```
/*
  A program for controlling a single stepper motor driving a rotary table.
  Uses a 4x4 matrix keypad for entry of degrees and direction or number of divisions to move the table.
  Serial I2C display, Pololu stepper driver.
*/


#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Keypad.h>

const byte COLS = 4;
const byte ROWS = 4;
char keys[COLS][ROWS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'.', '0', '#', 'D'}
};

byte rowPINS[ROWS] = {11, 10, 9, 8};
byte colPINS[COLS] = {7, 6, 5, 4};

Keypad kpd = Keypad(makeKeymap(keys), rowPINS, colPINS, ROWS, COLS);

LiquidCrystal_I2C lcd(0x27, 20, 4); // set the LCD address to 0x20 for a 16 chars and 2 line display

//setup vars
const int stp = 12;  //connect pin 12 to step
const int dir = 13;  // connect pin 13 to dir
const int StepsPerRotation = 800; //Set Steps per rotation of stepper

float TableRatio = 1; //ratio of rotary table
float Multiplier = (StepsPerRotation * TableRatio) / 360;

float stepdelay = 1;

float Degrees = 0;                //Degrees from Serial input
float ToMove = 0;                 //Steps to move
float Divisions;
float current = 0;

int Mode = 0;


void setup()
{
  lcd.init();                      // initialize the lcd
  pinMode(stp, OUTPUT);
  pinMode(dir, OUTPUT);

  // Print welcome message to the LCD.
  lcd.backlight();
```

```
  lcd.print("Rotary Table Control");
  lcd.setCursor(0, 2);
  lcd.print("  CrankyTechGuy CNC");
  lcd.setCursor(0, 3);
  lcd.print("   Copyright 2014");
  delay(2000);
  lcd.init();
  Mode = GetMode();
}

void software_Reset() // Restarts program from beginning but does not reset the peripherals and
registers
{
  asm volatile ("  jmp 0");
}

void rotationCW(float tm, int d)
{
  if (d == 0)
  {
    digitalWrite(dir, LOW);
  }
  else
  {
    digitalWrite(dir, HIGH);
  }

  for (int i = 0; i < tm; i++)
  {
    digitalWrite(stp, HIGH);
    delay(stepdelay);
    digitalWrite(stp, LOW);
    delay(stepdelay);
  }
}


void rotationCCW(float tm, int d)
{
  if (d == 0)
  {
    digitalWrite(stp, LOW);
  }
  else
  {
    digitalWrite(stp, HIGH);
  }

  for (int i = 0; i < tm; i++)
  {
    digitalWrite(dir, HIGH);
    delay(stepdelay);
    digitalWrite(dir, LOW);
```

```
107     delay(stepdelay);
108   }
109 }
110
111 float GetNumber()
112 {
113   float num = 0.00;
114   float decimal = 0.00;
115   float decnum = 0.00;
116   int counter = 0;
117   char key = kpd.getKey();
118   lcd.setCursor(0, 0); lcd.print("Enter degrees then"); lcd.setCursor(0, 3); lcd.print("    press
119 [#].");
120   lcd.setCursor(0, 30); lcd.print("Reset [D]");
121   lcd.setCursor(8, 2);
122   bool decOffset = false;
123
124   while (key != '#')
125   {
126     switch (key)
127     {
128       case NO_KEY:
129         break;
130
131       case '.':
132         if (!decOffset)
133         {
134           decOffset = true;
135         }
136         lcd.print(key);
137         break;
138
139       case '0': case '1': case '2': case '3': case '4':
140       case '5': case '6': case '7': case '8': case '9':
141         if (!decOffset)
142         {
143           num = num * 10 + (key - '0');
144           lcd.print(key);
145         }
146         else if ((decOffset) && (counter <= 1))
147         {
148           num = num * 10 + (key - '0');
149           lcd.print(key);
150           counter++;
151         }
152         break;
153
154       case 'D':
155         software_Reset();
156         break;
157     }
158
159     decnum = num / pow(10, counter);
```

```
160     key = kpd.getKey();
161   }
162   return decnum;
163 }
164
165 float GetDivisions()
166 {
167   float num = 0.00;
168   char key = kpd.getKey();
169   lcd.clear();
170   lcd.setCursor(0, 0); lcd.print("Enter Divisions then"); lcd.setCursor(0, 1); lcd.print("     press
171 [#].");
172   lcd.setCursor(0, 30); lcd.print("Reset [D]");
173   lcd.setCursor(8, 2);
174
175   while (key != '#')
176   {
177     switch (key)
178     {
179       case NO_KEY:
180         break;
181
182       case '0': case '1': case '2': case '3': case '4':
183       case '5': case '6': case '7': case '8': case '9':
184         num = num * 10 + (key - '0');
185         lcd.print(key);
186         break;
187
188       case 'D':
189         software_Reset();
190         break;
191     }
192     key = kpd.getKey();
193     // num = 360/num;
194   }
195   return num;
196 }
197
198 int GetMode()
199 {
200   int mode = 0;
201   lcd.setCursor(0, 1); lcd.print("Select Op Mode");
202
203   lcd.setCursor(0, 3);
204   lcd.print("    DIV[A] DEG[B]");
205   while (mode == 0)
206   {
207     char key = kpd.getKey();
208     if (key == 'A')
209     {
210       mode = 1;
211     }
212     else if (key == 'B')
```

```
213     {
214       mode = 2;
215     }
216   }
217
218   lcd.clear();
219   return mode;
220 }
221
222 void loop()
223 {
224   if (Mode == 1)
225   {
226     Divisions = GetDivisions();
227     Degrees = (360 / Divisions);
228   }
229   if (Mode == 2)
230   {
231     Degrees = GetNumber();
232   }
233   lcd.clear();
234   lcd.setCursor(0, 3);
235   lcd.print("FWD[A] REV[B] CAN[C]");
236   char key = kpd.getKey();
237   while (key != 'C')
238   {
239     lcd.setCursor(0, 0); lcd.print("POS:"); lcd.print(current); lcd.setCursor(0, 1);
240 lcd.print("DPM:"); lcd.print(Degrees);
241     key = kpd.getKey();
242     if (key == 'A')
243     {
244       if (current >= 360)
245       {
246         current = (current + Degrees) - 360;
247       } else {
248         current = current + Degrees;
249       }
250       ToMove = Degrees * Multiplier;
251       lcd.setCursor(0, 2);
252       lcd.print("       Moving       ");
253       rotationCW(ToMove, 2);
254       lcd.setCursor(0, 2); lcd.print("                    ");
255       lcd.setCursor(4, 0); lcd.print("      ");
256     }
257     if (key == 'B')
258     {
259       if (current <= 0)
260       {
261         current = 360 + (current - Degrees);
262       } else {
263         current = current - Degrees;
264       }
265       ToMove = Degrees * Multiplier;
```

```
266       lcd.setCursor(0, 2);
267       lcd.print("       Moving       ");
268       rotationCCW(ToMove, 2);
269       lcd.setCursor(0, 2); lcd.print("                    ");
270       lcd.setCursor(4, 0); lcd.print("      ");
271     }
272   }
273   lcd.clear();
274 }
275
```