**Mastering Embedded System Diploma**

**Embedded C- Lab 2: Write Bare metal SW on ARM Cortex-M3**

**32-bit microcontroller STM32F103C8T6 chip form scratch**

**A blue background with text and a diagram

Description automatically generated**

|  |
| --- |
| by: |
| Ehab Mohamed Abdelhamed |

|  |
| --- |
| Under the supervision of: |
| Eng. Keroles Shenouda |

**Content of lab:**

1) Makefile 2) C code files 3) Startup 4) Linker script 5) Simulation

**1) Makefile**

Makefile used to automate building process and minimize building time when we change some files.

**Consist of:**

(1) TARGET: (2) DEPENDENCY LIST

Command1………….

Command2…………. (3) RULES

**Notes**: 1) Make variable

Variable= value > to read it $(Variable)

2) $<: dependencies , $@: target 3) (Generic) Rules %

4) $(wildcard \*.c) every file end with .c 5) $(var:.o=.c) every file.o >> file.c

**Dependency tree for building process**

File.bin >> File.elf >> files.o + startup.o , Files.o >> files.c , Startup.o >> startup.s

**Makefile Configuration**

**CC**= tool chain name (arm-none-eapi-) **CF**= compiler flags (-g -mcpu=)

**INFS**= include files (-I .<folder path>) **LIBS**=libraries object files

**SRC**=source files (files.c) **As**=assembly files (file.s)

**OBJ**=object files (file.o) **NAME**=Output file name

A screenshot of a computer

Description automatically generated

**2) C code:**

We will Write c code for toggle led at STM32F103C8T6 board at pin 13 of port A:

We will access registers in two modules.

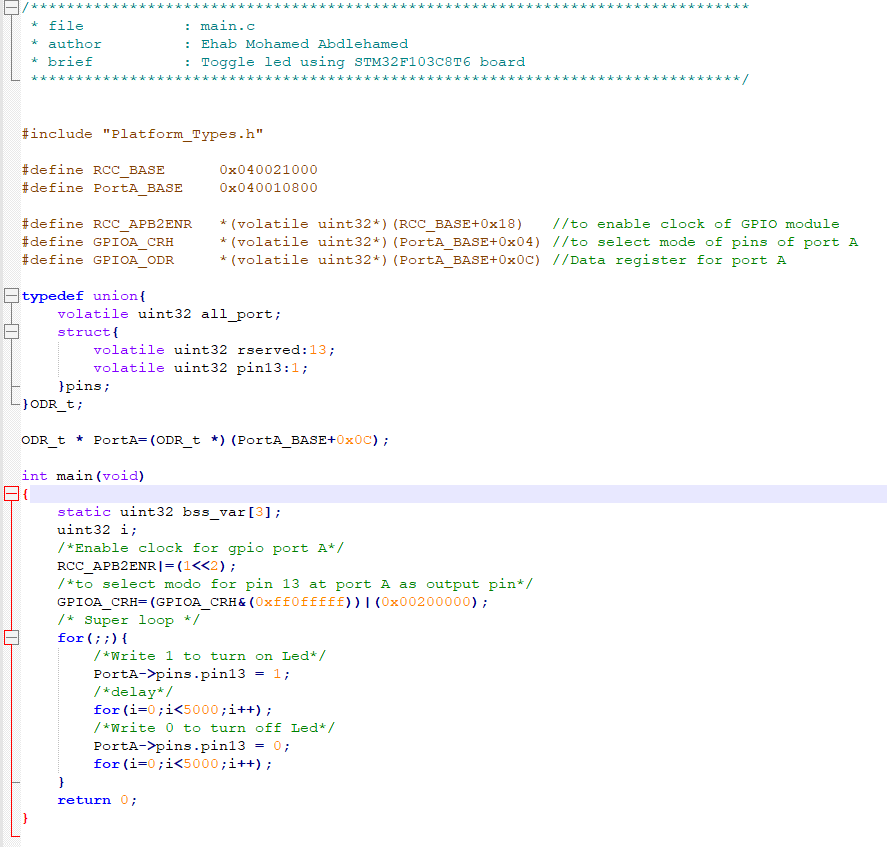
(1) RCC at base address 0x40021000.

Register APB2ENR at offset 0x18, write 1 at bit 2 to enable clock for Port A

2) GPIO Port A at base address 0x40010800

1) Register CRH at offset 0x04, write num 2 at [20 -23] bits to select output pin mode

2) Register ODR at offset 0x0c, data register for port A



**3) Startup:**

Startup is the code that runs before main function to do basic initialization of CPU and memory.

Startup code actins:

1) Disable interrupts

2) Create vector table for microcontroller.

3) Copy .data section that contains initialized data form ram to rom.

4) Reserve .bss section in ram and initialize it by zero.

5) initialize stack pointer by address of top stack.

6) Create and initialize heap (optional).

7) Enable interrupts.

8) jump to main function.

1- Startup.s

A screenshot of a computer

Description automatically generated

**2- Full C Startup file (Startup.c)**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**4) Linker\_Script.ld**

Linker script file describes the memory resources and memory map of the target microcontroller like number of memory and start address and size of each memory.

Define sections and start address of each section and define stack top address.

From linker script we can load section in specific address as reset section in entry point of CPU.

A screenshot of a computer

Description automatically generated

**5) Simulation**

**A screenshot of a computer

Description automatically generated**

**Binary Analysis**

**#Symbols**

1) main.o symbols:

A black screen with yellow and purple text

Description automatically generated

2) startup. o symbols

A screenshot of a computer

Description automatically generated

4) Ehab\_lab2.elf symbols

A screenshot of a computer

Description automatically generated

**#Sections**

**.text** Start\_add=0x08000000 Size=0x158 End\_add=0x08000158

**.data** Start\_add=0x20000000 Size=0x04 End\_add=0x20000004

**.bss** Start\_add=0x20000004 Size=0x0c End\_add=0x20000010

**.stack** Start\_add=0x20001010 Size=0x1000 End\_add=0x20000010

A screenshot of a computer screen

Description automatically generated