Unit3_Lesson(4)

write codes

main.c

```c
#define SYSCTL_RCGC2_R          (*((volatile unsigned long*)0x400FE108))
#define GPIO_PORTF_DATA_R       (*((volatile unsigned long*)0x400253FC))
#define GPIO_PORTF_DIR_R        (*((volatile unsigned long*)0x40025400))
#define GPIO_PORTF_DEN_R        (*((volatile unsigned long*)0x4002551C))

int main(void)
{
    volatile unsigned long delay_counter;
    SYSCTL_RCGC2_R = 0x00000020;
    for(delay_counter = 0; delay_counter <200; delay_counter++);
    GPIO_PORTF_DIR_R |=  (1<<3);
    GPIO_PORTF_DEN_R  |= (1<<3);
    while(1)
    {
        GPIO_PORTF_DATA_R |= (1<<3);
        for(delay_counter = 0; delay_counter <200; delay_counter++);
        GPIO_PORTF_DATA_R &= ~(1<<3);
        for(delay_counter = 0; delay_counter <200; delay_counter++);
    }
    return 0;
}

```

startUp.c

```c
/* learn-in-depth
Eng: Hazem Abd El-Halim
    */

#include <stdint.h>

extern int main(void);
void Reset_Handler(void);
void Default_Handler()
{
    Reset_Handler();
}
void NMI_Handler(void) __attribute__ ((weak, alias ("Default_Handler")));
void H_Fault_Handler(void) __attribute__ ((weak, alias ("Default_Handler")));


static unsigned long Stack_top[256];

void (* const g_p_fn_vectors[])() __attribute__((section(".vectors")))  ={
(void(*)())  ((unsigned long)Stack_top  + sizeof(Stack_top)),
 &Reset_Handler,
 &NMI_Handler,
 &H_Fault_Handler
};

extern unsigned int _S_DATA;
extern unsigned int _E_DATA;
extern unsigned int _S_bss;
extern unsigned int _E_bss;
extern unsigned int _E_text;

void Reset_Handler(void)
{
    volatile unsigned long conter;
    unsigned int DATA_size = (unsigned char*)&_E_DATA - (unsigned char*)&_S_DATA;
    unsigned char* P_src = (unsigned char*)&_E_text;
    unsigned char* P_dst = (unsigned char*)&_S_DATA;


    for (conter = 0; conter < DATA_size; conter++)
    {
        *((unsigned char*)P_dst++) = *((unsigned char *)P_src++);
    }

    unsigned int bss_size = (unsigned char*)&_E_bss - (unsigned char*)&_S_bss;
    P_dst = (unsigned char*)&_S_bss;
    for(conter = 0; conter< bss_size; conter++)
    {
        *((unsigned char*)P_dst++)= (unsigned char)0;
    }

    main();
}
```

linker_script.ld

```
/* learn-in-depth
Eng: Hazem Abd El-Halim
        */

MEMORY
{
flash(RX) : ORIGIN = 0x00000000 , LENGTH = 512m
sram(RWX) : ORIGIN = 0x20000000 , LENGTH = 512m
}

SECTIONS
{
    .text : {
            *(.vectors*)
            *(.text*)
            *(.rodata)
            _E_text = . ;
    } > flash

    .data : {
            _S_DATA = . ;
            *(.data)
            _E_DATA = . ;
    } >sram AT> flash

    .bss : {
            _S_bss = . ;
            *(.bss*)
            . = ALIGN(4);
            _E_bss = . ;
    } > sram
}
```

Makefile

```
CC=arm-none-eabi-
CFLAGS=-gdwarf-2 -mcpu=cortex-m4 -g
INCS= -I .
LIBS=
SRC=$(wildcard *.c)
OBJ=$(SRC:.c=.o)
As_s=$(wildcard *.s)
As_o=$(As_s:.s=.o)
project_name= toggle_LED_Arm_cortexM4

all:$(project_name).bin

%.o: %.s
    $(CC)as.exe $(CFLAGS) $< -o $@

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

$(project_name).elf: $(OBJ) $(OBJ)
    $(CC)ld.exe -T linker_script.ld  $(LIBS) $(OBJ)   -o  $@     -Map=Map_file.map
    cp $(project_name).elf    $(project_name).axf

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

clear_all:
    rm *.o *.elf *.bin
```

Mapefile.map

```
11
12  .text           0x00000000      0x124
13   *(.vectors*)
14   .vectors        0x00000000       0x10 startup.o
15                   0x00000000                g_p_fn_vectors
16   *(.text*)
17   .text           0x00000010       0x90 startup.o
18                   0x00000010                H_Fault_Handler
19                   0x00000010                Default_Handler
20                   0x00000010                NMI_Handler
21                   0x0000001c                Reset_Handler
22   .text           0x000000a0       0x84 main.o
23                   0x000000a0                main
24   *(.rodata)
25                   0x00000124                _E_text = .
26
```

```
45  .data           0x20000000        0x0 load address 0x00000124
46                   0x20000000                _S_DATA = .
47   *(.data)
48   .data           0x20000000        0x0 startup.o
49   .data           0x20000000        0x0 main.o
50                   0x20000000                _E_DATA = .
51
52  .igot.plt       0x20000000        0x0 load address 0x00000124
53   .igot.plt       0x20000000        0x0 startup.o
54
55  .bss            0x20000000      0x400 load address 0x00000124
56                   0x20000000                _S_bss = .
57   *(.bss*)
58   .bss            0x20000000      0x400 startup.o
59   .bss            0x20000400        0x0 main.o
60                   0x20000400                . = ALIGN (0x4)
61                   0x20000400                _E_bss = .
62  LOAD startup.o
63  LOAD main.o
```

main.o

mingw32-make.exe main.o

startUp.o

mingw32-make.exe startUp.o

To show sections for object_file

main.o

```
$ arm-none-eabi-objdump.exe -h main.o

main.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000084  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000000  00000000  00000000  000000b8  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000b8  2**0
                  ALLOC
  3 .debug_info   00000066  00000000  00000000  000000b8  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 0000005c  00000000  00000000  0000011e  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000038  00000000  00000000  0000017a  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  000001b2  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   00000061  00000000  00000000  000001d2  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    000000a0  00000000  00000000  00000233  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      0000007f  00000000  00000000  000002d3  2**0
                  CONTENTS, READONLY
 10 .debug_frame  0000002c  00000000  00000000  00000354  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
 11 .ARM.attributes 00000033  00000000  00000000  00000380  2**0
                  CONTENTS, READONLY
```

Startup.o

```
$ arm-none-eabi-objdump.exe -h startUp.o

startUp.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000090  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  000000c4  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000400  00000000  00000000  000000c4  2**2
                  ALLOC
  3 .vectors      00000010  00000000  00000000  000000c4  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, DATA
  4 .debug_info   00000188  00000000  00000000  000000d4  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  5 .debug_abbrev 000000c0  00000000  00000000  0000025c  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_loc    0000007c  00000000  00000000  0000031c  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_aranges 00000020  00000000  00000000  00000398  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_line   00000069  00000000  00000000  000003b8  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  9 .debug_str    00000185  00000000  00000000  00000421  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .comment      0000007f  00000000  00000000  000005a6  2**0
                  CONTENTS, READONLY
 11 .debug_frame  00000050  00000000  00000000  00000628  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
 12 .ARM.attributes 00000033  00000000  00000000  00000678  2**0
                  CONTENTS, READONLY
```

To show symbol table for main.o and startUp.o

```
$ arm-none-eabi-nm.exe  main.o
00000000 T main
```

```
$ arm-none-eabi-nm.exe  startUp.o
         U _E_bss
         U _E_DATA
         U _E_text
         U _S_bss
         U _S_DATA
00000000 T Default_Handler
00000000 R g_p_fn_vectors
00000000 W H_Fault_Handler
         U main
00000000 W NMI_Handler
0000000c T Reset_Handler
00000000 b Stack_top
```

use linker_script to get executable_file (toggle_LED_Arm_cortexM3.elf) and Map_file.map

mingw32-make-exe toggle_LED_cortexM3.elf

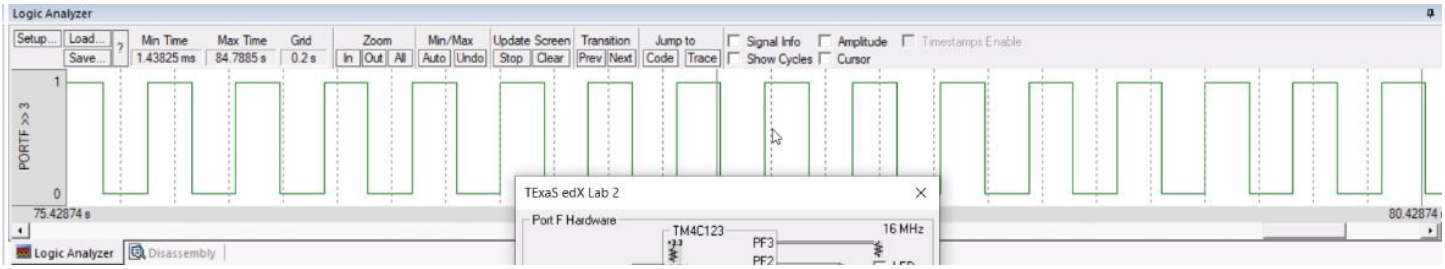
To show sections for toggle_LED_cortexM3.elf

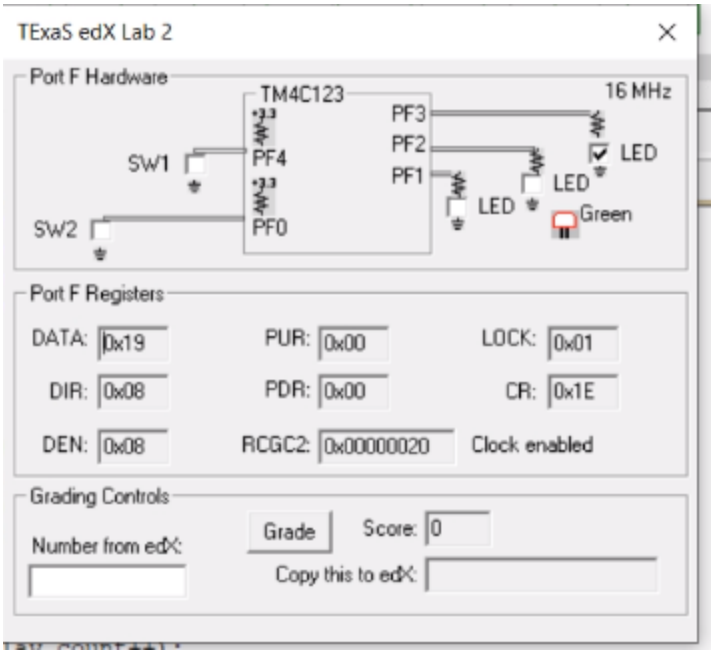
```
$ arm-none-eabi-objdump.exe -h toggle_LED_Arm_cortexM4.elf

toggle_LED_Arm_cortexM4.elf:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000124  00000000  00000000  00010000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .bss          00000400  20000000  00000124  00020000  2**2
                  ALLOC
  2 .debug_info   000001ee  00000000  00000000  00010124  2**0
                  CONTENTS, READONLY, DEBUGGING
  3 .debug_abbrev 0000011c  00000000  00000000  00010312  2**0
                  CONTENTS, READONLY, DEBUGGING
  4 .debug_loc    000000b4  00000000  00000000  0001042e  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_aranges 00000040  00000000  00000000  000104e2  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_line   000000ca  00000000  00000000  00010522  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_str    0000016b  00000000  00000000  000105ec  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .comment      0000007e  00000000  00000000  00010757  2**0
                  CONTENTS, READONLY
  9 .ARM.attributes 00000033  00000000  00000000  000107d5  2**0
                  CONTENTS, READONLY
 10 .debug_frame  0000007c  00000000  00000000  00010808  2**2
                  CONTENTS, READONLY, DEBUGGING
```

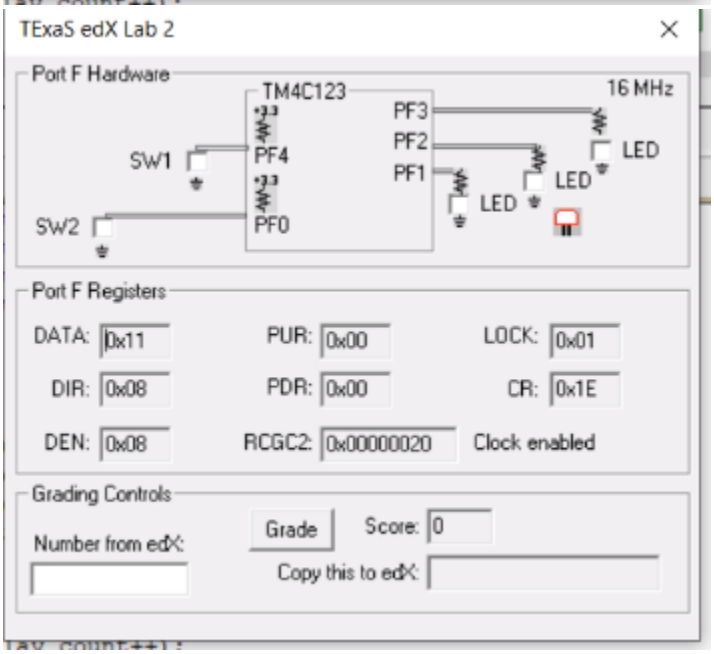
To show symbol table for toggle_LED_cortexM3.elf

```
$ arm-none-eabi-nm.exe  toggle_LED_Arm_cortexM4.elf
20000400 B _E_bss
20000000 T _E_DATA
00000124 T _E_text
20000000 B _S_bss
20000000 T _S_DATA
00000010 T Default_Handler
00000000 T g_p_fn_vectors
00000010 W H_Fault_Handler
000000a0 T main
00000010 W NMI_Handler
0000001c T Reset_Handler
20000000 b Stack_top
```

Keil simulation




TExaS edX Lab 2
Port F Hardware
TM4C123
16 MHz
SW1
SW2
PF4
PF0
PF3
PF2
PF1
LED
LED
LED
Green
Port F Registers
DATA: 0x19
PUR: 0x00
LOCK: 0x01
DIR: 0x08
PDR: 0x00
CR: 0x1E
DEN: 0x08
RCGC2: 0x00000020
Clock enabled
Grading Controls
Number from edX:
Grade
Score: 0
Copy this to edX:


TExaS edX Lab 2
Port F Hardware
TM4C123
16 MHz
SW1
SW2
PF4
PF0
PF3
PF2
PF1
LED
LED
LED
Port F Registers
DATA: 0x11
PUR: 0x00
LOCK: 0x01
DIR: 0x08
PDR: 0x00
CR: 0x1E
DEN: 0x08
RCGC2: 0x00000020
Clock enabled
Grading Controls
Number from edX:
Grade
Score: 0
Copy this to edX: