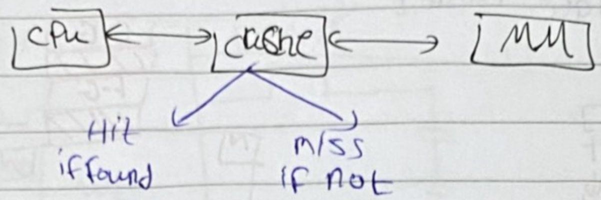
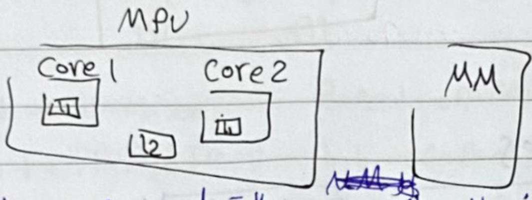
Page: Date: / /

Cache → SRAM



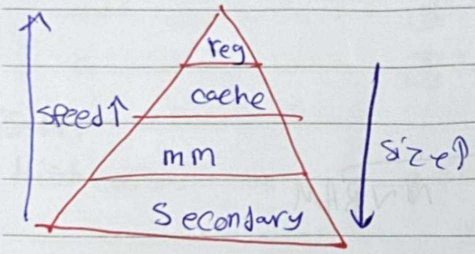


ممكن core يعدل في ال cache بس التعديل ده مش موجود ل core التاني . طب ايه الحل ؟؟

↓

Cache Coherence

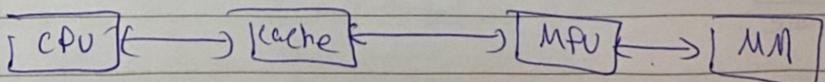
لو حصل تغيير في ال cache التغيير يوصل لكل ال cores

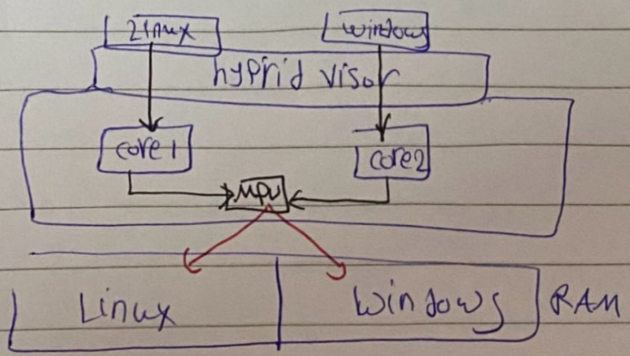


FPU - MPU - MMU :

FPU (Floating Point unit): العمليات على ال floating point numbers

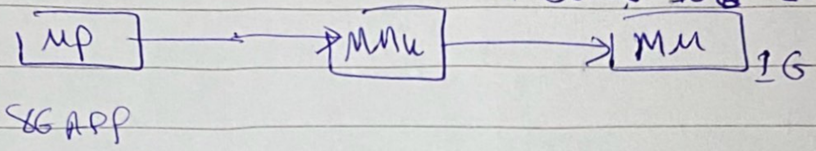
MPU (~~Micro Processor unit~~ memory protection unit): بيسمح لكل OS انه يتعامل مع الجزء الخاص بيه من ال RAM





Date: / /

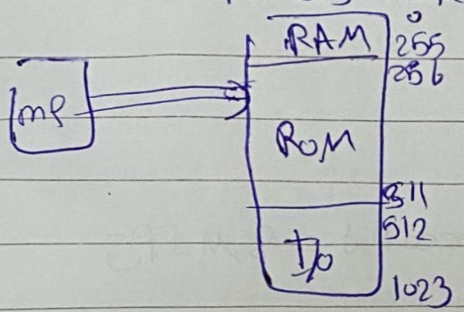
MMU (memory mangement unit): بتفهم ال APP 8G انه موجود في 8G لكن فعليا الي اتشال 1G بس على اد ال MM

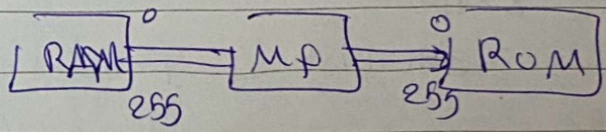


I/O (interface)

Arch types

- Von-neumen → PC
  - one memory sys tem
  - memory mapeed
  - عشان ال wires كتيرة جدا وده هيعمل مشاكل لو كان harvard
- Harvard → Micro controller
  - port mapped
  - عشان هو حطوط كله في chip صغيرة فا مكنش سعبوش سلوك كتير وبيساعد اكتر في الاداء ال MC



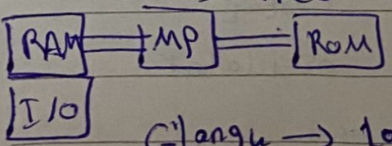


السوفت وير اللي بيحدد انا كده بكلم ال RAM ولا ال ROM

SW
- RAM: Assembly · Load/Store
- ROM: R/W

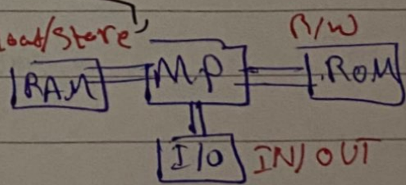
I/O (Harvard)

- memory mapped



C langu → load-store

- (port mapped)



Page: Date: / /

Pipe line

inst 1	f	d	e			
inst 2		f	d	e		
inst3			f	d	e	
inst4				f	d	e

Von-neu: can't support pipeline
harverad: can support pipeline

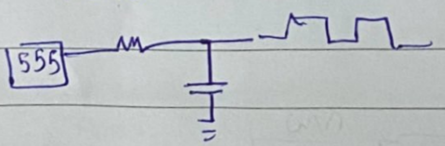
RISC: can support pipeline (معظم ال instructions بتخلص في clock cycle واحدة)
CISC: can't support pipeline (ال instructions بتخلص مو كلها clock cycle)

* if clk = 8 MHz
RISC → 1 ~~clk~~ inst → 1 cycle
∴ 8 MHz = 8 Mcycles/s → 8M instruction per second = 8 MIPS

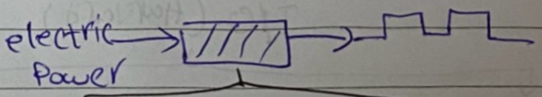
Clock systems

electrical — mechanical

electrical: RC oscillator



mechanical: meterial



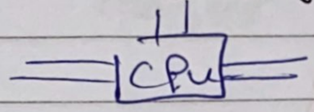
→ ceramic resonator
→ crystal oscillator

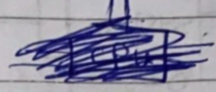
	RC	ceramic	crystal
cost	↓	in between	↑
accuracy	↓	in between	↑
settling time (الوقت الي بياخذه ال clock لحد ما تستقر)	↑	in between	↓
Noise imm (مناعة)			
→ temp	↓	↑	↑
→ EMI	↓	↑	↑
→ vibration	↑	↓	↓

Page: Date: / /

Connection

port mapped — memory mapped → ES





each device → (GPIO)
↓
range of addresses
↓
Base + offset

How to read Data sheets

TRM — tech reference manual
↳ specs

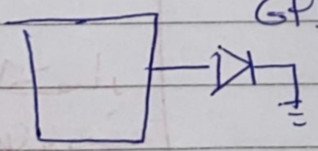
specs → memory map [GPIO → Base, ↳ END] → size + Base

↳ GPIO
- DR 0x0 + Base offset
- ODR 0x4 + Base

GPIO الي جوا ←

EX: need to write a(SW) to turn on a led

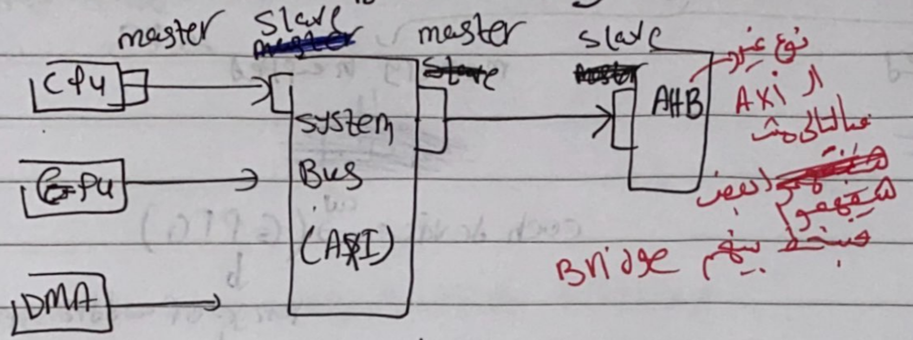
يبقى عايزين نطلع High عل GPIO bin



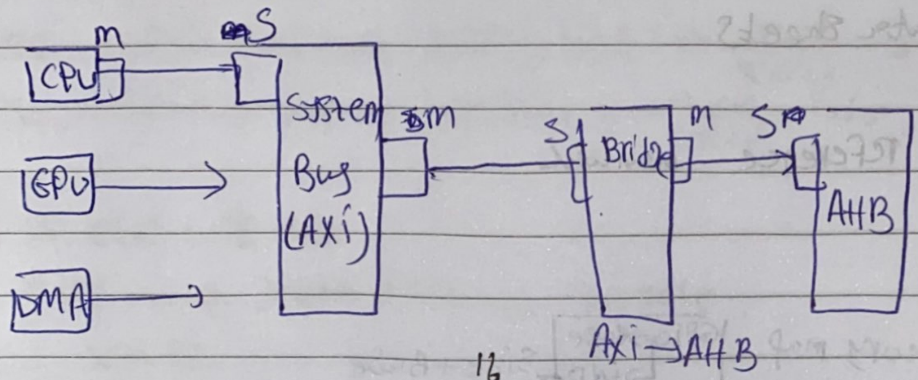
1) pointer → Direction register (DR) = 1 (O/P)
↓
Base + offset

2) pointer → ODR = 1 (H)
↓
Base + offset

Bus bridges



↓



EX: write (1) → [000 | 16 | 0000] ODR

1) define the ODR register

```
# define GPIO_ODR → Base + Offset
```

2) Pointer for GPIO_ODR

```
*(volatile unit32_t *)GPIO_ODR) |= (1<<16);
```

↓ It prevents optimization

يعني القيمة اللي على الـ register يشيلها caching وكل مره بيجي يشك على القيمه بيكون ثابتة ومش هي القيمة الحقيقية على الـ register

```
# define GPIO_ODR ((volatile unit32_t *) Base+offset)
void main () {
    * GPIO_ODR |= (1<<16);
}
```

اختصار للكود مرة

آخر update

```
# define GPIO_ODR (*(volatile uint32_t) Base+offset)
void main () {
    GPIO_ODR |= (1<<16);
```

HW port types

- Master
- Slave

HW port protocal types

- AXI
- AHB
- APB
- ACHI

· Master: هو اللى بيبدء ايا كانت العملية W/R
↓
transaction

AMBA the arm advanced Mcbus Arch

جوا كل انواع البروتوكولات عشان محتاجين نستخدم Bus bridges

* Bus bridges differ in
→ BW اقصى استخدام بيعني الطاقة (F_{max})
→ latency الوقت اللى بتاخده العملية سواء R/W لحد ما تخلص