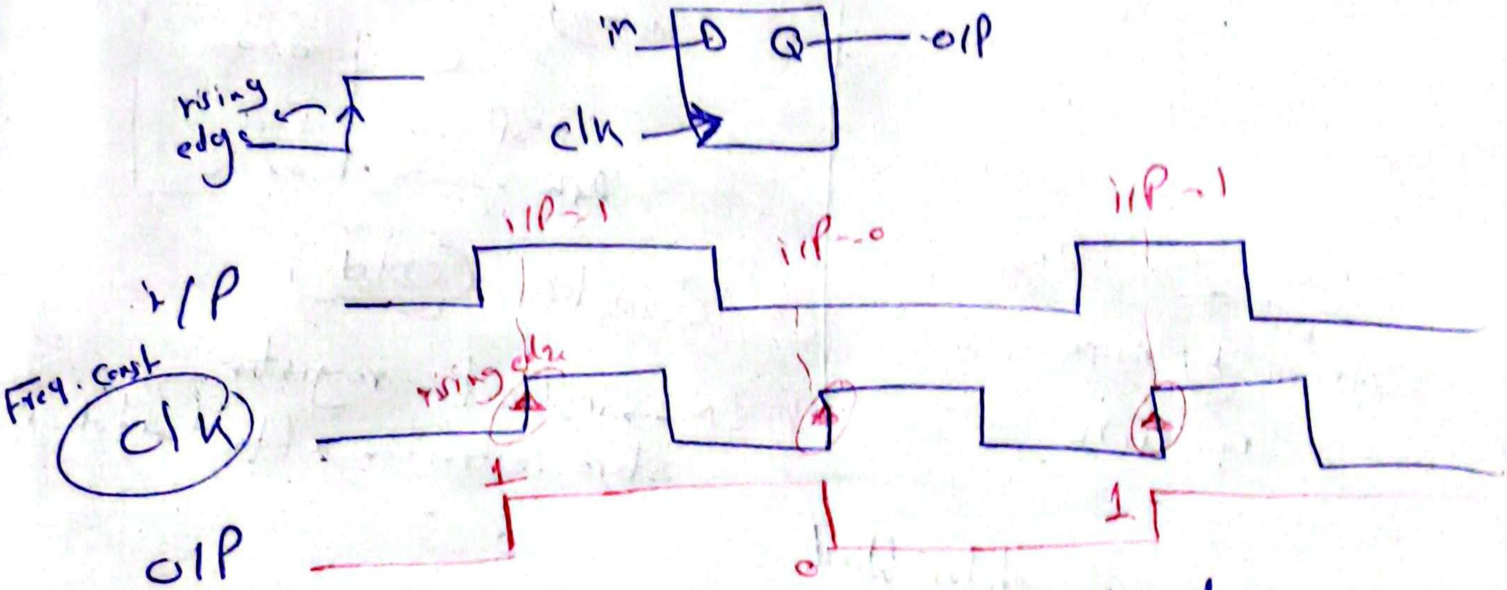
Memory :- Locations → Ex: Location is 8-bit size

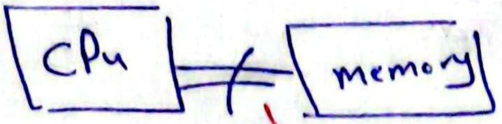
0 1

Access time (R/W)

Faster Slower

- Basic memory elements: Flip Flop → bit





Bus set

data → No. data lines (Bits) → No. of memory bits in Location

Address → 2^n Locations → No. of address Lines (Bits)

Control → R → 0, W → 1

Memory characteristics

1. Capacity size → Address Lines
2. speed Access time
3. organize 4K × 4 → 16K

data lines → Word size

Address lines → No. of accessable Locations

Memory types :-

memory

Volatile
RAM

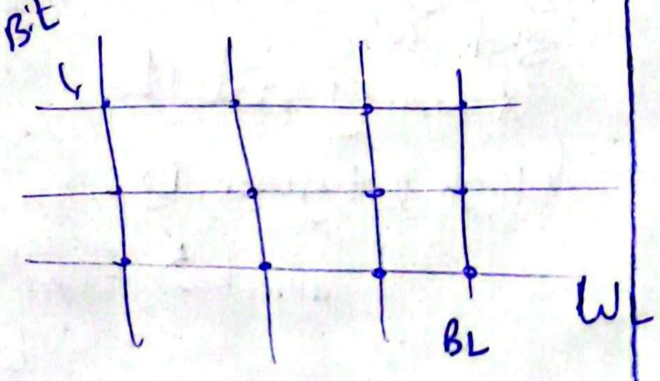
Non Volatile
ROM

Hybrid

① Volatile : RAM Random Access Memory
RWM Random Working memory

SRAm : Based on transistor
↳ Static

bit → flib flob
min: 6 Transistors
2 Mosfets & 2 inverters



* No need for refreshment circuit
↳ Faster

* High Cost Per bit

DRam
↳ Dynamic : Capacitor & simple Mosfet

discharged
Bit 0

Charged
Bit 1

المشكلة : المكثف بيفرغ شحنته بعد فترة
↳ Refreshment Circuit DRam Controller

- Higher Piority than CPu on Memory
- Access time inc. & sPeed dec.

SRam : الحل

Dram Advantages

- * Simple hardware
- * low Cost Per bit
- * high density
- * Low Power Consumption (Mosfet) → SRam is lower due to refreshment Circuit of Dram

Es → SRAM → Constraints
(Cash)

---

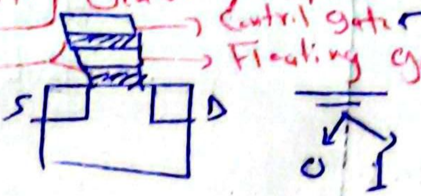
2 Non-Volatile Rom Read only memory (MP)

PC read From
(Program memory)
البروسيسور هو اللي مبيقدرش يكتب فيها

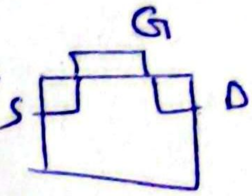
Slower than Ram

Rom Saves Program instr.

Based on Floating Gate Mosfet



Mosfet



High Voltage by Flasher

Programming State (0)

charge (F.G)

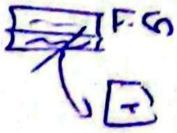
سالب ⇓ 0 → Programming State

موجب ⇓ 1 → erasing State

x erasing state (1)

High Voltage on D

attraction



Types of Rom

① Mask Programmable Rom

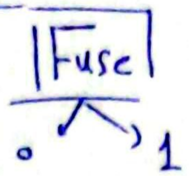
OTP → one time Program

* Bios chip
* Toys

② PRom

Programmable Rom

OTP → by (User)



③ EPRom → Erasable Programmable Rom

(UV) → لازم تمسح كله بعده

Noise & Radiation Sensitive → data Corruption

Hybrid → Ram, Rom — Read & write, Non Volatile

E²PRom

- Electrical Erasable
- Endurance → 100,000 write & eraw cycles
- Byte Access
- high cost Per bit

EEProm

internal — Peripheral

External — Communication Protocol

Flash

- Faster than E²PRom
- Block Access → Sector by Sector
- Endurance 10,000
- Low Cost Perbit

nVRam

SRAm + Battery / E²PRom + battery

- Faster than Rom
- High Cost Per bit

SRam + battery → Backup

SRam + E²PRom + battery → move data