Department of Computer Engineering

BLG 351E
Microcomputer Laboratory Experiment Report

Experiment No : 8

Experiment Date : 12.12.2016

Group Number : Monday -13

Group Members :

ID	Name	Surname
150130032	Baran	Kaya
40130051	Halil İbrahim	Onuş
150130002	Ahmet Seha	Çelenk

Laboratory Assistant : Gökhan Seçinti

• Introduction

In this experiment, we were expected to implement a game that drives 16x2 dot matrix LCD on the experiment board. Our game was going to use a predefined 2 words as an obstiacles and display the player triangle on LCD.

• Experiment

In this experiment, we were expected to play a game on LCD. For this purpose, first we have initialized registers to assign true connections of LCD. Then we have tried to implement right code to make the given 2 16-bit words (lineA&lineB) of the game written on LCD.

• First part

Code:

```
                clr.b &P2SEL
                clr.b &P2SEL2
Setup           mov.b #11111111b,&P1DIR
                mov.b #11111111b,&P2DIR
                call #Delay
                call #initLCD
printSCR
upline          mov.w #8000h, R5
jmp1            bit.w R5, &lineA
                jz shift
                mov.b #0FCh, r10
                call #sendDATA
                clrc
                rrc.w R5
                jmp jmp1
dwline          call #sendCMD
                mov.w #8000h, R5
jmp2            bit.w R5, &lineB
                jz shift1
                mov.b #0FCh, r10
                call #sendDATA
                clrc
                rrc.w R5
```

```
                    jmp jmp2
player              mov.b #0F7h, r10
                    call #sendDATA
                    jmp          finish
shift               clrc
                    rrc.w R5
                    mov.b #020h, r10
                    call #sendDATA
                    cmp #0001h, R5
                    jz player
                    cmp #0000h, R5
                    jz dwline
                    jmp jmp1
shift1              clrc
                    rrc.w R5
                    mov.b #020h, r10
                    call #sendDATA
                    cmp #0001h, R5
                    jz player
                    jmp jmp2

triggerEN   mov.b #01000000b, &P2OUT
                    mov.b #00000000b, &P2OUT
                    call #Delay
                    ret

triggerEEN  mov.b #11000000b, &P2OUT
                    mov.b #10000000b, &P2OUT
                    call #Delay
                    ret

Delay               mov.w #02h,R14 ; Delay to R14
L2                  mov.w #00500h,R15
L1                  dec.w R15 ; Decrement R15
                    jnz L1
```

```
                dec.w R14
                jnz L2
                ret

sendCMD     mov.b #00000000b, &P2OUT
                mov.b #0c0h, r10
                mov.b #0c0h, &P1OUT
                call #triggerEN
                rla.b r10
                rla.b r10
                rla.b r10
                rla.b r10
                mov.b r10, &P1OUT
                call #triggerEN
                ret

sendDATA    mov.b #10000000b, &P2OUT
                mov.b r10, &P1OUT
                call #triggerEEN
                rlc.b r10
                rlc.b r10
                rlc.b r10
                rlc.b r10
                mov.b r10, &P1OUT
                call #triggerEEN
                ret

finish          jmp finish
initLCD     mov.b #00110000b, &P1OUT
                call #triggerEN

                mov.b #00110000b, &P1OUT
                call #triggerEN

                mov.b #00110000b, &P1OUT
```

```
call #triggerEN

;4
mov.b #00100000b, &P1OUT
call #triggerEN


;5 FUNCTION NF
mov.b #00100000b, &P1OUT
call #triggerEN
;6
mov.b #10000000b, &P1OUT ;NF**
call #triggerEN

mov.b #00000000b, &P1OUT
call #triggerEN
mov.b #10000000b, &P1OUT
call #triggerEN

mov.b #00000000b, &P1OUT
call #triggerEN
mov.b #00010000b, &P1OUT
call #triggerEN

mov.b #00000000b, &P1OUT
call #triggerEN

mov.b #01100000b, &P1OUT ;I/D S
call #triggerEN

;init end
mov.b #00000000b, &P1OUT
call #triggerEN

mov.b #11100000b, &P1OUT
```

```
                    call #triggerEN
                    ret
;                   .data
;string .byte 0FCh,0Dh,0F7h,00h
;                   .data
;string1 .byte 020h
;                   .data
;string .byte " Game " ,0Dh ," Over !!! " ,00h
tpos .byte 00h ; 0 represents the bottom line , 1 rep the upper line
lineA .word 1802h ; obstacle bit -map of the upper line
lineB .word 8010h
;timerC .byte 00h
```

Firstly, we tried to print obstacle (#0FCh) and player (#0F7h) symbols to LCD and we succeed it (string array part). Then, we tried to implement lineA word for printing obstacles on the LCD. We assigned #8000h value to R5 register for checking lineA word's ones and zeros. For this part we used test instruction (bit.w) with R5 and lineA word then, if result of bit.w is 0 program jumps to shift label. In shift label, it clears carry then right shift R5 once and then print " " symbol (#020h) to the LCD. After that it jumps jmp1 label and tests R5 and lineA again. In jmp1 label, if bit.w operation's result is 1 then it prints obstacle symbol to the LCD, clears carry, right shifts R5 once and jumps back to the jmp1. In shift label it compares R5 with 0 and if R5 is 0 then program jumps dwline label which makes same operations for the second 16-bit line with lineB word. The only difference between the first and the second line'S printing labels is printing the player symbol. For player symbol it compares R5 with #0001h which is the last position of the second line and then it prints player symbol to the last position of the second line.

- ## Second & Third Part

In the second and the third part we could not do anything because of time. We worked on first part nearly two hours and we could not write any code for these two parts.

• Conclusion

To sum up, we made only the first part of the experiment because of time limitation. Maybe we can do the second and the third part if we have enough time.

I think this experiment takes too much time to implement the whole game on the board.