**Department of Computer Engineering**

BLG 351E  
Microcomputer Laboratory Experiment Report

Experiment No : 2

Experiment Date : 20.10.2017

Group Number : Friday - 13

Group Members :

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Surname** |
| 150160537 | OKAN | YILDIRIM |
| 150160546 | HASAN EMRE | ARI |
|  |  |  |
|  |  |  |

Laboratory Assistant : Yusuf Hüseyin Şahin

# Introduction

In this experiment we get used to use MSP430 Education Board, MSP430G2553 microcontroller and its assembly language in terms of input and output (GPIO). Before the experiment, we studied on Supplementary\_Chapter\_6\_General\_purpose \_IO document and get familiar with the experiment. We did preliminary work and reminded our background information.

# Experiment

## Part 1 Controlling Single Led via Push Button

According to the information given background information section and necessary supplementary material, we learn how to set, test and clear the input and outputs with “*bis.b, bit.c* and *bic.b”* commands. It is wanted from us to control LED 2 on Port 1 using the push button 3 on port 2. We wrote our assembly code given below:

SetupP1 bis.b #00000010b,&P1DIR ; OUTPUT/LED 2

SetupP2 bic.b #00000100b,&P2DIR ; INPUT/BUTTON3

bic.b #00000010b,&P1OUT ; CLEAR LED2

Mainloop bit.b #00000100b,&P2IN ; TEST BUTTON3

jnz ON

bic.b #00000010b,&P1OUT ; CLEAR LED2

jmp Mainloop

ON bis.b #00000010b,&P1OUT

jmp Mainloop

This code consists of three section as seen on the above. In first section , we set up the ports in P1 and P2 to be used as inputs and outputs. Then, we cleared the LED2 for precaution such as it has been on in advance. Then we wrote *Mainloop* section which is control the BUTTON3 on port2. If result of the test is zero, it means button is not pressed so that it must continue to control button and jumps to *Mainloop*. Otherwise it jumps to *ON* part which LED2 is turned on then jump to *Mainloop* again. In the *Mainloop* part, we clear the LED2 in order to turned it off.

## Part 2 Countıng Push Button Events

In this part, it is wanted to count how many times button is pressed and display the result on the LEDs on port1. Moreover, it is required to declare the variable instead of using register. We study on 1\_MSP430\_Intrdoduction document and learn how to declare the variable on assembly code. We add counter variable in data section as given below:

;-------------------------------------------------------------------------------

.text ; Assemble into program memory.

.retain ; Override ELF conditional linking

; and retain current section.

.retainrefs ; And retain any sections that have

; references to current section.

.data ; ADDED \*

counter .byte 0 : ADDED \*

;-------------------------------------------------------------------------------

Then we change the code on Part 1. It is given below:

SetupP1 bis.b #11111111b,&P1DIR ; ALL P1 LEDS` ACTIVE

SetupP2 bic.b #00000100b,&P2DIR ; INPUT/BUTTON3

mov.b counter,&P1OUT

Mainloop bit.b #00000100b,&P2IN ; TEST BUTTON3

jnz PRESS

jmp Timer

PRESS inc.b counter ;İncrement counter

mov.b counter,&P1OUT ; and display

Loop bit.b #00000100b,&P2IN ; TEST BUTTON3

jnz Loop ; if button is still pressed just jump loop again

jmp Timer

Timer mov.w #050000 , R15 ; Delay to R15

L1 dec.w R15 ; Decrement R15

jnz L1 ; Delay over ?

jmp Mainloop ; Again

This code consist of five section. First two section is similar to Part1 code. Additionally, we set up the counter for output. Then in *Mainloop* section, we control if the button is pressed or not. If it pressed, jump to the PRESS part and increment the counter and then display the value on LEDs as binary number.

Cornerstone of the program is measure how many times button is pressed. However, When we press in just a second, counter increasing multiple times. In order to solve this problem, we add two parts(*Loop* and *Timer*) to code. Actually it is a two layered approach to the solution. After the *PRESS* part, Loop part is working by the way. In *Loop* section, while button have been pressing, it jump to *Loop* again and counter is not incremented in this process. Moreover, we add a *Timer* to separate pressed and not pressed process from each other.

## Part 3 Addıng Reset Mechanısm to Counter

In this part, it wanted to add a reset mechanism to code which is written on the Part 2. We utilize another push button (Button2) to reset. When pressed it, counter is cleared. Code written by us, is given below:

SetupP1 bis.b #11111111b,&P1DIR ; ALL P1 LEDS` ACTIVE

SetupP2 bic.b #00000100b,&P2DIR ; INPUT/BUTTON3

SetupP3 bic.b #00000010b,&P2DIR ; INPUT/BUTTON3

mov.b counter,&P1OUT

Mainloop bit.b #00000010b,&P2IN ; TEST BUTTON3

jnz Reset

bit.b #00000100b,&P2IN ; TEST BUTTON3

jnz PRESS

jmp Timer

PRESS inc.b counter

mov.b counter,&P1OUT

Loop bit.b #00000100b,&P2IN ; TEST BUTTON3

jnz Loop

jmp Timer

Reset mov.b #0000000b,counter

mov.b counter,&P1OUT

jmp Mainloop

Timer mov.w #050000 , R15 ; Delay to R15

L1 dec.w R15 ; Decrement R15

jnz L1 ; Delay over ?

jmp Mainloop ; Again

In the first section of code we added a code to set second push button on the second port. We just improve the code given on the Part 2. We did some changes on *Mainloop* section and added Reset section. In *Mainloop* section, we check firstly, whether the second button (reset button) is pressed or not. If it pressed, jump to *Press* section and clear counter, display it (no LED is turn on) and jump to *Mainloop* again. If reset button is not pressed then check the third push button is pressed or not and it continues so on just like Part 1 or Part 2.

# Conclusion

We learn to how to use assembly code to controls input and output ports. We had a difficulty in solving a problem that counting by 1 counter when push button is pressed. Because counter increasing multiple times, when we press just one time in a second. However we implement good solution for it and it works. It was challenging but joyful also. On the other hand, it takes much more to time to declare and initialize variable due to the lack of knowledge about it. In spite of the fact that we searched it on the required document, it is not show exactly, how to and where write code.