

MAIN.C

```
1  // prepared by
2  //Eng.Moaz Omar
3  #include "stdint.h"
4  typedef volatile unsigned long vuint32_t;
5  #define SYSCTL_RCGC2_R      (*((vuint32_t *) 0x400FE108))
6  #define GPIO_PORTF_DIR_R    (*((vuint32_t *) 0x40025400))
7  #define GPIO_PORTF_DATA_R   (*((vuint32_t *) 0x400253FC))
8  #define GPIO_PORTF_DEN_R    (*((vuint32_t *) 0x4002551C))
9
10 int main()
11 {
12     vuint32_t delay_counter;
13     SYSCTL_RCGC2_R =0x20;
14     for(delay_counter=0 ;delay_counter<200;delay_counter++);
15     GPIO_PORTF_DIR_R |=1<<3;
16     GPIO_PORTF_DEN_R |=1<<3;
17     while(1){
18         GPIO_PORTF_DATA_R |=1<<3;
19         for(delay_counter=0;delay_counter<20000;delay_counter++);
20         GPIO_PORTF_DATA_R &=~(1<<3);
21         for(delay_counter=0;delay_counter<20000;delay_counter++);
22     }
23
24     return 0;
25 }
```

STARTUP.C

```
1  #include "stdint.h"
2
3  extern int main();
4  extern unsigned int E_text;
5  extern unsigned int S_data;
6  extern unsigned int E_data;
7  extern unsigned int S_bss;
8  extern unsigned int E_bss;
9
10 volatile unsigned long stack_top[256];
11
12 void reset_handler();
13 void Default_Handler(){
14     reset_handler();
15 }
16 void NMI_falut().__attribute__((weak,alias("Default_Handler")));
17 void hard_falut().__attribute__((weak,alias("Default_Handler")));
18 void memory_management_falut().__attribute__((weak,alias("Default_Handler")));
19 void bus_falut().__attribute__((weak,alias("Default_Handler")));
20 void usage_falut().__attribute__((weak,alias("Default_Handler")));
21 void reserved_0().__attribute__((weak,alias("Default_Handler")));
22 void reserved_1().__attribute__((weak,alias("Default_Handler")));
23 void reserved_2().__attribute__((weak,alias("Default_Handler")));
24 void reserved_3().__attribute__((weak,alias("Default_Handler")));
25 void SV_Call().__attribute__((weak,alias("Default_Handler")));
26 void reserved_for_debug().__attribute__((weak,alias("Default_Handler")));
27 void reserved_4().__attribute__((weak,alias("Default_Handler")));
28 void pendSV().__attribute__((weak,alias("Default_Handler")));
29 void systick().__attribute__((weak,alias("Default_Handler")));
30 void IRQ_0().__attribute__((weak,alias("Default_Handler")));
31 void IRQ_1().__attribute__((weak,alias("Default_Handler")));
```

STARTUP.C CONT_1

```
32
33 void (* const g_ptr_func_vectors[])()__attribute__((section(".vectors"))) =
34 {
35     (void (*)()) ((unsigned long)stack_top+sizeof(stack_top)),
36     &NMI_falut,
37     &hard_falut,
38     &memory_management_falut,
39     &bus_falut,
40     &usage_falut,
41     &reserved_0,
42     &reserved_1,
43     &reserved_2,
44     &reserved_3,
45     &SV_Call,
46     &reserved_for_debug,
47     &reserved_4,
48     &pendSV,
49     &systick,
50     &IRQ_0,
51     &IRQ_1
52 };
53
```


STARTUP.C CONT_2

```
53
54 void reset_handler()
55 {
56     unsigned int data_size;
57     data_size = ( unsigned char*)&E_data - ( unsigned char*)&S_data;
58     unsigned char * P_source = ( unsigned char*)&E_text;
59     unsigned char * P_destination = ( unsigned char*)&E_text;
60     int i;
61     for(i=0; i<data_size; i++)
62         *((unsigned char*)P_destination++) = *((unsigned char*)P_source++);
63     unsigned int bss_size ;
64     bss_size =(unsigned char*)&E_bss - (unsigned char*)&S_bss;
65     P_destination =(unsigned char *)&S_bss;
66     for(i=0; i<bss_size; i++)
67         *((unsigned char*)P_destination++) =(unsigned char) 0;
68     main();
69 }
70
```

LINKER_SCRIPT.LD

```
1  /*
2  prepared by
3  eng.Moaz Oamr*/
4
5  MEMORY
6  {
7      flash(RX) : ORIGIN = 0x00000000, LENGTH = 512M
8      sram(RWX) : ORIGIN = 0x20000000, LENGTH = 512M
9  }
10 SECTIONS
11 {
12     .text : {
13         *(.vectors*)
14         *(.text*)
15         *(.rodata)
16         E_text = . ;
17     }>flash
18
19     .data : {
20         S_data = . ;
21         *(.data*)
22         . = ALIGN(4);
23         E_data = . ;
24     }> sram AT> flash
25
26     .bss : {
27         S_bss = . ;
28         *(.bss*)
29         E_bss = . ;
30     }>sram
31 }
32
```

MAIN.O SECTIONS

```
Moaz Omar@Moaz-Omar MINGW64 /d/Courses/Embedded Systems/working_dir/Unit3_Embedded_C/lesson_4 (master)
```

```
$ 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	0000008c	00000000	00000000	00000034	2**2
	CONTENTS, ALLOC, LOAD, READONLY, CODE					
1	.data	00000000	00000000	00000000	000000c0	2**0
	CONTENTS, ALLOC, LOAD, DATA					
2	.bss	00000000	00000000	00000000	000000c0	2**0
	ALLOC					
3	.debug_info	000000a7	00000000	00000000	000000c0	2**0
	CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS					
4	.debug_abbrev	0000006b	00000000	00000000	00000167	2**0
	CONTENTS, READONLY, DEBUGGING, OCTETS					
5	.debug_loc	00000038	00000000	00000000	000001d2	2**0
	CONTENTS, READONLY, DEBUGGING, OCTETS					
6	.debug_aranges	00000020	00000000	00000000	0000020a	2**0
	CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS					
7	.debug_line	0000009c	00000000	00000000	0000022a	2**0
	CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS					
8	.debug_str	00000152	00000000	00000000	000002c6	2**0
	CONTENTS, READONLY, DEBUGGING, OCTETS					
9	.comment	0000004a	00000000	00000000	00000418	2**0
	CONTENTS, READONLY					
10	.debug_frame	0000002c	00000000	00000000	00000464	2**2
	CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS					
11	.ARM.attributes	0000002e	00000000	00000000	00000490	2**0
	CONTENTS, READONLY					

STARTUP.O SECTIONS

```
Moaz Omar@Moaz-Omar MINGW64 /d/Courses/Embedded systems/working_dir/Unit3_Embedded_C/lesson_4 (master)
```

```
$ 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	00000044	00000000	00000000	000000c4	2**2
	CONTENTS,		ALLOC,	LOAD,	RELOC,	READONLY, DATA
4	.debug_info	00000199	00000000	00000000	00000108	2**0
	CONTENTS,		RELOC,	READONLY,	DEBUGGING,	OCTETS
5	.debug_abbrev	000000d9	00000000	00000000	000002a1	2**0
	CONTENTS,		READONLY,	DEBUGGING,	OCTETS	
6	.debug_loc	0000007c	00000000	00000000	0000037a	2**0
	CONTENTS,		READONLY,	DEBUGGING,	OCTETS	
7	.debug_aranges	00000020	00000000	00000000	000003f6	2**0
	CONTENTS,		RELOC,	READONLY,	DEBUGGING,	OCTETS
8	.debug_line	000000bb	00000000	00000000	00000416	2**0
	CONTENTS,		RELOC,	READONLY,	DEBUGGING,	OCTETS
9	.debug_str	000001be	00000000	00000000	000004d1	2**0
	CONTENTS,		READONLY,	DEBUGGING,	OCTETS	
10	.comment	0000004a	00000000	00000000	0000068f	2**0
	CONTENTS,		READONLY			
11	.debug_frame	00000050	00000000	00000000	000006dc	2**2
	CONTENTS,		RELOC,	READONLY,	DEBUGGING,	OCTETS
12	.ARM.attributes	0000002e	00000000	00000000	0000072c	2**0
	CONTENTS,		READONLY			

MAIN.O SYMBOLS

```
Moaz Omar@Moaz-Omar MINGW64 /d/Courses/Embedded Systems/Working_dir/Unit3_Embedded_C/lesson_4 (master)
$ arm-none-eabi-nm.exe main.o
00000000 T main
```


STARTUP.O SYMBOLS

```
Moaz Omar@Moaz-Omar MINGW64 /d/Courses/Embedded Systems/working_dir/Unit3_Embedded_C/lesson_4 (master)
$ arm-none-eabi-nm.exe startup.o
00000000 w bus_falut
00000000 T Default_Handler
           U E_bss
           U E_data
           U E_text
00000000 R g_ptr_func_vectors
00000000 w hard_falut
00000000 w IRQ_0
00000000 w IRQ_1
           U main
00000000 w memory_management_falut
00000000 w NMI_falut
00000000 w pendsv
00000000 w reserved_0
00000000 w reserved_1
00000000 w reserved_2
00000000 w reserved_3
00000000 w reserved_4
00000000 w reserved_for_debug
0000000c T reset_handler
           U S_bss
           U S_data
00000000 B stack_top
00000000 w sv_call
00000000 w systick
00000000 w usage_falut
```

TIVAC_TOGGLING_LED SYMBOLS

```
Moaz Omar@Moaz-Omar MINGW64 /d/Courses/Embedded Systems/working_dir/Unit3_Embedded_C/lesson_4 (master)
$ arm-none-eabi-nm.exe tivac_toggling_led.elf
000000d0 w bus_falut
000000d0 T Default_Handler
20000400 B E_bss
20000000 D E_data
00000160 T E_text
00000000 T g_ptr_func_vectors
000000d0 w hard_falut
000000d0 w IRQ_0
000000d0 w IRQ_1
00000044 T main
000000d0 w memory_management_falut
000000d0 w NMI_falut
000000d0 w pendsv
000000d0 w reserved_0
000000d0 w reserved_1
000000d0 w reserved_2
000000d0 w reserved_3
000000d0 w reserved_4
000000d0 w reserved_for_debug
000000dc T reset_handler
20000000 B S_bss
20000000 D S_data
20000000 B stack_top
000000d0 w sv_call
000000d0 w systick
000000d0 w usage_falut
```

TIVAC_TOGGLING_LED.ELF SECTIONS

```
Moaz Omar@Moaz-Omar MINGW64 /d/Courses/Embedded Systems/Working_dir/Unit3_Embedded_C/lesson_4 (master)
$ arm-none-eabi-objdump.exe -h tivaC_toggling_led.elf
```

```
tivaC_toggling_led.elf:      file format elf32-littlearm
```

Sections:

Idx	Name	Size	VMA	LMA	File off	Algn
0	.text	00000160	00000000	00000000	00010000	2**2
	CONTENTS, ALLOC, LOAD, READONLY, CODE					
1	.data	00000000	20000000	00000160	00020000	2**0
	CONTENTS, ALLOC, LOAD, DATA					
2	.bss	00000400	20000000	00000160	00020000	2**2
	ALLOC					
3	.debug_info	00000240	00000000	00000000	00020000	2**0
	CONTENTS, READONLY, DEBUGGING, OCTETS					
4	.debug_abbrev	00000144	00000000	00000000	00020240	2**0
	CONTENTS, READONLY, DEBUGGING, OCTETS					
5	.debug_loc	000000b4	00000000	00000000	00020384	2**0
	CONTENTS, READONLY, DEBUGGING, OCTETS					
6	.debug_aranges	00000040	00000000	00000000	00020438	2**0
	CONTENTS, READONLY, DEBUGGING, OCTETS					
7	.debug_line	00000157	00000000	00000000	00020478	2**0
	CONTENTS, READONLY, DEBUGGING, OCTETS					
8	.debug_str	000001ae	00000000	00000000	000205cf	2**0
	CONTENTS, READONLY, DEBUGGING, OCTETS					
9	.comment	00000049	00000000	00000000	0002077d	2**0
	CONTENTS, READONLY					
10	.ARM.attributes	0000002e	00000000	00000000	000207c6	2**0
	CONTENTS, READONLY					
11	.debug_frame	0000007c	00000000	00000000	000207f4	2**2
	CONTENTS, READONLY, DEBUGGING, OCTETS					

SIMULATION

The screenshot displays the Keil uVision IDE interface during a simulation. The main window shows the disassembly and source code of a C program. The source code is as follows:

```
6 #define GPIO_PORTF_DIR_R (*(vuint32_t *) 0x40025400)
7 #define GPIO_PORTF_DATA_R (*(vuint32_t *) 0x40025404)
8 #define GPIO_PORTF_DEN_R (*(vuint32_t *) 0x4002540C)
9
10 int main()
11 {
12     vuint32_t delay_counter;
13     SYSCTL_RCGC2_R = 0x20;
14     for(delay_counter=0; delay_counter<20000; delay_counter++)
15     {
16         GPIO_PORTF_DIR_R |= 1<<3;
17         GPIO_PORTF_DEN_R |= 1<<3;
18         while(1) {
19             GPIO_PORTF_DATA_R |= 1<<3;
20             for(delay_counter=0; delay_counter<20000; delay_counter++)
21             {
22                 GPIO_PORTF_DATA_R &= ~(1<<3);
23                 for(delay_counter=0; delay_counter<20000; delay_counter++)
24                 {
25                     return 0;
26                 }
27             }
28         }
29     }
30 }
```

The 'TExaS edX Lab 2' dialog box is open, showing the following details:

- Port F Hardware:** A schematic diagram of the TM4C123 microcontroller's Port F. It shows pins PF0, PF1, PF2, PF3, PF4, and PF5. PF0 is connected to a switch SW1. PF1 is connected to a switch SW2. PF2 and PF3 are connected to LEDs. PF4 and PF5 are connected to a 16 MHz clock source.
- Port F Registers:** A table of registers and their values.
- Grading Controls:** A section for grading the simulation.

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

The 'Grading Controls' section shows a 'Number from edX' field and a 'Grade' button. The 'Score' is 0. The 'Copy this to edX' field is empty.

The 'Command' window shows the command: `Load "...\\tivaC_toggling_led.axf"`.

The 'Call Stack + Locals' window shows the following variables:

Name	Location/Value	Type
main	0x00000086	int f()
delay_counter	0x00004E20	auto - uint

The 'Simulation' window shows the simulation time: `t1: 0.24146531 sec`.

SIMULATION CONT_1

The screenshot displays the Keil uVision IDE interface for a project named 'Keil_uvision_unit3_lab4_project'. The main window shows the disassembly of the C code, with the following assembly instructions visible:

```
21: for(delay_counter=0;delay_counter<20000;delay_counter++);  
0x000000A8 2300 MOVS r3,#0x00  
0x000000AA 607B STR r3,[r7,#0x04]  
0x000000AC F002 R 0x000000B4
```

The C code in the main.c file is as follows:

```
6 #define GPIO_PORTF_DIR_R (*(vuint32_t *) 0x40025400)  
7 #define GPIO_PORTF_DATA_R (*(vuint32_t *) 0x40025404)  
8 #define GPIO_PORTF_DEN_R (*(vuint32_t *) 0x4002540C)  
9  
10 int main()  
11 {  
12     vuint32_t delay_counter;  
13     SYSCTL_RCGC2_R = 0x20;  
14     for(delay_counter=0;delay_counter<20000;delay_counter++)  
15     {  
16         GPIO_PORTF_DIR_R |= 1<<3;  
17         GPIO_PORTF_DEN_R |= 1<<3;  
18         while(1){  
19             GPIO_PORTF_DATA_R |= 1<<3;  
20             for(delay_counter=0;delay_counter<20000;delay_counter++)  
21             {  
22                 GPIO_PORTF_DATA_R &= ~(1<<3);  
23                 for(delay_counter=0;delay_counter<20000;delay_counter++)  
24                 {  
25                     return 0;  
26                 }  
27             }  
28         }  
29     }  
30 }
```

The 'Port F Hardware' window shows the TM4C123 microcontroller with Port F pins PF0, PF1, PF2, PF3, PF4, and PF5. It also shows the connection of SW1, SW2, and three LEDs to the pins. The 'Port F Registers' window shows the following values:

Register	Value
DATA	0x11
PUR	0x00
LOCK	0x01
DIR	0x08
PDR	0x00
CR	0x1E
DEN	0x08
RCGC2	0x00000020

The 'Grading Controls' window shows the following values:

Control	Value
Number from edX	0
Grade	0
Score	0

The 'Call Stack + Locals' window shows the following variables:

Name	Location/Value	Type
main	0x00000086	int f()
delay_counter	0x00004E20	auto - uint