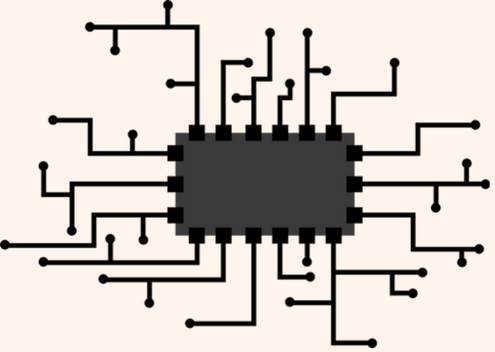
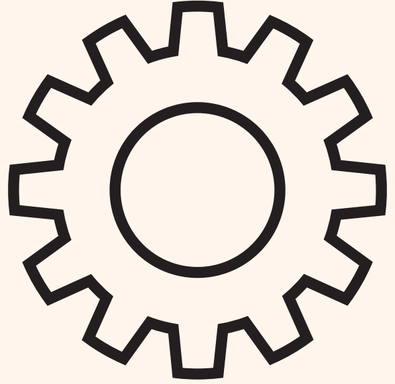
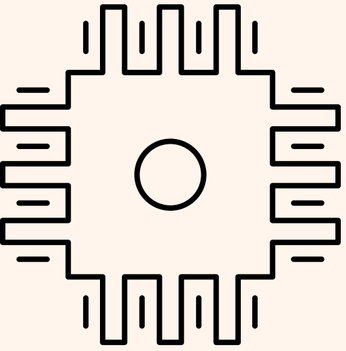


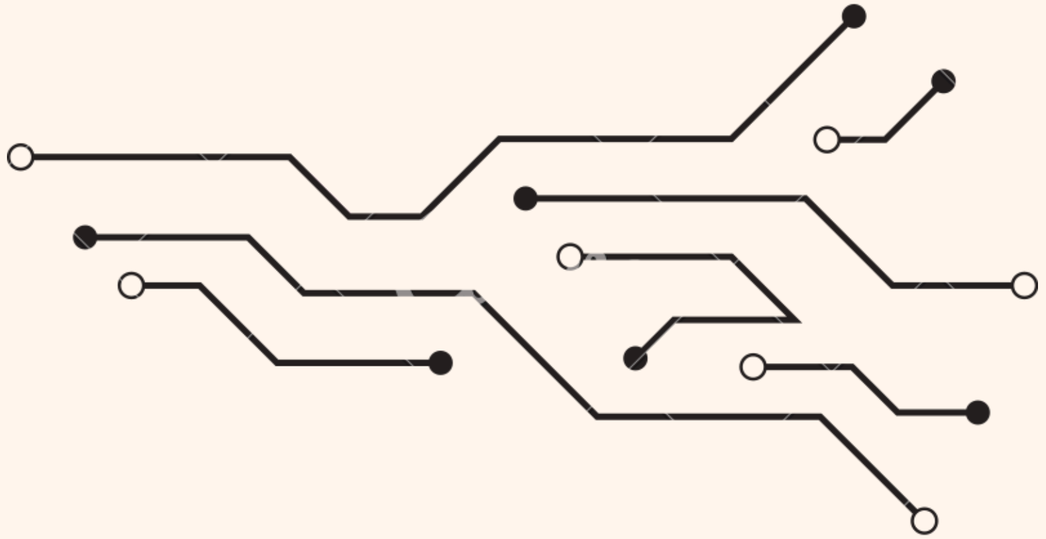


MICRO FINAL PROJECT

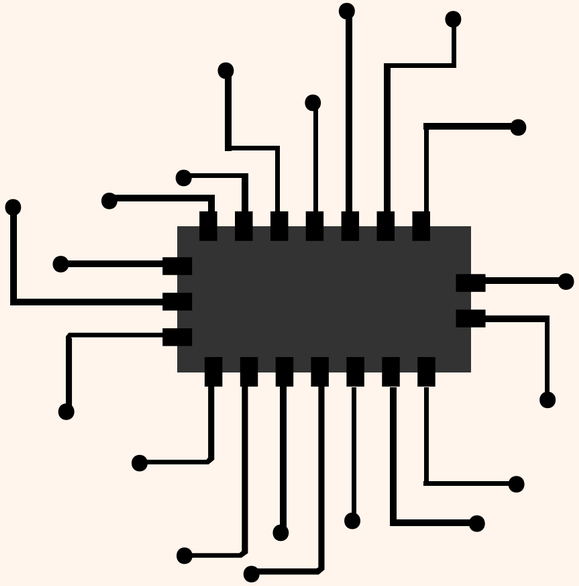




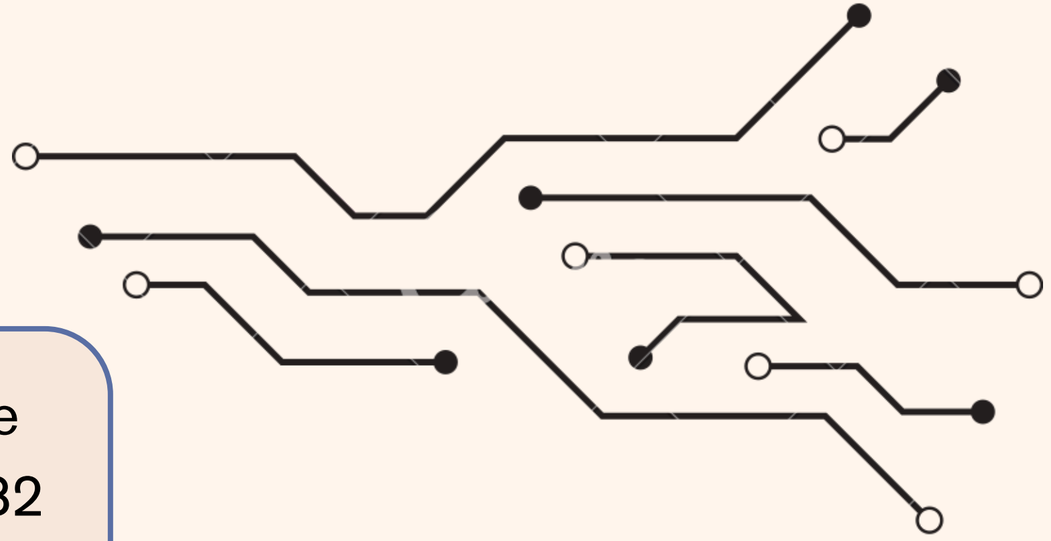
TEAM MEMBERS



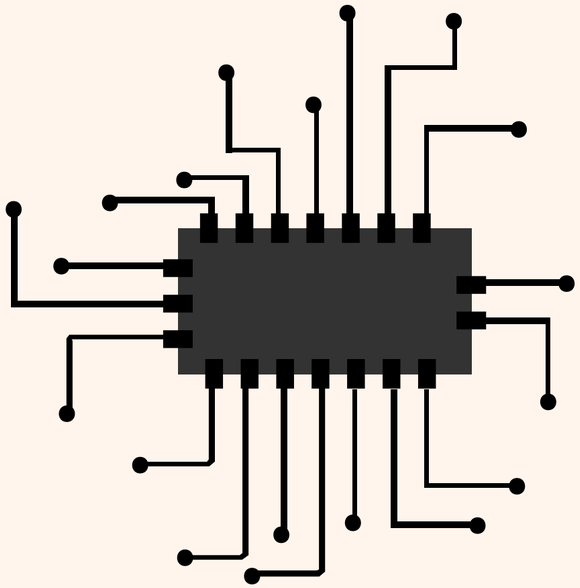
- Lucky Agarwal 67011698
- Napon Namchawat 67011179
- Chindanani Thongsaenkham 67011098
- Tanadol Lertwerasirikul 67110134
- Kanachok Pongwarin 67011604
- Poonyaphat Sukphew 67011275



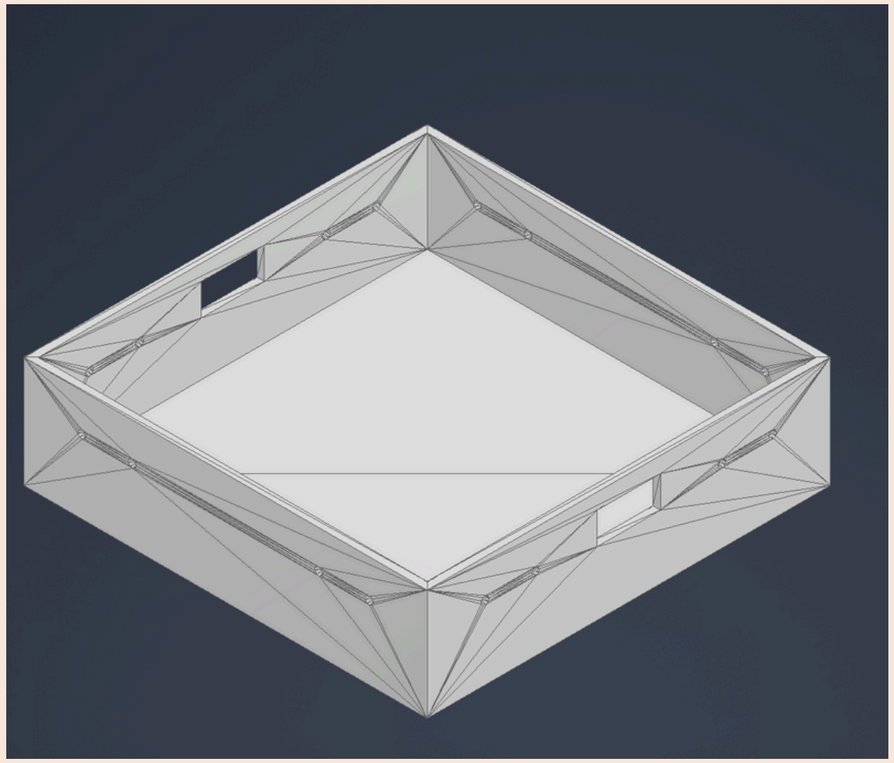
ABSTRACT



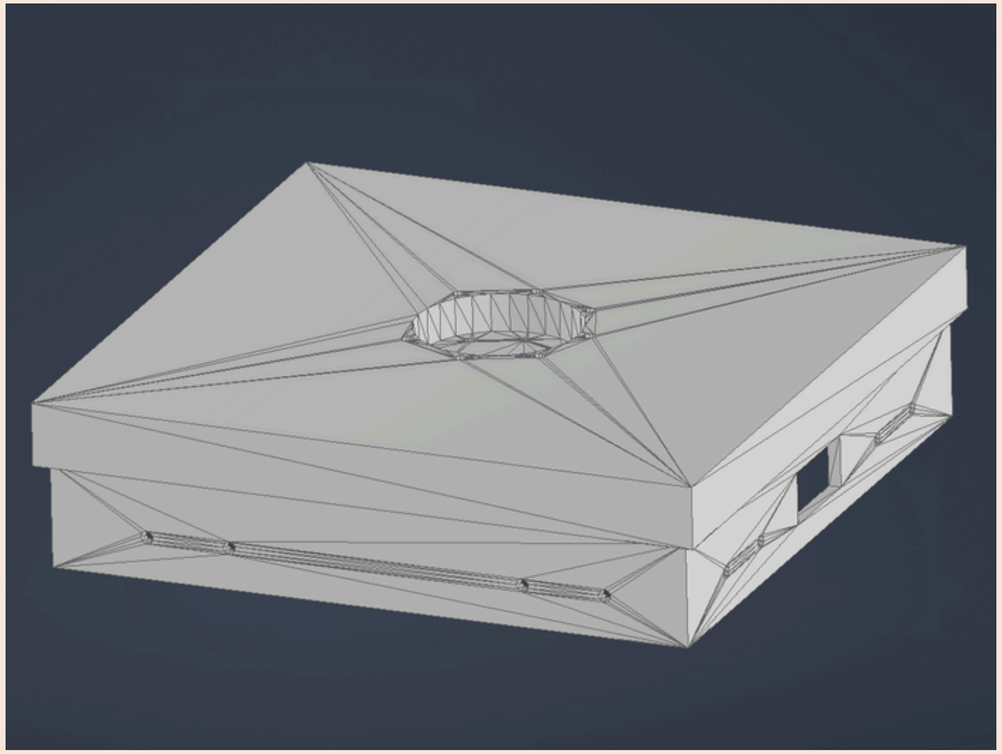
This project involves the control of a BLDC motor using the MODBUS protocol over an RS485 interface, with an STM32 microcontroller as the core controller. The MODBUS communication allows for speed control of the motor, while real-time data such as target speed, current speed, and motor angle is displayed on an OLED screen. An emergency stop button is integrated for immediate motor shutdown to enhance safety. A custom PCB was designed to interconnect all components, and a 3D-printed base was created to house the system.



Hardware design

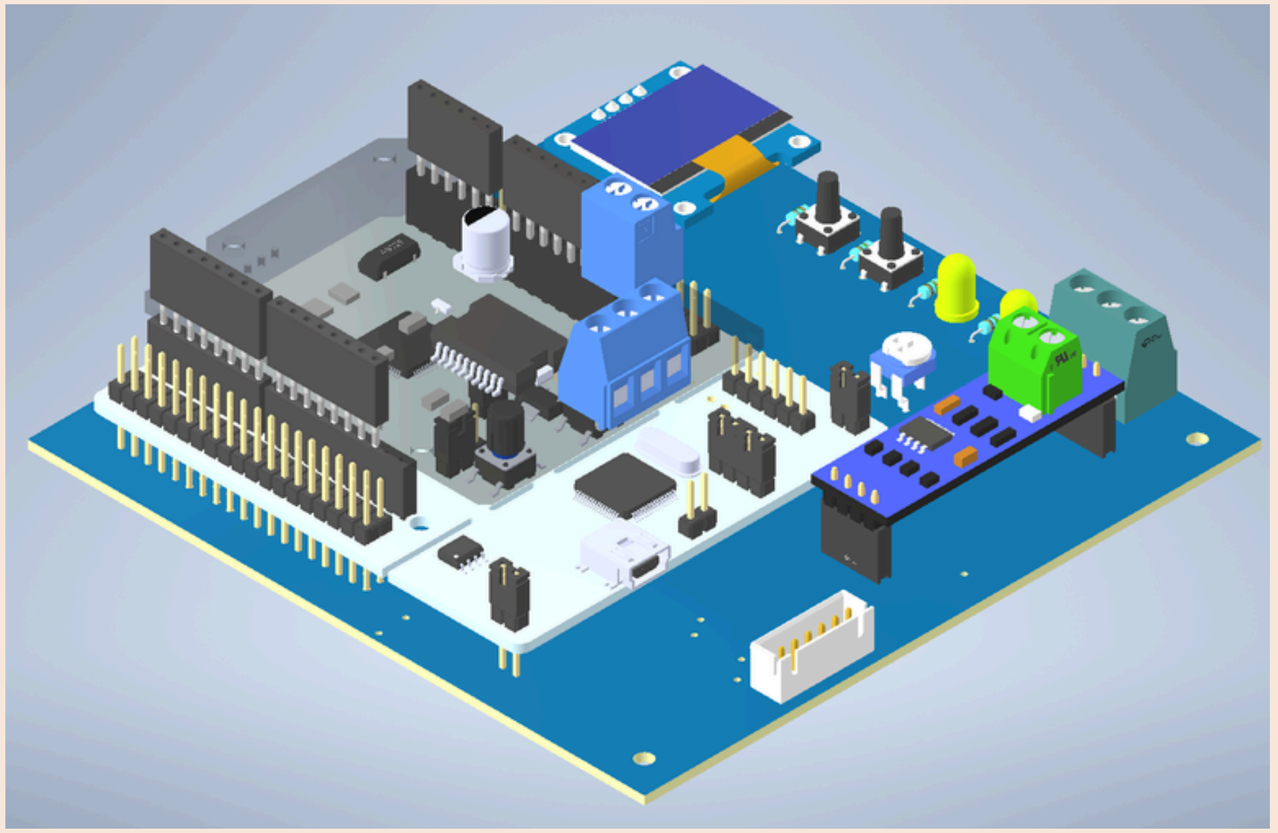


Base Holder



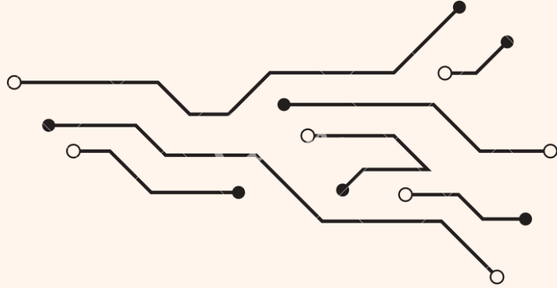
Top Holder

Hardware design



PCB Design

Software Design: Functions



```
int ButtonPressed(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin) {
   if (HAL_GPIO_ReadPin(GPIOx, GPIO_Pin) == GPIO_PIN_RESET) {
       HAL_Delay(50);
       if (HAL_GPIO_ReadPin(GPIOx, GPIO_Pin) == GPIO_PIN_RESET) {
           return 1;
       }
   }
   return 0;
}

void MAX485_EnableTransmit(){
  HAL_GPIO_WritePin(MAX485_DE_PIN_GPIO_Port,MAX485_DE_PIN_Pin, GPIO_PIN_SET);
  HAL_GPIO_WritePin(MAX485_RE_PIN_GPIO_Port,MAX485_RE_PIN_Pin, GPIO_PIN_SET);
}

void MAX485_EnableRecieve(){
  HAL_GPIO_WritePin(MAX485_DE_PIN_GPIO_Port,MAX485_DE_PIN_Pin, GPIO_PIN_RESET);
  HAL_GPIO_WritePin(MAX485_RE_PIN_GPIO_Port,MAX485_RE_PIN_Pin, GPIO_PIN_RESET);
}

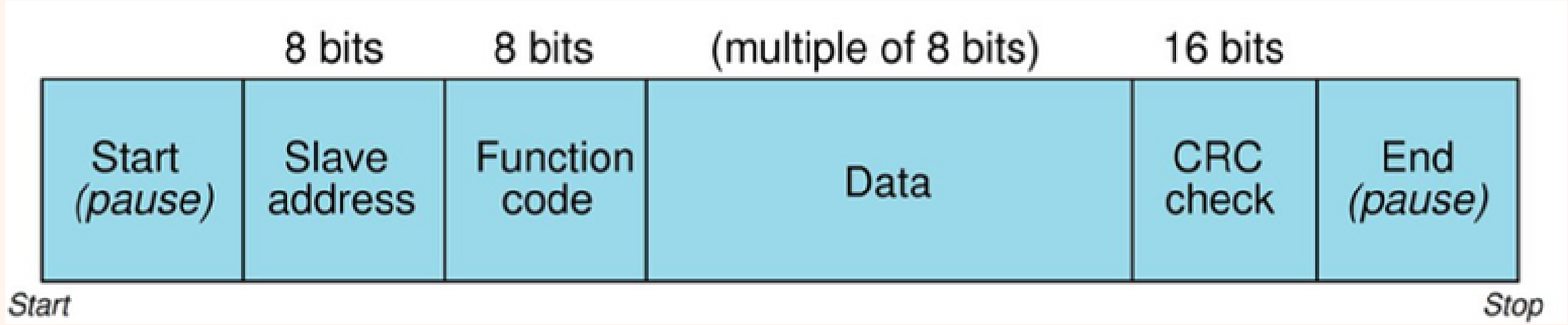
void HAL_UARTEx_RxEventCallback(UART_HandleTypeDef *huart, uint16_t Size) {
            memcpy(TxData, RxData, Size);
            MAX485_EnableTransmit();
            HAL_UART_Transmit_IT(&huart1, TxData,8);

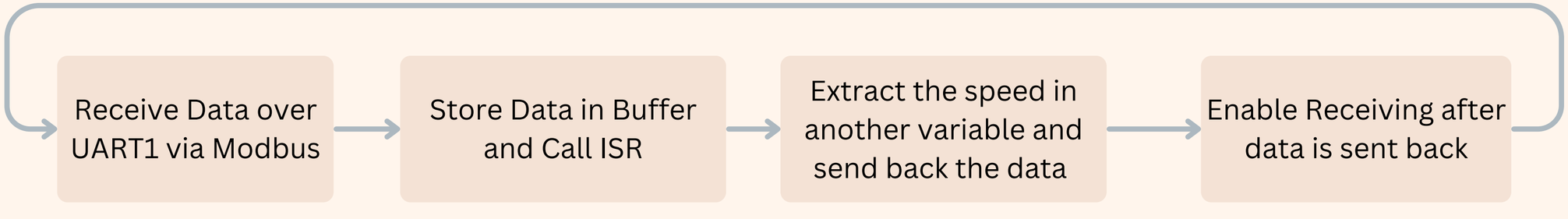
            data = RxData[4] << 8 | RxData[5];
    }

void HAL_UART_TxCpltCallback(UART_HandleTypeDef *huart) {
    if (huart->Instance == USART1) {
        MAX485_EnableRecieve();
        HAL_UARTEx_ReceiveToIdle_IT(&huart1, RxData, sizeof(RxData));
    }
}
```

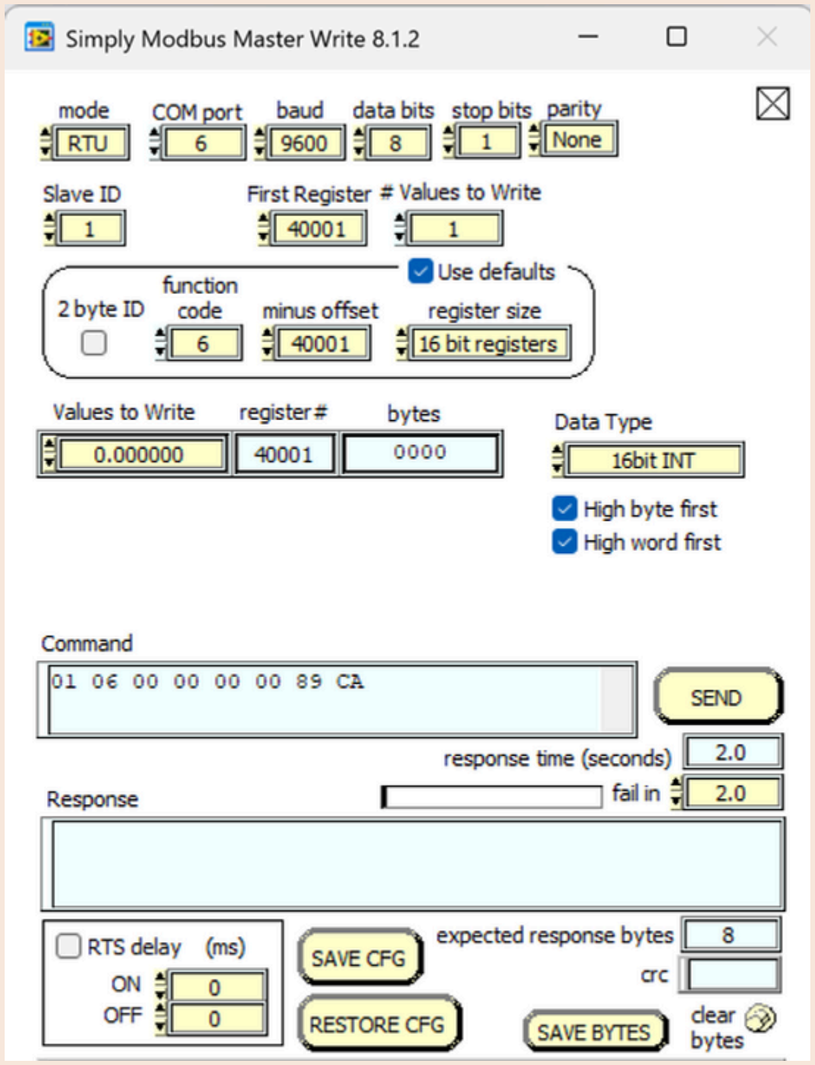
- The UART Interrupt functions handles the Modbus communication
- The button pressed function returns the state of the button and debounces any delays
- The transmitting and receiving of the MAX485 module is controlled by setting the DE and RE pin High or Low respectively

Software Design: MODBUS





Software Design - MODBUS



Modbus Control Application

Software Design - While Loop Code

```
while (1)
{

    if (is_stopping) {
      ssd1306_Fill(Black);
      ssd1306_SetCursor(10,10);
      ssd1306_WriteString("EMERGENCY" , Font_11x18, White);
      ssd1306_SetCursor(10,40);
      ssd1306_WriteString("STOP" , Font_11x18, White);
      ssd1306_UpdateScreen();
    }

    else {
        ssd1306_Fill(Black);
      sprintf(v , "VEL: %.2f rad/s" , get_vel());
      sprintf(s , "ANG: %.2f rad", get_angle());
      sprintf(dta, "VEL_TARGET: %d" , data);
      ssd1306_SetCursor(10,10);
      ssd1306_WriteString(s , Font_7x10, White);
      ssd1306_SetCursor(10,30);
      ssd1306_WriteString(v, Font_7x10, White);
      ssd1306_SetCursor(10,50);
      ssd1306_WriteString(dta, Font_7x10, White);
      ssd1306_UpdateScreen();
    }

      f_spd = (float) data;

      if (ButtonPressed(GPIOC, GPIO_PIN_13)) {
        speed = 0.0f;
        is_stopping = !is_stopping;
      }


    while (speed != f_spd && !is_stopping) {

        if (speed < f_spd) {

            speed += step;

            if (speed > f_spd)
              speed = f_spd;

        }

        else if (speed > f_spd) {

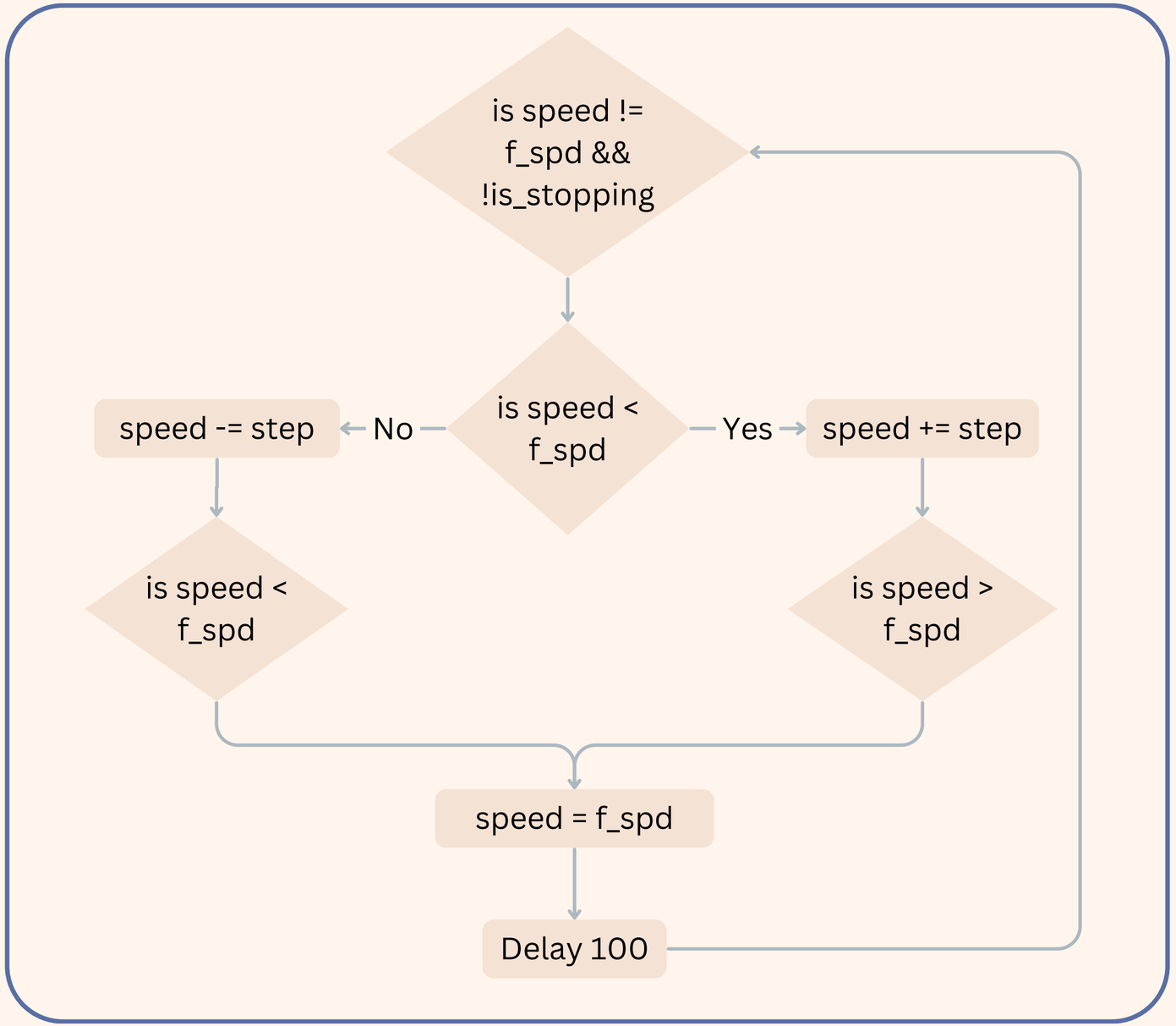
            speed -= step;

            if (speed < data)
              speed = f_spd;

        }
        HAL_Delay(100);
    }
  }
```

- When the button is pressed it toggles the is_stopping flag which switches the system from a stop state to a running state
- The speed of the motor increases gradually as it gets incremented or decremented by 1 until it reaches the target speed
- The delay ensures the change is speed is smooth
- The OLED screen displays the current angle, the current speed and the target speed

Software Design: Speed Change



Software Design - Motor Control

```
int main(){
as5048a_init(&hspi2, SPI_CS_GPIO_Port, SPI_CS_Pin);
driver_init(&htim2,
              TIM_CHANNEL_1,
              &htim3,
              TIM_CHANNEL_2,
              &htim2,
              TIM_CHANNEL_2,
              FOC_EN_GPIO_Port,
              FOC_EN_Pin,
              20.0f,
              20.0f);
bldc_open_loop_init(7, 10.0f);
foc_init();
HAL_TIM_Base_Start(&htim1);
HAL_TIM_Base_Start_IT(&htim4);
}

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
  if (htim->Instance == TIM4)
  {
    timer_counter = __HAL_TIM_GET_COUNTER(&htim1);
    __HAL_TIM_SET_COUNTER(&htim1, 0);

    const float dt_while = (float)timer_counter * 1e-6f;

    loop_foc();
    bldc_move(speed, dt_while);
  }
}
```

- The Encoder of motor and the motor driver are initialized in the main function and the foc algorithm is started along with an open loop control
- The motor is controlled periodically by TIM4 and uses timer 1 to measure the elapsed time
- The functions loop_foc() and bldc_move() start the motion of the motor and set the appropriate speed

Results



https://youtube.com/shorts/Twc-KzzQh6M?feature=share

- As seen in this video, our motor is successfully controlled using Modbus to adjust the speed
- Our OLED displays the necessary information and the emergency stop button works well too

Application of BLDC

- BLDC motors can be used for haptic feedback as they have precise control over their position
- This makes them ideal to be used as a knob to control a large variety of things such as gears in cars, using them as an input device, controlling a display etc.
- Our idea is to use the bldc motor as a gear selector in electrical cars allowing the driver to easily select the gear

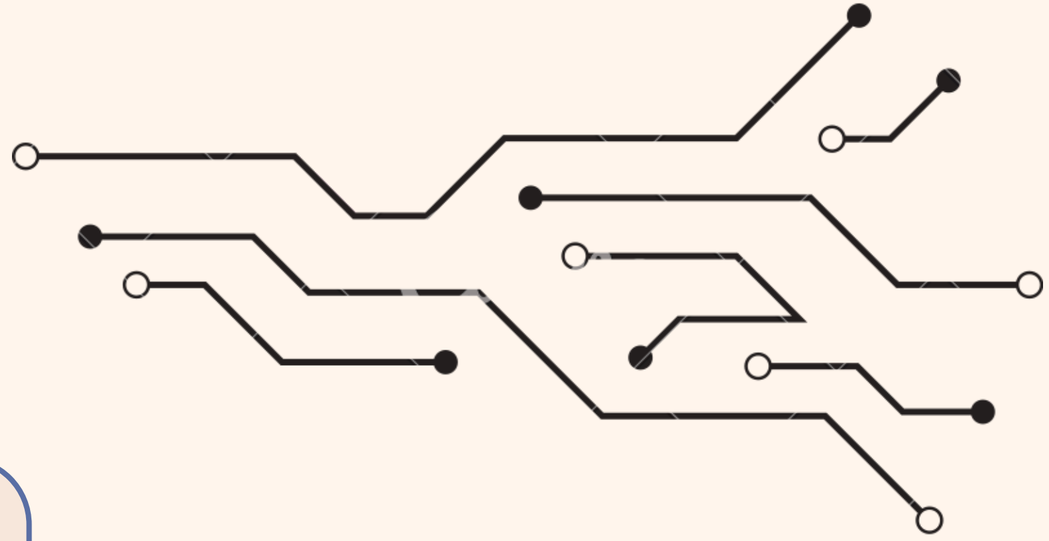


Application of BLDC

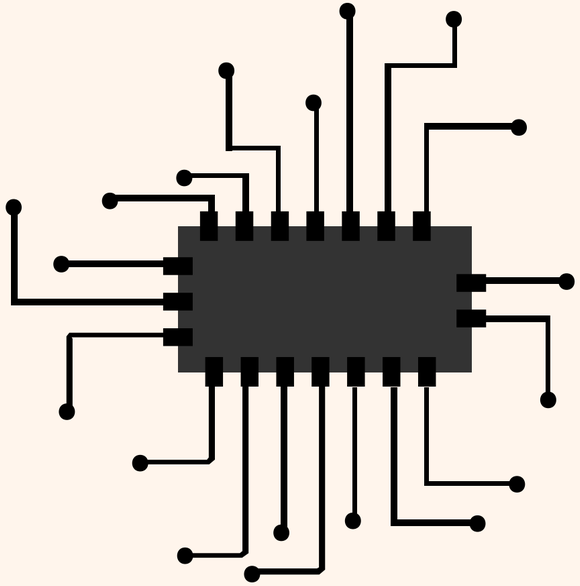
- Our 3d model shows our idea
- The central knob is the bldc motor which can rotate and the haptic feedback is sent to the car's control system and it changes the gears accordingly
- They offer several advantage such as precise control, fast response time and customizability
- As they can be controlled via software, the speed and torque can be easily adjusted for different cars

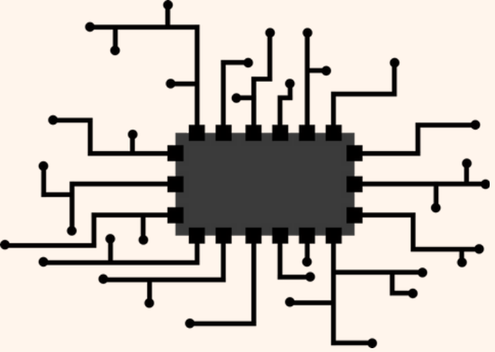


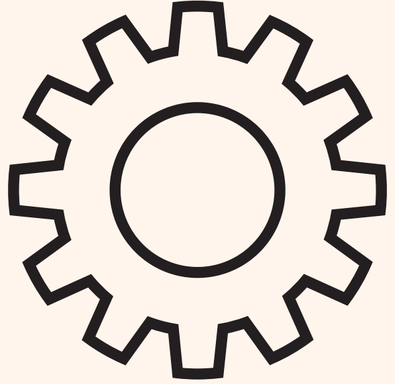
CONCLUSION



- In conclusion, our project successfully implemented the MODBUS communication between the STM32 board and the PC and successfully controlled the motor speed through it
- We successfully implemented an emergency stop button for safety and displayed the values on the OLED screen







THANK YOU

