گزارشکار جلسه پنجم ۲۰ آبان آزمایشگاه سیستمهای ریزپردازنده و مدارهای واسطه گروه ۳ آنوشا شریعتی ۹۹۲۳۰۴۱ مهشاد اکبری سریزدی ۹۹۲۳۰۹۳ امیرحسین منصوری

## آزمایش سری ۴:

بخش سوم: آزمایش روشن کردن LED که در سری ۳ انجام شد را با assembly تکرار کنید.

در ابتدا با دستور EQU. آدرس پایه رجیستر RCC\_AHB۱ که مربوط به کلاک است را تعریف میکنیم و با دستورات MOV و MOVT مقدار آن را در رجیستر عمومی  $r \cdot s$  ذخیره میکنیم. برای یک کردن بیت سوم این رجیستر مقدار هگز s در نهایت این مقدار را با دستور s در خانه با آدرس تعریف شده در رجیستر عمومی s ممراه آفست s در خیره میکنیم.

## //part2

".EQU clk\_adrsl,0X3800\n"
".EQU clk\_adrsh,0X4002\n"
".EQU value11 ,0X0008\n"
".EQU value1h,0X0000\n"
"MOV r0 , clk\_adrsl\n"
"MOVT r0 , clk\_adrsh\n"
"MOVT r1 , value1l\n"
"MOVT r1 , value1h\n"
"STR R1 , [R0 , #0X30]\n"

در ادامه آدرس رجیسترهای moder و moder را با دستور EQU. تعریف کرده و به ترتیب در رجیستر moder و r میریزیم. برای فعال کردن مود خروجی باید بیتهای r و r میریزیم. برای فعال کردن مود خروجی باید بیتهای r و r میریزیم. برای فعال کردن مود خروجی باید بیتهای r دخیره کرده و r کرد. برای انجام این کار عدد هگز r منتقل میکنیم. سپس مقدار این رجیستر را در خانه با آدرس ذخیره شده در r و آفست r دخیره میکنیم.

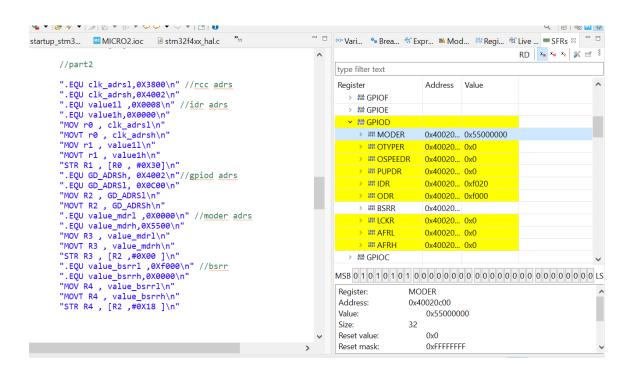
```
".EQU GD_ADRSh, 0X4002\n"//
".EQU GD_ADRS1, 0X0C00\n"
"MOV R2 , GD_ADRS1\n"
"MOVT R2 , GD_ADRSh\n"
".EQU value_mdrl ,0X0000\n"
".EQU value_mdrh,0X5500\n"
"MOV R3 , value_mdrl\n"
"MOVT R3 , value_mdrh\n"
"STR R3 , [R2 ,#0X00 ]\n"
```

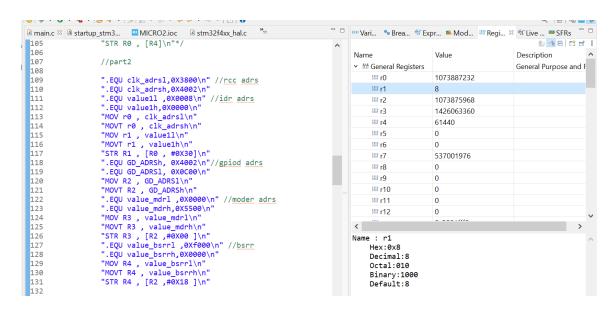
برای یک کردن بیتهای ۱۲و ۱۳ و ۱۴ و ۱۵ رجیستر bsrr را در در این رجیستر ۱۳ و ۱۳ و ۱۳ و ۱۳ و متغیر value\_bsrr ریخته و به رجیستر  $r \xi$  انتقال داده سپس مقدار این رجیستر را در خانه با آدرس ذخیره شده در  $r \xi$  و آفست  $x \xi$  د خیره میکنیم.

```
".EQU value_bsrrl ,0Xf000\n"
".EQU value_bsrrh,0X0000\n"
"MOV R4 , value_bsrrl\n"
"MOVT R4 , value_bsrrh\n"
"STR R4 , [R2 ,#0X18 ]\n"
```

توجه شود که کدهای بالا درون تابع ;(asm(); و پیش از حلقه تکرار میکروکنترلر نوشته شده است.

در ادامه با ران کردن برنامه و وارد شدن به مود دیباگ مقادیر رجیستر ها را در قسمت SFR مشاهده میکنیم. و همچنین با روشن شدن LED ها متوجه میشویم برنامه به درستی کار میکند.





آزمایش سری ۵:

بخش اول : تمرین جلسه قبل مربوط به روشن کردن LED را این بار در قالب فایل .s به صورت یک فایل اسمبلی جداگانه بنویسید.

یک فایل با پسوند S. ایجاد کرده و کد قسمت قبل را در آن کپی میکنیم. بالای برنامه خط یک و دو را اضافه میکنیم و در آخر برنامه نیز دستور BX LR که مقدار لینک رجیستر را ریترن میکند اضافه میکنیم.

```
🖹 😘 🎖 📅 🗖 🔞 stm32f4xx_hal.c 💆 project5.ioc 📑 assembly5.s 🔞 main.c 📑 *assmbly.s 🔯 🐧 startup_stm3... 🧦 *assmbly.s 🕏 *assmbly.s 🖂
Project Explorer ≅
                                                        _1.global led_on
2led_on:
> MICRO2
v 🔟 project5
   > 🗱 Binaries
                                                                 .EQU clk_adrs,0X40023800 @rcc adrs
                                                               LQU value1 ,0x000000008 @idr adrs
LDR r0 , =clk_adrs
LDR r1 ,= value1
STR R1 , [R0 , #0X30]
   🗸 🐸 Core
     > 🍅 Inc

→ Src

        assmbly.s
                                                                .EQU GD_ADRS, 0X40020C00 @gpiod adrs
                                                                LDR R2 , =GD_ADRS
.EQU value_mdr ,0X55000000 @moder adrs
         in main.c
        > @ stm32f4xx hal msp.c
                                                                LDR R3 , =value_mdr
STR R3 , [R2 ,#0X00 ]
        > @ stm32f4xx it.c
        > 🖟 sysmem.c
                                                                 .EQU value_bsrr ,0X0000f000 @bsrr
                                                                LDR R4 , =value_bsrr
STR R4 , [R2 ,#0X18 ]
        > 🖻 system_stm32f4xx.c
     > 🗁 Startup
   > @ CMSIS
```

سپس درون main.c اسم تابع نوشته شده به زبان اسمبلی که در اینجا led\_on است را با دستور extern تعریف کرده و داخل تابع وایل از آن استفاده میکنیم. و مشاهده میکنیم که برنامه به درستی کار میکند.

```
3 stm32f4xx_hal.c □ project5.ioc □ assembly5.s □ main.c □ □ assmbly.s
                                                               84
                                                                    /* USER CODE END SysInit */
                                                               85
 40 /* USER CODE END PM */
                                                               86
                                                                    /* Initialize all configured peripherals */
                                                               87
                                                                    MX_GPIO_Init();
 42 /* Private variables -----
                                                                    /* USER CODE BEGIN 2 */
                                                               88
                                                               89
 44 /* USER CODE BEGIN PV */
                                                                    /* USER CODE END 2 */
                                                               90
 45
                                                               91
 46 /* USER CODE END PV */
                                                               92
                                                                    /* Infinite loop */
                                                                    /* USER CODE BEGIN WHILE */
                                                               93
 48 /* Private function prototypes -----
                                                                    while (1)
                                                               95
 49 void SystemClock_Config(void);
                                                               96
 50 static void MX GPIO Init(void);
                                                               97
                                                                      /* USER CODE END WHILE */
 51 /* USER CODE BEGIN PFP *
                                                               98
                                                               99
 53 /* USER CODE END PFP */
                                                              100
                                                                        while(1){
                                                              101
                                                                           led_on();
 55⊖/* Private user code -----
                                                              102
 56 /* USER CODE BEGIN 0 */
 57 extern void led_on(void);
                                                                      /* USER CODE BEGIN 3 */
 58 /* USER CODE END 0 */
                                                                     * USER CODE END 3 */
 600 / **
                                                              107 }
 * @brief The application entry point.
     * @retval int
```