

LAB 2

- Main.c code

```
1 // @Copyright: Arasny
2
3 #include "stdint.h"
4 #define RCC_BASE      0x40021000
5 #define GPIOA_BASE    0x40010800
6 #define RCC_APB2ENR   *(volatile uint32_t *) (RCC_BASE + 0x18)
7 #define GPIOA_CRH     *(volatile uint32_t *) (GPIOA_BASE + 0x04)
8 #define GPIOA_ODR     *(volatile uint32_t *) (GPIOA_BASE + 0x0C)
9
10 unsigned char g_variabled[3]={1,2,3};
11 unsigned char const const_var[3]={1,2,3};
12 unsigned char bss_var[3];
13
14 int main(void)
15 {
16     int i;
17     RCC_APB2ENR |= (1<<2);
18     GPIOA_CRH   &= 0xFF0FFFFFFF;
19     GPIOA_CRH   |= 0x00200000;
20     while(1)
21     {
22         GPIOA_ODR |= 1<<13;
23         for(i=0;i<5000;i++);
24         GPIOA_ODR &= ~(1<<13);
25         for(i=0;i<5000;i++);
26     }
27 }
```

• Startup.c code

```
1  //startup.c
2  //End.Arsany
3  #include"stdint.h"
4  extern int main();
5  void Reset_Handler();
6  void Default_Handler()
7  {
8      Reset_Handler();
9  }
10 void NMI_Handler() __attribute__((weak,alias("Default_Handler")));
11 void H_Fault_Handler() __attribute__((weak,alias("Default_Handler")));
12 void MM_Fault_Handler() __attribute__((weak,alias("Default_Handler")));
13 void Bus_Fault_Handler() __attribute__((weak,alias("Default_Handler")));
14 void Usage_Fault_Handler() __attribute__((weak,alias("Default_Handler")));
15
16 extern unsigned int _stack_top;
17
18 uint32_t vectors[] __attribute__((section(".vectors"))) = {
19     (uint32_t) &_stack_top,
20     (uint32_t) &Reset_Handler,
21     (uint32_t) &NMI_Handler,
22     (uint32_t) &H_Fault_Handler,
23     (uint32_t) &MM_Fault_Handler,
24     (uint32_t) &Bus_Fault_Handler,
25     (uint32_t) &Usage_Fault_Handler
26 };
27
28
29 extern unsigned int _S_DATA;
30 extern unsigned int _E_DATA;
31 extern unsigned int _S_bss;
32 extern unsigned int _E_bss;
33 extern unsigned int _E_text;
34 int i;
35
36
37 void Reset_Handler()
38 {
39     unsigned int DATA_SIZE = (unsigned char*)&_E_DATA - (unsigned char*)&_S_DATA;
40     unsigned char* P_src = (unsigned char*)&_E_text;
41     unsigned char* P_dst = (unsigned char*)&_S_DATA;
42     for( i=0;i<DATA_SIZE;i++)
43     {
44         *((unsigned char*)P_dst++)=*((unsigned char*)P_src);
45     }
46     unsigned int bss_SIZE = (unsigned char*)&_E_bss - (unsigned char*)&_S_bss;
47     P_dst=(unsigned char*)&_S_bss;
48     for( i=0;i<DATA_SIZE;i++)
49     {
50         *((unsigned char*)P_dst++)=(unsigned char*)0;
51     }
52     main();
53 }
```

- Linker_script.ld

```
1 MEMORY
2 {
3     flash(RX) : ORIGIN = 0x08000000, LENGTH = 128k
4     sram(RWX) : ORIGIN = 0x20000000, LENGTH = 20k
5 }
6
7 SECTIONS
8 {
9     .text : {
10         *(.vectors*)
11         *(.text*)
12         *(.rodata*)
13         _E_text = . ;
14     }> flash
15
16     .data : {
17         _S_DATA = . ;
18         *(.data)
19         _E_DATA = . ;
20     }> sram AT> flash
21
22     .bss : {
23         _S_bss = . ;
24         *(.bss)
25         _E_bss = . ;
26         . = ALIGN(4) ;
27         . = . + 0x1000 ;
28         _stack_top = . ;
29     }>sram
30 }
31 }
```

- Modifying the makefile

```
1 #@copyright : Arsany
2 CC=arm-none-eabi-
3 CFLAGS=-mthumb -mcpu=cortex-m3 -gdwarf-2
4 INCS=-I .
5 LIBS=
6 SRC = $(wildcard *.c)
7 OBJ = $(SRC:.c=.o)
8 AS = $(wildcard *.s)
9 ASOBJ = $(AS:.s=.o)
10
11 Project_name=LED_TOGGLE
12
13 all: $(Project_name).bin
14     @echo "=====Build is Done=====
15
16 %.o: %.c
17     $(CC)gcc.exe -c $(INCS) $(CFLAGS) $< -o $@
18
19 $(Project_name).elf: $(OBJ) $(ASOBJ)
20     $(CC)ld.exe -T linker_script.ld $(LIBS) $(OBJ) $(ASOBJ) -o $@ -Map=Map_file.map
21
22 $(Project_name).bin: $(Project_name).elf
23     $(CC)objcopy.exe -O binary $< $@
24
25 clean_all:
26     rm *.o *.elf *.bin *.map
27     @echo "=====CLEAN=====
28
29 clean:
30     rm *.elf *.bin *.map
```

- Building Process

```
MINGW64:/e/Git/Github_Repo/Unit_3_Embedded_C/Lec_3/Lab_2

Arshy@Arsany MINGW64 /e/Git/Github_Repo/Unit_3_Embedded_C/Lec_3/Lab_2 (master)
$ make
arm-none-eabi-gcc.exe -c -I . -mthumb -mcpu=cortex-m3 -gdwarf-2 main.c -o main.o
arm-none-eabi-gcc.exe -c -I . -mthumb -mcpu=cortex-m3 -gdwarf-2 startup.c -o startup.o
startup.c: In function 'Reset_Handler':
startup.c:29: warning: assignment makes integer from pointer without a cast [enabled by default]
arm-none-eabi-ld.exe -T linker_script.ld main.o startup.o -o LED_TOGGLE.elf -Map=Map_file.map
arm-none-eabi-objcopy.exe -O binary LED_TOGGLE.elf LED_TOGGLE.bin
=====Build is Done=====
```

- Symbols

```
MINGW64:/e/Git/Github_Repo/Unit_3_Embedded_C/Lec_3/Lab_2

Arshy@Arsany MINGW64 /e/Git/Github_Repo/Unit_3_Embedded_C/Lec_3/Lab_2 (master)
$ arm-none-eabi-nm.exe main.o
00000003 C bss_var
00000000 R const_var
00000000 D g_variabld
00000000 T main

Arshy@Arsany MINGW64 /e/Git/Github_Repo/Unit_3_Embedded_C/Lec_3/Lab_2 (master)
$ 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
00000000 W Bus_Fault_Handler
00000000 T Default_Handler
00000000 W H_Fault_Handler
00000004 C i
                 U main
00000000 W MM_Fault_Handler
00000000 W NMI_Handler
0000000c T Reset_Handler
00000000 W Usage_Fault_Handler
00000000 D vectors

Arshy@Arsany MINGW64 /e/Git/Github_Repo/Unit_3_Embedded_C/Lec_3/Lab_2 (master)
$ arm-none-eabi-nm.exe LED_TOGGLE.elf
20000004 B _E_bss
20000004 D _E_DATA
080001cc T _E_text
20000004 B _S_bss
20000000 D _S_DATA
20001004 B _stack_top
20001004 B bss_var
080000d0 W Bus_Fault_Handler
080001c8 T const_var
080000d0 T Default_Handler
20000000 D g_variabld
080000d0 W H_Fault_Handler
20001008 B i
0800001c T main
080000d0 W MM_Fault_Handler
080000d0 W NMI_Handler
080000dc T Reset_Handler
080000d0 W Usage_Fault_Handler
08000000 T vectors

Arshy@Arsany MINGW64 /e/Git/Github_Repo/Unit_3_Embedded_C/Lec_3/Lab_2 (master)
$
```

• Testing

The image displays two screenshots of the Keil IDE, illustrating the testing process of a program. The top screenshot shows the initial state where the program is not running. The bottom screenshot shows the program running, with the LED indicator lit, indicating a successful test.

CM3 Registers - U1

Register	Value	Mode	Thread
PC	08000098	Privileged	
Pr1	00.200	Mode	Thread
R0	00000000	R7	20000FDC
R1	00000000	R8	00000000
R2	00000604	R9	00000000
R3	00001387	R10	00000000
R4	00000000	R11	00000000
R5	00000000	R12	00000000
R6	00000000	LR	080001C1
MSP	20000FDC	PSP	00000000
IRQ	0		
APSR	CNQVZ	EPSR	01000000
	10000	1c11t	000.00000

CM3 Source Code - U1

```

main.c
-----//@Copyright: Arasny
-----
#include "stdint.h"
#define RCC_BASE 0x40021000
#define GPIOA_BASE 0x40010800
#define RCC_APB2ENR (*(volatile uint32_t *) (RCC_BASE + 0x18))
#define GPIOA_CRH (*(volatile uint32_t *) (GPIOA_BASE + 0x04))
#define GPIOA_ODR (*(volatile uint32_t *) (GPIOA_BASE + 0x0C))

unsigned char g_variabled[3]={1,2,3};
unsigned char const const_var[3]={1,2,3};
unsigned char bss_var[3];

int main(void)
{
    int i;
    RCC_APB2ENR |= (1<<2);
    GPIOA_CRH |= 0xFF0FFFFFFF;
    GPIOA_CRH |= 0x00200000;
    while(1)
    {
        GPIOA_ODR |= 1<<13;
        for(i=0;i<5000;i++);
        GPIOA_CRH |= 1<<13;
        for(i=0;i<5000;i++);
    }
}
    
```

The top screenshot shows the initial state of the program. The LED indicator is not lit, and the program is not running. The bottom screenshot shows the program running, with the LED indicator lit, indicating a successful test.