CODE

```
/*
BTP- Automation and Optimization of mold cart movement in foundry
Guided By: Prof. Jaina Mehta, Dr. Maryam Kaveshgar, Dr. Shashi Prakash

Group Members:
Abutibyan Hawawala: AU1743038
Adit Gada:          AU1743058
Dhruval Mehta:      AU1743040
Nitya Padia:        AU1743026
*/

//Motor encoder
#define ENC_COUNT_REV 9840 //Motor encoder output pulse per rotation
#define PI 3.14159265358979323846264338327 95

//Library for OLED
#include <Adafruit_SSD1306.h>
#include <Adafruit_GFX.h>
#include <Wire.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 32 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
Adafruit_SSD1306 disp(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);



//Limit switch variables
const int limit_s = 7;
const int limit_s2 = 6;

// variable for loading and unloading of carts
```

```
int run = 0, nvar=0, mvar=0;
int rerun = 0;

// Pins connected to motor
int Pin1 = A1;
int Pin2 = A2;
int act1= A3;
int act2= A4;
int act3= A5;
int act4= A6;

//Ultrasonic sensor1
const int trigPin = 45;
const int echoPin = 44;
long duration;
float distance;

//Ultrasonic sensor2
const int trigPin1 = 40;
const int echoPin1 = 41;
long duration1;
float distance1;

//led variables for indication
int ledPin2 = 8;
int ledPin1 = 12;

//Push button varibales for manual input
int buttonState = 0;
int buttonPin = 5;

//Ir sensors variables for obsatcle detection
int IRSensor1 = 47;
```

```
int IRSensor2 = 36;

// Ir proximity sensor to detect ahead mold carts
int IRSensor3 = 46;

// Encoder Calibration
const int encoderPinA = 4; //encoder pin
volatile long total=0; //Total measures total number of pulses (sum of encoder values)
int interval = 50; // 50 milli-seconds interval for measurements
float radius= 2.5; // Variable to measure distance (in cm)

//PID calibration
//Define Variables we'll be connecting to
  double Setpoint = 31.7, input, output;
//Specify the links and initial tuning parameters
  double Kp = 10,Ki = 1, Kd = 0;
  unsigned long currentTime,previousTime;
  double elapsedTime;
  double error;
  double lastError;
  double cumError, rateError;

void setup()
{
  pinMode(Pin1, OUTPUT);//sets the pin connected to motor as an output
  pinMode(Pin2, OUTPUT);//sets the pin connected to motor as an output

  pinMode(act1, OUTPUT); //sets the pin connected to motor as an output
  pinMode(act2, OUTPUT);  //sets the pin connected to motor as an output
  pinMode(act3, OUTPUT); //sets the pin connected to motor as an output
  pinMode(act4, OUTPUT); //sets the pin connected to motor as an output
  pinMode(ledPin1, OUTPUT); // Sets the LedPin1 as an Output
  pinMode(ledPin2, OUTPUT); // Sets the LedPin2 as an Output
```

```
 pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
 pinMode(trigPin1, OUTPUT); // Sets the trigPin as an Output
 pinMode(echoPin, INPUT); // Sets the echoPin as an Input
 pinMode(echoPin1, INPUT); // Sets the echoPin as an Input
 pinMode(limit_s, OUTPUT); // Sets the limit switch-1 as an Output
 pinMode(limit_s2, OUTPUT); // Sets the limit switch-2 as an Output
 pinMode(buttonPin, INPUT); // Sets the Pin connected to Pushbutton as an Input

 pinMode(IRSensor1, INPUT); // Sets the Pin connected to Ir-sensor as an Input
 pinMode(IRSensor2, INPUT); // Sets the Pin connected to Ir-sensor as an Input
 pinMode(IRSensor3, INPUT); // Sets the Pin connected to Ir-sensor as an Input

 Serial.begin(9600); // initialize serial communications at 9600 bps:

 if(!disp.begin(SSD1306_SWITCHCAPVCC, 0x3C))
 { // Address 0x3D for 128x64
  Serial.println(F("SSD1306 allocation failed"));
  for(;;);
  }

 disp.setTextSize(2);
 disp.setTextColor(WHITE);

}

void loop()
{
 disp.clearDisplay();
 run = 0;
 rerun = 0;

  // read the state of the pushbutton value // check if pushbutton is pressed
 buttonState = digitalRead(buttonPin);
```

```
 digitalRead(IRSensor1);
 digitalRead(IRSensor2);

while (run <15)
{
   UltraS();//function for ultrasonic sensors

  digitalWrite(ledPin2, HIGH); //Red led to indicate turned on
  digitalWrite (ledPin1, LOW);
  //main motor off
  analogWrite(Pin2, 0);
  analogWrite(Pin1, 0);

 while (distance1<= 9||nvar==1) //if mold cart is detected
  {
   nvar=1;  //variable
   UltraS(); //function for ultrasonic sensors

   Serial.println("Main Condition started");
    while ((distance1> 7 && distance<= 7)||nvar==2)
    {
      nvar=2; //variable
      UltraS();//function for ultrasonic sensors
      if (distance1<= 7 && distance> 7)
      {
      Serial.println("Cart detected");
      digitalWrite(ledPin2, LOW);
      digitalWrite(ledPin1, HIGH); //Green led to indicate turned on
      Cart(); //calling the function when cart is detected
      nvar=3; //variable
      }
      else
```

```
      {
      analogWrite(Pin2, 0);
      analogWrite(Pin1, 0);
      digitalWrite(ledPin2, HIGH); //Red led to indicate turned on
      digitalWrite(ledPin1, LOW);
      No_cart(); //calling the function when cart is not detected
      }

    }

    if (distance1 > 7 || distance > 7)  //if mold cart is not detected
    {
      Serial.println("Cart not detected");
     //motor is off
      analogWrite(Pin2, 0);
      analogWrite(Pin1, 0);
      digitalWrite(ledPin2, HIGH); //Red led to indicate turned on
      digitalWrite(ledPin1, LOW);
      No_cart(); //calling the function when cart is not detected
    }
  }
 }
float i = logic2(); // calling the function
}

//Function to run if the cart is detected at Sensors end
void Cart()
{
  disp.clearDisplay();
  disp.print("Cart detected");
  Serial.println("Cart detected");
  disp.display();
```

```
 //Lead screw moment
 delay(1000);
 analogWrite(act1, 150);
 analogWrite(act2, 0);
 analogWrite(Pin1, 0);
 delay(2000);
 analogWrite(act2, 150);
 analogWrite(Pin1, 0);
 delay(1000);

  digitalRead(IRSensor1);
  digitalRead(IRSensor2);

 //till the limit switch is pressed or IR proximity detects the cart
while(digitalRead(IRSensor3)>0 && digitalRead(limit_s2) == 0 )
{
  digitalRead(IRSensor3);
  Serial.println("main forward loop");

//if there is no obstacle in the way of the platform
 while(digitalRead(IRSensor1)==1 && digitalRead(IRSensor3)==1 &&
digitalRead(limit_s2)<1 )
 {
   digitalRead(IRSensor3);
   Serial.println("forward motion");
   digitalRead(IRSensor1);
   disp.clearDisplay();
   disp.print("Travelling");
   disp.display();

 //if there is an obstacle in the way of the platform
      while(digitalRead(IRSensor1)==0)
      {
```

```
      analogWrite(Pin1, 0);
      analogWrite(Pin2, 0);
      Serial.println("Obstacle detected");
     }

     float dist= (total/ENC_COUNT_REV)*2*PI*radius;
    Serial.print("Distance travelled by the wheel is: ");
    Serial.print(dist);
    Serial.println(" cm");

    while(dist < 31.7)
    {
    input = dist;           //read from rotary encoder connected to A0
    output = computePID(input);
    delay(10); //to be removed
    analogWrite(Pin1, output);        //control the motor based on PID value
    analogWrite(Pin2, 0);
    }
    analogWrite(Pin1, 255);
    analogWrite(Pin2, 0);
 }

}

 analogWrite(Pin1, 0);
 Serial.println("LS going in");
 disp.clearDisplay();
 disp.print("Cart delivered");
 disp.display();

//Lead screw moment return
 delay(1500);
 analogWrite(act1, 0);
```

```
 analogWrite(act2, 150);
 analogWrite(Pin1, 0);
 delay(2000);
 analogWrite(act2, 0);
 analogWrite(Pin1, 0);
 delay(1000);

 disp.clearDisplay();
 disp.print("Returning");
 Serial.println("returning");
 disp.display();

 //return cycle
//till the limit switch is pressed
 while(digitalRead(limit_s) == 0)
{
 digitalRead(IRSensor3);
 Serial.println("return loop starts ");

//if there is an obstacle in the way of the platform
    while(digitalRead(IRSensor2)==0)
     {
      analogWrite(Pin1, 0);
      analogWrite(Pin2, 0);
      Serial.println("Obstacle detected at ir2");
     }

//if there is no obstacle in the way of the platform
 while((digitalRead(IRSensor2)==1 && digitalRead(limit_s) < 1  ))
 {
  Serial.println("return movement ");
  digitalRead(IRSensor2);
  digitalRead(IRSensor3);
```

```
    analogWrite(Pin1, 0);
    analogWrite(Pin2, 255);
  }

}
  analogWrite(Pin2, 0);
// return cycle ends
  disp.clearDisplay();
  disp.print("Reached");
  Serial.println("returned to home");
  delay(1000);
  run++;
}

// Function No cart
void No_cart()
{
     Serial.println("Empty Cart");
     disp.clearDisplay();
     disp.println("Empty Cart");
     disp.display();
}

//function which has logic to unload the carts
float logic2()
{
  //while loop to exit mold carts
 while (rerun< 16)
 {
    buttonState = digitalRead(buttonPin);

     //Serial.println(buttonState);
```

```
  if (buttonState == HIGH) // buttonState is HIGH
    {
     digitalWrite(ledPin2, LOW);
     digitalWrite(ledPin1, HIGH); //Green led to indicate turned on
     exit_cart(); // calling the function to exit the carts onto Column rail
    }

  if (buttonState == LOW) // buttonState is LOW
    {
     //motor stopped
     analogWrite(Pin2, 0);
     analogWrite(Pin1, 0);
     digitalWrite(ledPin2, HIGH); //Red led to indicate turned on
     digitalWrite(ledPin1, LOW);
     dontexit_cart(); // calling the function when no command of exit is given
    }
 }

}

//Function to run if no command is given to exit at Pushbutton end
void dontexit_cart()
{
  Serial.println("Waiting for Response");
  disp.clearDisplay();
  disp.setCursor(0, 10);
  disp.print("Waiting for");
  disp.setCursor(1, 10);
  disp.print("Response");
  disp.display();
}

//Function to run if command is given to exit at Pushbutton end
```

```
void exit_cart()
{
  disp.clearDisplay(); // clears the display:
  if (rerun >0 )
  {
    UltraS(); // function for ultrasonic sensors

  digitalWrite(ledPin2, HIGH); //Red led to indicate turned on
  digitalWrite(ledPin1, LOW);
  analogWrite(Pin2, 0);
  analogWrite(Pin1, 0);
 while (distance1<= 9||mvar==1)  //if mold cart is detected
  {
   mvar=1; //variable
   UltraS();// function for ultrasonic sensors
   analogWrite(Pin2, 120);
   analogWrite(Pin1, 0);

   Serial.println("Unloading cycle starts");
    while ((distance1> 7 && distance<= 7)||mvar==2)
     {
      mvar=2; //variable
      UltraS(); // function for ultrasonic sensors
      analogWrite(Pin2, 120);
      analogWrite(Pin1, 0);

      if (distance1> 6 && distance<= 8)
      {
      Serial.println("Cart detected at one sensor");
      analogWrite(Pin2, 0);
      digitalWrite(ledPin2, LOW);
      digitalWrite(ledPin1, HIGH);
      mainexit_cart(); //calling the function when cart is detected
```

```
      mvar=3; //variable
      }
     }

    while (distance1 > 7 && distance > 7) //if mold cart is not detected
     {
      UltraS();// function for ultrasonic sensors
      Serial.println("no mold cart present");
     //motor continues to rotate
      analogWrite(Pin2, 120);
      analogWrite(Pin1, 0);
      digitalWrite (ledPin2, 255); //Red led to indicate turned on
      digitalWrite (ledPin1, 0);
      dontexit_cart(); //calling the function when cart is not detected
     }
    }
   }
   else if (rerun =0)
   {
//till the limit switch is pressed
    while (digitalRead(limit_s2) == 0)
    {
     rerun++;
    analogWrite(Pin1, 255);
    analogWrite(Pin2, 0);
    digitalWrite(ledPin2, HIGH); //Red led to indicate turned on
    digitalWrite(ledPin1, HIGH);
    Serial.println("dummy run");
    }
   }

}
```

```
void mainexit_cart()
{
   delay(1000);
  //lead screw out
  analogWrite(act3, 150);
  analogWrite(act4, 0);
  analogWrite(Pin2, 0);
  analogWrite(Pin1, 0);
  delay(2000);

  analogWrite(act4, 150);
  analogWrite(Pin2, 0);
  delay(1000);

  //exiting command
    while (digitalRead(limit_s2) == 0)
    {
      analogWrite(Pin1, 255);
      analogWrite(Pin2, 0);
    }
  disp.clearDisplay();
  disp.print("Cart Exited");
  disp.display();

  //motor stops
  //lead screw going in
  delay(1000);
  analogWrite(act3, 0);
  analogWrite(act4, 150);
  analogWrite(Pin1, 0);
  delay(2000);
  analogWrite(act4, 0);
  analogWrite(Pin1, 0);
```

```
  delay(1000);

  disp.clearDisplay();
  disp.print("Waiting");
  disp.display();
  Serial.println("Waiting");
  delay(1000);
  rerun++;
}

void UltraS()
{
  // Clears the trigPin for 10 microseconds
   digitalWrite(trigPin, LOW);
   delayMicroseconds(10);

   // Sets the trigPin on HIGH state for 10 micro seconds
   digitalWrite(trigPin, HIGH);
   delayMicroseconds(10);
   digitalWrite(trigPin, LOW);


  // Reads the echoPin, returns the sound wave travel time1 in microseconds
  duration = pulseIn(echoPin, HIGH);

  // Calculating the distance
    distance= duration*0.034/2;

    Serial.print("reading from US-1 =");
    Serial.println(distance);

// Clears the trigPin for 10 microseconds
    digitalWrite(trigPin1, LOW);
```

```
    delayMicroseconds(10);

    // Sets the trigPin on HIGH state for 10 micro seconds
    digitalWrite(trigPin1, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin1, LOW);
    // Reads the echoPin, returns the sound wave travel time1 in microseconds
     duration1 = pulseIn(echoPin1, HIGH);
    // Calculating the distance
    distance1= duration1*0.034/2;
    Serial.print("reading from US-2 =");
    Serial.println(distance1);
}

void updateEncoder()
{
  //Total of encoderValue
  total++;
}

double computePID(double inp)
{
        currentTime = millis();  //get current time
        elapsedTime = (double)(currentTime - previousTime); //compute time elapsed from
previous computation

        error = Setpoint - inp;  // determine error
        cumError += error * elapsedTime; // compute integral
        rateError = (error - lastError)/elapsedTime;   // compute derivative

        double out = Kp*error + Ki*cumError + Kd*rateError;    //PID output

        lastError = error;  //remember current error
```

```
    previousTime = currentTime;  //remember current time
    return out;         //have function return the PID output
}
```