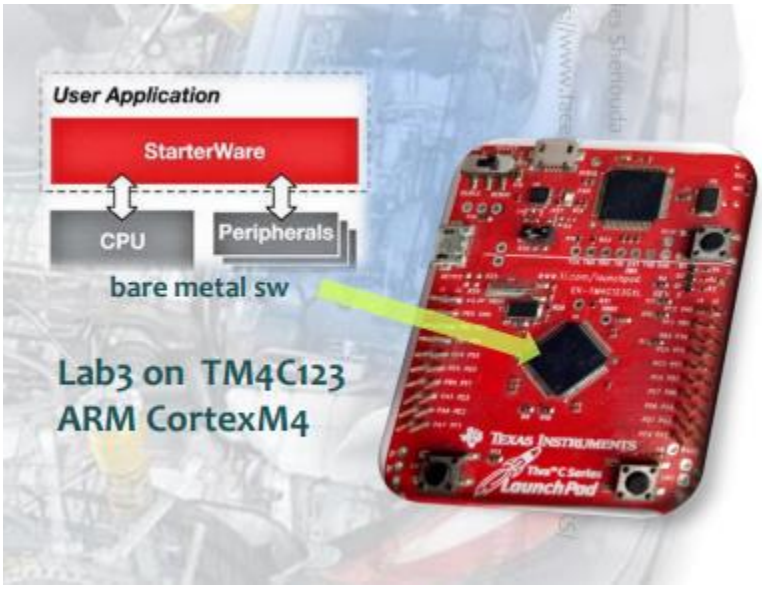
Lab 3

Description:

Create a bare-metal Software to toggle the green LED on TM4C123 ARM CORTEXM4 board.



Files Created:

- main.c
- linker_script.ld
- startup.c
- Makefile

Executable Files:

- learn-in-depth_cortex_m4.elf
- learn-in-depth_cortex_m4.bin
- learn-in-depth_cortex_m4.axf

Analysis Files:

- main.o
- Map_file.map
- startup.o

Main.c:

```c
//learn-in-depth
//Ebram Habib
//Cortex-M4

#include <stdint.h>

#define SYSCTCL_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 ()
{
    volatile unsigned delay_counter;
    SYSCTCL_RCGC2_R = 0x20; //initializing the GPIO
    for(delay_counter=0;delay_counter<200;delay_counter++); // waiting to make sure that the GPIO is up and running

    GPIO_PORTF_DIR_R |= 1<<3; //Seting the pin 3 direction to output
    GPIO_PORTF_DEN_R |= 1<<3; //Enabling PORT F pin 3

    while(1)
    {
        GPIO_PORTF_DATA_R |= 1<<3;
        for(delay_counter=0;delay_counter<200000;delay_counter++); // making the delay big enough to notice the LED blinking
        GPIO_PORTF_DATA_R &= ~(1<<3);
        for(delay_counter=0;delay_counter<200000;delay_counter++);
    }


    return 0;
}
```

Startup.c:

```c
/*      startup.c
    Eng.Ebram Habib
*/

#include <stdint.h>

extern int main(void);
void Reset_Handler();

void Default_Handler(){
    Reset_Handler();
}

void NMI_Handler(void)__attribute__((weak,alias("Default_Handler")));;
void H_Fault_Handler(void)__attribute__((weak,alias("Default_Handler")));;

//booking 1024 bytes located by .bss as an unintialized array of 256 elements (256*4=1024b)
static unsigned long stack_top[256];

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

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

void Reset_Handler(void){
    int i;
    //copy data from ROM to RAM
    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(i = 0; i < DATA_size; ++i){
        *((unsigned char*)P_dst++) = *((unsigned char*)P_dst++);
    }
```

```c
    //initialize the .bss section in SRAM with zeros
    unsigned int 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();
}
```

Makefile:

```
#Prepared by Ebram Habib
CC=arm-none-eabi-
CFLAGS=-mcpu=cortex-m4  -gdwarf-2 -g
INCS=-I .
LIBS=
SRC= $(wildcard *.c)
OBJ= $(SRC:.c=.o)
As= $(wildcard *.s)
AsOBJ= $(As:.s=.o)
Project_name=learn-in-depth_cortex_m4

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

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

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

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

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

clean:
    rm *.elf *.bin
```

Linkerscript:

```
/* learn-in-depth
Unit3_lesson4_Lab3
Ebram Habib
*/

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

Mapfile:

```
Memory Configuration

Name             Origin             Length             Attributes
flash            0x00000000         0x20000000         xr
sram             0x20000000         0x20000000         xrw
*default*        0x00000000         0xffffffff

Linker script and memory map


.text           0x00000000      0x12c
 *(.vectors*)
 .vectors       0x00000000       0x10 startup.o
                0x00000000                g_pt_func_vectors
 *(.text*)
 .text          0x00000010       0x8c main.o
                0x00000010                main
 .text          0x0000009c       0x90 startup.o
                0x0000009c                H_Fault_Handler
                0x0000009c                Default_Handler
                0x0000009c                NMI_Handler
                0x000000a8                Reset_Handler
 *(.rodata)
                0x0000012c                _E_text = .

.data           0x20000000        0x0 load address 0x0000012c
                0x20000000                _S_DATA = .
 *(.data)
 .data          0x20000000        0x0 main.o
 .data          0x20000000        0x0 startup.o
                0x20000000                . = ALIGN (0x4)
                0x20000000                _E_DATA = .

.igot.plt       0x20000000        0x0 load address 0x0000012c
 .igot.plt      0x20000000        0x0 main.o

.bss            0x20000000      0x400 load address 0x0000012c
                0x20000000                _S_bss = .
 *(.bss*)
 .bss           0x20000000        0x0 main.o
 .bss           0x20000000      0x400 startup.o
                0x20000400                _E_bss = .
LOAD main.o
LOAD startup.o
OUTPUT(learn-in-depth_cortex_m4.elf elf32-littlearm)
```

Symbols:

```
El_Amir Tech@DESKTOP-NCOG612 MINGW32 /e/Downloads/Embedded Here We Go Again/Ke
los Shenoda's Diploma/Code/Mastering_Embedded_Systems/Unit3/lesson4 (master)
$ arm-none-eabi-nm.exe learn-in-depth_cortex_m4.elf
20000400 B _E_bss
20000000 D _E_DATA
0000012c T _E_text
20000000 B _S_bss
20000000 D _S_DATA
0000009c T Default_Handler
00000000 T g_pt_func_vectors
0000009c W H_Fault_Handler
00000010 T main
0000009c W NMI_Handler
000000a8 T Reset_Handler
20000000 b stack_top
```

Sections:

```
El_Amir Tech@DESKTOP-NCOG612 MINGW32 /e/Downloads/Embedded Here We Go Again/Ker
los Shenoda's Diploma/Code/Mastering_Embedded_Systems/Unit3/lesson4 (master)
$ arm-none-eabi-objdump.exe -h learn-in-depth_cortex_m4.elf

learn-in-depth_cortex_m4.elf:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000012c  00000000  00000000  00010000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000000  20000000  0000012c  00020000  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000400  20000000  0000012c  00020000  2**2
                  ALLOC
```

Keil Uvision Simulation: