**Department of Computer Engineering**

BLG 351E  
Microcomputer Laboratory Experiment Report

Experiment No : 7

Experiment Date : 01.12.2017

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 implemented a program that is display the predefined char array on LCD by using MSP430 Education Board, MSP430G2553 microcontroller and its assembly language. Before the experiment, we studied on Background information and Experiment sheet in detail. Especially LCD initialization and commands links are given on the experiment sheets are beneficial understand the LCD architecture and how to initialize and manage the LCD. We also studied on datasheet of the LCD display (HD44780U named document). We did preliminary work well.

# Experiment

In this program of the experiment, we write code according to the Instruction Set and initialization flow chart given on the experiment sheet and other links given on the experiment sheet. We wrote general explanations below and detailed descriptions are given on the code as comment line.

We firstly wrote the general architecture of the code. We create our char array as "ITU - Comp. Eng.",0Dh,"MC Lab. 2017",00h. 0Dh corresponds to \n and 00h corresponds to \0. We wrote subroutine names which we would use. Then, we fulfilled these subroutines’ contents.

Firstly, we wrote the delay subroutines. We modified the code 1 second delay provided in Experiment 5 and wrote three different delay which are delays more than 100ms, 4.1ms and 100us. These delay are given on the initialization flow chart. Actually we can wrote only one delay, yet we give importance the details. These delays are important for LCD to function properly.

Then, we wrote the triggerEN subroutine which is just changes the value of EN to high (1) then it changes it back to low (0). We firstly set the seventh (sixth for from 0) element of the port 2 and then clear it. This subroutine is used frequently enable the commands when I want to send a command to LCD.

Then, we wrote the initLCD part which was the one of the most important part of the program. WE wrote this part according to the initialization flow chart. Step by step, commands are sent to the LCD and it initialized. we selected the values of the command according to the command table and link given below table. Detailed description is given as command line for this part.

Then, we wrote the Open part which is open the LCD by enabling the display. We set the D, C, B values so that all display, cursor and blink of cursor were enabled.

Then, we wrote Setup part which is made set required ports, clean flags and assign the first address of the array to R6 register.

Then we, wrote the Print part which is the another most important part of program to display the characters on the LCD. In this part we can use the sendDATA and sendCMD subroutines which are given the experiment explanation, yet we did not use due to being small part of code write this codes when we need it. When we send data to LCD ,we set RS whereas clear it when we want to send command. We firstly send Data so that set RS. In Print subroutine, we firstly check if the value on the R8 register is 00h. In other words, we check the char array is finished or not. If it finished jump to finish part and Print subroutine is return the main part. Otherwise it continues to display characters. Then we assigned the content of R8 (next character on the address) to R5 register which is hold the characters on the array in order for each loop. We give this data to P1OUT, yet just upper nibble 4 bits are send. Then we call the triggrEN as we need. Now, we should send remain 4 bits so that we us rla command four times and shift bits. Then we send data to LCD in similar way. For each loop, characters are send to the LCD in this way and then we check the if we pass the second line or not. We compare the content of the array address with the 0Dh and if it is equal it jumps to secondLİne label otherwise it continues to jumps continue label and increment the R8 (address of array) and jump to the start label. If we jump to secondLine label, firstly we clear the RS because we send command to the LCD to pass to second line. We set the DDRAM address to 40. Because second line starts from the 40th character. We did this part 4bits - 4bits again. Finally we set RS 1 again because we send data to LCD now. We increment the R8 in similar way to the continue part and jump Start to display next character on the second line. All remain characters are print on the second line as the same way on Print subrotine. ?????start2 mevzusu

Then we wrote the main part of program. We firstly initialize the LCD by calling the initLCD. Secondly, call to open for opening LCD and calling print to display the content of array. Finally it jumps to Setup and infinite loop continues so on.

Our program and detailed description is given below:

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

; MSP430 Assembler Code Template for use with TI Code Composer Studio

;

;

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

.cdecls C,LIST,"msp430.h" ; Include device header file

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

.def RESET ; Export program entry-point to

; make it known to linker.

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

.text ; Assemble into program memory.

.retain ; Override ELF conditional linking

; and retain current section.

.retainrefs ; And retain any sections that have

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

RESET mov.w #\_\_STACK\_END,SP ; Initialize stackpointer

StopWDT mov.w #WDTPW|WDTHOLD,&WDTCTL ; Stop watchdog timer

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

; Main loop here

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

Setup clr.b &P2SEL ;clear the flags

clr.b &P2SEL2

bis.b #0ffh, &P1DIR ;set all P1DIR

bis.b #11000000b, &P2DIR ;set only uppest two P2DIR

mov.w #string,R8 ; assign the first element address of array to the R8

Main call #initLCD ;initialize LCD

call #Open ;open LCD

call #Print ; Print the content of the array

jmp Setup ;turn setup

Print mov.b #10000000b,&P2OUT ;set RS due to the sending data to the LCD

start cmp.b #00h,0(R8) ;control if the array is finished

jeq finish ;if finish jump to finish and end up print

mov.b @R8,R5 ; assign charater which is on the array to the R5

mov.b R5,&P1OUT ; send the data to the LCD, yet only 4 bits

call #triggerEN ; enable it

rla R5 ; for sending other 4 bits of data

rla R5

rla R5

rla R5

mov.b R5,&P1OUT ;send remain 4 bits to the LCD

call #triggerEN ;enable it

cmp.b #0Dh,0(R8) ;compare if the character is "\n"

jne start2 ; if it is not continue to start2

jmp secondLine ; otherwise go to secondLine label

start2 cmp.b #00h,0(R8) ;control if the array is finished

jne continue ; if it is not finished conctinue

mov.b #00000000b,&P2OUT ; otherwise necessary codes to send command to the LCD

mov.b #00000000b,&P1OUT

call #triggerEN

mov.b #00000000b,&P1OUT

call #triggerEN

finish call #delay1 ;delay and end up the ğrint subrouitine

ret

continue inc.w R8 ;next char and go to start of print

jmp start

secondLine mov.b #00000000b,&P2OUT ; clear RS so that send commmand to LCD

mov.b #10100000b,&P1OUT ; set the DDRAM address to 40 - upper 4bits

call #triggerEN

mov.b #10000000b,&P1OUT ; lower 4bits

call #triggerEN

;call #delay1

mov.b #10000000b,&P2OUT ;set RS so that send data to LCD

inc.w R8 ; next character, remain characters are printed on second line

jmp start

initLCD mov.b #00000000b,&P2OUT ;clear RS so that send commmand to LCD

call #triggerEN

call #delay1 ;more than 100ms

mov.b #00110000b,&P1OUT ;Special case of 'Function Set' (lower four bits are irrelevant)

call #triggerEN

call #delay2 ; more than4.1 ms

mov.b #00110000b,&P1OUT ;Special case of 'Function Set' (lower four bits are irrelevant)

call #triggerEN

call #delay3 ;more than 100us

mov.b #00110000b,&P1OUT;Special case of 'Function Set' (lower four bits are irrelevant)

call #triggerEN

call #delay3 ;more than 100us

mov.b #00100000b,&P1OUT ;initial 'Function Set' to change interface

call #triggerEN

call #delay3 ;more than 100us

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*8 bit to 4 bit \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

mov.b #00100000b,&P1OUT ;upper 4bits

call #triggerEN

mov.b #10000000b,&P1OUT ;lower 4bits 'Function Set' (I = 1, N=1 it means I use secondline)

call #triggerEN

call #delay3 ;;more than 100us

;upper 4bits

mov.b #00000000b,&P1OUT ;'Display ON/OFF Control' (D=1, C=0, B=0 )

call #triggerEN

mov.b #10000000b,&P1OUT ;lower 4bits

call #triggerEN

call #delay3

;upper 4bits

mov.b #00000000b,&P1OUT ;'Clear Display' (no configurable bits )

call #triggerEN

mov.b #00010000b,&P1OUT ;lower 4bits

call #triggerEN

call #delay2

mov.b #00000000b,&P1OUT ;upper 4bits -- Entry mod set I/D =1 cursor move direction

call #triggerEN ;S=0 not shift the display

mov.b #01100000b,&P1OUT ;lower 4bits

call #triggerEN

ret

Open mov.b #00000000b,&P2OUT ;clear RS so that send commmand to LCD

mov.b #00000000b,&P1OUT ;upper 4bits -- 'Display ON/OFF Control'(D=1, C=1, B=1 )

call #triggerEN ; we want to display cursor and blink of cursor also

mov.b #11110000b,&P1OUT ;lower 4bits

call #triggerEN

ret

delay1 mov.w #0Ah ,R14

L2 mov.w #0C4E0h ,R15 ;3150\*10 = 31500 >= 100ms

L1 dec.w R15

jnz L1

dec.w R14

jnz L2

ret

delay2 mov.w #0Ah ,R14

L4 mov.w #0820h ,R15 ;130\*10 = 1300 >= 4.1ms

L3 dec.w R15

jnz L3

dec.w R14

jnz L4

ret

delay3 mov.w #08h ,R14

L6 mov.w #04h ,R15 ;4\*8 = 32 >= 100us

L5 dec.w R15

jnz L5

dec.w R14

jnz L6

ret

triggerEN bis.b #01000000b,&P2OUT

bic.b #01000000b,&P2OUT

ret

string .byte "ITU - Comp. Eng.",0Dh,"MC Lab. 2017",00h ; references to current section.

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

; Stack Pointer definition

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

.global \_\_STACK\_END

.sect .stack

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

; Interrupt Vectors

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

.sect ".reset" ; MSP430 RESET Vector

.short RESET

# Conclusion

We enhanced the practical experience much more on board. We learned how to initialize and manage the LCD display. It was the most difficult experiment. Especially we had difficulty in passing second line. However, it is joyful to achieve the success for a such an experiment.