# Department of Computer Engineering

# BLG 351E
# Microcomputer Laboratory Experiment Report

Experiment No : 5

Experiment Date : 17.11.2017

Group Number : Friday - 13

Group Members :

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

Laboratory Assistant : Yusuf Hüseyin Şahin

# 1 INTRODUCTION

In this experiment we get used to use MSP430 Education Board, MSP430G2553 microcontroller and its assembly language in terms of driving 7-segment display and initializing interrupts. We enhanced the practical experience. Before the experiment, we studied on MSP430 User Guide – Chapter 8 document and Background information on experiment sheet. We get familiar with using 7-segment display and interrupts. We did preliminary work.

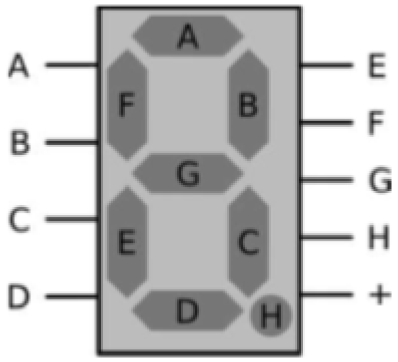
# 2 EXPERIMENT

## 2.1 PART 1 – COUNTER PROGRAM

In this program of the experiment, we implemented a counter that counts from 0 to 9 in ten seconds on 7-segment display. For delaying at each count, we used the part of code which is given on experiment sheet. It delays a second at the each count.

Before the experiment we drew a table to show representation of numbers from 0 to 9 on the 7-segment display. The table is given below:

| Integer | H | G | F | E | D | C | B | A |
|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| **1** | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| **2** | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| **3** | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| **4** | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| **5** | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| **6** | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| **7** | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| **8** | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **9** | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |



We used these values on the array in order and at the each count, 7-segmnet display displays the content of the this array.

Our program and detailed description is given below:

```
Setup       mov.b  #0000h,&P1OUT

            mov.b  #0FFFh,&P1DIR


Start       mov.w #array,R6       ; assign the address of the first element to R6 register

Count       mov.b @R6,&P1OUT      ; turn on the LEDs on the 7-segment display

            inc R6                ; next element

            call #Delay           ; delay a second
```

```
                cmp #lastElement,R6     ; compare if we are on the lest element or not
                jeq Start               ; jump to Start to count from 0
                jmp  Count              ; jump to count next element


Delay           mov.w   #0Ah ,R14                       ; Delay routine, wait a second
L2              mov.w   #07A00h ,R15
L1               dec.w   R15
                 jnz    L1
                 dec.w   R14
                 jnz    L2
                 ret
                ;Integer array
array    .byte  00111111b, 00000110b, 01011011b, 01001111b, 01100110b, 01101101b, 01111101b,
00000111b, 01111111b, 01101111b
lastElement
```

Firstly, we initialize GPIO Port 1 to activate and use 7-segment display. We assigned the address of the first element of the array to R6 resister. At the each iteration/count, we turn on the LEDs of 7-segment corresponds to these elements on the array which are binary representation of the numbers from 0 to 9. After we turned on the LEDs, we increment R6 register, thus next element is would turn on and we called delay routine to wait a second for each count. We compared the R6 register with the last element address to determine if the we come the last element  the array or not. If we came to the last element, it assigned 0 to R6 register and program starts to count from 0 again. Otherwise it jumps to turn on the next element of the  array.

## 2.2 Part 2 Interrupt subroutine

In this part, we implemented a interrupt subroutine by enhancing the main part of the previous part above. This subroutine able to count even numbers or odd numbers according to the external interrupt. We define a Boolean variable to determine count mode whether is counting even or odd number by toggling the this variable in the interrupt subroutine. We added required parts for using interrupt subroutine. We fallowed the steps given on the experiment sheet.

Our code is given below:

```
init_INT            bis.b #040h,&P2IE ; enable interrupt at P2.6
                    and.b #0BFh ,&P2SEL ; set 0 P2SEL.6
                    and.b #0BFh ,&P2SEL2 ; set 0 P2SEL2 .6
```

```
                              bis.b #040h,& P2IES ; high -to -low interrupt mode
                              clr &P2IFG ; clear the flag
                              eint ; enable interrupts
                              bis.b     #01d,&P2OUT

;-------------------------------------------------------------------------------
; Main loop here
;-------------------------------------------------------------------------------
Setup           mov.b   #0,&P1OUT
                mov.b   #0FFFh,&P1DIR
                mov.w   #array,R6
                mov.b   #0h,R10
                mov.b   #0h,R7
                mov.b   #lastElement,R8


Start1          mov.w #array,R6
                ;inc R6
Count1          mov.b @R6,&P1OUT
                cmp.b   R10,R7
                jnz     Inte1
                inc R6
                inc R6
                call #Delay
                cmp #lastElement,R6
                jge Start1
                jmp  Count1

Start2          mov.w #array,R6
                inc R6
```

```
Count2          mov.b @R6,&P1OUT
                cmp.b   R10,R7
                jnz       Inte2
                inc R6
                inc R6
                call #Delay
                cmp #lastElement,R6
                jge Start2
                jmp  Count2



Delay           mov.w   #0Ah ,R14
L2               mov.w   #07A00h ,R15
L1              dec.w   R15
                 jnz     L1
                 dec.w   R14
                 jnz     L2
                 ret

Inte1           dec R6
                mov.b   R10,R7
                jmp Count2

Inte2           dec R6
                mov.b   R10,R7
                jmp Count1

ISR             dint
                xor.b    #1h,R10
                clr       &P2IFG
```

```
                        eint

                        reti

                        ;Integer array

array     .byte  00111111b, 00000110b, 01011011b, 01001111b, 01100110b, 01101101b, 01111101b, 00000111b, 01111111b, 01101111b

lastElement

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

; Stack Pointer definition

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

          .global __STACK_END

          .sect   .stac

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

; Interrupt Vectors

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

          .sect   ".reset"              ; MSP430 RESET Vector

          .short  RESET

          .sect       ".int03"

           .short     ISR
```

# 3 CONCLUSION

We learn to how to use 7-segment display and interrupt. After the we implemented program on Part 2, we tested it. However, program had never enter the interrupt subroutine despite the fact that we press the push button for changing the push button. We thought that our program is correct. After we changed the kit and used another kit, program is worked really. We wasted too much time because of the kit.