Department of Computer Engineering

BLG 351E
Microcomputer Laboratory
Experiment Report

Experiment No : 2

Experiment Date : 21.10.2016

Group Number : Friday - 3

Group Members :

ID	Name	Surname
15140142	Gamze	Akyol
150130136	Mehmet Barış	Yaman
150130109	Güllü	Katık

Laboratory Assistant : Mustafa Ersen

1 INTRODUCTION

In this experiment, we learned I/O relation in the assembly language of MSP430 kit by implementing three different programs.

2 EXPERIMENT

2.1 FIRST PART - CONTROLLING SINGLE LED VIA PUSH BUTTON

The P1.1 pin (led) was controlled via pushing P2.0 button in this part of the experiment. If theP2.0 button was pushed, then P1.1 pin would be turned on, otherwise the output pin (P1.1) would be turned off.

The code with explanations for part 1:

```
Setup       clr.b   &P1OUT                  ; Clearing the data inside the P1 pin

            mov.b   #11111111b, &P1DIR      ; Initializing P1 as output

            mov.b   #00000000b, &P2DIR      ; Initializing P2 as input

Main        bit.b   #00000001b, &P2IN       ; Assigning P2.0 as input

            jnz     pushed                  ; If P2.0 is pushed (not zero), then branch to “pushed”

            jmp     Main                    ; If P2.0 is not pushed,

                                            ; then return to Main

pushed      bit.b   #00000010b, &P1OUT      ; Initializing P1.1 as output

            jnz     low                     ; If P1.1 is not pushed, then branch to “low”

            jmp     high                    ; If P1.1 is pushed, then branch to “high”

high        bis.b   #00000010b, &P1OUT      ; Set the P1.1 as output,

                                            ; and it turns the second pin (led) on.

            jmp     Main                    ; Return to Main

low         bic.b   #00000010b, &P1OUT      ; Clear P1.1, and it turns the second pin (led) off

            jmp     Main                    ; Return to Main
```

2.2 SECOND PART - COUNTING PUSH BUTTON EVENTS

In this part of the experiment, a counter was implemented which showed the numbers by led pins in P1. This counter counted in binary when every time the P2.0 button was pushed. (For example: 0000, 0001, 0010, 0011…)

The code for the part 2:

```
Setup       clr.b   &P1OUT               ; Clearing the data inside the P1 pin
            mov.b   #11111111b, &P1DIR   ; Initializing P1 as output
            mov.b   #00000000b, &P2DIR   ; Initializing P2 as input
            mov.b   #00000000b, R5       ; Clearing the all bits of the register (R5)
Main        bit.b   #00000001b, &P2IN    ; Assigning P2.0 as input
            jnz     Inc                  ; If P2.0 button is pushed (not zero),
                                         ; then branch to "Inc"
            jmp     Main                 ; If P2.0 button is not pushed,
                                         ; then return to the beginning of Main


Inc         mov.w    #060000, R6         ; Assigning a number to another register (R6)
                                         ; to wait for a while in the program flow
RetR        dec.w   R6                   ; Decreasing the number in the register (R6)…
             jnz    RetR                 ; …while it is not equal to zero
                                         ; (so the program will wait 60000 times to continue)


            inc.b   R5                   ; After waiting for a while,
                                         ; increase the number in R5 by 1…
            mov.b   R5, &P1OUT           ; ...and show this number as the output in P1
            jmp     Main                 ; Return to the beginning of Main
```

2.3 THIRD PART - ADDING RESET MECHANISM TO COUNTER

In the third part of the experiment, only reset mechanism is added to the second part.

The code for the part 3:

```
Setup   clr.b   &P1OUT              ; Clearing the data inside the P1 pin
        clr.b   &P2IN               ; Clearing the data inside the P2 pin
        mov.b   #11111111b, &P1DIR  ; Initializing P1 as output
        mov.b   #00000000b, &P2DIR  ; Initializing P2 as input
        mov.b   #00000000b, R5      ; Clearing the all bits of the register (R5)
Main    bit.b   #00000001b, &P2IN   ; Assigning P2.0 as input
        jnz     Inc                 ; If P2.0 button is pushed (not zero), then jump to “Inc"
        jmp     reset               ; If P2.0 button is not pushed, then jump to “reset”

reset   bit.b   #00000010b, &P2IN   ; Assigning P2.1 as input
        jnz     clear               ; If the P2.1 button is pushed (is not zero),
                                    ; then branch to “clear”
        jmp     Main                ; Else if, return to the beginning of Main
clear   clr.b   R5                  ; Clear the data in the register R5
        clr.b   &P1OUT              ; Clear the output P1, too.
                                    ; Because P1 shows the value of R5
        jmp     Main                ; Return to the beginning of Main

Inc     mov.w   #060000,R6          ; Assigning a number to another register (R6)
                                    ; to wait for a while in the program flow
RetR    dec.w   R6                  ; Decreasing the number in the register (R6)…
        jnz     RetR                ; …while it is not equal to zero
                                    ; (so the program will wait 60000 times to continue)

        inc.b   R5                  ; After waiting for a while,
                                    ; increase the number in R5 by 1…
        mov.b   R5, &P1OUT          ; ...and show this number as the output in P1
        jmp     Main                ; Return to the beginning of Main
```

3 CONCLUSION

In the last part of the experiment, we realized that some input pins in P2 were not worked in MSP430 kit. Because of that, we had to change these pins with another pins.

We have learned the implementation of input-output mechanism in the assembly language of MSP430 kit by different programs.