

Report

- **Lab3:**

- Board name : STM32f103c8t6 arm-cortex-m4 based .
- Flash starts with 0x00000000
- Sram starts with 0x20000000

1.Main .c

```
1
2 //Eng.Ziad
3
4
5 #define SYSCAL_RCGC2_R      (*((volatile unsigned Long*)0x400FE108))
6 #define GPIO_PORTF_DIR_R    (*((volatile unsigned Long*)0x40025400))
7 #define GPIO_PORTF_DEN_R    (*((volatile unsigned Long*)0x4002551C))
8 #define GPIO_PORTF_DATA_R   (*((volatile unsigned Long*)0x400253FC))
9 #define PIN_SET             (1<<3)
10
11
12 int main()
13 {
14     volatile unsigned long delay;
15
16     SYSCAL_RCGC2_R |= 0x20;
17
18     //to make sure GPIO is up and running
19
20     for(delay=0;delay<200;delay++);
21
22
23     GPIO_PORTF_DIR_R |= PIN_SET ;    //Dir is output for pin 3 port F
24
25     GPIO_PORTF_DEN_R |= PIN_SET ;
26
27     while(1)
28     {
29         GPIO_PORTF_DATA_R |= PIN_SET ;
30
31         for(delay=0;delay<200000;delay++);
32
33         GPIO_PORTF_DATA_R &= ~(PIN_SET) ;
34
35         for(delay=0;delay<200000;delay++);
36
37     }
38
39
40     return 0;
41
42 }
43
```

2.Startup.c

```
1 //startup.c
2 //Eng.Ziad
3
4 #include <stdint.h>
5 extern int main(void);
6
7 void Reset_Handler(void);
8 void Default_handler();
9
10
11 //to make it easy to overrid & all have the same handler to minimize the size
12
13 void NMI_Handler() __attribute__((weak,alias("Default_handler")));
14 void H_fault_Handler() __attribute__((weak,alias("Default_handler")));
15
16
17 //booking 1024 bytes located by .bss through uninitialized array of int 256 elements (256*4=1024)
18
19
20
21 static unsigned long Stack_top[256] ; //static to make the scope for this file only
22
23
24 void (*const g_P_fn_Vectors[])(void) __attribute__((section(".Vectors"))) =
25 {
26     (void (*)()) ((unsigned long)Stack_top + sizeof(Stack_top)),
27     &Reset_Handler,
28     &NMI_Handler,
29     &H_fault_Handler
30 };
31 extern unsigned int _S_DATA;
32 extern unsigned int _E_DATA;
33 extern unsigned int _S_bss;
34 extern unsigned int _E_bss;
35 extern unsigned int _E_text;
36
37 void Reset_Handler (void)
38 {
39     volatile int i;
40     //copy data from ROM to RAM
41     unsigned int DATA_size =(unsigned char*)&_E_DATA - (unsigned char*)&_S_DATA;
42     unsigned char* P_src = (unsigned char*)&_E_text;
43     unsigned char* P_dst = (unsigned char*)&_S_DATA;
44
45     for(i=0;i<DATA_size; i++)
46     {
47         *((unsigned char*)P_dst++)= *((unsigned char*)P_src++);
48     }
49
50
51     // initialize the .bss with zero
52     unsigned int bss_size = (unsigned char*)&_E_bss - (unsigned char*)&_S_bss ;
53     P_dst = (unsigned char*)&_S_bss ;
54     for(i=0 ; i<bss_size ; i++)
55     {
56         *((unsigned char*)P_dst++)=(unsigned char)0;
57     }
58
59     // jump to main
60
61     main();
62 }
63 void Default_handler()
64 {
65     Reset_Handler();
66 }
```

3.Linker script

```
1  /* linker script CortexM4
2  Eng.Ziad
3  */
4
5  MEMORY
6  {
7      flash(RX) : ORIGIN = 0x00000000, LENGTH = 512M
8      sram(RWX) : ORIGIN = 0x20000000, LENGTH = 512M
9  }
10
11  SECTIONS
12  {
13      .text : {
14          *(.Vectors*)
15          *(.text*)
16          *(.rodata)
17          _E_text = . ;
18      }> flash
19
20      .data : {
21          _S_DATA = . ;
22          *(.data)
23          _E_DATA = . ;
24      }> sram AT> flash
25
26      .bss : {
27          _S_bss = . ;
28          *(.bss)
29          . = ALIGN(4);
30          _E_bss = . ;
31      }> sram
32
33  }
34
35
36 }
```

4.Make file

```
1  #@ copyright : Eng.Ziad
2
3  CC=arm-none-eabi-
4  CFLAGS= -mcpu=cortex-m4 -mthumb -gdwarf-2 -g
5  INCS=-I .
6  LIBS=
7  SRC = $(wildcard *.c)
8  OBJ = $(SRC:.c=.o)
9  AS = $(wildcard *.s)
10 ASOBJ = $(AS:.s=.o)
11
12 project_name=unit3_lab4_cortexM4
13
14 all: $(project_name).bin
15     @echo "-----all build is done-----"
16
17 %.o: %.c
18     $(CC)gcc.exe -c $(CFLAGS) $(INCS) $< -o $@
19
20 %.o: %.s
21     $(CC)as.exe $(CFLAGS) $< -o $@
22
23 $(project_name).elf: $(OBJ) $(ASOBJ)
24     $(CC)ld.exe -T linker_script.ld -Map=map_file.map $(OBJ) $(ASOBJ) -o $@
25     cp $(project_name).elf $(project_name).axf
26
27 $(project_name).bin: $(project_name).elf
28     $(CC)objcopy.exe -O binary $< $@
29
30
31 clean_all:
32     rm *.o *.elf *.bin
33 clean:
34     rm *.bin *.elf
35
36
37
38
```

-Map file

```
1
2 Memory Configuration
3
4 Name          Origin          Length          Attributes
5 flash         0x00000000        0x20000000      xrw
6 sram          0x20000000        0x20000000      xrw
7 *default*     0x00000000        0xffffffff
8
9 Linker script and memory map
10
11
12 .text         0x00000000        0x1a4
13 *(.Vectors*)
14 .Vectors      0x00000000        0x10 startup.o
15              0x00000000        g_P_fn_Vectors
16 *(.text*)
17 .text         0x00000010        0xd4 main.o
18              0x00000010        main
19 .text         0x000000e4        0xc0 startup.o
20              0x000000e4        Reset_Handler
21              0x00000198        H_fault_Handler
22              0x00000198        Default_handler
23              0x00000198        NMI_Handler
24 *(.rodata)
25              0x000001a4        _E_text = .
26
27 .glue_7       0x000001a4        0x0
28 .glue_7       0x00000000        0x0 linker stubs
29
30 .glue_7t      0x000001a4        0x0
31 .glue_7t      0x00000000        0x0 linker stubs
32
33 .vfp11_veneer 0x000001a4        0x0
34 .vfp11_veneer 0x00000000        0x0 linker stubs
35
36 .v4_bx        0x000001a4        0x0
37 .v4_bx        0x00000000        0x0 linker stubs
38
39 .iplt         0x000001a4        0x0
40 .iplt         0x00000000        0x0 main.o
41
42 .rel.dyn      0x000001a4        0x0
43 .rel.iplt     0x00000000        0x0 main.o
44
45 .data         0x20000000        0x0 load address 0x000001a4
46              0x20000000        _S_DATA = .
47 *(.data)
48 .data         0x20000000        0x0 main.o
49 .data         0x20000000        0x0 startup.o
50              0x20000000        _E_DATA = .
51
52 .igot.plt     0x20000000        0x0 load address 0x000001a4
53 .igot.plt     0x00000000        0x0 main.o
54
55 .bss          0x20000000        0x400 load address 0x000001a4
56              0x20000000        _S_bss = .
57 *(.bss)
58 .bss          0x20000000        0x0 main.o
59 .bss          0x20000000        0x400 startup.o
60              0x20000400        . = ALIGN (0x4)
61              0x20000400        _E_bss = .
62 LOAD main.o
63 LOAD startup.o
64 OUTPUT(unit3_lab4_cortexM4.elf elf32-littlearm)
65
```

Handwritten notes:

- Red arrow pointing to `g_P_fn_Vectors` in line 15.
- Red bracket around lines 58-61.
- Red text: $400 = 1024$ with an arrow pointing to the `0x400` value in line 59.

-Keil Uvision Simulation

CAUsers\ziade\Downloads\Compressed\Keil_uvision_unit3_lab4_project\Keil_uvision_unit3_lab4_project\uvprojx - µVision [Non-Com]

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

Registers

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x000040AF
R3	0x0000003F
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x20000300
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000300
R14 (LR)	0x00000191
R15 (PC)	0x000000A2
xPSR	0x81000000
Banked	
System	
Internal	
Mode	Thread
Privilege	Privileged
Stack	MSP
States	3395871019
Sec	212.24193869
FPU	

Logic Analyzer

Setup... Load... Save... Min Time 104.9776 s Max Time 212.2419 s Grid 2 s In Out All Auto Undo Stop Clear Prev Next Code Trace Signal Info Amplitude Timestamps

PORTF

Disassembly Logic Analyzer

main.c

```
24 GPIO_PORTF_DEN_R |= PIN_SET ;
25
26 while(1)
27 {
28     GPIO_PORTF_DATA_R |= PIN_SET ;
29     for(delay=0;delay<200000;delay++);
30
31     GPIO_PORTF_DATA_R &= ~(PIN_SET) ;
32     for(delay=0;delay<200000;delay++);
33
34 }
35
36 return 0;
```

Texas edX Lab 2

Port F Hardware

TM4C123

SW1 PF4 PF3 PF2 PF1 LED LED Green

Port F Registers

Register	Value
DATA	0x19
DIR	0x08
DEN	0x08
PUR	0x00
PDR	0x00
RCGC2	0x00000020
LOCK	0x01
CR	0x1E

Grading Controls

Number from edX: Grade Score: 0 Copy this to edX

GPIOF

Property	Value
DATA	0x08080819
DIR	0x08080808
IS	0x00000000
IBE	0x00000000
IEV	0x00000000
IM	0
RIS	0
MIS	0
ICR	0
AFSEL	0x00000000
DR2R	0xFFFFFFFF
DR4R	0x00000000
DR8R	0x00000000
ODR	0x00000000
PUR	0x00000000
PDR	0x00000000
DEN	0x08080808
SLR	0x00000000
LOCK	0x01010101
CR	0x1E1E1E1E
AMSEL	0x00000000

Command

Load "D:\Lab_3\unit2_lab4_cortexM4.axf"

LA PORTF

main 0x0000008C int f0

delay 0x000040AF auto - uint

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE

Call Stack = Locals Memory 1

Simulation F1: 212.24193869 sec 1-13 C-1 LAP: NIML CFSI NUD: BAW