Embedded c

[1] typedef command.

- بنستخدم ال typedef علشان نغير اسماء الداتا types العاديه الي في ال c لاسماء تانيه نستخدمها في حالات معينه محتاجينو

EX: Volatile unsigned int → Vuint32_t

[2] Header protection.

لو بكون عامل فايل (.h) ومستخدمه في كذا فايل تاني انا مش عايز في كل مره كدا يروح يعمل replacement كدا هيبقى في عندي كذا multible deceleration

← فبستخدم حاجه اسمها Header protection

لو T_H_ مش متعرفه #define قبل كدا هيعرفو وهيتنفذ الي جوا

لو متعرف قبل كدا مش هيخش على الي جوا

```
// t.h file
#ifndef T_H_
#define T_H_


#endif
```

[3] Optimization

دا مستوي process كدا ال compiler بيعملهولك على الكود بتاعك لتحقيق اهداف منها

1- reduce # of instr_

2- reduce memory access time

3- reduce Power consumption.

عندنا 4 levels

- 00 ← مفيش اي opt بيحصل ← debugging friendly
- 01 ← بيسرع ال Mem access
- 02 ← opt كامل دا
- 03 ← opt كامل + شوية حاجات = aggressive

4 volatile keyword:

باستخدمه عشان اعرف ال compiler ان ال variable دا ال value بتاعته ممكنه تتغير في اي وقت بعيداً عن الكود (ممكن يغيره ال H.w مثلاً) ← فملكش دعوه بيه و متعملهوش optimization

used with

- memory mapped Pheripheral registers
- global variable modified by interrupt
- global variable accessed by multiple tasks within a multiple treaded application.

- Volatile int foo ✓

or int volatile foo ✓

مش هيفرق مكان volatile لو بيعرف var عادي

هيفرق لو بيعرف pointer

* Volatile pointers.

في الحالة دي بعامل حاجة volatile نفس معاملة حاجة const

لو قبل (*)

- Volatile unit8-t * PReg;
- unit8-t volatile * PReg;

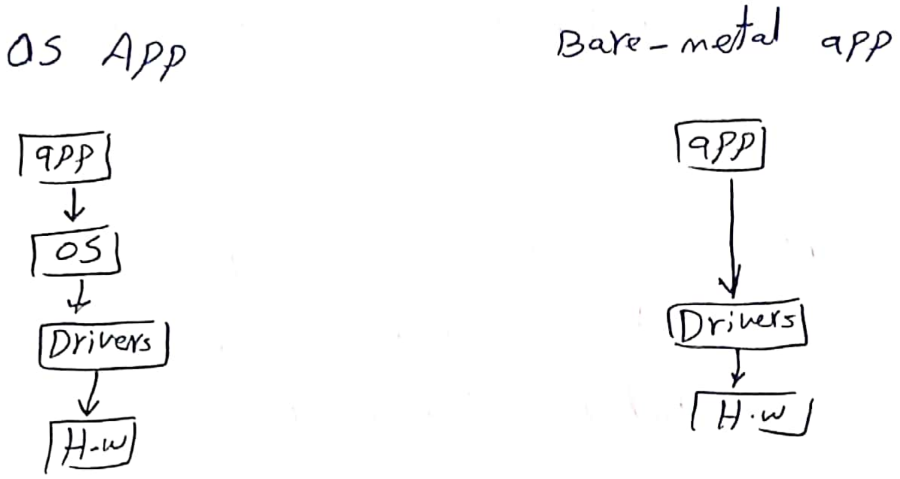
Pointer to Volatile data

بعد ال (*)

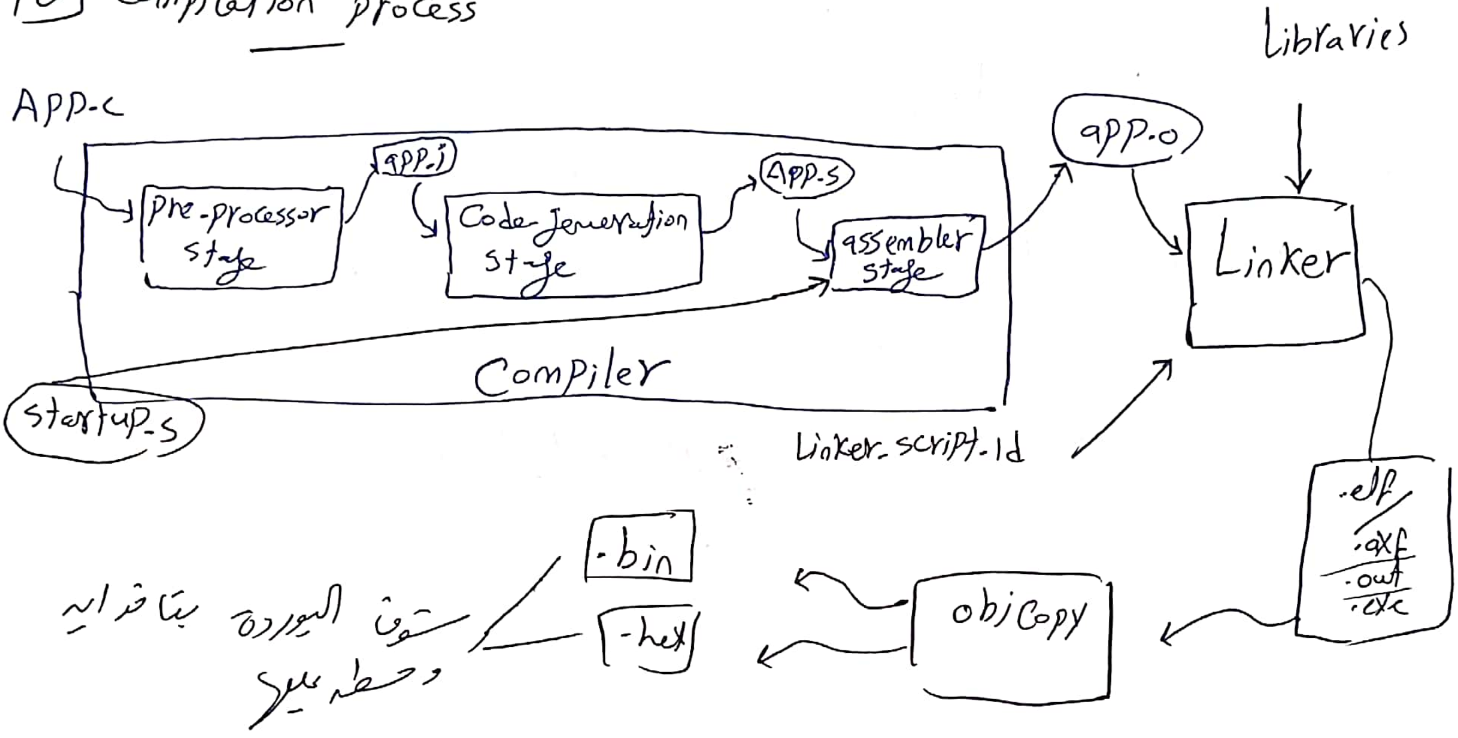
- unit8-t * volatile preg;

- Volatile Pointer to non Volatile data.

5 OS apps Vs Bare-metal SW



6 Compilation process



شوف البوردة بتاخد ايه
وحطه عليها

7 Booting sequence.

- البروسيسور لازم يكون عنده default address او اسمه Entry point

بيبدأ منه عندما يكون لسه بادئ او لو حصله reset.

الـ Entry point دى بيكون عندها code كدا بتاع الـ startup.s

عشان يـ init الـ .data ، يحط اصفار فى الـ bss.

و يحدد مكان الـ top بتاع الـ stack

بعديها يخش على الـ main()

Case 2: Bootloader

فى حالة انك عندك (OS) او حاجة

انت محتاج من الاول تروح تعمل init للـ H.W الى موجود عند

كود الـ (OS) بنفس الطريقة الى فاتت ،

بعد ما يخلص يرجع يعمل الـ Boot sequence العادى

دى jump على الـ main

8) to write a Bare-matel sw

You need:

- startup.s
- linkerscript
- C Code files
- makefile

Cross tool chain

9) simple startup.s

```
.grobal reset
reset:
        ldr sp, = stack_top
        bl  main
stop: b stop
```

10) simple linker script.

~~Entry~~ ENTRY (reset)

```
MEMORY
{
mem (rwx) : ORIGIN = 0X0 , LENGTH = 64M
}
SECTIONS
{
 . = 0X10000
 .startup . :
 {
 startup.o (.text)
 } > mem
 .text :
 {
 *(.text) *(.rodata)
 } > mem
 .data ~~ß~~ :
 {
  *(.data)
 } > mem
 .bss
 {
  *(.bss)
 }> mem
 . = . + 0X1000
 stack_top = . ; }
```