Embedded C

Assignment lesson 3

-Toggle led on Arm_cortex m3 32-bit stm32f103c6 chip :

-Write codes

1-main.c

main.c

```c
#include "Header_Platform.h"

#define RCC_BASE    0x40021000
#define PORTA_BASE  0x40010800

#define APB2ENB             *((vuint32_t *)(RCC_BASE + 0x18))
#define GPIOA_CRH           *((vuint32_t *)(PORTA_BASE + 0x04))
#define GPIOA_ODR           *((vuint32_t *)(PORTA_BASE + 0x0c))

uint8_t g_variables[]={1,2,3};
uint8_t const const_variables[]={1,2,3};
uint32_t bss_var[7];

int main(void)
{
    APB2ENB |= 1<<2;
    GPIOA_CRH &= 0xff0fffff;
    GPIOA_CRH |= 0x00200000;

    while(1)
    {
        vuint32_t i;
        GPIOA_ODR |= 1<<13;
        for(i=0; i<5000; i++);      //delay
        GPIOA_ODR &= ~(1<<13);
        for(i=0; i<5000; i++);      //delay
    }
    return 0;
}
```

2-Headers_platform.h

```c
/*
 * Header_Platform.h
 *
 *  Created on: Mar 30, 2021
 *      Author: mostafa
 */

#ifndef HEADER_PLATFORM_H_
#define HEADER_PLATFORM_H_


#define CPU_TYPE        CPU_TYPE_32
#define CPU_BIT_ORDER   MSB_FIRST
#define CPU_BYTE_ORDER  HIGH_BYTE_FIRST

typedef unsigned char       uint8_t;
typedef signed char         sint8_t;
typedef unsigned short      uint16_t;
typedef signed short        sint16_t;
typedef unsigned int        uint32_t;
typedef signed int          sint32_t;
typedef unsigned long long  uint64_t;
typedef signed long long    sint64_t;
typedef float               float32_t;
typedef double              float64_t;

typedef volatile unsigned char          vuint8_t;
typedef volatile signed char            vsint8_t;

typedef volatile unsigned short         vuint16_t;
typedef volatile signed short           vsint16_t;

typedef volatile unsigned int           vuint32_t;
typedef volatile signed int             vsint32_t;

typedef volatile unsigned long long     vuint64_t;
typedef volatile signed long long       vsint64_t;

typedef volatile float                  vfloat32_t;
typedef volatile double                 vfloat64_t;


#endif /* HEADER_PLATFORM_H_ */
```

3-startup.c

startup.c

```c
//startup.c
//Eng:Mostafa Besher

#include <stdint.h>
//prototypes
extern int main(void);
void Default_handler();
void Reset_handler();
void NMI_handler()__attribute__((weak,alias("Default_handler")));;
void HARD_FAULT_handler()__attribute__((weak,alias("Default_handler")));;
void MM_handler()__attribute__((weak,alias("Default_handler")));;
void BUS_handler()__attribute__((weak,alias("Default_handler")));;
void USAGE_FAULT_handler()__attribute__((weak,alias("Default_handler")));;

//declaration of symbols
extern uint32_t _stack_top;
extern uint32_t _E_text;
extern uint32_t _S_data;
extern uint32_t _E_data;
extern uint32_t _S_bss;
extern uint32_t _E_bss;

//.vectors section
uint32_t vectors[] __attribute__((section(".vectors"))) = {

    (uint32_t)&_stack_top,
    (uint32_t)&Reset_handler,
    (uint32_t)&NMI_handler,
    (uint32_t)&HARD_FAULT_handler,
    (uint32_t)&MM_handler,
    (uint32_t)&BUS_handler,
    (uint32_t)&USAGE_FAULT_handler
};

void Reset_handler()
{
    int i = 0;
    //copy .data from flash to sram
    uint32_t Data_size = (unsigned char *)&_E_data - (unsigned char *)&_S_data;
    unsigned char *p_src = (unsigned char *)&_E_text;   //starting address of .data in flash
    unsigned char *p_dst = (unsigned char *)&_S_data;   //starting address of .data in sram

```

```c
void Reset_handler()
{
    int i = 0;
    //copy .data from flash to sram
    uint32_t Data_size = (unsigned char *)&_E_data - (unsigned char *)&_S_data;
    unsigned char *p_src = (unsigned char *)&_E_text;   //starting address of .data in flash
    unsigned char *p_dst = (unsigned char *)&_S_data;   //starting address of .data in sram

    for(i = 0; i < Data_size; i++)
    {
        *((unsigned char *)p_dst++) = *((unsigned char *)p_src++);
    }

    //initialize .bss with zero
    uint32_t bss_size = (unsigned char *)&_E_bss - (unsigned char *)&_S_bss;
    p_dst = (unsigned char *)&_S_bss;

    for(i = 0; i < bss_size; i++)
    {
        *((unsigned char *)p_dst++) = (unsigned char)0;
    }

    //jump to main
    main();
}

void Default_handler()
{
    Reset_handler();
}
```

5-linker_script.ld

```
/* linker_script cortexM3
Eng.Mostafa_Besher
*/


MEMORY
{
    FLASH(RX): ORIGIN = 0x08000000, LENGTH = 128k
    SRAM(RWX): ORIGIN = 0x20000000, LENGTH = 20k
}

SECTIONS
{
    .text :
    {
        *(.vectors*)
        *(.text*)
        *(.rodata)
        . = ALIGN(4);
        _E_text = .;
    }>FLASH
    .data :
    {
        _S_data = .;
        *(.data)
        . = ALIGN(4);
        _E_data = .;
    }>SRAM AT> FLASH
    .bss :
    {
        _S_bss = .;
        *(.bss)
        _E_bss = .;

        . = ALIGN(4);
        . = . + 0x1000;
        _stack_top = .;
    }>SRAM


}
```

6-Makefile

Makefile

```
#@copyright : Mostafa_Besher
CC=arm-none-eabi-
CFLAGS=-mcpu=cortex-m3 -gdwarf-2
INCS= -I .
LIBS=
SRC= $(wildcard *.c)
OBJ= $(SRC:.c=.o)
As= $(wildcard *.s)
AsOBJ= $(As:.s=.o)
Project_name= toggle_led_lab2


all: $(Project_name).bin
    @echo "=======Build is complete======="

%.o: %.c
    $(CC)gcc.exe -c $(CFLAGS) $(INCS) $< -o $@

$(Project_name).elf: $(OBJ) $(AsOBJ)
    $(CC)ld.exe -T linker_script.ld $(LIBS) $(OBJ) $(AsOBJ) -o $(Project_name).elf -Map=map_file.map

$(Project_name).bin: $(Project_name).elf
    $(CC)objcopy.exe -O binary $< $@

clean_all:
    rm *.o *.elf *.bin
clean:
    rm *.elf *.bin


```

-Get object_files using Makefile

```
mostafa@DESKTOP-6K5T62N MINGW32 /d/Embedded_Diploma/assignments/2_Embedded c/lesson 3/lab 2
$ make
arm-none-eabi-gcc.exe -c -mcpu=cortex-m3 -gdwarf-2 -I . main.c -o main.o
arm-none-eabi-gcc.exe -c -mcpu=cortex-m3 -gdwarf-2 -I . startup.c -o startup.o
arm-none-eabi-ld.exe -T linker_script.ld  main.o startup.o  -o toggle_led_lab2.elf -Map=map_file.map
arm-none-eabi-objcopy.exe -O binary toggle_led_lab2.elf toggle_led_lab2.bin
```

-Show Symbols For :

1-main.o

```
mostafa@DESKTOP-6K5T62N MINGW32 /d/Embedded_Diploma/assignments/2_Embedded c/lesson 3/lab 2
$ arm-none-eabi-nm.exe main.o
0000001c C bss_var
00000000 R const_variables
00000000 D g_variables
00000000 T main
```

2- startup.o

```
mostafa@DESKTOP-6K5T62N MINGW32 /d/Embedded_Diploma/assignments/2_Embedded c/lesson 3/lab 2
$ arm-none-eabi-nm.exe startup.o
         U _E_bss
         U _E_data
         U _E_text
         U _S_bss
         U _S_data
         U _stack_top
00000088 W BUS_handler
00000088 T Default_handler
00000088 W HARD_FAULT_handler
         U main
00000088 W MM_handler
00000088 W NMI_handler
00000000 T Reset_handler
00000088 W USAGE_FAULT_handler
00000000 D vectors
```

3-toggle_led_lab2.elf

```
mostafa@DESKTOP-6K5T62N MINGW32 /d/Embedded_Diploma/assignments/2_Embedded c/lesson 3/lab 2
$ arm-none-eabi-nm.exe toggle_led_lab2.elf
20000004 B _E_bss
20000004 D _E_data
08000130 T _E_text
20000004 B _S_bss
20000000 D _S_data
20001004 B _stack_top
20001004 B bss_var
08000120 W BUS_handler
0800012c T const_variables
08000120 T Default_handler
20000000 D g_variables
08000120 W HARD_FAULT_handler
0800001c T main
08000120 W MM_handler
08000120 W NMI_handler
08000098 T Reset_handler
08000120 W USAGE_FAULT_handler
08000000 T vectors
```

-Proteus simulation