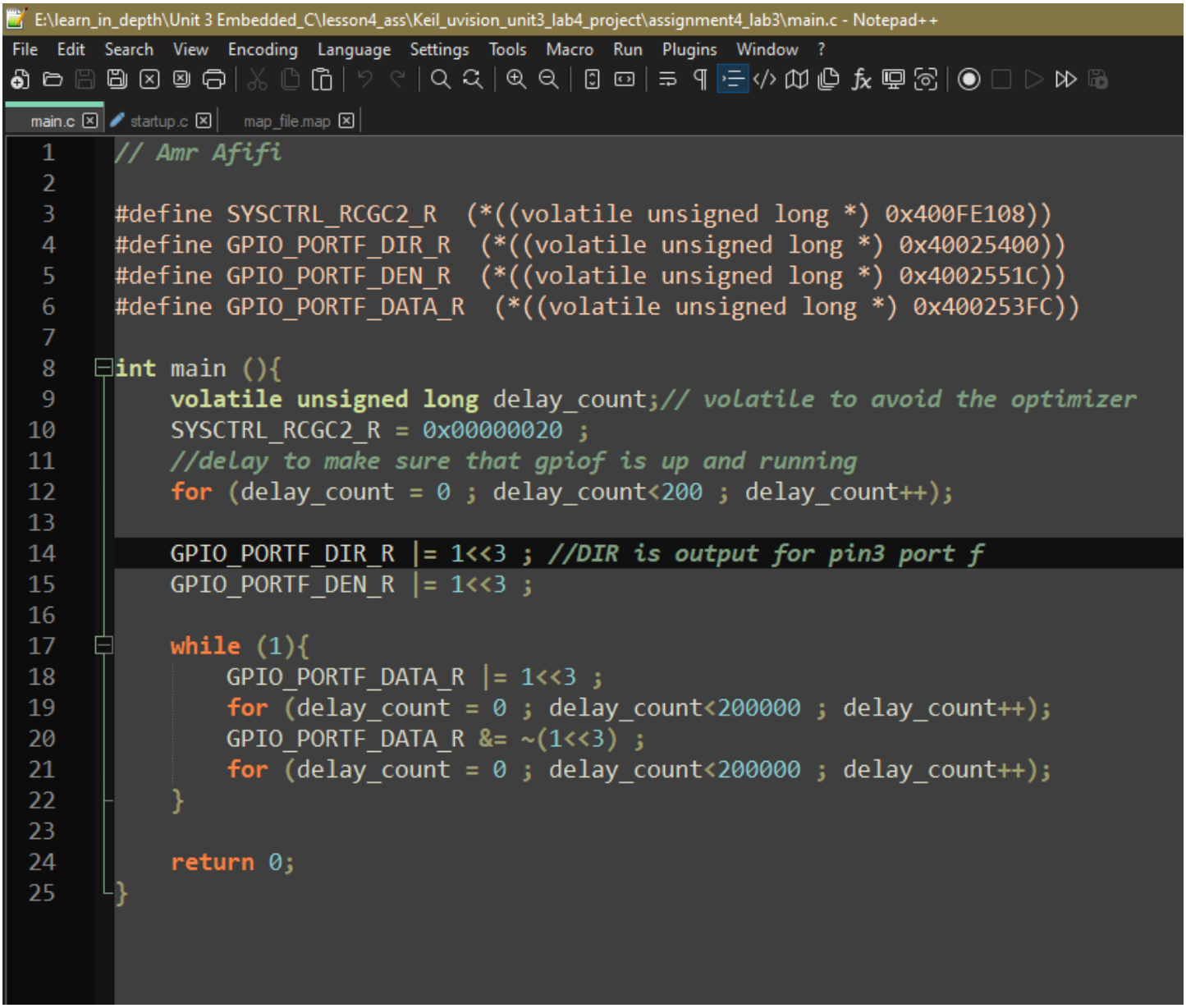
Lab 4 report

In this scope we will toggle the LED and simulate the code on Tiva_C Kit that has tm4c123 and arm-cortexM4 processor .

The LED connected to pin3 of PORTF.

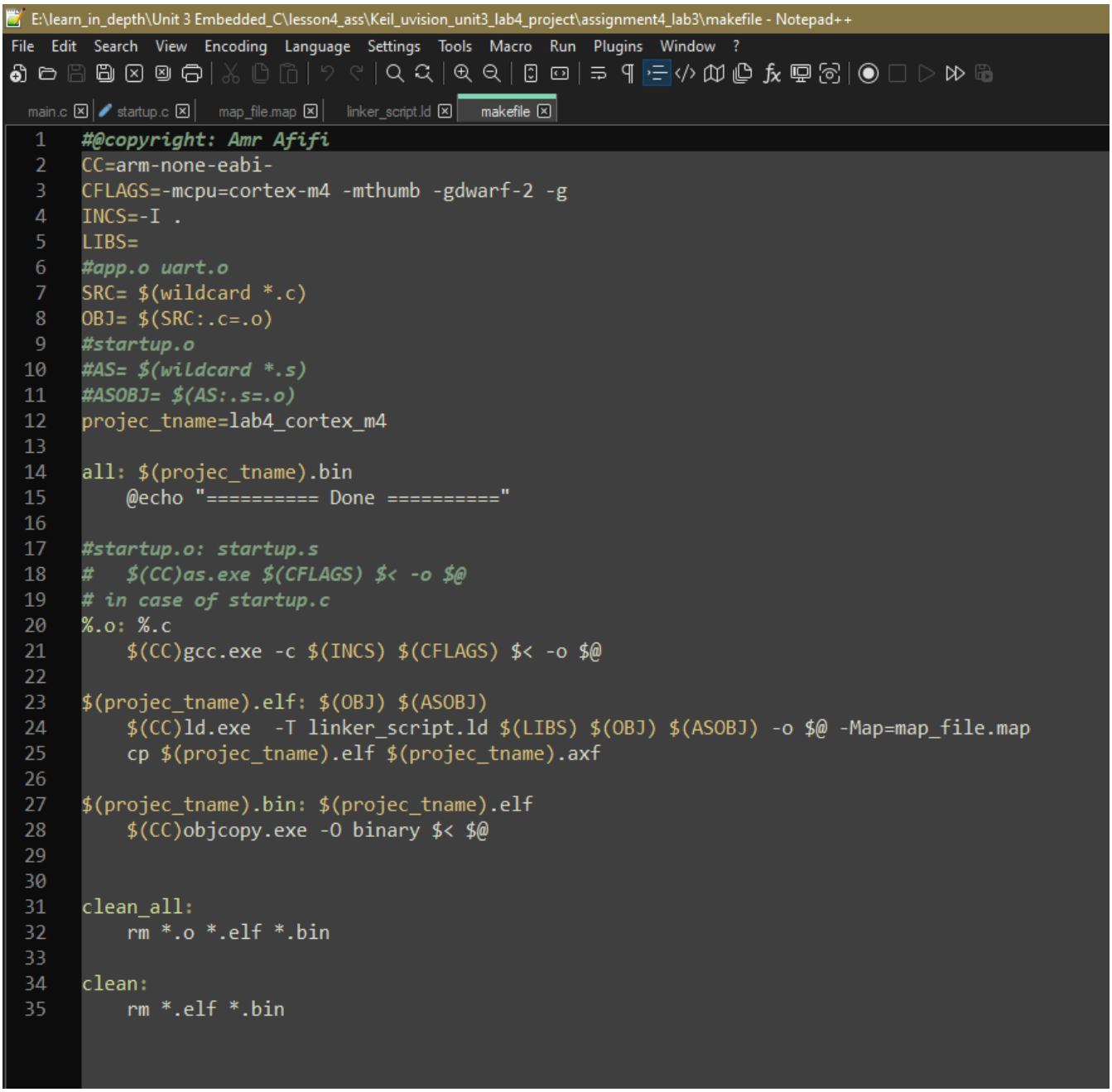
From specs :

- ✓ Flash memory starts with address 0x00000000 and has size of 512M.
- ✓ Sram memory starts 0x20000000 and has size of 512M.
- ✓ SYSCTL is system control module that we will use to enable clock for PORTF has base address of 0x400FE000
- ✓ SYSCTL_RCGC2_R has offset address of 0x108 under SYSCTL we will assign this register with value of 0x00000020 to enable clock for PORTF
- ✓ GPIO module has base address of 0x40025000 and we will use three registers inside
  First GPIO_PORTF_DIR_R has offset of 0x400 and we will assign value of 1 in pin3 to define this pin as an output
  First GPIO_PORTF_DEN_R has offset of 0x51c and we will assign value of 1 in pin3 to enable this pin
- ✓ GPIO_PORTF_DR_R has offset of 0x400 and we will assign value of 1 in pin3 and 0 to toggle the output.


E:\learn_in_depth\Unit 3 Embedded_C\lesson4_ass\Keil_uvision_unit3_lab4_project\assignment4_lab3\main.c - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
main.c
startup.c
map_file.map
1 // Amr Afifi
2
3 #define SYSCTRL_RCGC2_R (*((volatile unsigned long *) 0x400FE108))
4 #define GPIO_PORTF_DIR_R (*((volatile unsigned long *) 0x40025400))
5 #define GPIO_PORTF_DEN_R (*((volatile unsigned long *) 0x4002551C))
6 #define GPIO_PORTF_DATA_R (*((volatile unsigned long *) 0x400253FC))
7
8 int main (){
9 volatile unsigned long delay_count;// volatile to avoid the optimizer
10 SYSCTRL_RCGC2_R = 0x00000020 ;
11 //delay to make sure that gpiof is up and running
12 for (delay_count = 0 ; delay_count<200 ; delay_count++);
13
14 GPIO_PORTF_DIR_R |= 1<<3 ; //DIR is output for pin3 port f
15 GPIO_PORTF_DEN_R |= 1<<3 ;
16
17 while (1){
18 GPIO_PORTF_DATA_R |= 1<<3 ;
19 for (delay_count = 0 ; delay_count<200000 ; delay_count++);
20 GPIO_PORTF_DATA_R &= ~(1<<3) ;
21 for (delay_count = 0 ; delay_count<200000 ; delay_count++);
22 }
23
24 return 0;
25 }

make file : we will make simple changes

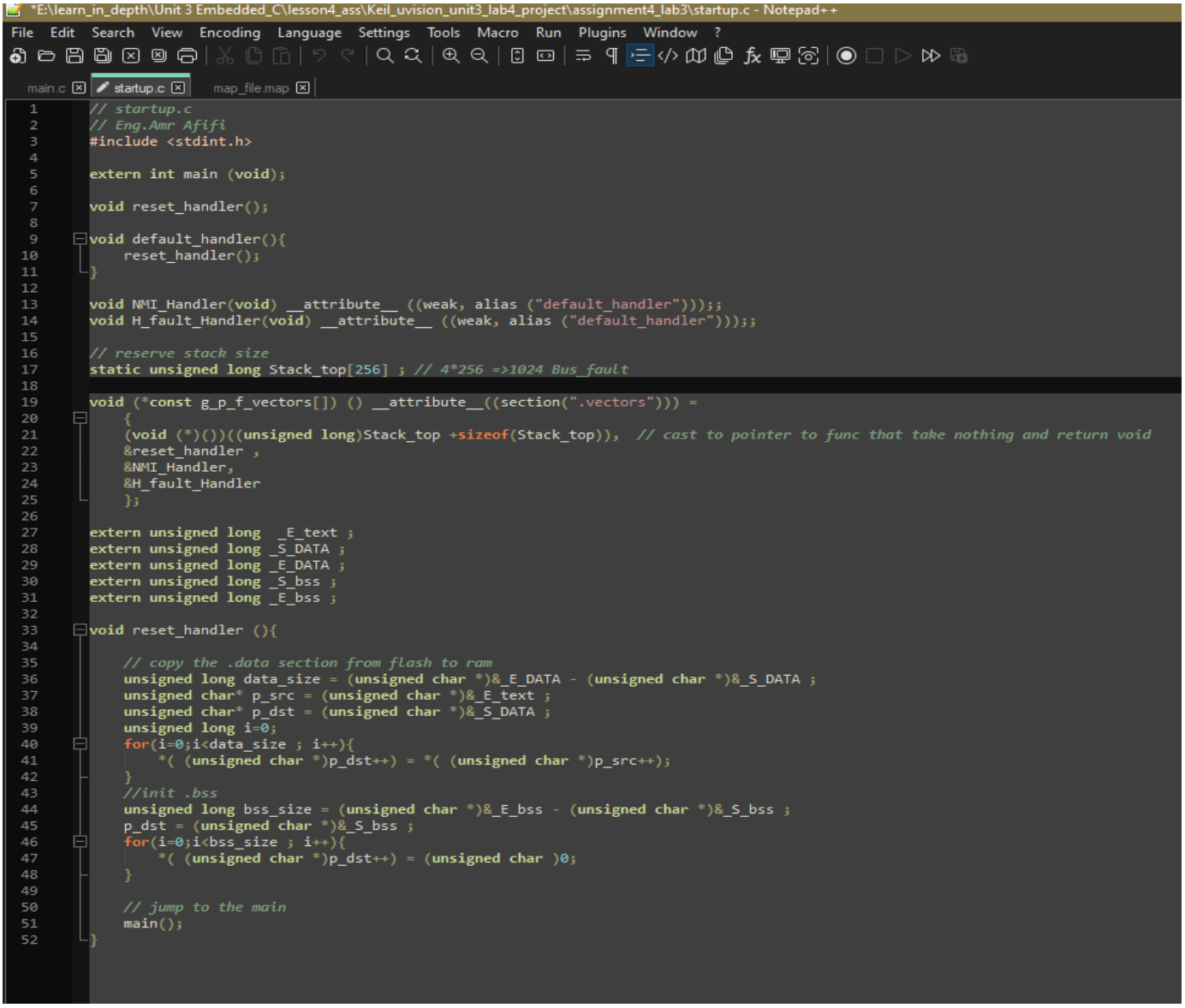
- ✓ we will copy the .axf file to run on kiel micro vision tool
- ✓ change the project name and processor name



startup.c ;

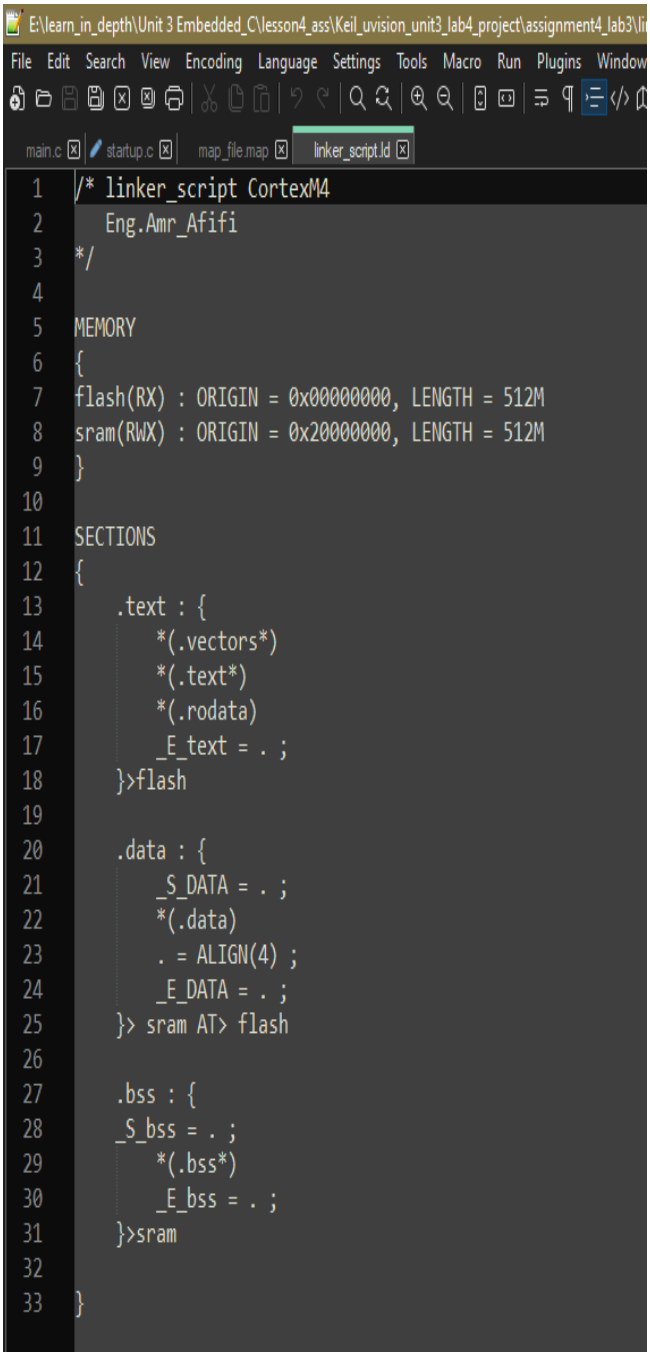
using new approach by initializing the stack pointer in startup.c

- ✓ We will fix SP after 1Kbyte of .bss that by using uninitialized array of int with 256 elements so the total size of the array will be 1024 where the SP will be at the end of the array
- ✓ Array elements will be pinter to function that take nothing and return void
- ✓ The pointers will point to the functions which handle the interrupts according to IVT.



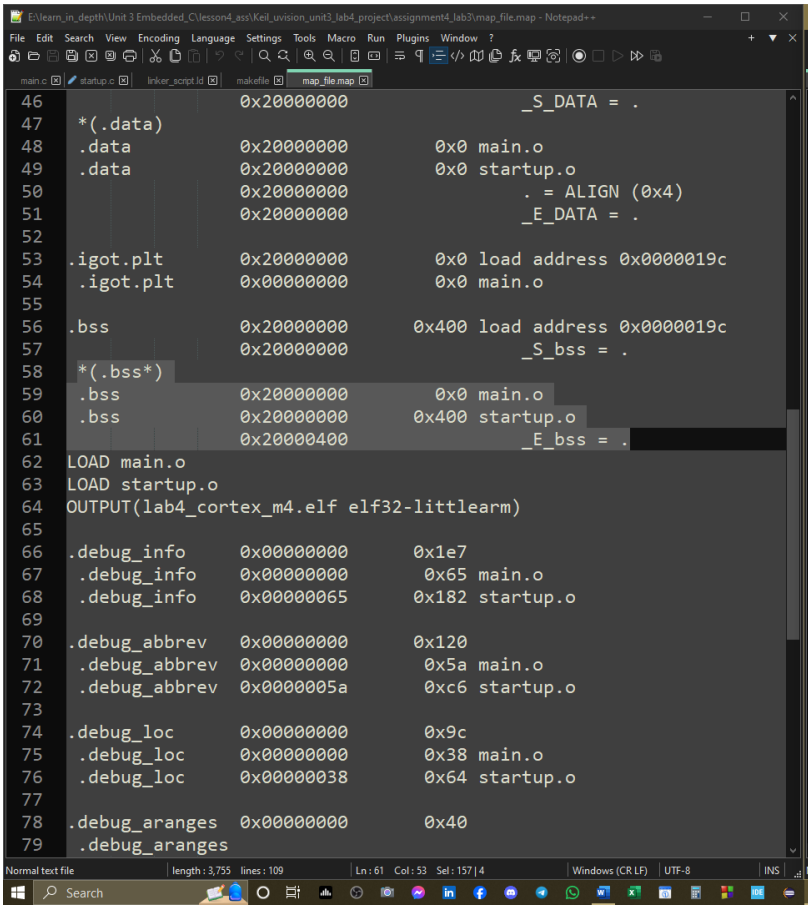
Linker_script :

- ✓ delete stack_top symbol
- ✓ edit the size

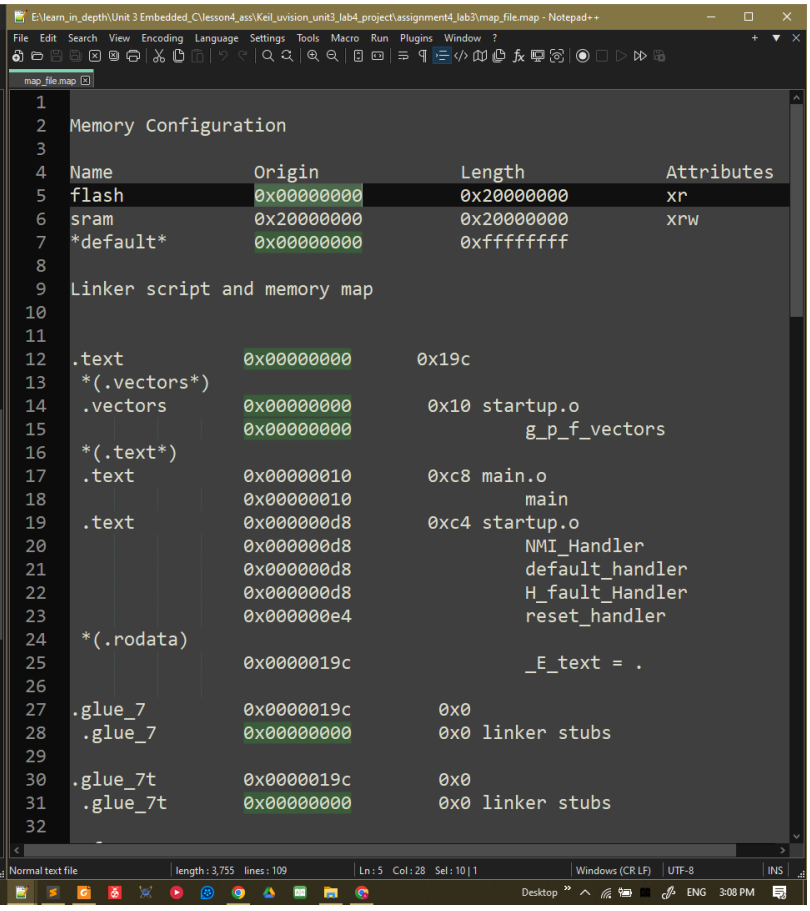


Map file :

- ✓ Flash starts with 0x00000000
- ✓ .bss starts with 0x200000000 and ends with 0x200000400
- ✓ Note that the incremented is 0x400 which equivalent to 1024 in decimal



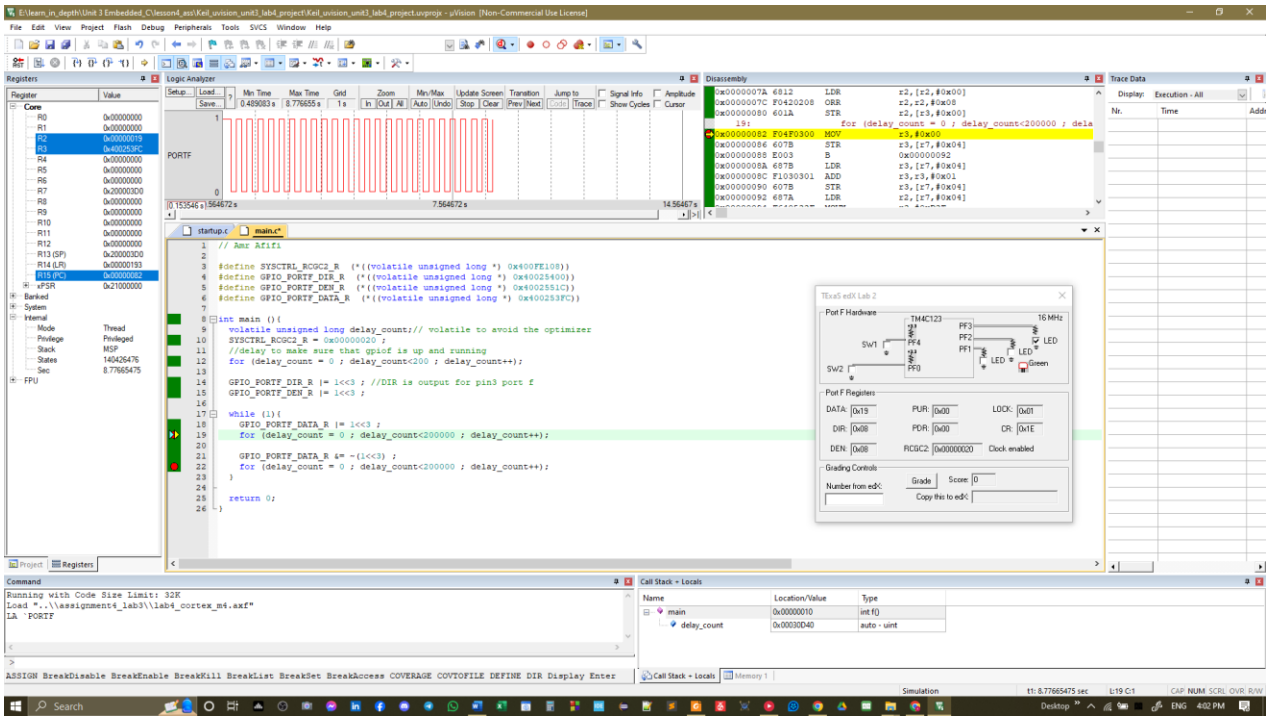
```
                0x20000000                _S_DATA = .
 *(.data)
 .data          0x20000000        0x0 main.o
 .data          0x20000000        0x0 startup.o
                0x20000000                . = ALIGN (0x4)
                0x20000000                _E_DATA = .

.igot.plt       0x20000000        0x0 load address 0x0000019c
 .igot.plt      0x00000000        0x0 main.o

.bss            0x20000000      0x400 load address 0x0000019c
                0x20000000                _S_bss = .
 *(.bss*)
 .bss           0x20000000        0x0 main.o
 .bss           0x20000000      0x400 startup.o
                0x20000400                _E_bss = .
LOAD main.o
LOAD startup.o
OUTPUT(lab4_cortex_m4.elf elf32-littlearm)

.debug_info     0x00000000      0x1e7
 .debug_info    0x00000000       0x65 main.o
 .debug_info    0x00000065      0x182 startup.o

.debug_abbrev   0x00000000      0x120
 .debug_abbrev  0x00000000       0x5a main.o
 .debug_abbrev  0x0000005a       0xc6 startup.o

.debug_loc      0x00000000       0x9c
 .debug_loc     0x00000000       0x38 main.o
 .debug_loc     0x00000038       0x64 startup.o

.debug_aranges  0x00000000       0x40
 .debug_aranges
```



```

Memory Configuration

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

Linker script and memory map


.text           0x00000000      0x19c
 *(.vectors*)
 .vectors       0x00000000       0x10 startup.o
                0x00000000                g_p_f_vectors
 *(.text*)
 .text          0x00000010       0xc8 main.o
                0x00000010                main
 .text          0x000000d8       0xc4 startup.o
                0x000000d8                NMI_Handler
                0x000000d8                default_handler
                0x000000d8                H_fault_Handler
                0x000000e4                reset_handler
 *(.rodata)
                0x0000019c                _E_text = .

.glue_7         0x0000019c        0x0
 .glue_7        0x00000000        0x0 linker stubs

.glue_7t        0x0000019c        0x0
 .glue_7t       0x00000000        0x0 linker stubs
```

Debugging using kiel microvision:

At high level: led is on



At low level: led is off