Department of Computer Engineering

BLG 351E
Microcomputer Laboratory
Experiment Report

Experiment No : 8

Experiment Date : 16.12.2016

Group Number : Friday - 3

Group Members :

ID	Name	Surname
150130136	Mehmet Barış	Yaman
150130109	Güllü	Katık
150140142	Gamze	Akyol

Laboratory Assistant : Mustafa Esengün

1 INTRODUCTION

In this experiment, we created a simple space game on the MSP 430 board by using 3 subprograms that work with interrupts having the functionality of printing to screen, Port2 control and timing.

2 EXPERIMENT

Firstly, we implemented the printSCR subroutine to print the requested symbols which are obstacles and the spaceship to LCD screen. The character codes of the desired characters are given in the booklet.

In the second part of the experiment, Port 2 was adjusted to steer the spaceship in LCD screen. Start/stop and up/down functionalities should be implemented and the spaceship should be controlled by using 3 inputs of Port 2, but we succeeded to implement only start/stop.

In the last part, the timer functionality is requested to add. However, we could not attach this functionality due to time constraints.

```
Setup       mov.w  #string, R5              ; Make R5 string array pointer
            bis.b  #11111111b, &P1DIR       ; P1 is directed to writeable
            bis.b  #11110000b, &P2DIR       ; P2 is directed to half writeable
; Slectors are cleared for interrupt operations
            clr.b  &P2SEL
            clr.b  &P2SEL2

; Initializing LCD by using the instructions explained in Experiment 7 by flowchart
; Delay and triggerEN is necessary for initializing the LCD

InitLCD     call   #Delay
            mov.w #00110000b, &P1OUT
            call   #triggerEN
            call   #Delay
            mov.w #00110000b, &P1OUT
            call   #triggerEN
            call   #Delay
            mov.w #00110000b, &P1OUT
            call   #triggerEN
            call   #Delay
```

```
mov.w #00100000b, &P1OUT
call    #triggerEN
call    #Delay
 mov.w #00100000b, &P1OUT
call    #triggerEN
mov.w #10000000b, &P1OUT
call    #triggerEN
call    #Delay
mov.w #00000000b, &P1OUT
call    #triggerEN
mov.w #10000000b, &P1OUT
call    #triggerEN
call    #Delay
mov.w #00000000b, &P1OUT
call    #triggerEN
mov.w  #00010000b, &P1OUT
call    #triggerEN
call    #Delay
mov.w #00000000b, &P1OUT
call    #triggerEN
mov.w  #01100000b, &P1OUT
call    #triggerEN
 call    #Delay
mov.w #00000000b, &P1OUT
call    #triggerEN
mov.w #11000000b, &P1OUT
call    #triggerEN
call    #Delay
mov.w #10000000b, &P1OUT
call    #triggerEN
mov.w #00000000b, &P1OUT
call    #triggerEN
```

```
; Printing the space game to LCD with respect to 2 words in the string array
                call    #check              ; Make game start with a button in P2IN
printSCR        mov.w   #lineA, R14         ; Make R14 a pointer that holds the word “Game”
                mov     @r14, R9            ; R9 is the value that R14 holds
                mov.w   #lineB, R14         ; Make R14 a pointer that holds the word “Over”
                mov     @r14, R10           ; R10 is the value that R14 holds
beginA          mov.w   #8000h, R11         ; Initialize R11 as holding first square in the LCD
loopA           cmp     #000h, R11          ; If R11 prints all of the squares in LCD
                jeq     secondLine          ; Jump the second line, for printing the word “Over”
                bit.w   R11, R9             ; If R9 has 1 in the square bit that R11 holds
                jz      emptyA              ; If not print emptiness
                call    #obstacle           ; If the bit is 1, print an obstacle to LCD
                clrc                        ; Clear the carry bit for the command “rrc”
                rrc.w   R11                 ; Make R11 hold the next bit of the LCD
                jmp     loopA
emptyA          call    #emptiness          ; Print emptiness to LCD
                clrc                        ; Clear the carry bit for the command “rrc”
                rrc.w   R11
                bit.b   #00000010b, &P2IN   ; Check the second bit for the P2IN in order to clarify
                                            ;whether to stop or not
                jz      loopA               ; If the button is not pushed go on printing line A
                jmp     shouldStopA         ; Otherwise, stop printing line A
; Changing the line of the LCD
secondLine      mov.w   #11000000b,&P1OUT
                call    #triggerEN
                mov.w   #00000000b,&P1OUT
                call    #triggerEN
                call    #Delay
                mov.w   #8000h, R11
; Instructions are the same as loopA
loopB           cmp     #001h, R11
                jeq     newLoop
```

```
                bit.w   R11, R10
                jz      emptyB
                call    #obstacle
                clrc
                rrc.w   R11
                bit.b   #00000010b, &P2IN
                jz      loopB
                jmp     shouldStopB
emptyB          call    #emptiness
                clrc
                rrc.w   R11
                jmp     loopB
newLoop         call    #triangle
                clrc
                rrc.w   R9
                clrc
                rrc.w   R10
; Make the program start to print in first line, in other words lineA
firstLine       mov.w   #00000000b,&P1OUT
                call    #triggerEN
                mov.w   #00000010b,&P1OUT
                call    #triggerEN
                call    #Delay
                jmp     beginA

; If P2.2 button is pressed stop subprogram is called

shouldStopA     call    #stop
                jmp     loopA

shouldStopB     call    #stop
                jmp     loopB
```

```
triggerEN       mov.w  #01000000b, &P2OUT
                mov.w  #00000000b, &P2OUT
                ret

; These subprograms work according to the MSP430 rules, in which least important four bits and most
; important four bits are printed in order
obstacle        mov.w  #10000000b, &P2OUT
                mov.b  #0FCh, R7              ; Obstacle ASCII value
                mov.b  R7, &P1OUT
; sendData subroutine
                mov.w  #10000000b, &P2OUT
                mov.w  #11000000b, &P2OUT
                mov.w  #10000000b, &P2OUT
                rla.w  R7
                rla.w  R7
                rla.w  R7
                rla.w  R7
                mov.w  R7, &P1OUT
                mov.w  #10000000b, &P2OUT
                mov.w  #11000000b, &P2OUT
                mov.w  #10000000b, &P2OUT
                call   #Delay
                ret

triangle        mov.w  #10000000b, &P2OUT
                mov.b  #0F7h, R7              ; Triangle ASCII value
                mov.b  R7, &P1OUT
; sendData subroutine
                mov.w  #10000000b, &P2OUT
                mov.w  #11000000b, &P2OUT
                mov.w  #10000000b, &P2OUT
                rla.w  R7
```

```
                rla.w   R7
                rla.w   R7
                rla.w   R7
                mov.w   R7, &P1OUT
                mov.w   #10000000b, &P2OUT
                mov.w   #11000000b, &P2OUT
                mov.w   #10000000b, &P2OUT
                call    #Delay
                ret

emptiness       mov.w   #10000000b, &P2OUT
                mov.b   #020h, R7               ; Emptiness ASCII value
                mov.b   R7, &P1OUT
; sendData subroutine
                mov.w   #10000000b, &P2OUT
                mov.w   #11000000b, &P2OUT
                mov.w   #10000000b, &P2OUT
                rla.w   R7
                rla.w   R7
                rla.w   R7
                rla.w   R7
                mov.w   R7, &P1OUT
                mov.w   #10000000b, &P2OUT
                mov.w   #11000000b, &P2OUT
                mov.w   #10000000b, &P2OUT
                call    #Delay
                ret

Delay           mov.w   #01h,R15
T1              mov.w   #07A00h,R14
T2              dec.w   R14
                jnz     T2
```

```
                dec.w   R15
                jnz     T1
                ret

check           clr.b   r3                  ; Not important! Is just written for distinguishing
                                            ; check subprogram and keep branch in it
keep            bit.b   #00000010b, &P2IN   ; check the second bit of P2
                jz      keep                ; stop goes on if not pressed
                jmp     action              ; action is needed if pressed
action          ret

stop            clr.b   r3                  ; used for distinguishing like “check” subprogram
halt            bit.b   #00000010b, &P2IN   ; check for second bit of P2 to stop or not
                jz      halt
                ret

.data
string  .byte " Game " ,0Dh," Over  !!! " ,00h
tpos    .byte 00h       ; 0 represents  the  bottom  line, 1 represents the upper line
lineA   .word  1802h    ; obstacle bit -map of the  upper line
lineB   .word  8010h
timerC  .byte 00h
```

3 CONCLUSION

In this experiment, we have learned how to create a game with just two words. We have learned how to print obstacles, emptiness and a triangular from the first part of the experiment.

We had some difficulties in the second part. We have adjusted the P2 inputs so that, when the user presses the second button of P2, sometimes it stopped the game and sometimes the game went on. We understood that this problem came from the Delay subprogram, since p2 button is pressed generally on delay subprogram. That’s why, the button did not work always as expected.

We could not do the third part of the experiment and the up down part of the second part.