

```
1: sbit LCD_RS at RB0_bit;
2: sbit LCD_EN at RB1_bit;
3: sbit LCD_D4 at RB2_bit;
4: sbit LCD_D5 at RB3_bit;
5: sbit LCD_D6 at RB4_bit;
6: sbit LCD_D7 at RB5_bit;
7: //
8: sbit LCD_RS_Direction at TRISB0_bit;
9: sbit LCD_EN_Direction at TRISB1_bit;
10: sbit LCD_D4_Direction at TRISB2_bit;
11: sbit LCD_D5_Direction at TRISB3_bit;
12: sbit LCD_D6_Direction at TRISB4_bit;
13: sbit LCD_D7_Direction at TRISB5_bit;
14: //
15: // Define pins for stepper motor
16: #define STEP1 PORTC.F0
17: #define STEP2 PORTC.F1
18: #define STEP3 PORTC.F2
19: #define STEP4 PORTC.F3
20:
21: //
22:
23: // Function to initialize the microcontroller
24: void init() {
25:     // Set TRIS registers
26:     TRISA = 0xFF; // Set PORTA as input (for ADC)
27:     TRISC = 0x00; // Set PORTC as output (for stepper motor)
28:     TRISD = 0xC0; // Set RD6 and RD7 as input (for switches)
29:     TRISB = 0x00; // Set PORTB as output (for LCD)
30:     //TRISA = 0x08; // Set RA3 as input (AN3)
31:
32:     // Initialize ADC
33:     ADCON1 = 0x80; // Configure A/D converter
34:
35:     // Initialize LCD
36:     Lcd_Init();
37:     Lcd_Cmd(_LCD_CURSOR_OFF);
38:
39: }
40:
41: // Function to display temperature on LCD
42: void displayTemperature(unsigned int temperature) {
43:     char tempStr[7];
44:     IntToStr(temperature, tempStr);
45:     Ltrim(tempStr);
46:
47:     Lcd_Out(1, 1, "Temp: ");
48:     Lcd_Out(1, 7, tempStr);
49:     Lcd_Out(1, 10, "C");
50: }
51:
52:
53: // Function to read temperature from LM35 sensor
54: unsigned int readTemperature() {
55:     unsigned int adc_value = 0;
56:     float temperature = 0.0;
57: }
```

```
58:     adc_value = ADC_Read(0); // Read from channel 0 (AN0)
59:     temperature = adc_value * 0.488; // Convert ADC value to temperature (assuming V
Vref = 5V)
60:     return (unsigned int)temperature;
61: }
62:
63: // Function to rotate stepper motor clockwise
64: void Clockwise() {
65:     Lcd_Cmd(_LCD_CLEAR); // Clear LCD
66:     Lcd_Out(2, 1, "Motor: Forward    ");
67:
68:     STEP1 = 1;
69:     STEP2 = 0;
70:     STEP3 = 0;
71:     STEP4 = 0;
72:     Delay_ms(50);
73:
74:     STEP1 = 0;
75:     STEP2 = 1;
76:     STEP3 = 0;
77:     STEP4 = 0;
78:     Delay_ms(50);
79:
80:     STEP1 = 0;
81:     STEP2 = 0;
82:     STEP3 = 1;
83:     STEP4 = 0;
84:     Delay_ms(50);
85:
86:     STEP1 = 0;
87:     STEP2 = 0;
88:     STEP3 = 0;
89:     STEP4 = 1;
90:     Delay_ms(50);
91: }
92:
93: // Function to rotate stepper motor counterclockwise
94: void rotateStepperCounterclockwise() {
95:     Lcd_Cmd(_LCD_CLEAR); // Clear LCD
96:     Lcd_Out(2, 1, "Motor: Reverse    ");
97:
98:     STEP1 = 0;
99:     STEP2 = 0;
100:    STEP3 = 0;
101:    STEP4 = 1;
102:    Delay_ms(50);
103:
104:    STEP1 = 0;
105:    STEP2 = 0;
106:    STEP3 = 1;
107:    STEP4 = 0;
108:    Delay_ms(50);
109:
110:    STEP1 = 0;
111:    STEP2 = 1;
112:    STEP3 = 0;
113:    STEP4 = 0;
```

```
114:     Delay_ms(50);
115:
116:     STEP1 = 1;
117:     STEP2 = 0;
118:     STEP3 = 0;
119:     STEP4 = 0;
120:     Delay_ms(50);
121: }
122: //Function to stop motor in case error
123: void motorStop() {
124:     Lcd_Cmd(_LCD_CLEAR);           // Clear LCD
125:     Lcd_Out(2, 1, "Motor: Stopped Error ");
126:
127:     STEP1 = 0;
128:     STEP2 = 0;
129:     STEP3 = 0;
130:     STEP4 = 0;
131:     Delay_ms(50);
132: }
133: //Funtion to read ldr readings
134: unsigned int readLDR() {
135:     unsigned int adc_ldr = 0;
136:     float ldr = 0.0;
137:
138:     adc_ldr = ADC_Read(1);         // Read from channel 0 (AN0)
139:     ldr = adc_ldr ;               // Convert ADC value to ldr
140:
141:     return (unsigned int)ldr;
142: }
143: //Funtion to display readings of ldr (Only for testing)
144: void displayLDR(unsigned int ldr) {
145:     char ldrStr[7];
146:
147:     IntToStr(ldr, ldrStr);
148:     Ltrim(ldrStr);
149:     Lcd_Out(1, 7, ldrStr);
150:     Lcd_Out(1, 10, "LDR");
151: }
152: //Funtion to close window at night mode
153: void rotateStepperCounterclockwiseNightMode() {
154:     int i;
155:     Lcd_Cmd(_LCD_CLEAR);           // Clear LCD
156:     Lcd_Out(2, 1, "Motor: Reverse ");
157:     for (i = 0; i < 27; i++) {     // loop for 27 steps
158:         if (PORTD.F7 == 0 && PORTD.F6 == 1) { // Check if switch on RD7 is de-activ
vated and RD6 is activated
159:             Lcd_Cmd(_LCD_CLEAR);     // Clear LCD
160:             Lcd_Out(2, 1, "Window Closed ");
161:         }
162:         else { //close window by turning stepper motor clockwise
163:             Lcd_Cmd(_LCD_CLEAR);     // Clear LCD
164:             Lcd_Out(2, 1, "Window closing ");
165:             STEP1 = 0;
166:             STEP2 = 0;
167:             STEP3 = 0;
168:             STEP4 = 1;
169:             Delay_ms(50);
```

```
170:
171:         STEP1 = 0;
172:         STEP2 = 0;
173:         STEP3 = 1;
174:         STEP4 = 0;
175:         Delay_ms(50);
176:
177:         STEP1 = 0;
178:         STEP2 = 1;
179:         STEP3 = 0;
180:         STEP4 = 0;
181:         Delay_ms(50);
182:
183:         STEP1 = 1;
184:         STEP2 = 0;
185:         STEP3 = 0;
186:         STEP4 = 0;
187:         Delay_ms(50);
188:
189:         if(i==25 ) {
190:             while(1){ //stop motor in case of error of window not closing.
191:                 motorStop();
192:             }
193:         }
194:
195:     }
196: }
197: }
198: void SoilSensor () {
199:     unsigned int adcValueSoil;
200:     unsigned int threshold = 500; // Threshold value for soil moisture
201:
202:     // Read ADC value from AN3 (RA3)
203:     adcValueSoil = ADC_Read(3); // ADC_Read(channel)
204:
205:     // Check if soil is dry
206:     if (adcValueSoil > threshold) {
207:         // Soil is dry, turn on the relay (motor ON)
208:         PORTC.F7 = 1;
209:     } else {
210:         // Soil is wet, turn off the relay (motor OFF)
211:         PORTC.F7 = 0;
212:     }
213:
214:     Delay_ms(1000);
215: }
216: void main() {
217:     //list of variables
218:     unsigned int temperature = 0;
219:     unsigned int ldr = 0;
220:     int b;
221:     unsigned int ldrValue = ADC_Read(1); // Read ADC value of channel 1 (RA1)
222:     int oldData = 0;
223:     init(); // Initialize the microcontroller
224:     Lcd_Cmd(_LCD_CLEAR); // Clear LCD
225:
226:     while (1) {
```

```

227:
228:     delay_ms(1000);
229:     ldr = readLDR();
230:     SoilSensor();
231:     if(ldr < 100 ) {          //if night detected then activate night mode, close
window.
232:         Lcd_Cmd(_LCD_CLEAR);          // Clear LCD
233:         Lcd_Out(2, 1, "Night Mode");    // Display message
234:         Lcd_Out(3, 1, "Activated");
235:         Delay_ms(1000);                // Delay to avoid flickering
236:         rotateStepperCounterclockwiseNightMode();    //call for funtcion to clo
ose window using stepper motor.
237:     }
238:     else {
239:         temperature = readTemperature(); // Read temperature from LM35
240:         displayTemperature(temperature); // Display temperature on LCD
241:         if(temperature != oldData){
242:             for (b = 0; b < 27; b++) {
243:                 if (temperature > 22) {
244:                     // If temperature is above 22 degrees
245:                     if (PORTD.F6 == 0 && PORTD.F7 == 1) { // Check if switch on
n RD6 is de-activated and RD7 activated
246:                         Lcd_Cmd(_LCD_CLEAR); // Clear LCD
247:                         displayTemperature(temperature); // Display temperature on
LCD
248:                         Lcd_Out(2, 1, "Window Open ");
249:                     } else { //turn stepper motor Clockwise to open window
250:                         Clockwise();
251:                         if(b==25 ) {          //at step 25 error occurs
252:                             motorStop();
253:                             return;
254:                         }
255:                     }
256:                 } else {
257:                     // If temperature is less than or equal to 22 degrees
258:                     if (PORTD.F7 == 0 && PORTD.F6 == 1) { // Check if switch
h on RD6 is activated and RD7 de-activated
259:                         Lcd_Cmd(_LCD_CLEAR); // Clear LCD
260:                         displayTemperature(temperature); // Display temperature
on LCD
261:                         Lcd_Out(2, 1, "Window Closed ");
262:                     } else {
263:                         rotateStepperCounterclockwise();
264:                         if(b==25) {          //at step 25 error occurs
265:                             motorStop();
266:                             return;
267:                         }
268:                     }
269:                 }
270:             }
271:             oldData = temperature;
272:         }
273:         else { //do nothing
274:         }
275:         Delay_ms(1000); // Small delay to avoid bouncing effects
276:     }

```

```
277:     }  
278: }
```