```
/*****************************************************************************
***
 * main.c
 *
 * Example to drive the Seiko L1682 LCD module
 *
 * Connections
 *  Gnd pin 1
 *  VCC pin 2
 *  VLC pin 3 on the L1682 is connected to ground.
 *  Connections between the eZ80 and L1682
 *     Port A[2] -> E pin 6
 *     Port A[1] -> RS pin 4
 *     Port A[0] -> R/W pin 5
 *     Port B[0-7] -> DB0 - DB7 (pins 7 to 14) for 8 bit data length
 *     Port B[4-7] -> DB4 - DB7 (pins 11 to 14) for 4 bit data length
 *
 * Enoch Hwang May 2002

*****************************************************************************
**/

#include <simple_ez80.h>                //Must include the register definitions which
                                        //will be used for PB configuration and control

/* Function to toggle the Enable line
   RS, R/W and Data are latched at the falling edge of the Enable signal, so
   after each data change, the Enable line must be pulled low.
*/
ToggleE(int RS){
   int i;
   if(RS == 0){  // for RS=0
      PA_DR = 0x04; // set RS=R/W=0; E=1
      // This delay works for the eZ80 development board.
      // If the 500 is changed to 100, then the character address is not set properly.
      for(i=0;i<500;i++);     // delay
      PA_DR = 0x00; // set E low to latch data
   }else{           // for RS=1
      PA_DR = 0x06; // set RS=1; R/W=0; E=1
      for(i=0;i<500;i++);     // delay
      PA_DR = 0x02; // set E low to latch data
   }
   for(i=0;i<15000;i++);     // delay to cause the characters to appear slowly
}

main(){
// VLC pin 3 on the L1682 is connected to ground
// Connections between the eZ80 and L1682
   // Port A[2] -> E
   // Port A[1] -> RS
   // Port A[0] -> R/W
   PA_DR = 0xFF; // set PA as output
   PA_DDR = 0x00;
   PA_ALT1 = 0x00;
   PA_ALT2 = 0x00;

   // Port B[0-7] -> DB0 - DB7 for 8 bit data length
   // Port B[4-7] -> DB4 - DB7 for 4 bit data length
   PB_DR = 0xFF; // set PB as output
   PB_DDR = 0x00;
   PB_ALT1 = 0x00;
   PB_ALT2 = 0x00;

/* The following is for 4 bit data length. Tested to work EH
   Data is sent through DB7 to DB4. The upper 4 bits are transferred first,
   then the lower 4 bits. */

//  Initialization sequence
//  ToggleE(0);       // this line causes an error
   PB_DR = 0x30; // Function Set: 4 bit data length; 2 line; 5x7 dot matrix
   ToggleE(0);
   PB_DR = 0x30;
   ToggleE(0);
   PB_DR = 0x30;
   ToggleE(0);
   PB_DR = 0x20;
   ToggleE(0);
   PB_DR = 0x20;
   ToggleE(0);
   PB_DR = 0x80;
   ToggleE(0);
   PB_DR = 0x00; // Entry mode set: Increment One; No Shift
   ToggleE(0);
   PB_DR = 0x60; // Entry mode set: Increment One; No Shift
   PB_DR = 0x70; // Entry mode set: Increment One; Shift
   ToggleE(0);
   PB_DR = 0x00; // Display control: Display ON; Cursor ON; Blink ON
   ToggleE(0);
   PB_DR = 0xE0; // Display control: Display ON; Cursor ON; Blink ON
   ToggleE(0);
// don't need for either shift or no shift
//  PB_DR = 0x10; // Cursor or display shift:
//  ToggleE(0);
//  PB_DR = 0x40; // Cursor or display shift:
//  ToggleE(0);
   PB_DR = 0x00; // Display clear
   ToggleE(0);
   PB_DR = 0x10; // Display clear
   ToggleE(0);

// Setup address for 1st line
//  ToggleE(0);       // set RS=R/W=0 this line causes an error
   // set address to rightmost position for shifting from right to left
   PB_DR = 0x80; // 81 is address of second position on first line
   ToggleE(0);
   PB_DR = 0xF0;
   ToggleE(0);

for(;;){
// Write data to first line of display
//  ToggleE(1);   // this line causes an error
   PB_DR = 0x50; // T
```

```
110     ToggleE(1);
111     PB_DR = 0x40; // T
112     ToggleE(1);
113     PB_DR = 0x60; // h
114     ToggleE(1);
115     PB_DR = 0x80; // h
116     ToggleE(1);
117     PB_DR = 0x60; // i
118     ToggleE(1);
119     PB_DR = 0x90; // i
120     ToggleE(1);
121     PB_DR = 0x70; // s
122     ToggleE(1);
123     PB_DR = 0x30; // s
124     ToggleE(1);
125     PB_DR = 0x20; // space
126     ToggleE(1);
127     PB_DR = 0x00; // space
128     ToggleE(1);
129     PB_DR = 0x60; // i
130     ToggleE(1);
131     PB_DR = 0x90; // i
132     ToggleE(1);
133     PB_DR = 0x70; // s
134     ToggleE(1);
135     PB_DR = 0x30; // s
136     ToggleE(1);
137     PB_DR = 0x20; // space
138     ToggleE(1);
139     PB_DR = 0x00; // space
140     ToggleE(1);
141     PB_DR = 0x50; // S
142     ToggleE(1);
143     PB_DR = 0x30; // S
144     ToggleE(1);
145     PB_DR = 0x60; // e
146     ToggleE(1);
147     PB_DR = 0x50; // e
148     ToggleE(1);
149     PB_DR = 0x60; // i
150     ToggleE(1);
151     PB_DR = 0x90; // i
152     ToggleE(1);
153     PB_DR = 0x60; // k
154     ToggleE(1);
155     PB_DR = 0xB0;// k
156     ToggleE(1);
157     PB_DR = 0x60; // o
158     ToggleE(1);
159     PB_DR = 0xF0; // o
160     ToggleE(1);
161     PB_DR = 0x20; // space
162     ToggleE(1);
163     PB_DR = 0x00; // space
164     ToggleE(1);
165
166   // Setup address for second line
167   //  ToggleE(0);       // set RS=R/W=0 this line causes an error
168   /*
169       PB_DR = 0xC0;// C0 is address of first position on second line
170       ToggleE(0);
171       PB_DR = 0x00;
172       ToggleE(0);
173   */
174
175   // Write data to second line of display
176   //  ToggleE(1);       // set RS=1; R/W=0 this line causes an error
177       PB_DR = 0x40; // L
178       ToggleE(1);
179       PB_DR = 0xC0;// L
180       ToggleE(1);
181       PB_DR = 0x40; // C
182       ToggleE(1);
183       PB_DR = 0x30; // C
184       ToggleE(1);
185       PB_DR = 0x40; // D
186       ToggleE(1);
187       PB_DR = 0x40; // D
188       ToggleE(1);
189       PB_DR = 0x20; // space
190       ToggleE(1);
191       PB_DR = 0x00; // space
192       ToggleE(1);
193       PB_DR = 0x60; // m
194       ToggleE(1);
195       PB_DR = 0xD0;// m
196       ToggleE(1);
197       PB_DR = 0x60; // o
198       ToggleE(1);
199       PB_DR = 0xF0; // o
200       ToggleE(1);
201       PB_DR = 0x60; // d
202       ToggleE(1);
203       PB_DR = 0x40; // d
204       ToggleE(1);
205       PB_DR = 0x70; // u
206       ToggleE(1);
207       PB_DR = 0x50; // u
208       ToggleE(1);
209       PB_DR = 0x60; // l
210       ToggleE(1);
211       PB_DR = 0xC0;// l
212       ToggleE(1);
213       PB_DR = 0x60; // e
214       ToggleE(1);
215       PB_DR = 0x50; // e
216       ToggleE(1);
217       PB_DR = 0x20; // space
218       ToggleE(1);
219       PB_DR = 0x00; // space
220       ToggleE(1);
221       PB_DR = 0x40; // L
```

```
222     ToggleE(1);
223     PB_DR = 0xC0;// L
224     ToggleE(1);
225     PB_DR = 0x30; // 1
226     ToggleE(1);
227     PB_DR = 0x10; // 1
228     ToggleE(1);
229     PB_DR = 0x30; // 6
230     ToggleE(1);
231     PB_DR = 0x60; // 6
232     ToggleE(1);
233     PB_DR = 0x30; // 8
234     ToggleE(1);
235     PB_DR = 0x80; // 8
236     ToggleE(1);
237     PB_DR = 0x30; // 2
238     ToggleE(1);
239     PB_DR = 0x20; // 2
240     ToggleE(1);
241
242  } // end for
243
244  /* The following is for 8 bit data length. Tested to work EH
245  //  Initialization sequence
246  //  ToggleE(0);       //  this line causes an error
247     PB_DR = 0x38; // Function Set: 8 bit data length; 2 line; 5x7 dot matrix
248     ToggleE(0);
249     PB_DR = 0x38;
250     ToggleE(0);
251     PB_DR = 0x38;
252     ToggleE(0);
253     PB_DR = 0x06; // Entry mode set: Increment One; No Shift
254     ToggleE(0);
255     PB_DR = 0x0E; // Display control: Display ON; Cursor ON; Blink ON
256     ToggleE(0);
257     PB_DR = 0x01; // Display clear
258     ToggleE(0);
259
260  // Setup address for 1st line
261  //  ToggleE(0);       // set RS=R/W=0 this line causes an error
262     PB_DR = 0x81; // 81 is address of second position on first line
263     ToggleE(0);
264
265  // Write data to first line of display
266  //  ToggleE(1);       // this line causes an error
267     PB_DR = 0x54; // T
268     ToggleE(1);
269     PB_DR = 0x68; // h
270     ToggleE(1);
271     PB_DR = 0x69; // i
272     ToggleE(1);
273     PB_DR = 0x73; // s
274     ToggleE(1);
275     PB_DR = 0x20; // space
276     ToggleE(1);
277     PB_DR = 0x69; // i
278     ToggleE(1);
279     PB_DR = 0x73; // s
280     ToggleE(1);
281     PB_DR = 0x20; // space
282     ToggleE(1);
283     PB_DR = 0x53; // S
284     ToggleE(1);
285     PB_DR = 0x65; // e
286     ToggleE(1);
287     PB_DR = 0x69; // i
288     ToggleE(1);
289     PB_DR = 0x6B;// k
290     ToggleE(1);
291     PB_DR = 0x6F; // o
292     ToggleE(1);
293
294  // Setup address for second line
295  //  ToggleE(0);       // set RS=R/W=0 this line causes an error
296     PB_DR = 0xC0;// C0 is address of first position on second line
297     ToggleE(0);
298
299  // Write data to second line of display
300  //  ToggleE(1);       // set RS=1; R/W=0 this line causes an error
301     PB_DR = 0x4C;// L
302     ToggleE(1);
303     PB_DR = 0x43; // C
304     ToggleE(1);
305     PB_DR = 0x44; // D
306     ToggleE(1);
307     PB_DR = 0x20; // space
308     ToggleE(1);
309     PB_DR = 0x6D;// m
310     ToggleE(1);
311     PB_DR = 0x6F; // o
312     ToggleE(1);
313     PB_DR = 0x64; // d
314     ToggleE(1);
315     PB_DR = 0x75; // u
316     ToggleE(1);
317     PB_DR = 0x6C;// l
318     ToggleE(1);
319     PB_DR = 0x65; // e
320     ToggleE(1);
321     PB_DR = 0x20; // space
322     ToggleE(1);
323     PB_DR = 0x4C;// L
324     ToggleE(1);
325     PB_DR = 0x31; // 1
326     ToggleE(1);
327     PB_DR = 0x36; // 6
328     ToggleE(1);
329     PB_DR = 0x38; // 8
330     ToggleE(1);
331     PB_DR = 0x32; // 2
332     ToggleE(1);
333  */
```

```
334 }
```