KING MONGKUT'S INSTITUTE OF TECHNOLOGY LATKRABANG
SCHOOL OF ENGINEERING GROUP OF
ROBOTICS & AI

01416318 - MICROPROCESSOR AND INTERFACE

FINAL PROJECT

INSTRUCTED BY: DR. POOM KONGHUAYROB

NAME: NAPAT, THITIWUT, KUNLANITH
ID NUMBER: 65011376, 65011601, 65110141
DATE OF SUB.: 29/05/2023

Purpose

1. To understand how to design PCB, soldering.
2. To have a better understanding of how to use STM32.

Project scope

1. Design PCB on EasyEDA.
2. Soldering all components onto PCB.
3. Programming the line-following robot.

Theory

1. **STM32**
   STM32 is a family of 32-bit microcontrollers integrated based on the Arm Cortex-M processor designed to offer new degrees of freedom to MCU users. It offers products combining very high performance, real-time capabilities, digital signal processing, low-power / low-voltage operation, and connectivity, while maintaining full integration and ease of development.
2. **EasyEDA**
   EasyEDA is an easier and powerful online PCB design tool that allows electronics engineers, educators, students, makers, and enthusiasts to design and share their projects. This is a design tool integrated with the LCSC components catalogue and JLCPCB PCB service that helps users to save time to make their ideas into real products.
3. **PID controller**
   A PID controller is an instrument used in industrial control applications to regulate temperature, flow, pressure, speed and other process variables. PID (proportional integral derivative) controllers use a control loop feedback mechanism to control process variables and are the most accurate and stable controller.

Background research

1. **3D printing**
   3D printing is a common term for a manufacturing process where parts are made by adding layers of material one upon other to form the final product. The process was developed in the 1980s, and is commonly known in the commercial industry as *additive manufacturing* or *rapid prototyping*. Prototyping is the process of creating a product by improving on a series of designs. Each change in the design is called an iteration.
2. **PID control**
   PID control is the most common control algorithm used in industry and has been universally accepted in industrial control. The popularity of PID controllers can be attributed partly to their robust performance in a wide range of operating conditions and partly to their functional simplicity, which allows engineers to operate them in a simple, straightforward manner.

   P: instantaneous error = setpoint - input
   I: cumulative error = + = error * elapsed time

D: rate of error = (error - error last calculation)/elapsed time

Methodology

1. **List of components**
   - 1.1. **LED 0402 SMD**
   - 1.2. **DRV8833 Motor Driver**
   - 1.3. **STM32F411RE Microcontroller**
   - 1.4. **CH-N20-3 Geared DC Motors**
   - 1.5. **220Ω 0603 SMD Resistors**
   - 1.6. **1kΩ 0603 SMD Resistors**
   - 1.7. **TOIR-30A94CXAA Infrared LEDs**
   - 1.8. **TOPS-030TB2 Infrared Phototransistors**
   - 1.9. **1N4007 Rectifier Diode**
   - 1.10. **GNB5502S70A 2S 7.4V 70C/140C 550 mAh LiPo Battery**
2. **Design**
   - 2.1. **Schematic**

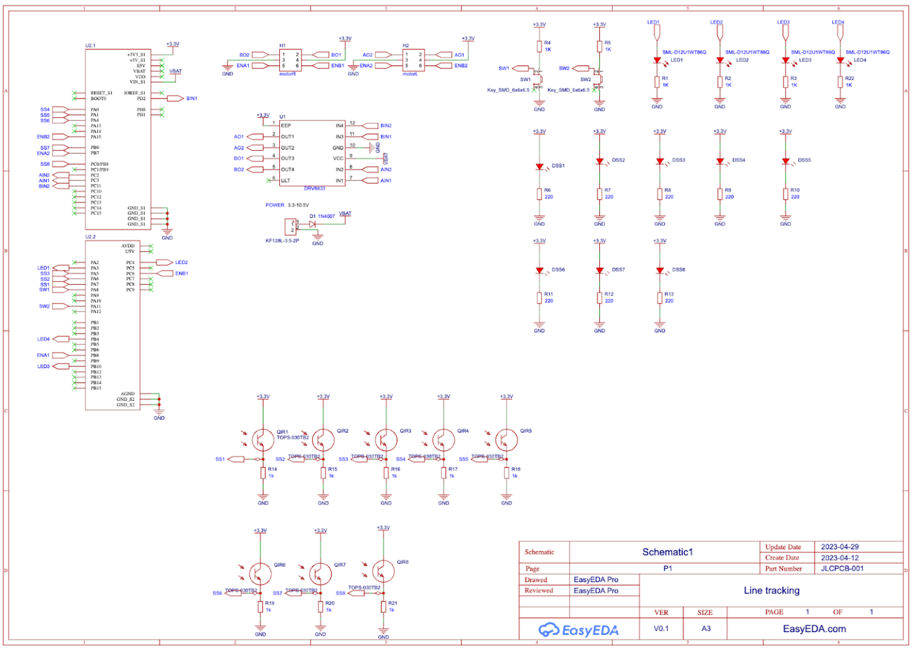


Fig 2.1.1 schematic for line following robot

   - 2.2. **PCB design**

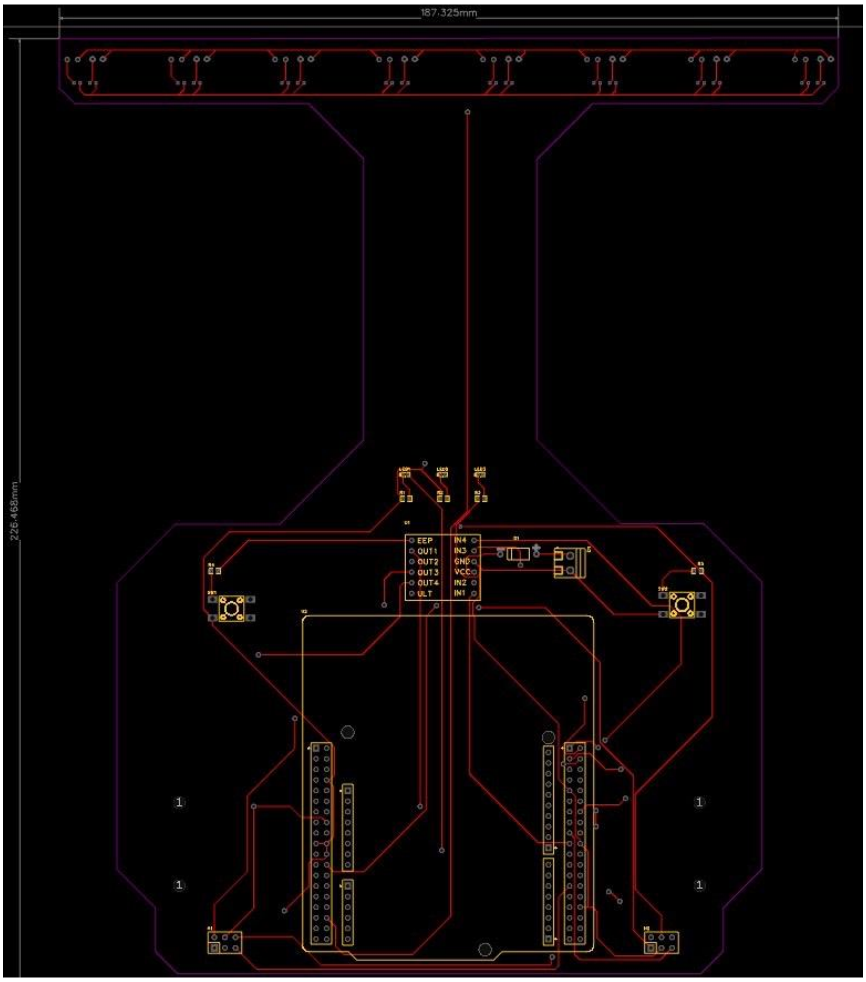


Fig 2.2.1 bottom layer of PCB

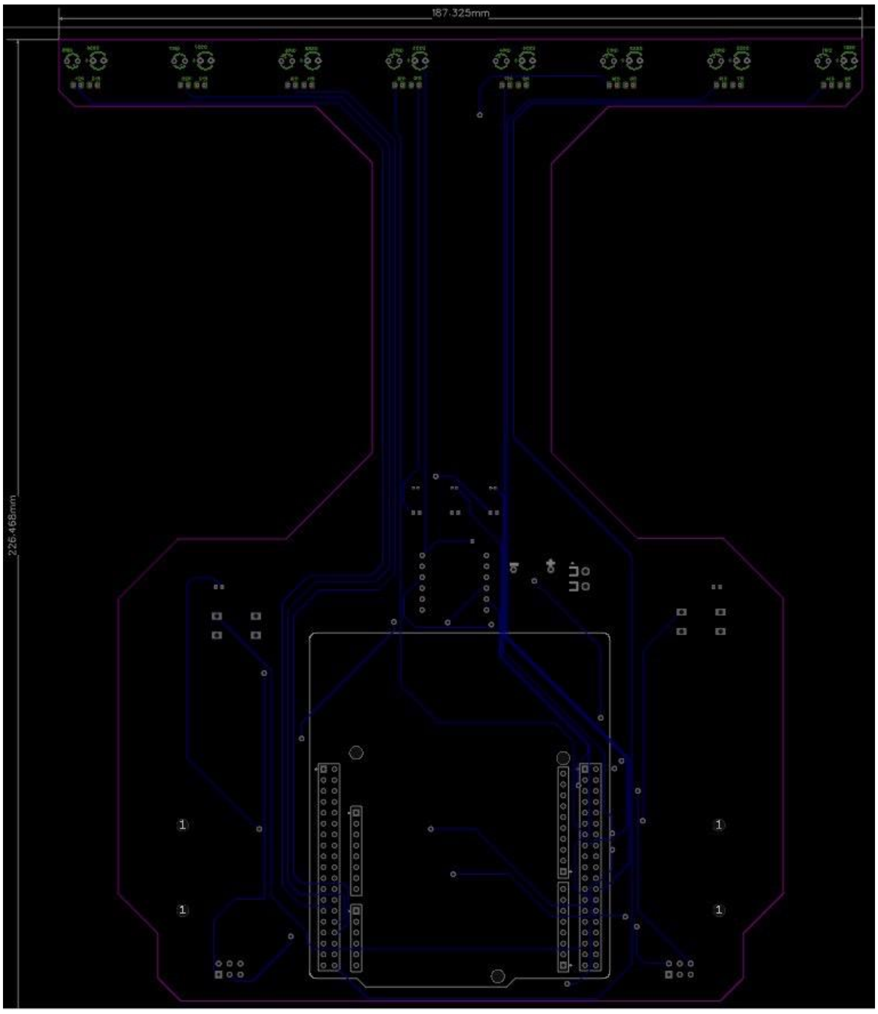


Fig 2.2.2 top layer of PCB

2.3. 3D printing

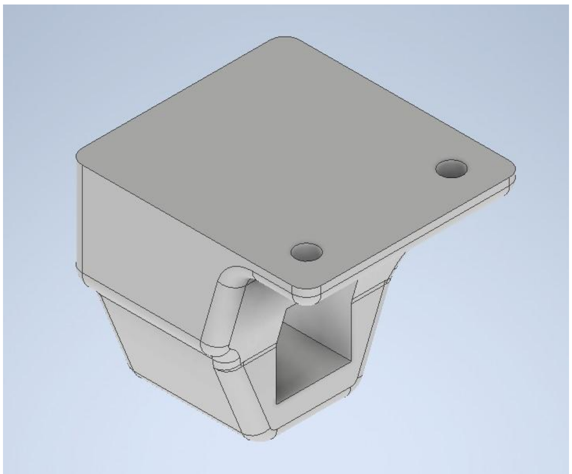


Fig 2.3.1 3D CAD for motor gripper

3. Programming

```
while(HAL_GPIO_ReadPin(GPIOC,GPIO_PIN_13) == 0){
 for(int i=0;i<8;i++){
  if (sensors_max[i]<adc_current[i]){
   sensors_max[i]=adc_current[i];
  }
  if(sensors_min[i]>adc_current[i]){
   sensors_min[i]=adc_current[i];
  }
 }
}
```

Fig 3.1 Tracks minimum and maximum values of each sensor and calculates the average value.

```
if(sensors_avg[1]<adc_current[1]&&sensors_avg[2]<adc_current[2]&&sensors_avg[3]<adc_current[3]&&senso
rs_avg[4]<adc_current[4]&&sensors_avg[5]<adc_current[5]&&sensors_avg[6]<adc_current[6]){
     Error=0;
         }//allwhite

if(sensors_avg[1]>adc_current[1]&&sensors_avg[2]>adc_current[2]&&sensors_avg[3]>adc_current[3]&&senso
rs_avg[4]>adc_current[4]&&sensors_avg[5]>adc_current[5]&&sensors_avg[6]>adc_current[6]){
         Error=0;
         }//allblack

if(sensors_avg[1]<adc_current[1]&&sensors_avg[2]<adc_current[2]&&sensors_avg[3]>adc_current[3]&&senso
rs_avg[4]>adc_current[4]&&sensors_avg[5]<adc_current[5]&&sensors_avg[6]<adc_current[6]){
             Error=0;
         }//black3,4

if(sensors_avg[1]<adc_current[1]&&sensors_avg[2]<adc_current[2]&&sensors_avg[3]<adc_current[3]&&senso
rs_avg[4]>adc_current[4]&&sensors_avg[5]<adc_current[5]&&sensors_avg[6]<adc_current[6]){
         Error=500;
         }//black4

if(sensors_avg[1]<adc_current[1]&&sensors_avg[2]<adc_current[2]&&sensors_avg[3]<adc_current[3]&&senso
rs_avg[4]>adc_current[4]&&sensors_avg[5]>adc_current[5]&&sensors_avg[6]<adc_current[6]){
         Error=1000;
                                  }//black4,5

if(sensors_avg[1]<adc_current[1]&&sensors_avg[2]<adc_current[2]&&sensors_avg[3]<adc_current[3]&&senso
rs_avg[4]<adc_current[4]&&sensors_avg[5]>adc_current[5]&&sensors_avg[6]<adc_current[6]){
         Error=1500;
         }//black5

if(sensors_avg[1]<adc_current[1]&&sensors_avg[2]<adc_current[2]&&sensors_avg[3]<adc_current[3]&&senso
rs_avg[4]<adc_current[4]&&sensors_avg[5]>adc_current[5]&&sensors_avg[6]>adc_current[6]){
         Error=2000;
         }//black5,6

if(sensors_avg[1]<adc_current[1]&&sensors_avg[2]<adc_current[2]&&sensors_avg[3]<adc_current[3]&&senso
rs_avg[4]<adc_current[4]&&sensors_avg[5]<adc_current[5]&&sensors_avg[6]>adc_current[6]){
         Error=2500;
         }//black6

if(sensors_avg[1]<adc_current[1]&&sensors_avg[2]<adc_current[2]&&sensors_avg[3]>adc_current[3]&&senso
rs_avg[4]<adc_current[4]&&sensors_avg[5]<adc_current[5]&&sensors_avg[6]<adc_current[6]){
         Error=-500;
        }//black3

if(sensors_avg[1]<adc_current[1]&&sensors_avg[2]>adc_current[2]&&sensors_avg[3]>adc_current[3]&&senso
rs_avg[4]<adc_current[4]&&sensors_avg[5]<adc_current[5]&&sensors_avg[6]<adc_current[6]){
        Error=-1000;
        }//black2,3

if(sensors_avg[1]<adc_current[1]&&sensors_avg[2]>adc_current[2]&&sensors_avg[3]<adc_current[3]&&senso
rs_avg[4]<adc_current[4]&&sensors_avg[5]<adc_current[5]&&sensors_avg[6]<adc_current[6]){
        Error=-1500;
        }//black2

if(sensors_avg[1]>adc_current[1]&&sensors_avg[2]>adc_current[2]&&sensors_avg[3]<adc_current[3]&&senso
rs_avg[4]<adc_current[4]&&sensors_avg[5]<adc_current[5]&&sensors_avg[6]<adc_current[6]){
        Error=-2000;
        }//black1,2

if(sensors_avg[1]>adc_current[1]&&sensors_avg[2]>adc_current[2]&&sensors_avg[3]<adc_current[3]&&senso
rs_avg[4]<adc_current[4]&&sensors_avg[5]<adc_current[5]&&sensors_avg[6]<adc_current[6]&&sensors_avg[7
]<adc_current[7]){
```

Fig 3.2 Determine the value of error based on the comparison with sensor values and current values.

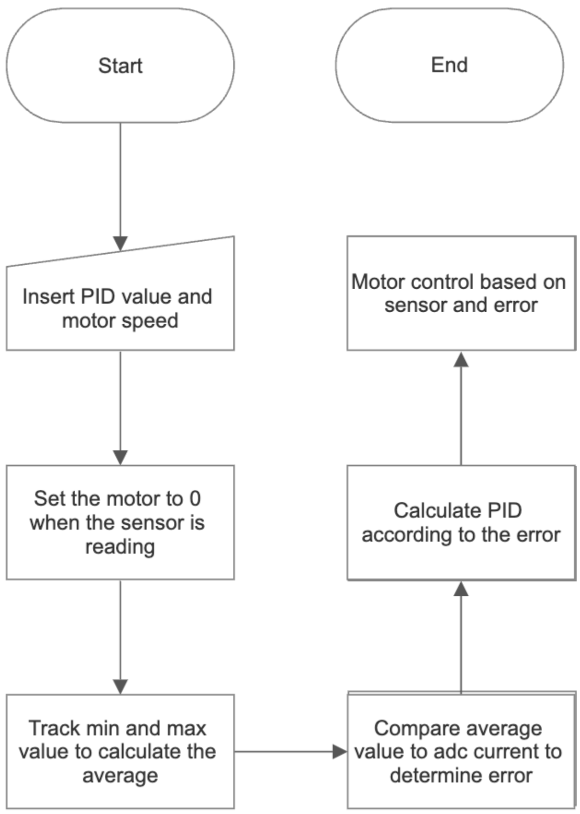
```
Error=-2500;
}//black1
P=Error;
I=I+Error;
D=Error-Lasterror;
Lasterror=Error;
float PID =(0.875*P)+(0*I)+(3*D);
sprintf(&text1,"%.2f\r\n",(float)PID);
HAL_UART_Transmit(&huart2,text1, strlen(text1),1000);
L=275+PID;
R=275-PID;
```

Fig 3.3 Insert the motor speed and a number into a float to calculate the PID.

```
L=275+PID;
R=275-PID;
if(L>=0&&R>=0){
TIM3->CCR3=0;//backward.L
TIM3->CCR4=L;//forward.L
TIM3->CCR1=0;//backward.L
TIM3->CCR2=R;//forward.L
}
if(L>=0&&R<0){
TIM3->CCR3=0;//backward.L
TIM3->CCR4=L;//forward.L
TIM3->CCR1=-R;//backward.L
TIM3->CCR2=0;//forward.L
}
if(L<0&&R>=0){
TIM3->CCR3=-L;//backward.L
TIM3->CCR4=0;//forward.L
TIM3->CCR1=0;//backward.L
TIM3->CCR2=R;//forward.L
}
if(L<0&&R<0){
TIM3->CCR3=-L;//backward.L
TIM3->CCR4=0;//forward.L
TIM3->CCR1=-R;//backward.L
TIM3->CCR2=0;//forward.L
```

Fig 3.4 Control motor output based on sensor readings and error values.

4. Flowchart



5. **Results**

This is the result video for work as a whole. Click here

6. **Conclusions**

In conclusion, the STM32 microcontroller's quick response time, the PID control method, and precisely positioning the sensors were all necessary for the successful building of a line follower robot. Eight IR sensors were used, which allowed for precise line tracking as well as real-time changes and corrections. In the disciplines of robotics and control systems, this research has made significant contributions that have laid the groundwork for further development. In addition, Track A took the robot 29 seconds to follow, whereas Track B took 1.48 minutes.

Percentage

33.33% - 65011376 NAPAT
ณภัทร

33.33% - 65011601 THITIWUT
ฐิติวุฒิ

33.33% - 65110141 KUNLANITH
Kulanith

Reference

ST, (2023), STM32 32-bit Arm Cortex MCUs, STM32 32-bit Arm Cortex MCUs

EasyEDA, (2023), About EastEDA, About EasyEDA

Packt, (2023), A short background of 3D printing, A short background of 3D printing

NI, (30/3/2023)The PID Controller & Theory Explained, The PID Controller & Theory Explained - NI