```
// CONTROL OF STEPPER MOTOR - GROUP SPARTANS
// CHANATI CHAITANYA 21JE0259


#include <LiquidCrystal_I2C.h>
#include <AccelStepper.h>

//const int stepsperrevolution = 200;   // NO. OF STEPS PER REVOLUTION
int msf = 16;                           // MICROSTEPPING FACTOR


//  LCD SCREEN SPECS
#define LCD_ADDR 0x27
#define LCD_COLUMNS 20
#define LCD_ROWS 4

// LABELLING OF DIGITAL PINS

const int power = 2;                  // POWER SUPPLY TO MOTOR (A4998 DRIVER) - ON/OFF
const int type_rot = 3;               // TYPE OF ROTATION (INDIVIDUAL DEGREES / RPM MODE)
const int rot = 4;                    // TO OBTAIN DIRECTION OF ROTATION

const int buttoninc = 5;              // TO INCREASE DIGITAL INPUT
const int buttondec = 6;              // TO DECREASE DIGITAL INPUT

const int dirpin = 8;                 // DIRECTION PIN TO DRIVER (A4998)
const int steppin = 9;                // STEP PIN TO DRIVER (A4998)

const int enter = 10;                 // BUTTON FOR PROCEEDING STEPS FURTHER
const int res = 13;                   // TO SET RESOLUTION OF DIGITAL INPUT

// ACCELSTEPPER MOTOR LIBRARY STEPPER DECLERATION
AccelStepper stepper(AccelStepper::DRIVER,steppin,dirpin);

// LCD CRYSTAL DISPLAY LIBRARY
LiquidCrystal_I2C lcd (LCD_ADDR, LCD_COLUMNS, LCD_ROWS);
```

```
unsigned long previousMillis = 0;           // FOR DELAY IN UPDATING PARAMETERS ON LCD

const long interval = 100;                  // Update interval (lag of inc and dec buttons) in milliseconds

float required_value;                       // DIGITAL + ANALOG INPUT

void update_lcd_m(int value1,int value2){   // FUNCTION FOR UPDATING MODE AND DIRECTION ON LCD

  //lcd.setCursor(16,0);
  // lcd.print("MO: ");
  //lcd.print("   ");

  if(value1 == HIGH){
    lcd.setCursor(17,0);
    lcd.print("RPM");
  }
  if(value1 == LOW){
    lcd.setCursor(17,0);
    lcd.print("DEG");
  }

  //lcd.setCursor(13,1);
  // lcd.print("MO: ");
  //lcd.print("   ");

  if(value2 == HIGH){
    lcd.setCursor(13,0);
    lcd.print(" CW");
  }
  if(value2 == LOW){
    lcd.setCursor(13,0);
    lcd.print("ACW");
  }

}
```

```
void update_lcd_a(){                        // FUNCTION FOR UPDATING ANALOG INPUT
  int value;
  lcd.setCursor(0,0);
  lcd.print("A.I: ");
  lcd.setCursor(5,0);
  lcd.print("     ");
  lcd.setCursor(5,0);
  value=calibrate();
  lcd.print(value);
}

void update_lcd_d(float value){             // FUNCTION FOR UPDATING DIGITAL INPUT
  lcd.setCursor(0,1);
  lcd.print("D.I:");
  lcd.setCursor(5,1);
  lcd.print("       ");
  lcd.setCursor(5,1);
  lcd.print(value);
}

void update_lcd_r(float value){             // FUNCTION FOR UPDATING REQUIRED VALUE
  // lcd.setCursor(12,1);
  // lcd.print("N:");

  lcd.setCursor(13,1);
  //cd.print("     ");
  lcd.setCursor(13,1);
  lcd.print(value);
}

void update_lcd_deg(float required_value){  // FUNCTION FOR UPDATING DEG VALUE IN DEG MODE
  lcd.setCursor(0,2);
  lcd.print("DEG : ");
  lcd.setCursor(5,2);
  lcd.print("      ");
```

```
  lcd.setCursor(5,2);
  lcd.print(required_value);
}

void update_lcd_rpm(float required_value){  // FUNCTION FOR UPDATING RPM IN DEG AND RPM MODE
  lcd.setCursor(0,3);
  lcd.print("RPM : ");
  lcd.setCursor(5,3);
  lcd.print("      ");
  lcd.setCursor(5,3);
  lcd.print(required_value);
}

float update_res(float a){                  // FUNCTION FOR UPDATING RESOLUTION IN SYSTEM AND ON LCD

  if((digitalRead(res)==HIGH)&&(a>=100)){
    a=0.01;
    delay(200);
  }

  else if((digitalRead(res)==HIGH)&&(a<100 )){
    a=a*10;
    delay(200);
  }

  else if((digitalRead(res)==HIGH)){
    a=1;
    delay(200);
  }

  if(a>100){
    a=0.01;
    delay(200);
  }

  lcd.setCursor(13,2);
```

```
  lcd.print("       ");
  lcd.setCursor(13,2);
  lcd.print(a);
  return a;
}

int calibrate(){                          // FUNCTION FOR CALIBRATION OF ANALOG INPUT
    int input;
    int ana_in = analogRead(A0);
    input = map(ana_in, 0, 1023, 30, 3000);
    return input;
}


void setup() {

  stepper.setMaxSpeed(10000000);
  stepper.setAcceleration(500);

  // ASSIGNMENT OF INPUT AND OUTPUT TO DIGITAL PINS

  pinMode(power, INPUT);
  pinMode(type_rot, INPUT);
  pinMode(rot, INPUT);

  pinMode(buttoninc, INPUT);
  pinMode(buttondec, INPUT);

  pinMode(enter, INPUT);

  pinMode(dirpin, OUTPUT);
  pinMode(steppin, OUTPUT);

  pinMode(res, INPUT);

  // ASSIGNMENT OF ANALOG INPUT
```

```
  pinMode(A0, INPUT);

  // LCD CUSTOMIZATION
  lcd.begin(20,4);
  lcd.init();
  lcd.backlight();

  //Serial.begin(9600);
  //Serial.begin(115200);

  //LCD INITIAL OPUTPUT
  lcd.setCursor(4,0);
  lcd.print("STEPPER MOTOR");
  lcd.setCursor(6,1);
  lcd.print("CONTROL");
  lcd.setCursor(3,2);
  lcd.print("GROUP SPARTANS");
  delay(5000);
  lcd.clear();

  // for microstepping
  digitalWrite(7, HIGH);

}

int ana_value;                                          // ANALOG VALUE DECLARATION (INPUT : POTENTIOMETER)
float dig_value = 170;                                  // DECLERATION AND INITIAL ASSIGNMENT OF DIGITAL VALUE
int b=0;                                                // PARAMETER FOR SETTING THE PROCEDURAL STEPS
static float as=1;                                      // DECLERATION OF RESOLUTION AND INITIAL ASSIGNMENT


void loop() {

    // UPDATING TYPE OF ROTATION ON LCD and DIRECTION
    update_lcd_m(digitalRead(type_rot),digitalRead(rot));
```

```
 // UPDATING ANALOG VALUE AND DIGITAL VALUE TO SYSTEM
 ana_value = calibrate();
 as = update_res(as);

 // UPDATING ANALOG VALUE ON LCD
 if (millis() - previousMillis >= interval){
  update_lcd_a();
  update_lcd_d(dig_value);                          // FOR INITIALIZATION OF DIGITAL VALUE

  if(digitalRead(buttoninc)==HIGH){                 // UPDATING DIGITAL INCREASING VALUE ON SYSTEM AND LCD
     dig_value = dig_value + as;
     update_lcd_d(dig_value);
   }
  if(digitalRead(buttondec)==HIGH){                 // UPDATING DIGITAL DECREASING VALUE ON SYSTEM AND LCD
     dig_value = dig_value - as;
     update_lcd_d(dig_value);
   }
  required_value = ana_value+ dig_value ;
  update_lcd_r(required_value);                     // UPDATING NET REQUIRED VALUE ON SYSTEM AND LCD

  previousMillis = millis();
 }


// DEG MODE
float step;
float deg;
int rpm;
float sps;

if((digitalRead(type_rot) == LOW)){
   if(digitalRead(power) == HIGH){
     if(digitalRead(rot)==HIGH)
       digitalWrite(dirpin, HIGH);
     else
       digitalWrite(dirpin, LOW);
```

```
if(b==0){                                    // STEP 0 : START

    lcd.setCursor(0,2);
    lcd.print("            ");
    lcd.setCursor(0,3);
    lcd.print("                ");

    if (digitalRead(enter) == HIGH) {
      delay(100);
      b=1;
    }
}

if(b==1){                                   // STEP 1 : ENTRY OF DEG INPUT
update_lcd_deg(required_value);
if((digitalRead(enter)==HIGH)){
  delay(100);
  deg=required_value;
  update_lcd_deg(deg);
  b=2;
}
}

if(b==2){                                   // STEP 2 : ENTRY OF RPM INPUT
update_lcd_rpm(required_value);
if((digitalRead(enter)==HIGH)){
  delay(100);
  rpm=required_value;
  update_lcd_rpm(rpm);
  b=3;
}
}


if(b==3){                                  // STEP 3 : EXECUTION OF ROTATION
```

```
      step = deg * (200.0 / 360.0) * msf;           // convert degrees to steps
      sps = (rpm) * (200.0 / 60.0) * msf;           // convert RPM to steps per second
      //Serial.print(sps);
      //Serial.print(stepper.currentPosition());
      int x = stepper.currentPosition();
      for(;(stepper.currentPosition()-x)<=step;){
        stepper.run();
        stepper.setSpeed(sps);                      // speed in steps per second
       }
       //Serial.print(stepper.currentPosition());
        stepper.stop();
        b=4;
      }

      if(b==4){                                  // STEP 4 : COMPLETION MESSAGE
        lcd.setCursor(13,3);
        lcd.print("DONE");

        if((digitalRead(enter)==HIGH)){               // returning to zero position ??
        delay(100);
        b=5;
        }
      }

      if(b==5){                                 // RECURSION
        if((digitalRead(enter)==HIGH)){
        delay(100);
        b=0;
        }
      }

      //Serial.print(b);
    }

  }
```

```
 // RPM MODE
if (digitalRead(type_rot) == HIGH) {
 if (digitalRead(power) == HIGH) {
     if (digitalRead(rot) == HIGH)
         digitalWrite(dirpin, HIGH);
     else
         digitalWrite(dirpin, LOW);

     int rpm = 0;                            // Initialize rpm variable

     if (b == 0) {                           // STEP 0 : START
         lcd.setCursor(0,2);
         lcd.print("            ");
         lcd.setCursor(0,3);
         lcd.print("                ");

         if (digitalRead(enter) == HIGH) {
           delay(100);
           b=1;
         }
     }

     if (b == 1) {                           // STEP 1 : ENTRY OF RPM INPUT
         update_lcd_rpm(required_value);         // display of required RPM on LCD
         if (digitalRead(enter) == HIGH) {
             delay(100);
             rpm = required_value;               // setting RPM value from the user input
             update_lcd_rpm(rpm);                // display the chosen RPM on LCD
             b = 2;
         }
     }

     sps = (rpm) * (200.0 / 60.0) * msf;     // convert RPM to steps per second
     //Serial.print(sps);
     for(;b==2;){                            // STEP 2 : EXECUTION OF ROTATION
```

```
          stepper.setSpeed(sps);                        // Setting speed of the stepper motor
          stepper.runSpeed();                                // run the stepper motor
          if (digitalRead(enter) == HIGH){
            delay(100);
            b=3;
            }
        }

        for(;b==3;){                                  // STEP 3 : COMPLETION MESSAGE
          lcd.setCursor(13,3);
          lcd.print("DONE");
          if((digitalRead(enter)==HIGH)){
            delay(100);
            b=4;
          }
        }


        if(b==4){                                      // RECURSION
          if((digitalRead(enter)==HIGH)){
          delay(100);
          b=0;
          lcd.clear();
          }
        }
     }
    }

}
```