Experiment 2: LCD Display Interface

Estimated Time: *12 hours*

Prescribed Reading

Code of Conduct for use of Laboratory Equipment	*(PM-07-001.PDF)*
Policy on Academic Misconduct	*(PM-07-002.PDF)*
Introduction to ...	*(PM-07-003.PDF)*
Getting to know the Adapt9S12X Development Board	*(AM-07-001.PDF)*
The Freescale (HS12) Assembler	*(AM-07-002.PDF)*

Familiarity With

Full Assembly Instruction Set for the MC9S12XDP512	*(RU-07-001.PDF)*
Aapt9S12X Memory Map	*(RU-07-002.PDF)*
Usable Routines provided by Dbug12	*(RU-07-003.PDF)*
MC9S12XDP512 Interrupt Vector Map	*(RU-07-004.PDF)*
MC9S12XDP512: On Chip Peripherals	*(RU-07-005.PDF)*

Aim

To interface an LCD with the microprocessor to perform functions of entering strings and editing them by scrolling.

Introduction

LCD displays are low powered and are often used in portable instrumentation. The LCD unit we will be using in this experiment is able to store 2 lines of 40 characters showing 16 characters on each line of the screen at one time. These lines are automatically wrapped so that moving the characters to the right or left will result in some characters leaving the screen and others appearing at the opposite side. The LCD is interfaced to the Adapt912 via a custom built PCB. The PCB takes care of powering the LCD off the Adapt9S12X's supply and sets the contrast of the LCD.

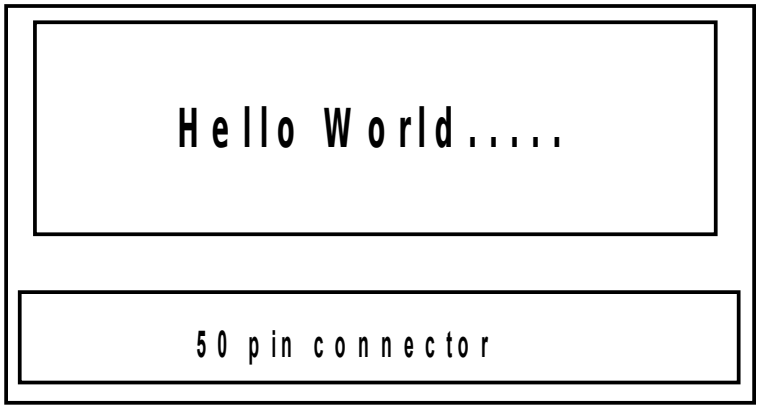

Figure 1: Diagram of the custom built LCD

Command Structure

The LCD unit recognizes several commands that are used to control the display. The commands are sent to the LCD from the microprocessor, using a combination of high/low signals to its register select (RS), read/write (R/W) and data byte (DB) lines.

To send a command to the LCD unit the command is loaded onto port PA connected to the 8 bit port DB, and R/W and RS lines connected to port PB. The LCD is enabled through the transition low-high-low (see the timing diagram) of RS. Note: Because the enable signal must be longer than a set time, a delay must be used between turning the enable on and off. The transformation of the data lines from the Adapt9S12X to the LCD is shown in the following table.

Adapt9S12X	LCD	Description
PA0-PA7	*DB*	*8 bit Data Line*
PB7	*E*	*Enable Signal*
PB6	*R/W*	*Read / Write Signal*
PB5	*RS*	*Resource Select Line*

The configurations of the lines to the LCD for all the instructions you will be using are shown below. The final program to perform these functions should follow the basic structure of the data flow diagram given at the end of this experiment.

	Code											E.T. (Fosc)
Instruction	RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Description	= 270kHz
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM and set DDRAM address to 00H from AC	1.53ms
Return Home	0	0	0	0	0	0	0	0	1	–	Sets DD RAM address to "0011" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.53ms
Entry Mode SET	0	0	0	0	0	0	0	1	1/D	SH	Assign cursor moving direction and enable shifting of entire display	39μ S
Display ON/OFF Control	0	0	0	0	0	0	1	D	C	B	Set display (D), cursor (C) and blinking cursor (B) control bit	39μ S
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	–	–	Set cursor moving and display shift control bit and direction without changing DDRAM data.	39μ S
Function Set	0	0	0	0	1	DL	N	F	–	–	Sets interface data length (DL: 8-bit / 4-bit), number of display lines (N: 2-lines / 1-line) and display font type (F: 5x11 dots / 5x8 dots)	39μ S
Set CG RAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Sets CG RAM address in address counter	39μ S
Set DD RAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Sets DD RAM address in address counter	39μ S
Read 'BUSY' Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not, can be known by reading BF. The contents of the address counter can also be read.	0μ S
Write DATA to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Writes data into internal RAM (DD RAM / CG RAM)	43μ S
Read DATA from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Reads data from internal RAM (DD RAM / CG RAM)	43μ S

Figure 2:LCD Instruction Set [taken from Dick Smith Data Sheet].

Part A: Initializing the LCD

Connect the LCD to the H2 header of the Adapt9S12X. Once it is connected you should see that the top line of the LCD is partially lit. The LCD is connected to the Adapt9S12X through the GPIO ports A and B, so the first task that your program must do is to set up ports A and B as outputs. Create a new project in Code Warrior called "C2" and modify it as per the previous experiment. Write a subroutine called Init_Ports, that performs the task outlined in the pseudo code below.

```
Module Init_Ports(IN: OUT:)
Begin
      //Set both port A and B for Output
      Write $FF to DDRA
      Write $FF to DDRB
      Set E = 0
End
```

In order to get the LCD to initialize properly a strict initialization sequence must be followed. The initialization procedure is outlined below in pseudo code or pictorially in figure 4. Each of these control words must be written to the display in the correct order with the adequate delays between them. The delays mentioned on initialization sequence are minimum delays that must be enforced.

```
Program PartA
Begin
      Delay(IN: 30000)
      Init_Ports(IN: OUT:)
      set A=%00110000
      set B=%00000000
      Write_LCD(IN: A,B OUT:)
      Delay(IN: 39)
      set A=%00001111
      set B=%00000000
      Write_LCD(IN: A,B OUT:)
      Delay(IN: 39)
      set A=%00000001
      set B=%00000000
      Write_LCD(IN: A,B OUT:)
      Delay(IN: 1530)
      set A=%00000110
      set B=%00000000
      Write_LCD(IN: A,B OUT:)
      Delay(IN: 1530)
End
```

When writing data to the LCD, whether it is a character or one of the initialization strings the write cycle timing diagram must be followed as shown in figure 5. The LCD write cycle (Figure 5) requires you to set the values of RS, RW and DB and wait 240ns the E line is then set and left high for an additional 240ns. The E line is then set low and the write cycle is complete. This explanation assumes that the E line is originally set low.

Write a subroutine called Write_LCD to implement the LCD write cycle. The subroutine must allow the values of RS, RW and DB to be passed to it via accumulators A and B. The pseudo code for Write_LCD is shown below.

```
Module Write_LCD(IN: A, B OUT:)
Begin
      Write A to PortA Data Register
      Write B to PortB Data Register
      Wait for atleast 40ns
      Set E high (PB7)
      Wait for atleast 240ns
      Set E low (PB7)
      Wait for atleast 240ns
End
```

Using the subroutine WriteLCD implement the LCD initialization procedure. This will also require you to write a delay subroutine.

```
Module Delay(IN: value_in_useconds)
Begin
      //See exercise 11 in Experiment C1.
End
```

After successfully completing this procedure the LCD should have an empty screen with a flashing cursor in its upper-left corner.

It is encourage that you view the data sheet for the LCD for additional information not included in this lab script. The data sheet can be found on the Maxwell website.

Part B: Displaying a String of Text

*Using the program created in part A, add an additional subroutine called **lputchar** to display a single character on the LCD. The character to be printed should be passed to this routine via accumulator B. Test this program by print the character 'A" to the LCD.*

Show this working program to your demonstrator.

*Next modify your program by adding a subroutine called **lprint** that prints a string to the LCD. This subroutine should use the subroutine **lputchar** to display the characters that make up the string. The address of the string should be passed to the subroutine through accumulator X and the string should be null terminated (0x0). Use the string "Griffith" to test this program.*

Next modify this program to display a string that is entered by the user. You may use the subroutine GetS that you used in Experiment 1 to accommodate the entering of a string. The string should be echoed to the LCD after the carriage return (CR) character has been typed.

Show this working program to your demonstrator.

Part C: Adding the Menu

Modify the program in Part B so that once the LCD has been initialized, it prompts the user to press a key. If the 'E' key is pressed then runs the text entry subroutine you developed in Part B. If the 'C' key is pressed then it clears the LCD screen.

Part D: Scrolling

Write a subroutine to scroll the screen, when the letter 'L' is pressed it scrolls to the left or scrolls right when the letter 'R' is pressed. The screen should scroll until it returns to the original position. i.e. Shift the screen 40 times to the left or the right.
Finally, the instructions for each key should be displayed on the PC terminal as part of the menu subroutine.

Show this working program to your demonstrator. A report of this experiment must include the code of the final working program.

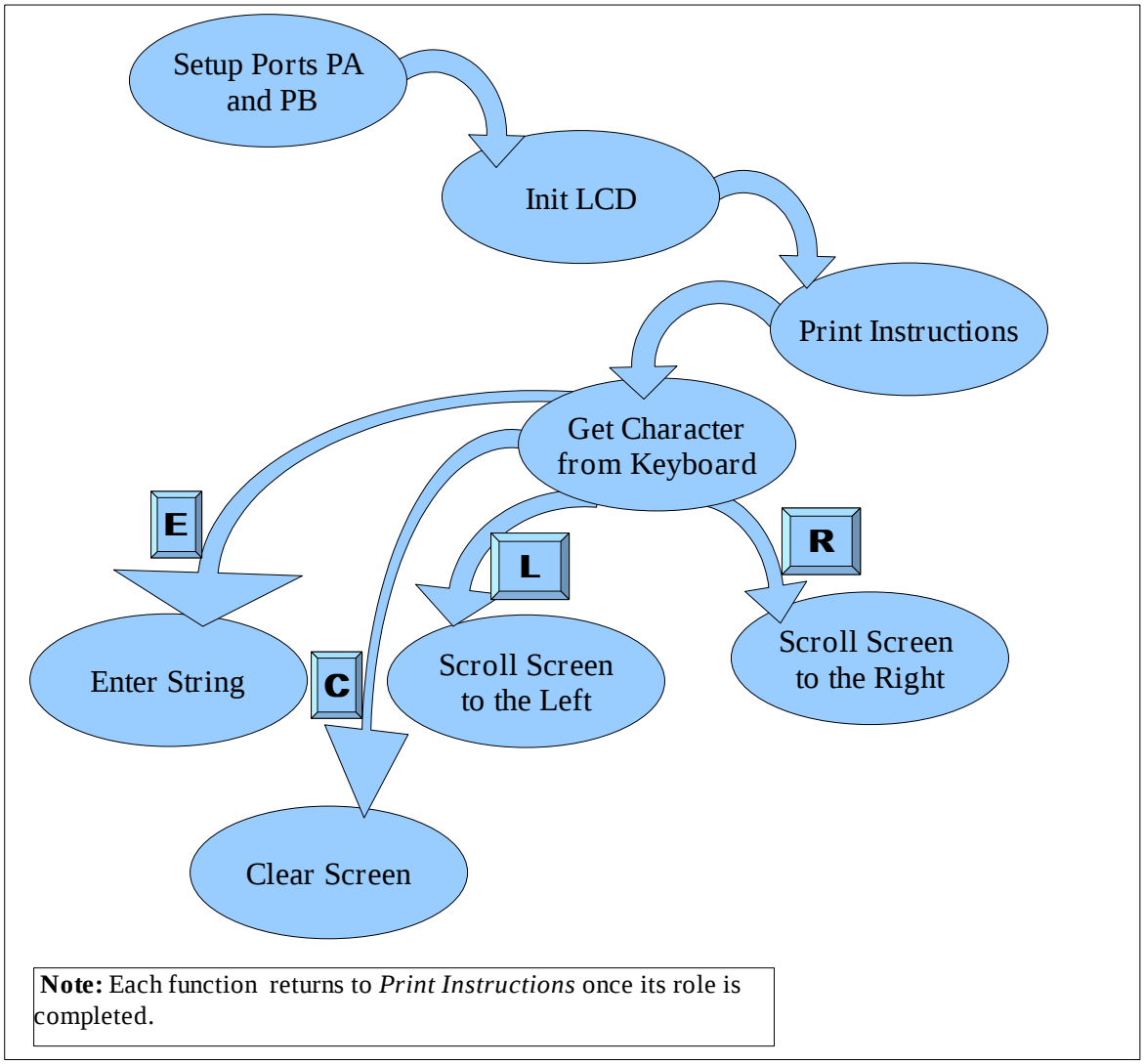

Figure 3: Diagram showing the flow of the proposed system.

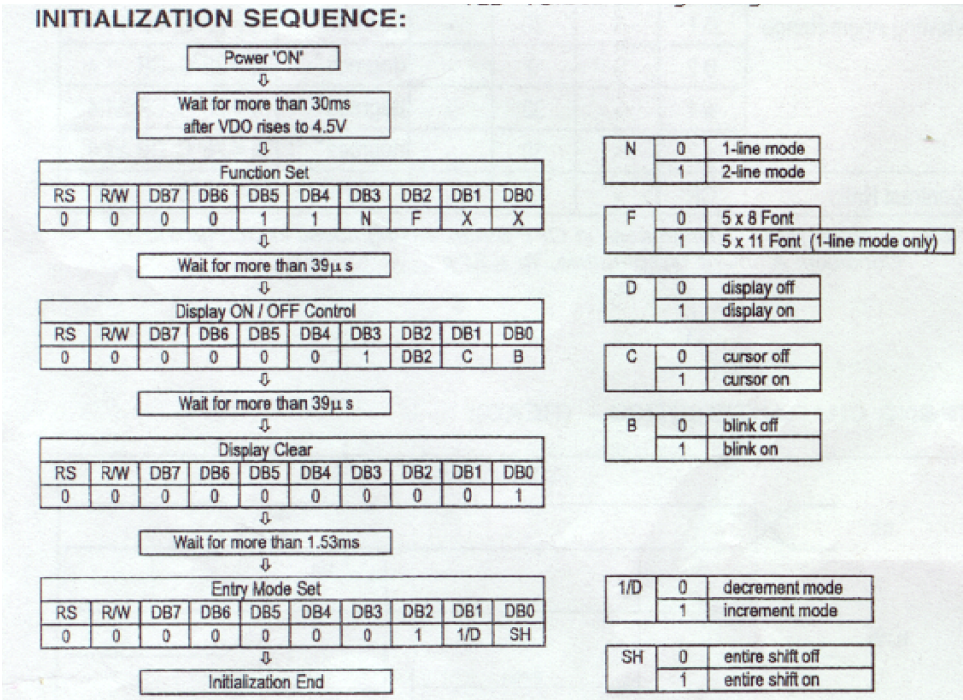

Figure 4: Initialization Sequence and Timing Diagram for Write Operations to the LCD [from Dick Smith Data Sheet]

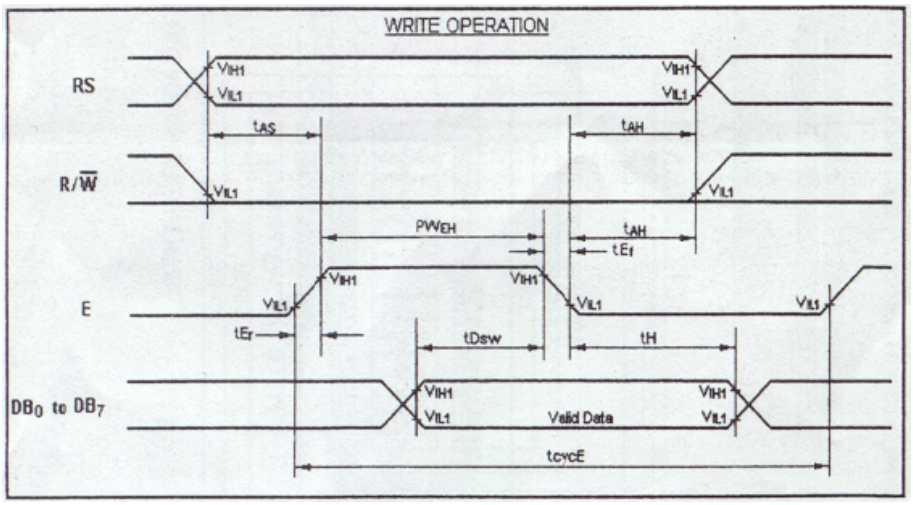

Figure 5: Writing timing Diagram[From Dick Smith Data Sheet]

Timing Characteristics - Read / Write Operation

Item	Symbol	Min	Typ	Max	Unit
Enable Cycle Time	tcycE	500	--	--	ns
Enable Pulse Width (high level)	PWEH	230	--	--	ns
Enable Rise / Fall Time	tEr, tEf	--	--	20	ns
Address Setup Time (RS, R/W to E)	tAS	40	--	--	ns
Address Hold Time	tAH	10	--	--	ns
Data Delay Time - Read	tDDR	--	--	120	ns
Data Hold Time - Read	tDRH	5	--	--	ns
Data Setup Time - Write	tDSW	80	--	--	ns
Data Hold Time - Write	tH	10	--	--	ns