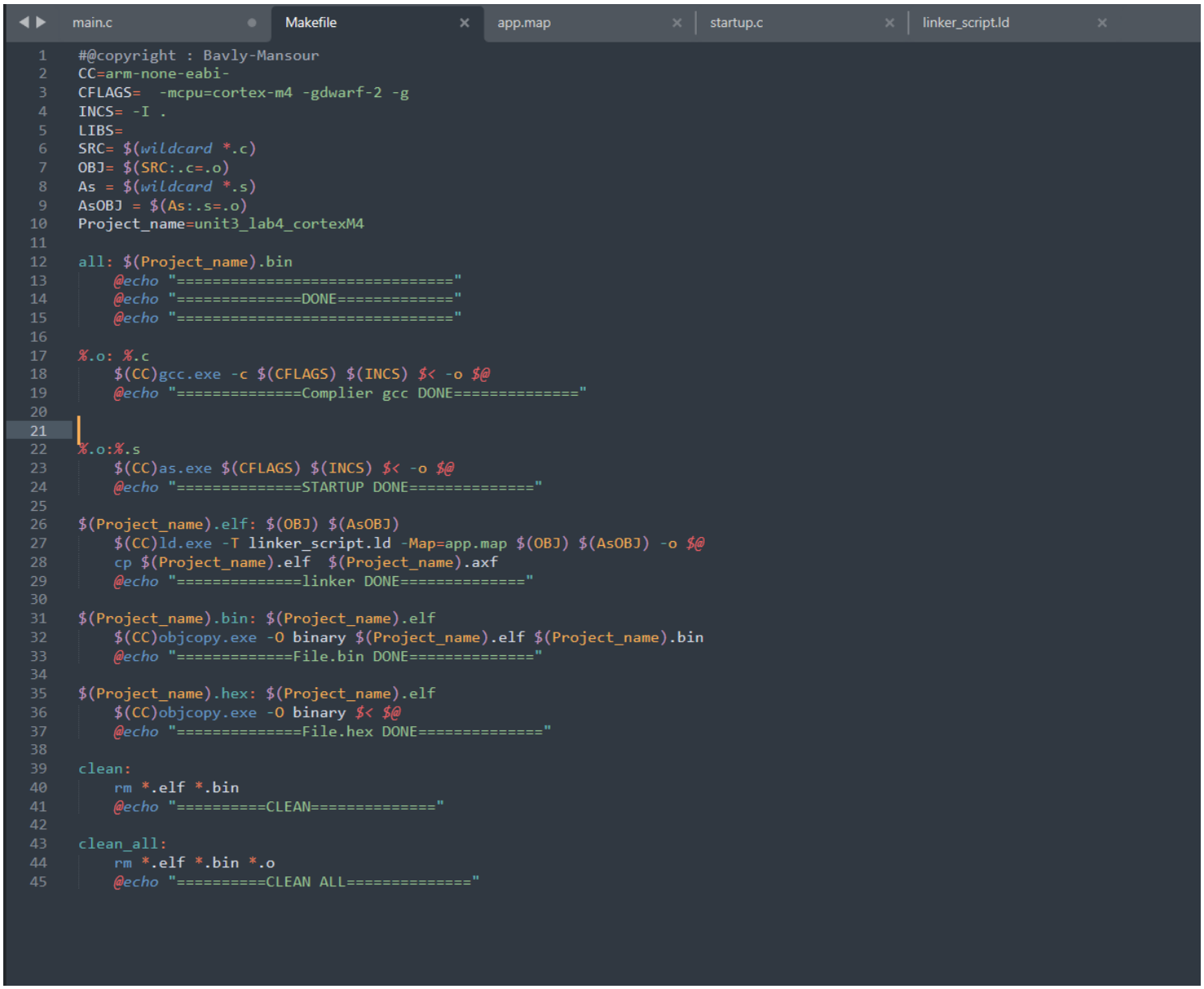
Lesson 4 - LAB 3 – Report

Makefile:

*This Makefile compatible with CPU: Cortex-M4



* -gdwarf-2 instead of -g with Cortex-M3.

* $@ is for target, $< for dependency.

* % is a generalization means all of.

Output of Makefile:

```
Bavly@DESKTOP-CNE79H4 MINGW64 ~/Desktop/Lab 3
$ make
arm-none-eabi-gcc.exe -c -mcpu=cortex-m4 -gdwarf-2 -g -I . main.c -o main.o
==============Complier gcc DONE==============
arm-none-eabi-gcc.exe -c -mcpu=cortex-m4 -gdwarf-2 -g -I . startup.c -o startup.
o
==============Complier gcc DONE==============
arm-none-eabi-ld.exe -T linker_script.ld -Map=app.map main.o startup.o  -o unit3
_lab4_cortexM4.elf
cp unit3_lab4_cortexM4.elf      unit3_lab4_cortexM4.axf
==============linker DONE==============
arm-none-eabi-objcopy.exe -O binary unit3_lab4_cortexM4.elf unit3_lab4_cortexM4.
bin
=============File.bin DONE==============
============================
==============DONE=============
============================

Bavly@DESKTOP-CNE79H4 MINGW64 ~/Desktop/Lab 3
$
```

Main.c file:

main.c | Makefile | app.map | startup.c | linker_script.ld

```c
/**
 ******************************************************************************
 * @file           : main.c
 * @author         : Eng Bavly Mansour
 * @brief          : Main program body
 ******************************************************************************
 */

//first define registers
//*************************************************************
#define SYSCTL_RCGC2_R      (*((volatile unsigned long*)0x400FE108))
#define GPIO_PORTF_DIR_R    (*((volatile unsigned long*)0x40025400))
#define GPIO_PORTF_DEN_R    (*((volatile unsigned long*)0x4002551C))
#define GPIO_PORTF_DATA_R   (*((volatile unsigned long*)0x400253FC))

int main ()
{
    volatile unsigned long delay_count ;

    SYSCTL_RCGC2_R  =0x20 ;
    //set system ctl enable the gpio port SYSCTL register = 0x20
    //delay to make sure GPIOF is up and running
    for (delay_count = 20 ; delay_count < 200 ; delay_count ++);
    GPIO_PORTF_DIR_R |=1<<3 ;
    //dir is output for pin 3 port f , to set pf3 as an output
    GPIO_PORTF_DEN_R |=1<<3 ;
    while (1)
    {
        GPIO_PORTF_DATA_R |= 1<<3 ;
        for (delay_count =0 ; delay_count < 200000 ; delay_count ++);
        GPIO_PORTF_DATA_R &=~(1<<3);
        for (delay_count =0 ; delay_count < 200000 ; delay_count ++);

    }
    return 0;

}
```

- SYSCTL_RCGC2_R = 0x20; // set system ctl enable the gpio port SYSCTL register = 0x20, delay to make sure GPIOF is up and running

Startup.c file for Cortex-M4:

main.c | Makefile | app.map | startup.c | linker_script.ld

```c
/*
**************************************
**************************************
learn-in-depth
By:Bavly Mansour
**************************************
**************************************
*/

#include <stdint.h>
//#define STACK_Start_SP 0x20001000  //txt replacement of STACK_Start_SP "stack pointer" to address 0x20001000
extern int main (void);            // define external int main (void)
void Reset_Handler (void);          //define rest_handler function


/*Copying data from flash to SRAM and allocate space
 for bss and stack according to the memory bourders */


void Default_Handler ()
{
    Reset_Handler();
}

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


/*booking 1024 B locate by .bss through unintialized
array of int 256 element (256*4=1024)*/

static unsigned long Stack_top[256];

//array of pointer to function take nothing and return void
void ( * const g_p_fn_Vectors[])() __attribute__ ((section(".vectors"))) ={
 (void (*)())   ((unsigned long)Stack_top + sizeof(Stack_top)),
 &Reset_Handler ,
 &NMI_Handler ,
 &H_Fault_Handler
};

```

main.c | Makefile | app.map | startup.c | linker_script.ld

```
42
43  extern unsigned int _E_text;
44  extern unsigned int _S_DATA;
45  extern unsigned int _E_DATA;
46  extern unsigned int _S_bss;
47  extern unsigned int _E_bss;
48  extern unsigned int _stack_top;
49
50
51  void Reset_Handler (void)
52  {
53      // copying data from ROM to RAM
54      unsigned int DATA_size = (unsigned char*)&_E_DATA - (unsigned char*)&_S_DATA;
55      unsigned char* P_src = (unsigned char*)&_E_text ;
56      unsigned char* P_dst = (unsigned char*)&_S_DATA ;
57
58      for (int i =0 ; i < DATA_size ; i++){
59          *((unsigned char *)P_dst++) = *((unsigned char*)P_src++) ;
60      }
61
62
63
64      // bss with zero
65      unsigned int bss_size = (unsigned char*)&_E_bss - (unsigned char*)&_S_bss ;
66      P_dst = (unsigned char*)&_S_bss ;
67      for (int i =0 ; i < bss_size ; i++ ){
68          *((unsigned char*)P_dst++) = *((unsigned char*)P_src++) ;
69      }
70      //go to main()
71      main();
72  }
73
74
```

- Alias attribute causes the declaration to be emitted as an alias for another symbol.
- Weak attribute causes the declaration to be emitted as a weak symbol rather than a global.
- booking 1024 B locate by .bss through uninitialized array of int 256 element (256*4=1024)

- **Functions in MapFile :**

```
.text           0x00000000      0x12b
 *(.vectors*)
 .vectors       0x00000000       0x10 startup.o
                0x00000000                g_p_fn_Vectors
 *(.text*)
 .text          0x00000010       0x80 main.o
                0x00000010                main
 .text          0x00000090       0x98 startup.o
                0x00000090                H_Fault_Handler
                0x00000090                Default_Handler
                0x00000090                NMI_Handler
                0x0000009c                Reset_Handler
 *(.rodata)
 .rodata        0x00000128        0x3 main.o
                0x00000128                const_var
                0x0000012b                _E_text = .
```

- Main at 0x0000010
- g_p_fn_Vectors at 0x0000000 instead of _stack_top symbol.
- And all Handler functions at the same address of the default address due to the alias. Default_Handler at 0x00000090, while the Reset_handler has a different address at 0x000009c.

Linker script for Cortex-M4:

main.c | Makefile | app.map | startup.c | linker_script.ld

```
/*This is linker script for LAB3 CPU=CORTEX-M4 ,by : ENG_BAVLY*/

MEMORY
{
    flash(RX) : ORIGIN = 0x00000000, LENGTH = 512M
    sram(RWX) : ORIGIN = 0x20000000, LENGTH = 512M
    /*flash memory origin and length , flash: read and execute*/
    /*sRAM origin and length , sram : read write execute*/

}
/*sections*/
SECTIONS {
    .text :{
            *(.vectors*)
            *(.text*)
            *(.rodata)
            _E_text = .;           /*end of text section in flash @ 0x08000b3*/

        }> flash                   /*everything inside .text put it in flash */

    .data :{
            _S_DATA = .;           /*start of data section @ 0x08000b3 the same  address of end text section */
            *(.data)
            _E_DATA = .;           /*end of data section */
        }>sram AT> flash           /*virtual memory :sram */

    .bss :{
            _S_bss = . ;          /*start of bss section */
            *(.bss*)
            _E_bss = . ;
        }>sram
                                  /* bss are the uninitialized variables = zeros so ignore them from the flash */


}
```

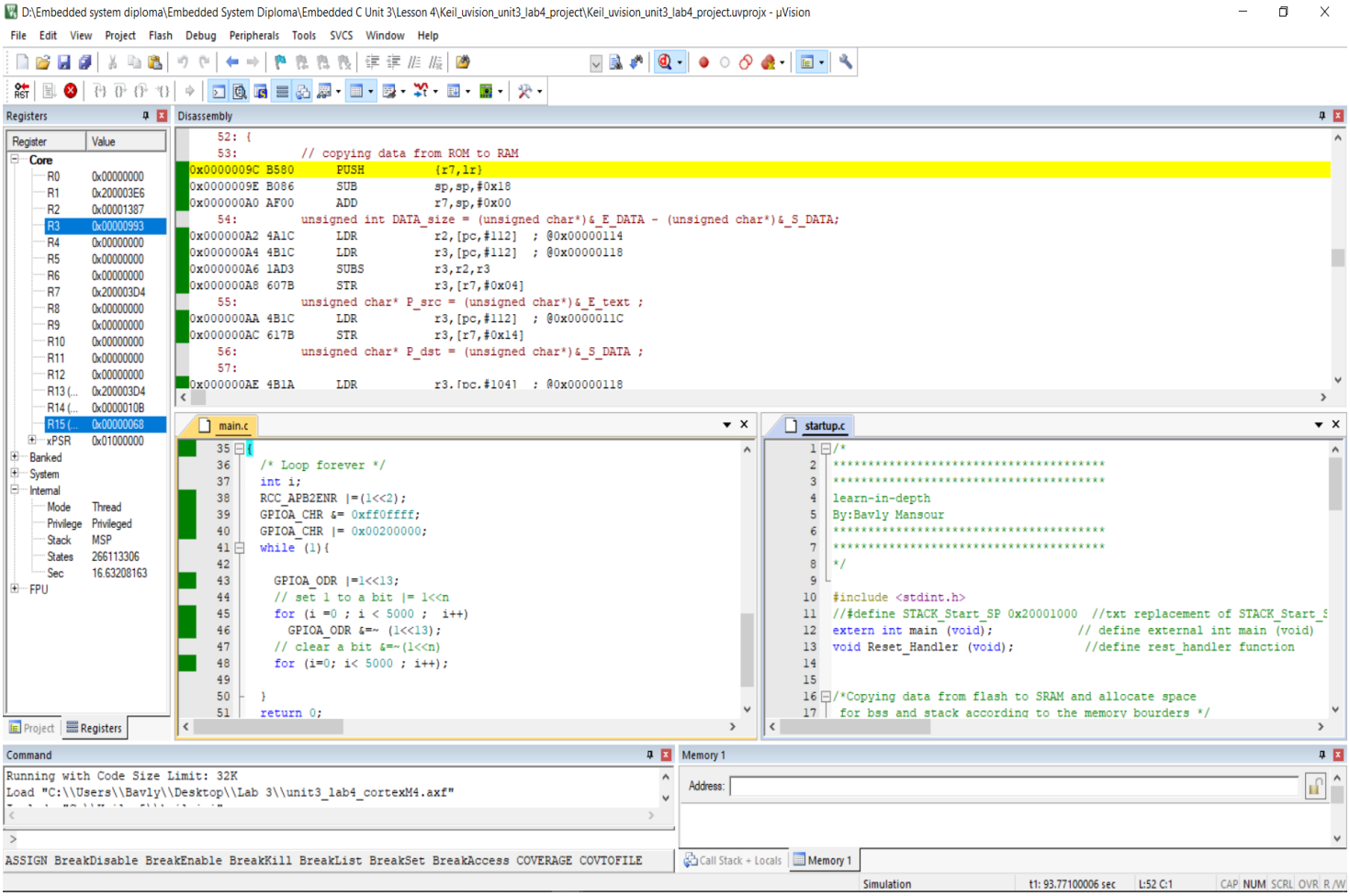
- According to memory borders:
  _E_text is the end of .text section
  _S_DATA is the start of .data section
  _S_bss is the start of .bss section
- .text section includes .rodata section.

- Memory configuration according to linker script memory borders:

main.c | Makefile | app.map | startup.c | linker_script.ld

```
1
2   Allocating common symbols
3   Common symbol       size              file
4
5   bss_var             0x3               main.o
6
7   Memory Configuration
8
9   Name             Origin             Length             Attributes
10  flash            0x00000000         0x20000000         xr
11  sram             0x20000000         0x20000000         xrw
12  *default*        0x00000000         0xffffffff
13
14  Linker script and memory map
15
16
17  .text           0x00000000      0x12b
18   *(.vectors*)
19   .vectors       0x00000000       0x10 startup.o
20                  0x00000000                g_p_fn_Vectors
21   *(.text*)
22   .text          0x00000010       0x80 main.o
23                  0x00000010                main
24   .text          0x00000090       0x98 startup.o
25                  0x00000090                H_Fault_Handler
26                  0x00000090                Default_Handler
27                  0x00000090                NMI_Handler
28                  0x0000009c                Reset_Handler
29   *(.rodata)
30   .rodata        0x00000128        0x3 main.o
31                  0x00000128                const_var
32                  0x0000012b                _E_text = .
33
34  .glue_7         0x0000012c        0x0
35   .glue_7        0x0000012c        0x0 linker stubs
36
37  .glue_7t        0x0000012c        0x0
38   .glue_7t       0x0000012c        0x0 linker stubs
39
40  .vfp11_veneer   0x0000012c        0x0
41   .vfp11_veneer  0x0000012c        0x0 linker stubs
42
43  .v4_bx          0x0000012c        0x0
44   .v4_bx         0x0000012c        0x0 linker stubs
45
46  .iplt           0x0000012c        0x0
47   .iplt          0x0000012c        0x0 main.o
48
49  .rel.dyn        0x0000012c        0x0
50   .rel.iplt      0x0000012c        0x0 main.o
```

- Debugging in Keil uvision output:



-