

LCD Interfacing with a PIC24
7th Laboratory Report for ECE 383
Microcomputers

Submitted by
Patrick Brooks
11650957

The University of Alabama
Tuscaloosa, AL 35487

4/2/21

Abstract:

The main objectives of this lab are to introduce students to LCD interfacing with the PIC24. They will be working in the PIC24 ecosystem while learning how to implement and use an LCD screen. Task one instructs students to build the schematic given on their breadboard. Task two asks students to output their name and school email to the LCD. Task three requires students to create a counter program that has certain characteristics described in the lab instructions.

Introduction:

The goal of this lab is to introduce students to LCD interfacing with the PIC24. They will be working in the PIC24 ecosystem while creating code to output what is required according to the lab instructions. Task one instructs students to build the schematic, given in the lab instructions, on their breadboard. Task two asks students to output their name and school email to the LCD by creating a C program inside of MPLAB. Task three requires students to create a counter program that has certain characteristics described in the lab instructions.

Procedure/Results:Task 1: Connecting the LCD to the PIC24

- a. Recreate given schematic

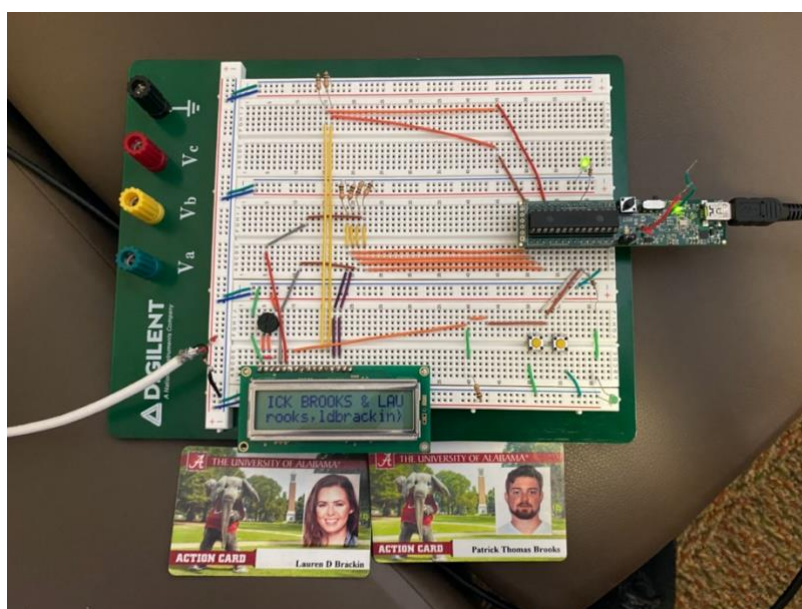


Figure 1. Deliverable 1

Task 2: Character Output to LCD

a. Use *lcd4bit.mcp* in “chap8” files for a template for this task

i. Create a program that outputs your name and email to the screen

<https://youtu.be/eW3qBe9AUa>

Figure 2. Deliverable 2

```

1 #include "pic24_all.h"
2
3 #define RS_HIGH() _LATB8 = 1
4 #define RS_LOW() _LATB8 = 0
5 #define CONFIG_RS() CONFIG_RB8_AS_DIO_OUTPUT()
6
7 #define RW_HIGH() _LATB9 = 1
8 #define RW_LOW() _LATB9 = 0
9 #define CONFIG_RW() CONFIG_RB9_AS_DIO_OUTPUT()
10
11 #define E_HIGH() _LATB13 = 1
12 #define E_LOW() _LATB13 = 0
13 #define CONFIG_E() CONFIG_RB13_AS_DIO_OUTPUT()
14
15 #define LCD4C _LATB4
16 #define LCD5C _LATB5
17 #define LCD6C _LATB6
18 #define LCD7C _LATB7
19 #define LCD7I _RB7
20
21 #define CONFIG_LCD4_AS_INPUT() CONFIG_RB4_AS_DIO_INPUT()
22 #define CONFIG_LCD4_AS_OUTPUT() CONFIG_RB4_AS_DIO_OUTPUT()
23 #define CONFIG_LCD5_AS_INPUT() CONFIG_RB5_AS_DIO_INPUT()
24 #define CONFIG_LCD5_AS_OUTPUT() CONFIG_RB5_AS_DIO_OUTPUT()
25 #define CONFIG_LCD6_AS_INPUT() CONFIG_RB6_AS_DIO_INPUT()
26 #define CONFIG_LCD6_AS_OUTPUT() CONFIG_RB6_AS_DIO_OUTPUT()
27 #define CONFIG_LCD7_AS_INPUT() CONFIG_RB7_AS_DIO_INPUT()
28 #define CONFIG_LCD7_AS_OUTPUT() CONFIG_RB7_AS_DIO_OUTPUT()
29
30 #define GET_BUSY_FLAG() LCD7I
31
32 /**
33  * Functions above this line must be redefined for
34  * your particular PICmicro-to-LCD interface
35  */
36
37 //Configure 4-bit data bus for output
38 void configBusAsOutLCD(void) {
39     RW_LOW(); //RMW to stop LCD from driving pins
40     CONFIG_LCD4_AS_OUTPUT(); //D4
41     CONFIG_LCD5_AS_OUTPUT(); //D5
42     CONFIG_LCD6_AS_OUTPUT(); //D6
43     CONFIG_LCD7_AS_OUTPUT(); //D7
44 }
45
46 //Configure 4-bit data bus for input
47 void configBusAsInLCD(void) {
48     CONFIG_LCD4_AS_INPUT(); //D4
49     CONFIG_LCD5_AS_INPUT(); //D5
50     CONFIG_LCD6_AS_INPUT(); //D6
51     CONFIG_LCD7_AS_INPUT(); //D7
52 }
53
54 // R/W = 1, for read
55
56 //Output lower 4-bits of u8_c to LCD data lines
57 void outputToBusLCD(uint8_t u8_c) {
58     LCD4C = u8_c & 0x01; //D4
59     LCD5C = u8_c >> 1 & 0x01; //D5
60     LCD6C = u8_c >> 2 & 0x01; //D6
61     LCD7C = u8_c >> 3 & 0x01; //D7
62 }
63
64 //Configure the control lines for the LCD
65 void configControlLCD(void) {
66     CONFIG_RS(); //RS
67     CONFIG_RW(); //RW
68     CONFIG_E(); //E
69     RW_LOW();
70     E_LOW();
71     RS_LOW();
72 }
73
74 //Pulse the E clock, 1 us delay around edges for
75 //setup/hold times
76 void pulseE(void) {
77     DELAY_US(1);
78     E_HIGH();
79     DELAY_US(1);
80     E_LOW();
81     DELAY_US(1);
82 }
83
84 /* Write a byte (u8_Cmd) to the LCD.
85  * u8_DataFlag is '1' if data byte, '0' if command byte
86  * u8_CheckBusy is '1' if must poll busy bit before write, else simply delay bc
87  * u8_SendBits is '1' if must send all 8 bits, else send only upper 4-bits
88  */
89 void writeLCD(uint8_t u8_Cmd, uint8_t u8_DataFlag,
90             uint8_t u8_CheckBusy, uint8_t u8_SendBits) {
91     uint8_t u8_BusyFlag;
92     uint8_t u8_u8tState;
93     if (u8_CheckBusy) {
94         RS_LOW(); //RS = 0 to check busy
95         // check busy
96         configBusAsInLCD(); //set data pins all inputs
97         u8_u8tState = _WDT; //save WDT enable state
98         CLRWDT(); //clear the WDT timer
99         _WDTEN = 1; //enable WDT to escape infinite wait
100         do {
101             E_HIGH();
102             DELAY_MS(10); // read upper 4 bits
103         } while (u8_u8tState == 0);
104         RS_HIGH(); // read upper 4 bits
105     }
106     outputToBusLCD(u8_Cmd >> 4); // send upper 4 bits
107     pulseE(); //pulse again for lower 4-bits
108     while (u8_BusyFlag); //wait for WDT enable state
109     _WDTEN = u8_u8tState; //restore WDT enable state
110 } else {
111     DELAY_MS(10); // don't use busy, just delay
112 }
113
114 configBusAsOutLCD();
115 if (u8_DataFlag & RS_HIGH(); // RS=1, data byte
116     else RS_LOW(); // RS=0, command byte
117     outputToBusLCD(u8_Cmd >> 4); // send upper 4 bits
118     pulseE();
119     if (u8_SendBits) {
120         outputToBusLCD(u8_Cmd); // send lower 4 bits
121         pulseE();
122     }
123 }
124
125 // Initialize the LCD, modify to suit your application and LCD
126 void initLCD() {
127     DELAY_MS(50); //wait for device to settle
128     writeLCD(0x28, 0, 0, 0); // 4 bit interface
129     writeLCD(0x28, 0, 0, 1); // 2 line display, 5x7 font
130     writeLCD(0x28, 0, 0, 1); // repeat
131     writeLCD(0x04, 0, 0, 1); // enable display
132     writeLCD(0x0C, 0, 0, 1); // turn display on; cursor, blink is off
133     writeLCD(0x01, 0, 0, 1); // clear display, move cursor to home
134     DELAY_MS(5);
135 }
136
137 //Output a string to the LCD
138 void outStringLCD(char *psz_s) {
139     while (*psz_s) {
140         writeLCD(*psz_s, 1, 1, 1);
141         psz_s++;
142     }
143 }
144
145 // added for lab requirements
146 void drawFrames() {
147     outStringLCD("PATRICK BROOKS & LAUREN BRACKIN");
148     writeLCD(0x0D, 0, 1, 1); // cursor to 2nd line
149     outStringLCD("[ptbrooks,ldbrackin@crimson.ua.edu]");
150 }
151
152 int main(void) {
153     //configure outputs to LCD as open-drain
154     _ODCB8 = 1;
155     _ODCB9 = 1;
156     _ODCB13 = 1;
157     _ODCB4 = 1;
158     _ODCB5 = 1;
159     _ODCB6 = 1;
160     _ODCB7 = 1;
161
162     configBasic(MELLO_M95); // Set up heartbeat, UART, print hello messa
163
164     configControlLCD(); //configure the LCD control lines
165     initLCD(); //initialize the LCD
166     drawFrames();
167
168     while (1) {
169         writeLCD(0x18, 0, 1, 1); // shift left
170         DELAY_MS(500);
171         doHeartbeat();
172     }
173 }
174

```

Figure 3. Deliverable 3

Task 3: Counter Output to LCD

a. Use *lcd4bit.mcp* in “chap8” files for a template for this task

i. Create a program that outputs a counter from *000-015*

ii. Once the counter hit *015*, counter resets

iii. If SW1 is pressed the counter will reset to *000*

<https://youtu.be/Sg8ta7pwU5M>

Figure 4. Deliverable 4

```

1  #include "pic24_all.h"
2  #include "stdio.h"
3
4  #define RS_HIGH()    _LATB0 = 1
5  #define RS_LOW()     _LATB0 = 0
6  #define CONFIG_RS()  CONFIG_RB0_AS_DIO_OUTPUT()
7
8  #define SW_HIGH()    _LATB9 = 1
9  #define SW_LOW()     _LATB9 = 0
10 #define CONFIG_SW()  CONFIG_RB9_AS_DIO_OUTPUT()
11
12 #define E_HIGH()     _LATB13 = 1
13 #define E_LOW()      _LATB13 = 0
14 #define CONFIG_E()   CONFIG_RB13_AS_DIO_OUTPUT()
15
16 #define LCD40        _LATB4
17 #define LCD50        _LATB6
18 #define LCD60        _LATB6
19 #define LCD70        _LATB7
20 #define LCD71        _RB7
21
22 #define LED1_LATB10 // MicroStick II definitions
23 #define CONFIG_LED1() CONFIG_RB10_AS_DIO_OUTPUT()
24
25 #define LED2_LATB1 // MicroStick II definitions
26 #define CONFIG_LED2() CONFIG_RB1_AS_DIO_OUTPUT()
27
28 #define CONFIG_LCD4_AS_INPUT() CONFIG_RB4_AS_DIO_INPUT()
29 #define CONFIG_LCD5_AS_INPUT() CONFIG_RB5_AS_DIO_INPUT()
30 #define CONFIG_LCD6_AS_INPUT() CONFIG_RB6_AS_DIO_INPUT()
31 #define CONFIG_LCD7_AS_INPUT() CONFIG_RB7_AS_DIO_INPUT()
32
33 #define CONFIG_LCD4_AS_OUTPUT() CONFIG_RB4_AS_DIO_OUTPUT()
34 #define CONFIG_LCD5_AS_OUTPUT() CONFIG_RB5_AS_DIO_OUTPUT()
35 #define CONFIG_LCD6_AS_OUTPUT() CONFIG_RB6_AS_DIO_OUTPUT()
36 #define CONFIG_LCD7_AS_OUTPUT() CONFIG_RB7_AS_DIO_OUTPUT()
37
38 #define SW1_RB14 // switch state
39 #define SW1_PRESSED() (SW1==0)
40 #define SW1_RELEASED() (SW1==1)
41
42 #define SW2_RB12 // switch state
43 #define SW2_PRESSED() (SW2==0)
44 #define SW2_RELEASED() (SW2==1)
45
46 inline void CONFIG_SW1() {
47     CONFIG_RB14_AS_DIO_INPUT();
48     ENABLE_RB14_PULLUP();
49     ENABLE_RB14_CN_INTERRUPT();
50 }
51
52 inline void CONFIG_SW2() {
53     CONFIG_RB12_AS_DIO_INPUT();
54     ENABLE_RB12_PULLUP();
55     ENABLE_RB12_CN_INTERRUPT();
56 }
57
58 #define GET_BUSY_FLAG() LCD71
59
60 /**
61  * Functions above this line must be redefined for
62  * your particular PICmicro-to-LCD interface
63  */
64
65 //Configure 4-bit data bus for output
66 void configBusOutLCD(void) {
67     SW_LOW(); //SW=0 to stop LCD from driving pins
68     CONFIG_LCD4_AS_OUTPUT(); //D4
69     CONFIG_LCD5_AS_OUTPUT(); //D5
70     CONFIG_LCD6_AS_OUTPUT(); //D6
71     CONFIG_LCD7_AS_OUTPUT(); //D7
72 }
73
74 //Configure 4-bit data bus for input
75 void configBusInLCD(void) {
76     CONFIG_LCD4_AS_INPUT(); //D4
77     CONFIG_LCD5_AS_INPUT(); //D5
78     CONFIG_LCD6_AS_INPUT(); //D6
79     CONFIG_LCD7_AS_INPUT(); //D7
80     SW_HIGH(); // R/W = 1, for read
81 }
82
83 //Output lower 4-bits of u8_c to LCD data lines
84 void outputToBusLCD(uint8_t u8_c) {
85     LCD40 = u8_c & 0x01; //D4
86     LCD50 = (u8_c >> 1) & 0x01; //D5
87     LCD60 = (u8_c >> 2) & 0x01; //D6
88     LCD70 = (u8_c >> 3) & 0x01; //D7
89 }
90
91 //Configure the control lines for the LCD
92 void configControlLCD(void) {
93     CONFIG_RS(); //RS
94     CONFIG_SW(); //SW
95     SW_LOW(); //E
96     E_LOW();
97     RS_LOW();
98 }
99
100 //Pulse the E clock, 1 us delay around edges for
101
102 //Pulse the E clock, 1 us delay around edges for
103 //setup/hold times
104 void pulseE(void) {
105     DELAY_US(1);
106     E_HIGH();
107     DELAY_US(1);
108     E_LOW();
109     DELAY_US(1);
110 }
111
112 /** Write a byte (u8_Cmd) to the LCD.
113  * u8_DataFlag is '1' if data byte, '0' if command byte
114  * u8_CheckBusy is '1' if must poll busy bit before write, else simply delay
115  * u8_Send8Bits is '1' if must send all 8 bits, else send only upper 4-bits
116  */
117 void writeLCD(uint8_t u8_Cmd, uint8_t u8_DataFlag,
118              uint8_t u8_CheckBusy, uint8_t u8_Send8Bits) {
119     uint8_t u8_BusyFlag;
120     uint8_t u8_u8State;
121     if (u8_CheckBusy) { //RS = 0 to check busy
122         RS_LOW();
123         // check busy
124         configBusInLCD(); //set data pins all inputs
125         u8_u8State = _SWTST; //save WDT enable state
126         CLSWDT(); //clear the WDT timer
127         _SWTST = 1; //enable WDT to escape infinite wait
128         do {
129             E_HIGH();
130             DELAY_US(1); // read upper 4 bits
131             u8_BusyFlag = GET_BUSY_FLAG();
132             E_LOW();
133             DELAY_US(1);
134             pulseE(); //pulse again for lower 4-bits
135         } while (u8_BusyFlag);
136         _SWTST = u8_u8State; //restore WDT enable state
137     } else {
138         DELAY_MS(10); // don't use busy, just delay
139     }
140     configBusOutLCD();
141     if (u8_DataFlag & RS_HIGH(); // RS=1, data byte
142     else RS_LOW(); // RS=0, command byte
143     outputToBusLCD(u8_Cmd >> 4); // send upper 4 bits
144     pulseE();
145     if (u8_Send8Bits) {
146         outputToBusLCD(u8_Cmd); // send lower 4 bits
147         pulseE();
148     }
149 }
150
151 // Initialize the LCD, modify to suit your application and LCD
152 void initLCD() {
153     DELAY_MS(50); //wait for device to settle
154     writeLCD(0x20, 0, 0, 0); // 4 bit interface
155     writeLCD(0x28, 0, 0, 1); // 2 line display, 5x7 font
156     writeLCD(0x01, 0, 0, 1); // reset
157 }

```

Figure 5. Deliverable 5.1

```

154 writeLCD(0x20,0,0,0); // 4 bit interface
155 writeLCD(0x20,0,0,1); // 2 line display, 5x7 font
156 writeLCD(0x20,0,0,1); // repeat
157 writeLCD(0x06,0,0,1); // enable display
158 writeLCD(0x0C,0,0,1); // turn display on; cursor, blink is off
159 writeLCD(0x01,0,0,1); // clear display, move cursor to home
160 DELAY_MS(3);
161 }
162
163 //Output a string to the LCD
164 void outStringLCD(char *psz_s) {
165     while (*psz_s) {
166         writeLCD(*psz_s, 1, 1, 1);
167         psz_s++;
168     }
169 }
170
171 int count = 0; // declaration of global counter and string variable
172 char str[3];
173
174 void printCount() {
175     writeLCD(0x01,0,0,1); // clear display, move cursor to home
176     sprintf(str, "%d", count);
177     outStringLCD(str);
178     if (count == 15) LED1 = 1;
179     else LED1 = 0;
180 }
181
182 void _ISRFAST__CHInterrupt(void) {
183     _CNIF = 0;
184     if (SW1_PRESSED()) {
185         while (SW1_PRESSED()) //debounce
186             count++;
187     }
188     if (SW2_PRESSED()) {
189         while (SW2_PRESSED()) //debounce
190             count = 0;
191     }
192     if (count == 14) {
193         count = 0;
194     }
195     printCount();
196 }
197
198 int main (void) {
199     //config switches and LEDs
200
201     CONFIG_SW1();
202     CONFIG_SW2();
203     CONFIG_LED1();
204     CONFIG_LED2();
205
206     LED1 = 1;
207
208     _CNIF = 0; //clear int flag
209     _CNIP = 2; //choose priority
210     _CNIE = 1; // enable CN general
211 }

```

```

204 CONFIG_LED1();
205 CONFIG_LED2();
206
207 LED1 = 1;
208
209 _CNIF = 0; //clear int flag
210 _CNIP = 2; //choose priority
211 _CNIE = 1; // enable CN general
212
213 configBasic(HELLO_MSG); // Set up heartbeat, UART, print hello mess
214
215 configControlLCD(); //configure the LCD control lines
216 initLCD(); //initialize the LCD
217
218 printCount();
219 while (1) {
220 }
221 }

```

Figure 5. Deliverable 5.2 (cont.)

Conclusion:

After completion of the lab students learned how to connect the LCD screen to the PIC24 system, revisited important foundational C language techniques, learned how to incorporate interrupts, and learned how to critically think through these problems. Students also became familiar with the pins on the PIC24 and how they can be used to perform different tasks. Students also revisited useful software tools implemented in MPLAB. Students also deepened their understanding of the lab report format which will be used throughout their academic endeavors.