EXPERIMENT 3: GENERAL-PURPOSE INPUT/OUTPUT (GPIO)

Objectives

The objectives of Experiment 3 are

- to learn how to use
- ✔ GPIO Output Data Register (ODR),
- ✔ Reading Button Value using Input Data Register (IDR),
- ✔ Debugger,
- ✔ Bit Set Reset Register (BSRR),
- ✔ GPIO_ReadPin function

Apparatus Required:

- STM32CubeMx
- Keil µVision (MDK ARM)
- STM32 ST-Link Utility
- STM32F4 Microcontroller
- STM32F4 Reference Manual
- STM32F4 User Manual

Preliminary Work:

1. Study the GPIO (lecture 4) notes.
2. Write the codes of the experimental work in Keil µVision.

Experimental Work:

1. Reading Button Value (Button debouncing). You can understand whether your code is running and control the changes of the variable using debug (Figure 1->Start/Stop Debug Session) using the debugger. Come to the i variable and right click. We select “Add i to” and “Watch 1” (Figure 2). Then click to “Run” and follow the changes of variable i (Figure 1). Right-click on the i which is in the Watch 1

window to convert the i displayed as hexedecimal to decimal. Here you can reset the i value by pushing the reset button on STM32F4G-DISC card and doing a hardware reset.

```
// Program that increase the value of the variable i by one each time the button is pushed

if (GPIOA->IDR&0x0001) //Checking if the button is pushed using IDR
  {
   i=i+1; // Increase the value of the variable i by one each time the button is pushed
  }
```

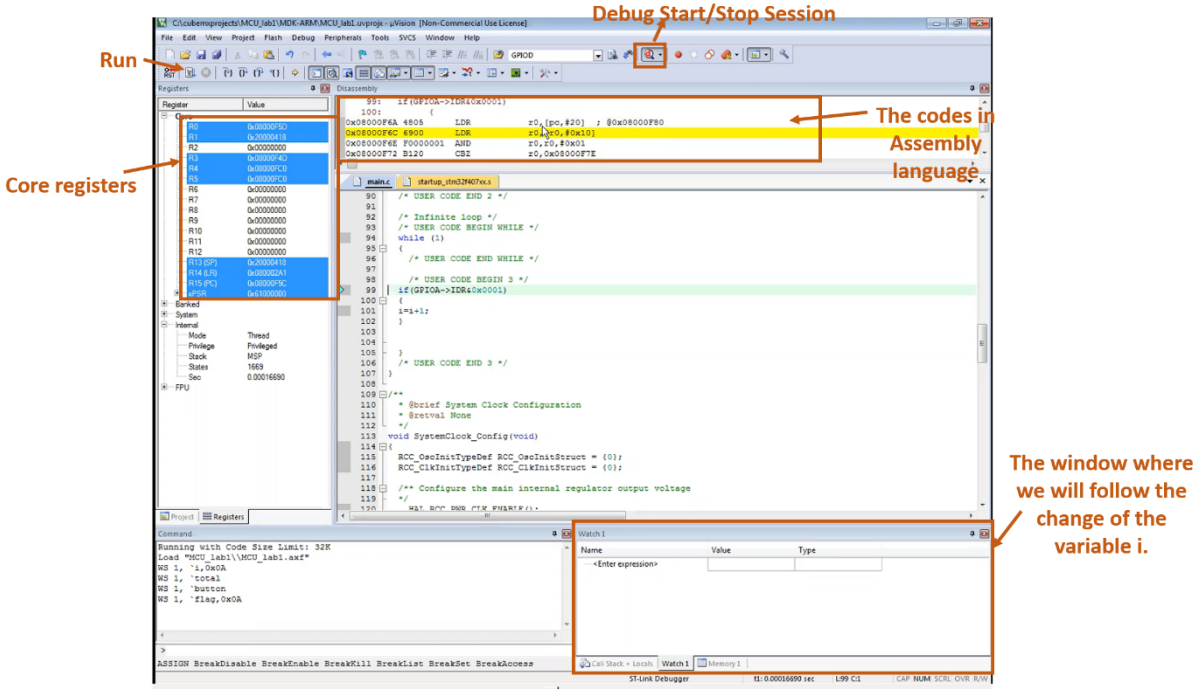

Figure 1

dim

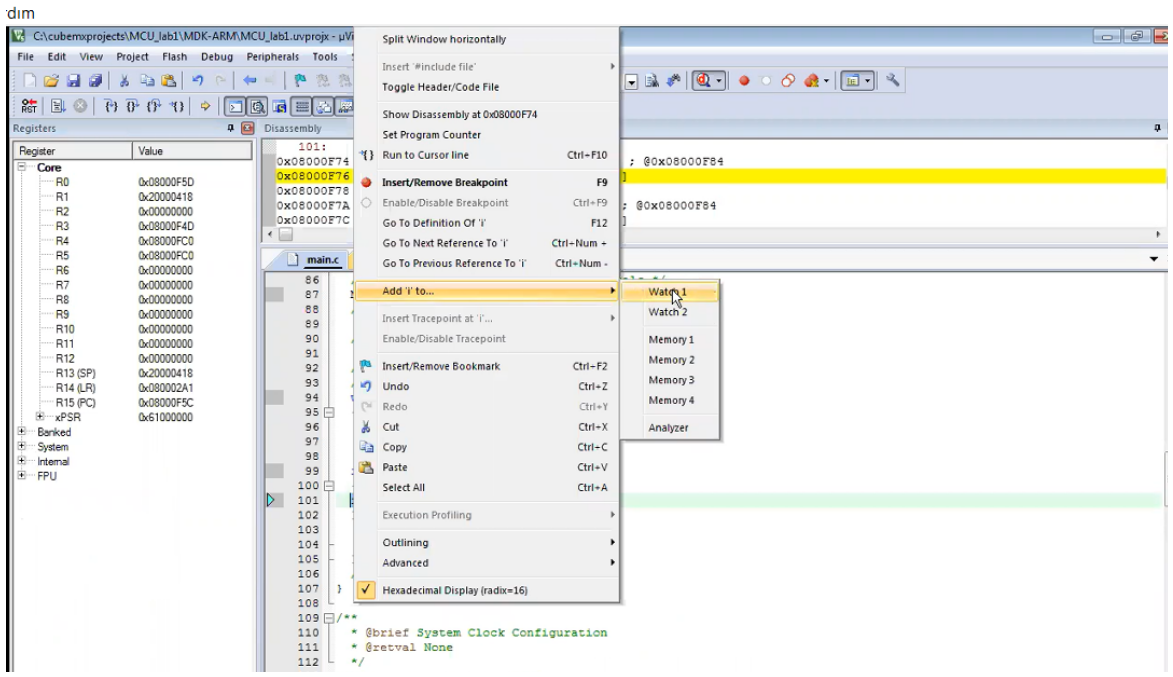

Figure 2

2. Reading Button Value (Prevent button debouncing using HAL_Delay)

```
// Program that increments the value of the variable i by one each time the button is pushed

        if (GPIOA->IDR&0x0001) //Checking if the button is pushed using IDR
          {
           i=i+1;
           HAL_Delay(200); //Wait 200 ms
           }
```

3. The program that turns on the LEDs when we push the button.

```
if (GPIOA->IDR&0x0001) //Check if the button is pushed
{
 i=i+1;
 HAL_Delay(200); //Wait 200 ms
 GPIOD->ODR=0xF000; //Assign 1 to the PD12, PD13, PD14 & PD15 pins and assign 0 to other PD pins
}
```

4. The program that turns the LEDs on when we push the button otherwise turns the LEDs off (use BSRR to assign logic 0 to the relevant pins)

```
GPIOD->BSRR=0xFFFF0000; //Reset ODR pins of the D port using BSRR
if (GPIOA->IDR&0x0001) //Check if the button is pushed using IDR
   {
    GPIOD->ODR=0xF000; ////Assign 1 to the PD12, PD13, PD14 & PD15 pins and assign 0 to other PD pins
    HAL_Delay(2000); //Wait 2 second
   }
```

5. The program that turns the LEDs on when we push the button otherwise turns the LEDs off (use ODR to assign logic 0 to the relevant pins)

```
GPIOD->ODR=0x0000; //Assign logic 0 to the pins at the D port
if (GPIOA->IDR&0x0001) //Checking if the button is pushed using IDR
  {
   i=i+1;
   HAL_Delay(200); //Wait 0.2 second
   GPIOD->ODR=0xF000; //Assign 1 to the PD12, PD13, PD14 & PD15 pins pins, assign 0 to other PD pins
    HAL_Delay(2000); //Wait 2 second
   }
```

6. The program that turns the LEDs on when we push the button otherwise turns the LEDs off (use ODR to assign logic 0 to the relevant pin and use ReadPin function to read the button).

```
 GPIOD->ODR=0x0000; //Assign logic 0 to the pins at the D port

if(HAL_GPIO_ReadPin(GPIOA,GPIO_PIN_0)) //Check if the button is pushed using ReadPin function

{
  i=i+1;
  HAL_Delay(200); //Wait 0.2 second
  GPIOD->ODR=0xF000; //Assign 1 to the PD12, PD13, PD14 & PD15 pins, assign 0 to other PD pins
  HAL_Delay(2000); //Wait 2 second
 }
```

7. A program that increases the value of i by one each time a button is pushed, at the same time, if i is an even number, toggles the related LED(which is connected to the PD12 pin), otherwise it turns off the LED (Control the i value using debugger). Use debug to see how to change the i variable.

```
while (1)
{

if(GPIOA->IDR&0x0001) //Check if the button is pushed using IDR

{

i=i+1; // Increase the value of i by one each time a button is pushed.

HAL_Delay(200);

}

// If i is an even number, let the led toggle otherwise led is off
```

```
if(i%2==0) //Check for an even number
{
 HAL_GPIO_TogglePin(GPIOD,GPIO_PIN_12); //Toggle 12th pin of the D port.
 HAL_Delay(200); //Wait 0.2 second
}
else
{
 GPIOD->BSRR=0xFFFF0000; //Reset ODR's pins of the D port using BSRR
}
}
```

8. Create a function that checks if the value of the variable i is an even number. If the value of the variable i is an even number, this function will turn on the LEDs connected to the 12th. and 14th pins of the D port, otherwise LEDs which are connected to 13th. and 15th pins of the D port. Write a program that then check if the button is pushed and increments the value of the variable i by one each time the button is pushed. Call the button function here. Each time you push the button, use the debugger to monitor what the value of the variable i is, and also observe which led is lit according to this value.

   (Write the button function part of the Private function prototypes (PFP) in the main.c file.
   Write it in while loop to check if button is pushed and call button function)

   ```
   //If it is an even number, the function that turn on the LEDs connected to the 12th. and 14th pins of the D port, otherwise the function that turns on the LEDs connected to the 13th. and 15th pins of the D port.

   void button (int a) // Creating a function named button
   {
   if (a%2==0) //Check if a is an even number
   {
   ```

```
        GPIOD->ODR=0x5000; //Turn on the LEDs which are connected to 12th. and 14th pins of the D port
}
else
{
GPIOD->ODR=0xA000; //Turn on the LEDs which are connected to 13th. and 15th pins of the D port
}
}


while (1)
{
//Checking if the button is pushed and calling the button function
if( GPIOA->IDR&0x0001) // Check if the button is pushed
{
i=i+1;
HAL_Delay(200);
}
button(i); //Call the button function
}
```