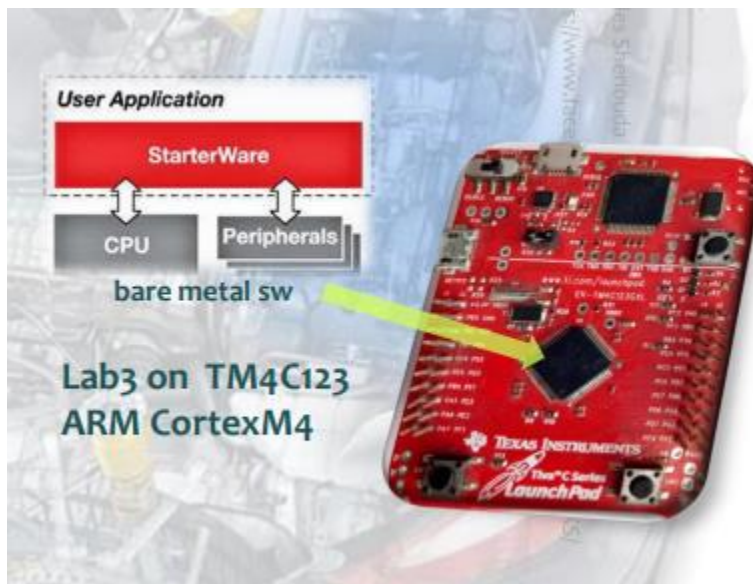


Lab 3

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:

```
1 //learn-in-depth
2 //Ebram Habib
3 //Cortex-M4
4
5 #include <stdint.h>
6
7 #define SYSTCL_RCGC2_R      (*((volatile unsigned Long*)0x400FE108))
8 #define GPIO_PORTF_DATA_R  (*((volatile unsigned Long*)0x400253FC))
9 #define GPIO_PORTF_DIR_R   (*((volatile unsigned Long*)0x40025400))
10 #define GPIO_PORTF_DEN_R   (*((volatile unsigned Long*)0x4002551C))
11
12 int main ()
13 {
14     volatile unsigned delay_counter;
15     SYSTCL_RCGC2_R = 0x20; //initializing the GPIO
16     for(delay_counter=0;delay_counter<200;delay_counter++); // waiting to make sure that the GPIO is up and running
17
18     GPIO_PORTF_DIR_R |= 1<<3; //Setting the pin 3 direction to output
19     GPIO_PORTF_DEN_R |= 1<<3; //Enabling PORT F pin 3
20
21     while(1)
22     {
23         GPIO_PORTF_DATA_R |= 1<<3;
24         for(delay_counter=0;delay_counter<200000;delay_counter++); // making the delay big enough to notice the LED blinking
25         GPIO_PORTF_DATA_R &= ~(1<<3);
26         for(delay_counter=0;delay_counter<200000;delay_counter++);
27     }
28
29     return 0;
30 }
31
```

Startup.c:

```
1  /*      startup.c
2      Eng.Ebram Habib
3  */
4
5  #include <stdint.h>
6
7  extern int main(void);
8  void Reset_Handler();
9
10 void Default_Handler(){
11     Reset_Handler();
12 }
13
14 void NMI_Handler(void) __attribute__((weak, alias("Default_Handler")));
15 void H_Fault_Handler(void) __attribute__((weak, alias("Default_Handler")));
16
17 //booking 1024 bytes located by .bss as an uninitialized array of 256 elements (256*4=1024b)
18 static unsigned long stack_top[256];
19
20 void (* const g_pt_func_vectors[])() __attribute__((section(".vectors"))) =
21 {
22     (void (*)()) ((unsigned long)stack_top + sizeof(stack_top)),
23     &Reset_Handler,
24     &NMI_Handler,
25     &H_Fault_Handler,
26 };
27
28 extern int _E_text;
29 extern int _S_DATA;
30 extern int _E_DATA;
31 extern int _S_bss;
32 extern int _E_bss;
33
34 void Reset_Handler(void){
35     int i;
36     //copy data from ROM to RAM
37     unsigned int DATA_size = (unsigned char*)&_E_DATA - (unsigned char*)&_S_DATA;
38     unsigned char* P_src = (unsigned char*)&_E_text;
39     unsigned char* P_dst = (unsigned char*)&_S_DATA;
40     for(i = 0; i < DATA_size; ++i){
41         *((unsigned char*)P_dst++) = *((unsigned char*)P_src++);
42     }
43
44     //initialize the .bss section in SRAM with zeros
45     unsigned int bss_size = (unsigned char*)&_E_bss - (unsigned char*)&_S_bss;
46     P_dst = (unsigned char*)&_S_bss;
47     for(i = 0; i < bss_size; i++){
48         *((unsigned char*)P_dst++) = (unsigned char) 0;
49     }
50
51     //jump to main
52     main();
53 }
```

Makefile:

```
1  #Prepared by Ebram Habib
2  CC=arm-none-eabi-
3  CFLAGS=-mcpu=cortex-m4 -gdwarf-2 -g
4  INCS=-I .
5  LIBS=
6  SRC= $(wildcard *.c)
7  OBJ= $(SRC:.c=.o)
8  As= $(wildcard *.s)
9  AsOBJ= $(As:.s=.o)
10 Project_name=learn-in-depth_cortex_m4
11
12 all: $(Project_name).bin
13     @echo "====Build is Done====="
14
15 %.o: %.c
16     $(CC)gcc.exe $(CFLAGS) $(INCS) -c $< -o $@
17
18 $(Project_name).elf: $(OBJ) $(AsOBJ)
19     $(CC)ld.exe -T linker_script.ld $(LIBS) $(OBJ) $(AsOBJ) -o $(Project_name).elf -Map=Map_file.map
20     cp $(Project_name).elf $(Project_name).axf
21
22 $(Project_name).bin: $(Project_name).elf
23     $(CC)objcopy.exe -O binary $< $@
24
25 clean_all:
26     rm *.o *.elf *.bin
27
28 clean:
29     rm *.elf *.bin
```

Linkerscript:

```
1  /* learn-in-depth
2  Unit3_lesson4_Lab3
3  Ebram Habib
4  */
5
6  MEMORY
7  {
8      flash(RX)      : ORIGIN = 0X00000000, LENGTH = 512M
9      sram(RWX)       : ORIGIN = 0X20000000, LENGTH = 512M
10 }
11
12 SECTIONS
13 {
14     .text : {
15         *(.vectors*)
16         *(.text*)
17         *(.rodata)
18         _E_text = .;
19     }> flash
20     .data : {
21         _S_DATA = .;
22         *(.data)
23         . = ALIGN(4);
24         _E_DATA = .;
25     }> sram AT> flash
26     .bss : {
27         _S_bss = .;
28         *(.bss*)
29         _E_bss = .;
30     }> sram
31 }
```

Mapfile:

```
1
2 Memory Configuration
3
4 Name          Origin          Length          Attributes
5 flash         0x00000000      0x20000000      xr
6 sram          0x20000000      0x20000000      xrw
7 *default*     0x00000000      0xffffffff
8
9 Linker script and memory map
10
11
12 .text         0x00000000      0x12c
13 *(.vectors*)
14 .vectors      0x00000000      0x10 startup.o
15              0x00000000      g_pt_func_vectors
16 *(.text*)
17 .text         0x00000010      0x8c main.o
18              0x00000010      main
19 .text         0x0000009c      0x90 startup.o
20              0x0000009c      H_Fault_Handler
21              0x0000009c      Default_Handler
22              0x0000009c      NMI_Handler
23              0x000000a8      Reset_Handler
24 *(.rodata)
25              0x0000012c      _E_text = .
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45 .data         0x20000000      0x0 load address 0x0000012c
46              0x20000000      _S_DATA = .
47 *(.data)
48 .data         0x20000000      0x0 main.o
49 .data         0x20000000      0x0 startup.o
50              0x20000000      . = ALIGN (0x4)
51              0x20000000      _E_DATA = .
52
53 .igot.plt     0x20000000      0x0 load address 0x0000012c
54 .igot.plt     0x20000000      0x0 main.o
55
56 .bss          0x20000000      0x400 load address 0x0000012c
57              0x20000000      _S_bss = .
58 *(.bss*)
59 .bss          0x20000000      0x0 main.o
60 .bss          0x20000000      0x400 startup.o
61              0x20000400      _E_bss = .
62 LOAD main.o
63 LOAD startup.o
64 OUTPUT(learn-in-depth_cortex_m4.elf elf32-littlearm)
65
```

Symbols:

```
El_Amir Tech@DESKTOP-NC0G612 MINGW32 /e/Downloads/Embedded Here We Go Again/Ker
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-NC0G612 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:

Keil Uvision Simulation interface showing the Logic Analyzer window. The Logic Analyzer displays a waveform for PORTF, showing a square wave signal. The registers window shows the status of various registers, including R0 through R15, and the GPIOF register. The code window shows the main.c file with the following code:

```
23 GPIO_PORTF_DATA_R |= 1<<3;
24 for(delay_counter=0;delay_counter<200000;delay_counter++); // making the delay big enough to notice
25 GPIO_PORTF_DATA_R ^= ~1<<3;
26 for(delay_counter=0;delay_counter<200000;delay_counter++);
27
28
29 return 0;
30
31
```

The GPIOF register window shows the following values:

Property	Value
DATA	0x08080819
DIR	0x08080808
IS	0x00000000

The Call Stack + Locals window shows the current function call stack:

Name	Location/Value
main	0x00000010
delay_counter	0x0001D92E

The Command window shows the following commands:

```
Running with Code Size Limit: 32K
Load *.\\..\\..\\..\\Code\\Mastering_Embedded_Systems\\Unit3\\lesson4\\learn-in-depth
LA 'PORTF
LA 'PORTF
```

The bottom status bar shows the simulation time: t1: 10.53600006 sec, L26 C:1, CAP: NUM SCRL OVR: R / W.