LAB2 REPORT

OBJECTIVE1: write LED toggle baremetal application on stm32f103c8t6 with custom startup, linker script and make file.

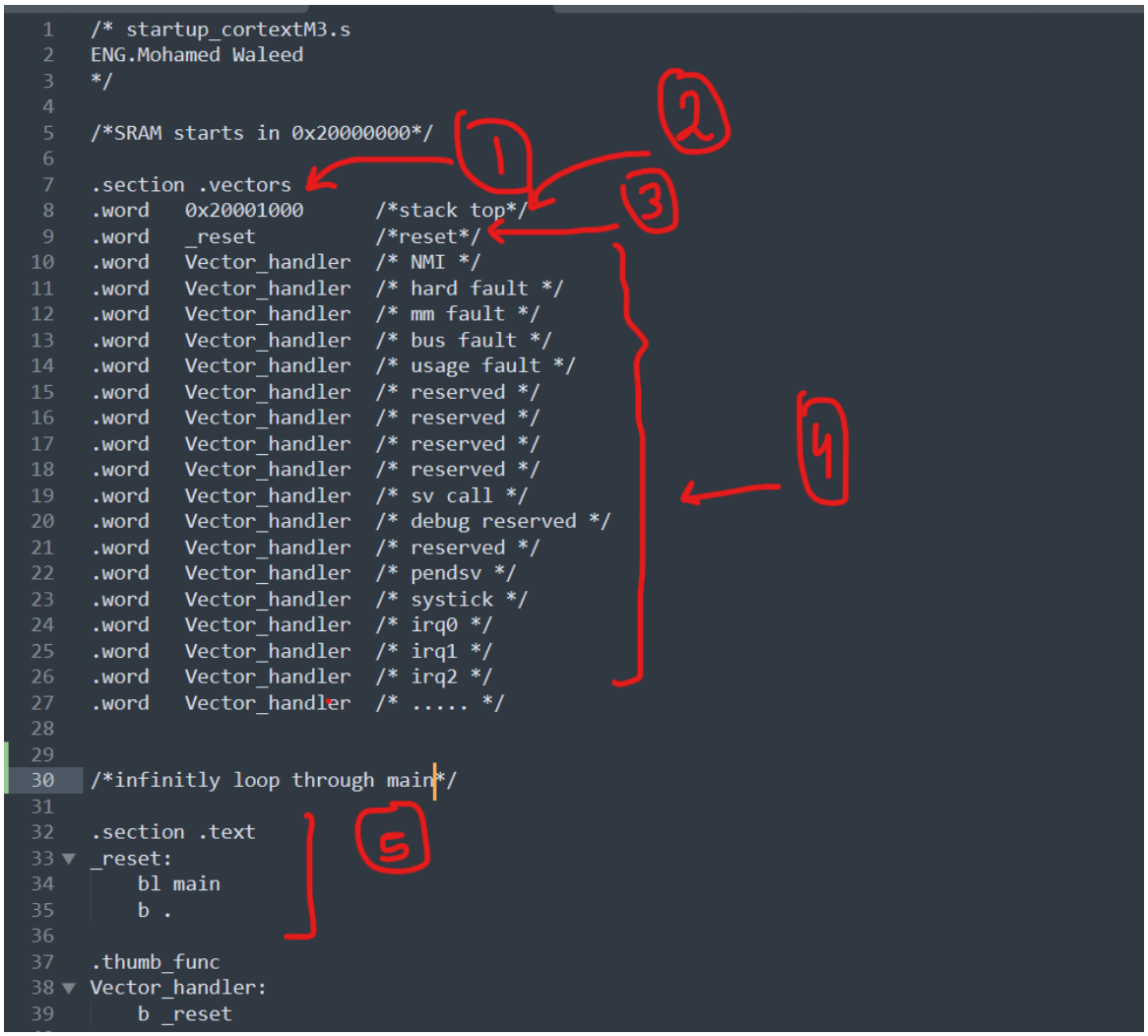
OBJECTIVE2: implement same thing with a c file startup instead of assembly.

NAME: Mohamed Waleed Elsayed.

APPLICATION

```c
#include "stdint.h"

#define RCC_BASE 0x40021000
#define GPIOA_BASE 0x40010800

#define RCC_APB2ENR *(volatile uint32_t *) (RCC_BASE + 0x18)
#define GPIOA_CRH   *(volatile uint32_t *) (GPIOA_BASE + 0x04)
#define GPIOA_ODR   *(volatile uint32_t *) (GPIOA_BASE + 0x0C)

typedef union{
    volatile uint32_t all_pins;
    struct {
        volatile uint32_t reserved:13;
        volatile uint32_t P_13:1;
    }Pins;
}R_ODR_t;

volatile R_ODR_t* R_ODR = (volatile R_ODR_t*) (GPIOA_BASE + 0X0C);
unsigned char g_variables [3] = {1,2,3};
unsigned char const con_variables [3] = {1,2,3};


int main(void)
{
    RCC_APB2ENR |= (1<<2);
    GPIOA_CRH   &= 0xFF0FFFFF;
    GPIOA_CRH   |= 0x00200000;

    while(1)
    {
        R_ODR->Pins.P_13 = 1;
        for(int i =0; i< 1000; i++);
        R_ODR->Pins.P_13 = 0;
        for(int i =0; i< 1000; i++);
    }
}
```

STARTUP.S



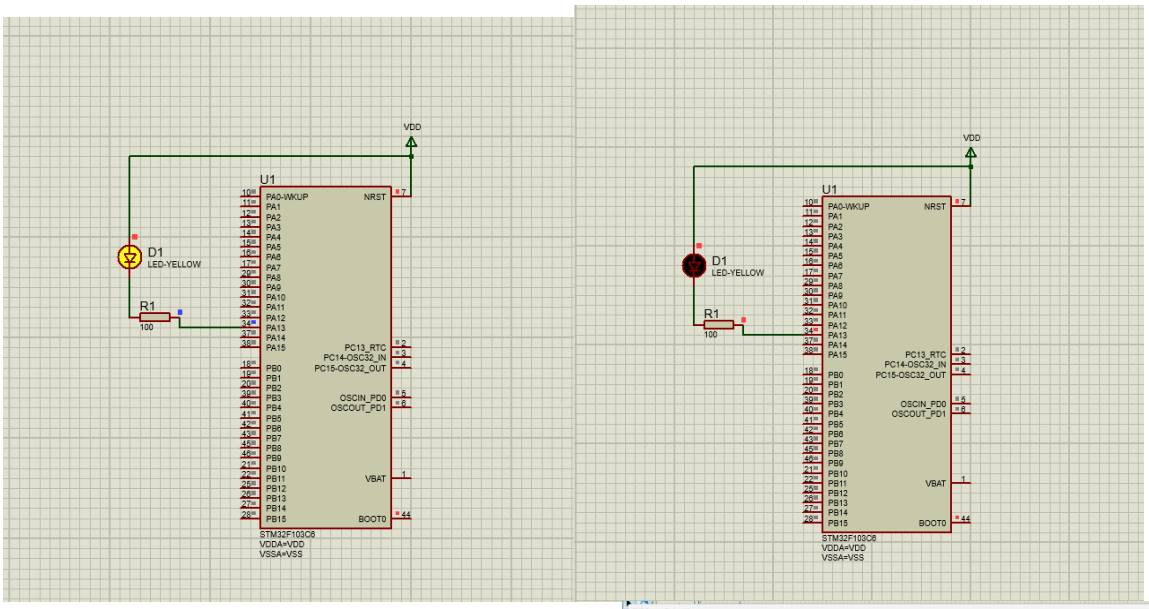
```
/* startup_cortextM3.s
ENG.Mohamed Waleed
*/

/*SRAM starts in 0x20000000*/

.section .vectors
.word   0x20001000      /*stack top*/
.word   _reset          /*reset*/
.word   Vector_handler  /* NMI */
.word   Vector_handler  /* hard fault */
.word   Vector_handler  /* mm fault */
.word   Vector_handler  /* bus fault */
.word   Vector_handler  /* usage fault */
.word   Vector_handler  /* reserved */
.word   Vector_handler  /* reserved */
.word   Vector_handler  /* reserved */
.word   Vector_handler  /* reserved */
.word   Vector_handler  /* sv call */
.word   Vector_handler  /* debug reserved */
.word   Vector_handler  /* reserved */
.word   Vector_handler  /* pendsv */
.word   Vector_handler  /* systick */
.word   Vector_handler  /* irq0 */
.word   Vector_handler  /* irq1 */
.word   Vector_handler  /* irq2 */
.word   Vector_handler  /* ..... */


/*infinitly loop through main*/

.section .text
_reset:
    bl main
    b .

.thumb_func
Vector_handler:
    b _reset
```

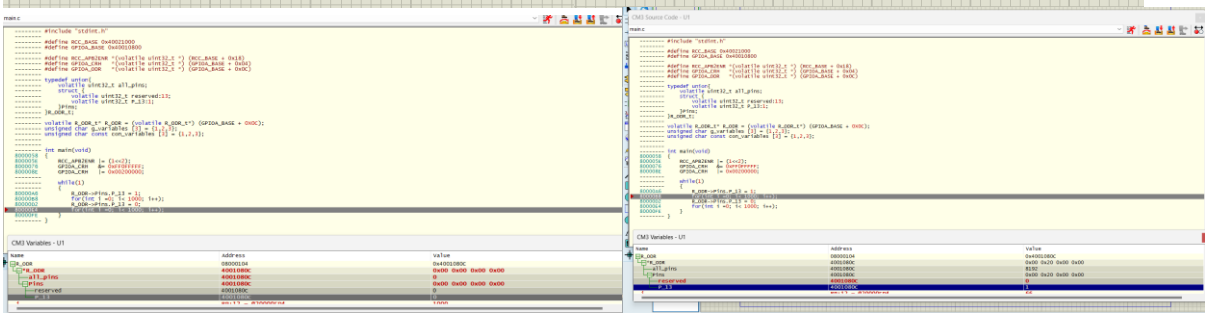
1- WE DEFINE VECTOR TABLE WHICH WILL BE THE FIRST THING FITCHED BY CPU AS A SECTION HEADER CALLED VECTORS
2- AS DICTATED BY DATASHEET FIRST FITCH IS SP ADDRESS WHICH IS A RANDOM MEMORY ADDRESS IN SRAM
3- SECOND IS RESET HANDLER WHICH IS 4 BYTES AFTER THE SP DUE TO THE USAGE OF ".WORD", IT JUMPS TO MAIN DIRECTLY
4- ERROR AND INT HANDLERS GROUPED TO ONE DEFAULT HANDLER THE REDIRECTS TO RESET
5- .TEXT SECTION COMES AFTER .VECTORS, IT HAS DEFINITIONS FOR RESET AND VECTOR HANDLER

```
Linker script and memory map


.text           0x08000000      0x184
 *(.vectors*)
 .vectors       0x08000000      0x1c startup.o
                0x08000000              vectors
 *(.text*)
 .text          0x0800001c      0xa8 main.o
                0x0800001c              main
 .text          0x080000c4      0xbc startup.o
                0x080000c4              h_fault_handler
                0x080000c4              mm_fault_handler
                0x080000c4              usage_fault_handler
                0x080000c4              defualt_handler
                0x080000c4              nmi_handler
                0x080000c4              bus_fault_handler
                0x080000d0              reset_handler
 *(.rodata*)
 .rodata        0x08000180       0x4 main.o
                0x08000180              con_variables
                0x08000184              _E_TEXT = .

.glue_7         0x08000184       0x0
 .glue_7        0x00000000       0x0 linker stubs
```

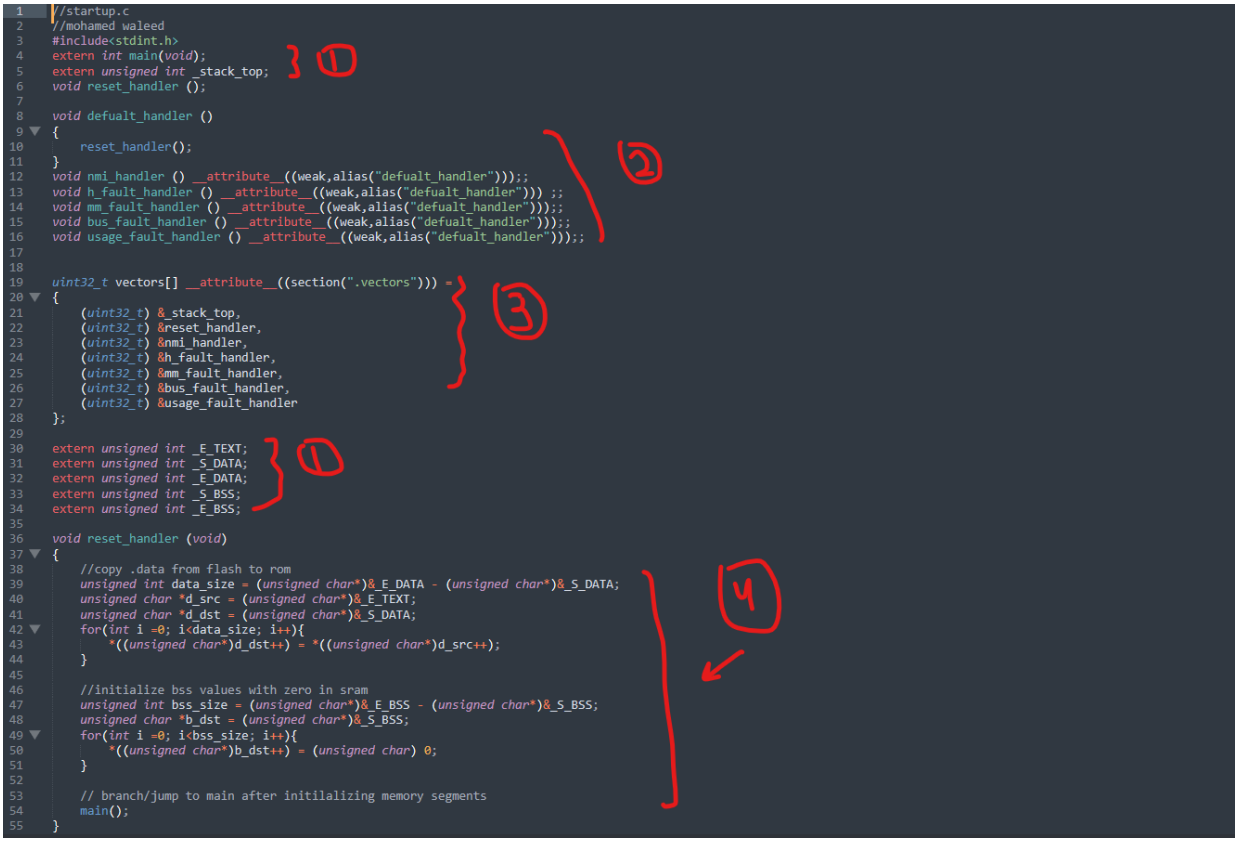
1-VECTORS IS FIRST ACCESSED

2-ALL DEFAULT HANDLERS HAVE THE SAME LOCATION





STARTUP.C



```c
//startup.c
//mohamed waleed
#include<stdint.h>
extern int main(void);
extern unsigned int _stack_top;
void reset_handler ();

void defualt_handler ()
{
    reset_handler();
}
void nmi_handler () __attribute__((weak,alias("defualt_handler")));;
void h_fault_handler () __attribute__((weak,alias("defualt_handler"))) ;;
void mm_fault_handler () __attribute__((weak,alias("defualt_handler")));;
void bus_fault_handler () __attribute__((weak,alias("defualt_handler")));;
void usage_fault_handler () __attribute__((weak,alias("defualt_handler")));;


uint32_t vectors[] __attribute__((section(".vectors"))) =
{
    (uint32_t) &_stack_top,
    (uint32_t) &reset_handler,
    (uint32_t) &nmi_handler,
    (uint32_t) &h_fault_handler,
    (uint32_t) &mm_fault_handler,
    (uint32_t) &bus_fault_handler,
    (uint32_t) &usage_fault_handler
};

extern unsigned int _E_TEXT;
extern unsigned int _S_DATA;
extern unsigned int _E_DATA;
extern unsigned int _S_BSS;
extern unsigned int _E_BSS;

void reset_handler (void)
{
    //copy .data from flash to rom
    unsigned int data_size = (unsigned char*)&_E_DATA - (unsigned char*)&_S_DATA;
    unsigned char *d_src = (unsigned char*)&_E_TEXT;
    unsigned char *d_dst = (unsigned char*)&_S_DATA;
    for(int i =0; i<data_size; i++){
        *((unsigned char*)d_dst++) = *((unsigned char*)d_src++);
    }

    //initialize bss values with zero in sram
    unsigned int bss_size = (unsigned char*)&_E_BSS - (unsigned char*)&_S_BSS;
    unsigned char *b_dst = (unsigned char*)&_S_BSS;
    for(int i =0; i<bss_size; i++){
        *((unsigned char*)b_dst++) = (unsigned char) 0;
    }

    // branch/jump to main after initilalizing memory segments
    main();
}
```

1- Declaration of global variables defined and changed in other files in the same directory , main in main, stack_top in linker, start and end of sections also in llinker.
2- Handlers prototypes, weak and alias attributes are attached to allow redifiintions and grouping into default handlers for ease of access modification and also efficient memory
3- Vectors section seen before in startup.s with the .vectors attribute, first thing being stack top, and then reset handler and then all other handlers
4- Reset has more functionality now, it copies data from flash to ram and initilaizes and zeroes out .bss in ram