

Lab 02

# LCD and Seven Segment Display

Logan Barber, Ian Nail

January 28, 2022

*Ian Nail*

*Logan Barber*

ME-4370 - Stephen Canfield

# 1 Executive Summary

In this lab, the objective is to output a recipe to a Liquid Crystal Display (LCD) and include a seven segment display as a timer. The minimum objectives were:

1. Tell the user what recipe is to follow.
2. Provide the recipe steps in order, timed.
3. Provide a countdown timer for any timed events in the recipe.
4. Display all instructions on the LCD screen
5. Display countdown time on one or more 7-segment displays
6. Test your product on someone outside the class, record their use of your product.

For objective 1, 2, and 4. the recipe is displayed on the LCD one item at a time. The next recipe item is displayed with the push button. For objective 3 and 5 the timed portion the timer will start and stop with a push button. The 4 digit seven segment display is used for the timer. The first two digits are the minutes, and the second two digits are the seconds.

An I2C module was used for the LCD to cut down on wires and ports needed from the Arduino Mega. The library LiquidCrystal\_I2C.h was utilized. Communication happens over 4 wires. Power, ground, and 2 communication lines.

On wiring the seven segment that has 4 displays. 12 GPIO pins are required from the Arduino. 8 are required for the segments (7 segments + 1 for the decimal point). The other 4 GPIO pins select which display to turn on. Only one display should be on at a time (unless you want the same digit to be written to other displays). This means that we have to continuously write to each display faster than we can visually see. This gives the appearance that each display is being on all at ounce. To use the GPIO pins, the registers were written to directly. PORTB GPIO pins were used. This happen is lines 92 to 94 in the C code.

```
0  *      PORT PIN: 2
   *
2  * PIN 10:
```

Lab02.ino

AutoCad was used to draw up the circuit diagram of all the connections, components, and devices used.

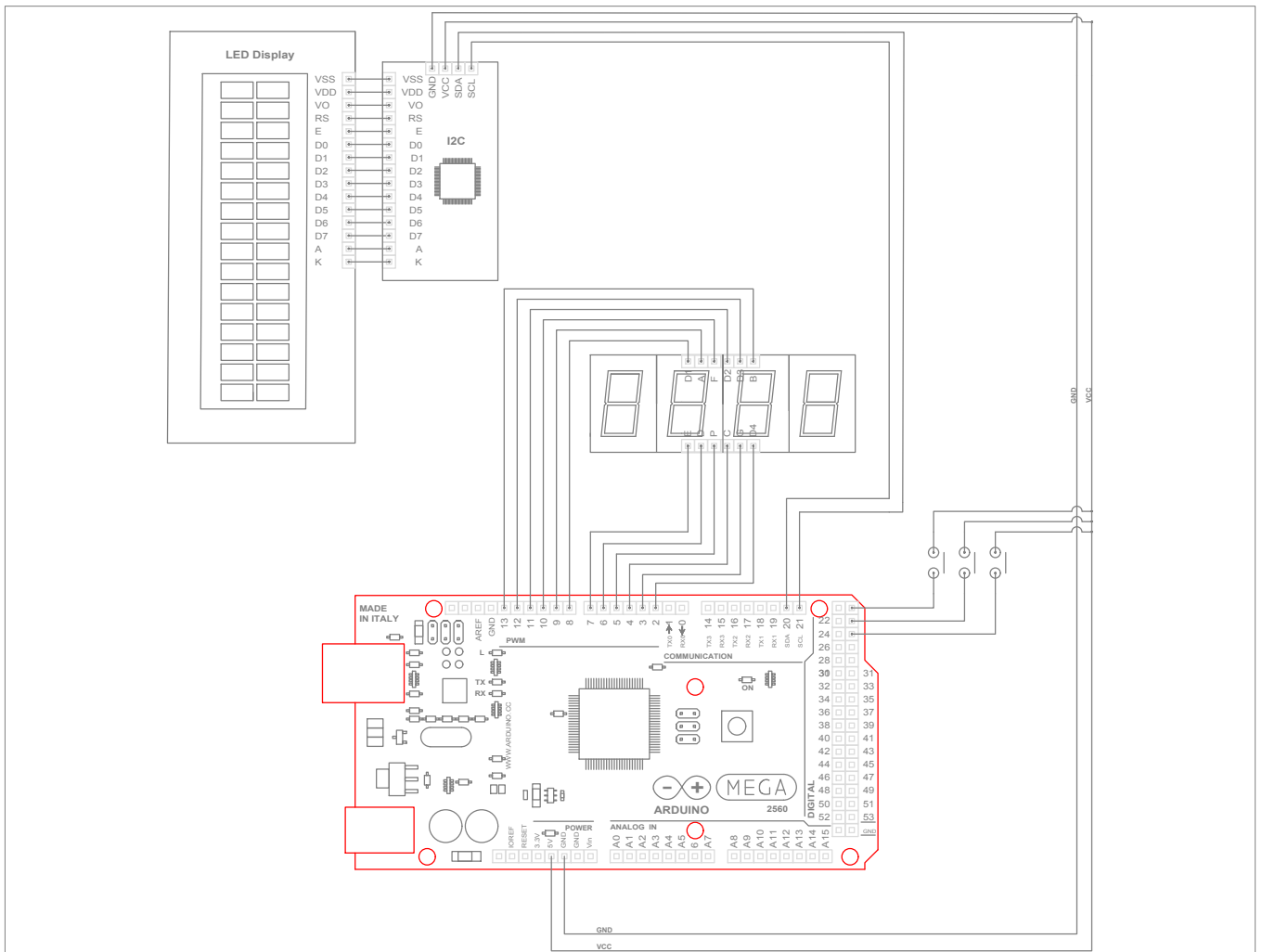


Figure 1: Circuit Diagram

## 2 Source Code

```
0 // AUTHORS: A. LOGAN BARBER; IAN NAIL
1 <<<<<<<< HEAD
2 // FILE NAME: LCD_Dpad.ino
3 // LAST UPDATED: 19 JANUARY 2022
4 //
5 // PURPOSE: THIS FILE IS THE MAIN FILE FOR DISPLAYING A RECIPE ON AN LCD AND
6 // BEING ABLE TO SCROLL USING TWO BUTTONS.
7 //
8 //=====
9 // FILE NAME: Lab02.ino
10 // LAST UPDATED: 28 JANUARY 2022
11 /*
12  * PURPOSE: THIS FILE IS THE MAIN FILE FOR DISPLAYING A RECIPE ON AN LCD AND
13  * BEING ABLE TO SCROLL USING TWO BUTTONS.
14  * THIS FILE ALSO UTILIZES AND 4-7 SEGMENT DISPLAY AS A 10 MINUTE TIMER.
15  */
16
17 /*
18  * BUTTON0:
19  *
20  * DIGITAL PIN: 27
21  *
22  * PORT: A
23  *
24  * PORT PIN: 5
25  */
26
27 /*
28  * BUTTON1:
29  *
30  * DIGITAL PIN: 28
31  *
32  * PORT: A
33  *
34  * PORT PIN: 6
35  */
36
37 /*
38  * BUTTON2:
39  *
40  * DIGITAL PIN: 29
41  *
42  * PORT: A
43  *
44  * PORT PIN: 7
45  */
46
47 /*
48  * LCD 4-7 SEGMENT DISPLAY
49  *
50  * PIN 1:
51  *     DIGITAL PIN: 10
52  *     PORT: B
53  *     PORT PIN: 4
54  *
55  * PIN 2:
56  *     DIGITAL PIN: 50
```

```

56  *      PORT: B
    *      PORT PIN: 3
    *
58  * PIN 3:
    *      DIGITAL PIN: 13
60  *      PORT: B
    *      PORT PIN: 7
62  *
    * PIN 4:
64  *      DIGITAL PIN: 51
    *      PORT: B
66  *      PORT PIN: 2
    *
68  * PIN 5:
    *      DIGITAL PIN: 12
70  *      PORT: B
    *      PORT PIN: 6
72  *
    * PIN 6:
74  *      DIGITAL PIN: 22
    *      PORT: A
76  *      PORT PIN: 0
    *
78  * PIN 7:
    *      DIGITAL PIN: 52
80  *      PORT: B
    *      PORT PIN: 1
82  *
    * PIN 8:
84  *      DIGITAL PIN: 23
    *      PORT: A
86  *      PORT PIN: 1
    *
88  * PIN 9:
    *      DIGITAL PIN: 24
90  *      PORT: A
    *      PORT PIN: 2
92  *
    * PIN 10:
94  *      DIGITAL PIN: 11
    *      PORT: B
96  *      PORT PIN: 5
    *
98  * PIN 11:
    *      DIGITAL PIN: 53
100 *      PORT: B
    *      PORT PIN: 0
102 *
    * PIN 12:
104 *      DIGITAL PIN: 25
    *      PORT: A
106 *      PORT PIN: 3
    */
108 >>>>>> refs/remotes/origin/master

110 // INCLUDE LIBRARIES
#include <LiquidCrystal_I2C.h>

112
// DEFINE PIN NUMBERS
114 #define ADDRESS 0x27

```

```

116 #define COLS 16
117 #define ROWS 2
118 // TIMER PARAMTERS
uint8_t buttonState = 0;
120 uint8_t minutes = 10; //start time -> CAN CHANGE TO WHATEVER TIME YOU WANT
uint8_t seconds = 0; //start time -> CAN CHANGE TO WHATEVER TIME YOU WANT
122 uint8_t totalMinutes = 0;
uint8_t minutesTens = 0;
124 uint8_t minutesOnes = 0;
uint8_t secondsTens = 0;
126 uint8_t secondsOnes = 0;
uint8_t secondsTemp = 0;
128 float totalSeconds = minutes*60 + seconds;
float totalMilliseconds = totalSeconds*1000;
130 float totalMicroseconds = totalMilliseconds*1000;

132 // DEFINE LETTERS FOR 7 SEGMENT DISPLAY
const uint8_t ZERO = 0x3F;
134 const uint8_t ZERO_DEC = 0xBF;
const uint8_t ONE = 0x06;
136 const uint8_t ONE_DEC = 0x86;
const uint8_t TWO = 0x5B;
138 const uint8_t TWO_DEC = 0xDB;
const uint8_t THREE = 0x4F;
140 const uint8_t THREE_DEC = 0xCF;
const uint8_t FOUR = 0x66;
142 const uint8_t FOUR_DEC = 0xE6;
const uint8_t FIVE = 0x6D;
144 const uint8_t FIVE_DEC = 0xED;
const uint8_t SIX = 0x7D;
146 const uint8_t SIX_DEC = 0xFD;
const uint8_t SEVEN = 0x07;
148 const uint8_t SEVEN_DEC = 0x87;
const uint8_t EIGHT = 0x7F;
150 const uint8_t EIGHT_DEC = 0xFF;
const uint8_t NINE = 0x67;
152 const uint8_t NINE_DEC = 0xE7;

154 // DEFINE THE DISPLAY SELECTION NUMBERS
const uint8_t D1 = 0xE7; // 0b11100111
156 const uint8_t D2 = 0xEB; // 0b11101011
const uint8_t D3 = 0xED; // 0b11101101
158 const uint8_t D4 = 0xEE; // 0b11101110
const uint8_t arrD[4] = {D4, D3, D2, D1};
160

// MESSAGE TO PRINT
162 char message0[] = "Hello World!";
char message1[] = "Hallo!";
164 char message2[] = "Wie gehts";
const uint8_t msgArrSize = 3;
166 char* msgArr[msgArrSize] = {message0, message1, message2};

168 // INDEX VARIABLES
uint8_t index = 0; // HOLDS INDEX FOR MESSAGE
170 uint8_t i = 0; // HOLDS INDEX IN for LOOPS FOR SCROLLING
uint8_t t = 0; // HOLDS INDEX IN for LOOP FOR THE TIMER
172

// CREATE LiquidCrystal OBJECT
174 LiquidCrystal_I2C lcd(ADDRESS, COLS, ROWS);

```

```

176 // RUN THIS PROGRAM
void setup()
178 {
    // INITIALIZE THE LCD SCREEN
180    lcd.begin();

    // PRINT MESSAGE
182    lcd.print(msgArr[0]);
184    lcd.setCursor(0, 1);
    lcd.print(msgArr[1]);

186    // SETUP BUTTON PINS AS INPUTS
188    // SETUP 7-SEGMENT SELECTOR PINS AS OUTPUT
    DDRA = 0x0F; // 0b00001111

190    // ENABLE INTERNAL PULL-UP RESISTOR FOR BUTTONS
192    PORTA = 0xE0; // 0b11100000

194    // SETUP PORT B AS OUTPUT FOR THE LCD
    DDRB = 0xFF;
196    PORTB = 0x00;

198    // CALCULATE INDIVIDUAL DIGITS
    totalMinutes = totalSeconds/60;
200    minutesTens = totalMinutes/10;
    minutesOnes = totalMinutes%10;
202    secondsTemp = int(totalSeconds)%60;
    secondsTens = secondsTemp/10;
204    secondsOnes = secondsTemp%10;
}

206 // LOOP FOREVER
void loop()
{
210    // ELSE IF BUTTON0 IS LOW SCROLL DOWN
    if((PINA & 0xE0) == 0xC0)
212    {
        // DEBOUNCE BUTTON2
214        delay(100);
        if((PINA & 0xE0) == 0xC0)
216        {
            scroll_down();
218        }
    }

220    // IF BUTTON1 IS LOW THEN SCROLL UP
    else if((PINA & 0xE0) == 0xA0)
222    {
        // DEBOUNCE THE BUTTON1
224        delay(100);
        if((PINA & 0xE0) == 0xA0)
226        {
            scroll_up();
228        }
    }

230    // IF BUTTON 2 IS LOW CHANGE THE BUTTON STATE
    else if((PINA & 0xE0) == 0x60)
232    {
234

```

```

236     delay(100);
237     if((PINA & 0xE0) == 0x60)
238     {
239         switch(buttonState)
240         {
241             case 0:
242                 buttonState = 1;
243                 break;
244
245             case 1:
246                 buttonState = 2;
247                 break;
248
249             case 2:
250                 buttonState = 0;
251
252                 // RESET TIME
253                 totalSeconds = minutes*60 + seconds;
254                 totalMilliseconds = totalSeconds*1000;
255                 totalMicroseconds = totalMilliseconds*1000;
256
257                 // CALCULATE INDIVIDUAL DIGITS
258                 totalMinutes = totalSeconds/60;
259                 minutesTens = totalMinutes/10;
260                 minutesOnes = totalMinutes%10;
261                 secondsTemp = int(totalSeconds)%60;
262                 secondsTens = secondsTemp/10;
263                 secondsOnes = secondsTemp%10;
264                 break;
265             }
266         }
267     }
268
269     // RUN TIMER IF BUTTON STATE IS IN STATE 1
270     if(buttonState == 1)
271     {
272         // TIME CALCULATIONS
273         totalMicroseconds = totalMicroseconds - 2000; //totalMilliseconds++
274         for stopwatch
275             totalMilliseconds = totalMicroseconds/1000;
276             totalSeconds = (totalMilliseconds/1000+1);
277
278         // CALCULATE INDIVIDUAL DIGITS
279         totalMinutes = totalSeconds/60;
280         minutesTens = totalMinutes/10;
281         minutesOnes = totalMinutes%10;
282         secondsTemp = int(totalSeconds)%60;
283         secondsTens = secondsTemp/10;
284         secondsOnes = secondsTemp%10;
285     }
286
287     // TIMER
288     for(t = 0; t < 4; ++t)
289     {
290         switch(t)
291         {
292             case 0:
293                 PORTA = arrD[t];
294                 pickNumber(minutesTens);
295                 delayMicroseconds(500);

```



```

294         break;

296     case 1:
297         PORTA = arrD[t];
298         pickNumber(minutesOnes);
299         PORTB |= 0x80;
300         delayMicroseconds(500);
301         break;
302
303     case 2:
304         PORTA = arrD[t];
305         pickNumber(secondsTens);
306         delayMicroseconds(500);
307         break;
308
309     case 3:
310         PORTA = arrD[t];
311         pickNumber(secondsOnes);
312         delayMicroseconds(500);
313         break;
314     }
315 }
316
317 /*
318  * TYPE: FUNCTION
319  * NAME: scroll_up
320  * RETURN: void
321  * NUMBER OF PARAMETERS: 2
322  * PARAMETER NAMES: char* messagePtr, uint8_t sizeOfArray
323  * PURPOSE: THIS FUNCTION SCROLLS THROUGH THE RECEIPE DISPLAYED ON THE LCD
324  */
325 void scroll_up()
326 {
327     // DECREMENT INDEX BY ONE
328     —index;
329
330     // CHECK THE BOUNDS OF INDEX (REMEMBER index IS UNSIGNED)
331     if(index > (msgArrSize - 2))
332         index = 0;
333
334     // CLEAR THE LCD SCREEN AND PRINT MESSAGES TO THE LCD
335     lcd.clear();
336     for(i = 0; i < 2; ++i)
337     {
338         lcd.setCursor(0, i);
339         delayMicroseconds(1000);
340         lcd.print(msgArr[index + i]);
341         delayMicroseconds(1000);
342     }
343 }
344
345 /*
346  * TYPE: FUNCTION
347  * NAME: scroll_down
348  * RETURN: void
349  * NUMBER OF PARAMETERS: 2
350  * PARAMETER NAMES: char* messagePtr, uint8_t sizeOfArray
351  * PURPOSE: THIS FUNCTION SCROLLS THROUGH THE RECEIPE DISPLAYED ON THE LCD
352  */

```

```

354 void scroll_down()
355 {
356     // INCREMENT INDEX
357     ++index;
358
359     // CHECK THE BOUNDS OF INDEX
360     if(index > (msgArrSize - 2))
361         index = msgArrSize - 2;
362
363     // CLEAR THE LCD SCREEN AND PRINT MESSAGES TO THE LCD
364     lcd.clear();
365     for(i = 0; i < 2; ++i)
366     {
367         lcd.setCursor(0, i);
368         delayMicroseconds(1000);
369         lcd.print(msgArr[index + i]);
370         delayMicroseconds(1000);
371     }
372 }
373
374 /*
375  * TYPE: FUNCTION
376  * NAME: pickNumber
377  * RETURN: void
378  * NUMBER OF PARAMETERS: 1
379  * PARAMETER NAMES: int x
380  * PURPOSE: THIS FUNCTION PICK THE NUMBER FOR THE LCD
381  */
382 void pickNumber(int x) //changes value of number
383 {
384     switch(x)
385     {
386         default:
387             PORTB = ZERO;
388             break;
389         case 1:
390             PORTB = ONE;
391             break;
392         case 2:
393             PORTB = TWO;
394             break;
395         case 3:
396             PORTB = THREE;
397             break;
398         case 4:
399             PORTB = FOUR;
400             break;
401         case 5:
402             PORTB = FIVE;
403             break;
404         case 6:
405             PORTB = SIX;
406             break;
407         case 7:
408             PORTB = SEVEN;
409             break;
410         case 8:
411             PORTB = EIGHT;
412             break;
413         case 9:

```

```
414         PORTB = NINE;  
415         break;  
416     }  
}
```

Lab02.ino