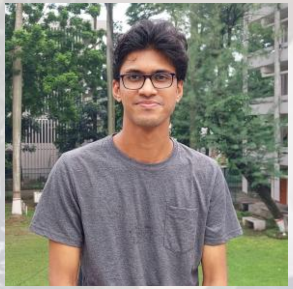
EEE 416-Microprocessor and Embedded Systems Laboratory
July 2023 Level-4 Term-1 Section C2
Final Project Presentation

STM32 Black Pill Dev Board

SUBMITTED BY – GROUP 07

Md. Ridwanul Haque
1906165

Md. Toky Tazwar
1906174


Md. Julkar Naim Joy
1906181

Raihan Amin Rana
1906186

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY
DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

Outline

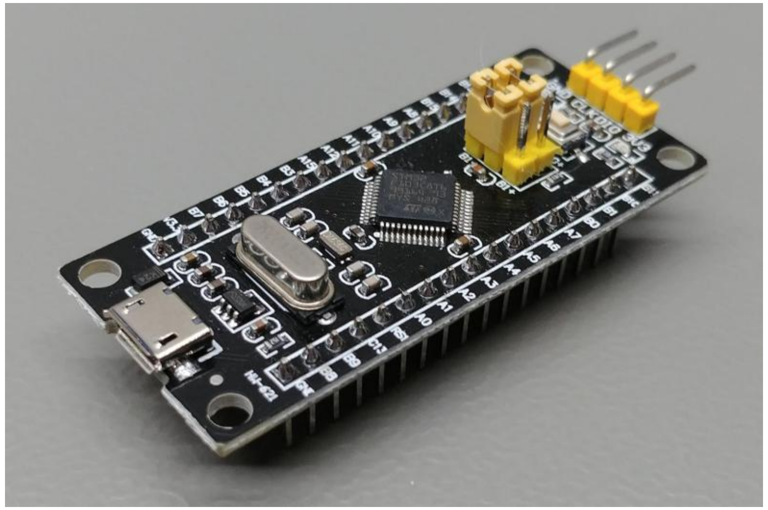
1. Summary
2. Introduction
3. Design
4. Implementation
5. Analysis and Evaluation
6. References

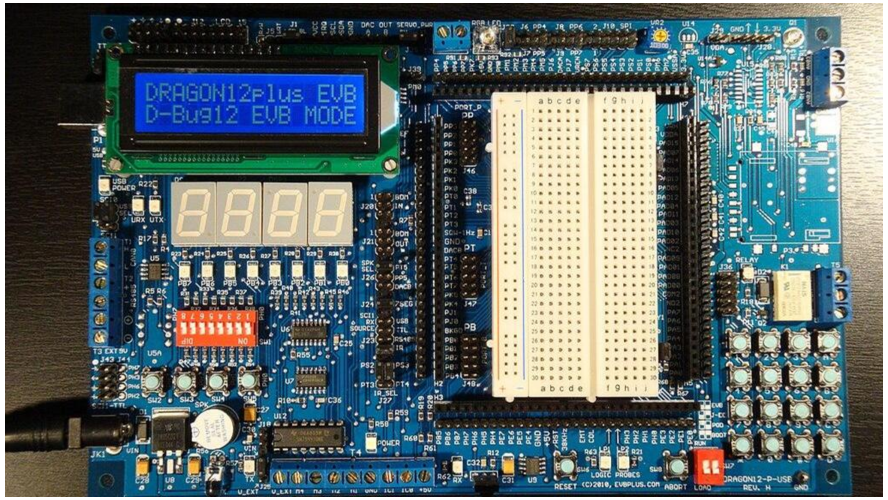
1. Summary / Abstract

Our Project is a compact design and implementation of a development board with stm32 black pill microprocessor which can perform all the experiments of EEE416 microprocessor and embedded systems laboratory. The peripheral devices are placed in a way so that it takes minimum floor area and the GPIO pins are used efficiently to make sure that all the devices are connected to suitable pin.

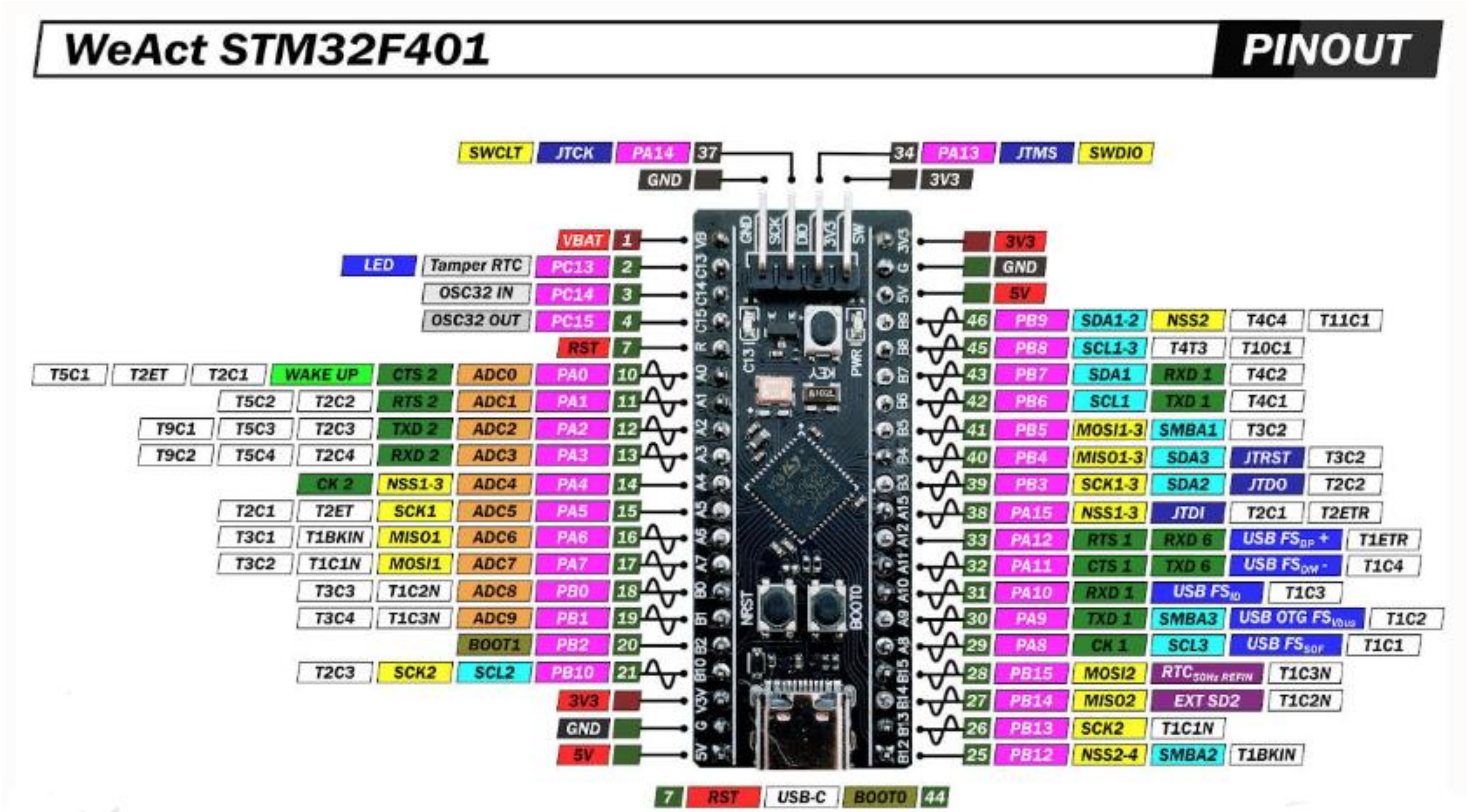
2. Introduction

Compact design is extremely important for cost reduction, space efficiency, reliability and so on. This project is an example of efficient compact design and implementation using stm32 black pill. .





2. STM32 Black Pill


WeAct STM32F401
PINOUT
SWCLT JTCK PA14 37
34 PA13 JTMS SWDIO
GND
3V3
VBAT 1
3V3
LED Tamper RTC PC13 2
GND
OSC32 IN PC14 3
5V
OSC32 OUT PC15 4
46 PB9 SDA1-2 NSS2 T4C4 T11C1
RST 7
45 PB8 SCL1-3 T4T3 T10C1
T5C1 T2ET T2C1 WAKE UP CTS 2 ADC0 PA0 10
43 PB7 SDA1 RXD 1 T4C2
T5C2 T2C2 RTS 2 ADC1 PA1 11
42 PB6 SCL1 TXD 1 T4C1
T9C1 T5C3 T2C3 TXD 2 ADC2 PA2 12
41 PB5 MOSI1-3 SMBA1 T3C2
T9C2 T5C4 T2C4 RXD 2 ADC3 PA3 13
40 PB4 MISO1-3 SDA3 JTRST T3C2
CK 2 NSS1-3 ADC4 PA4 14
39 PB3 SCK1-3 SDA2 JTDO T2C2
T2C1 T2ET SCK1 ADC5 PA5 15
38 PA15 NSS1-3 JTDI T2C1 T2ETR
T3C1 T1BKIN MISO1 ADC6 PA6 16
33 PA12 RTS 1 RXD 6 USB FSDP + T1ETR
T3C2 T1C1N MOSI1 ADC7 PA7 17
32 PA11 CTS 1 TXD 6 USB FSDM - T1C4
T3C3 T1C2N ADC8 PB0 18
31 PA10 RXD 1 USB FSID T1C3
T3C4 T1C3N ADC9 PB1 19
30 PA9 TXD 1 SMBA3 USB OTG FSVbus T1C2
BOOT1 PB2 20
29 PA8 CK 1 SCL3 USB FSSOF T1C1
T2C3 SCK2 SCL2 PB10 21
28 PB15 MOSI2 RTC50Hz REFIN T1C3N
3V3
27 PB14 MISO2 EXT SD2 T1C2N
GND
26 PB13 SCK2 T1C1N
5V
25 PB12 NSS2-4 SMBA2 T1BKIN
7 RST USB-C BOOT0 44
GND SCK DIO 3V3 SW
C13 PWR KEY NRST BOOT0

3.1 Design: Methodology

❑ Familiarize with STM32F401CDU6 Microcontroller

❑ Familiarize with STM32 Black Pill & CubeIDE

❑ Buying Hardware

❑ Coding & Debugging with STM32 Black Pill

❑ PCB Design

❑ PCB Manufacturing

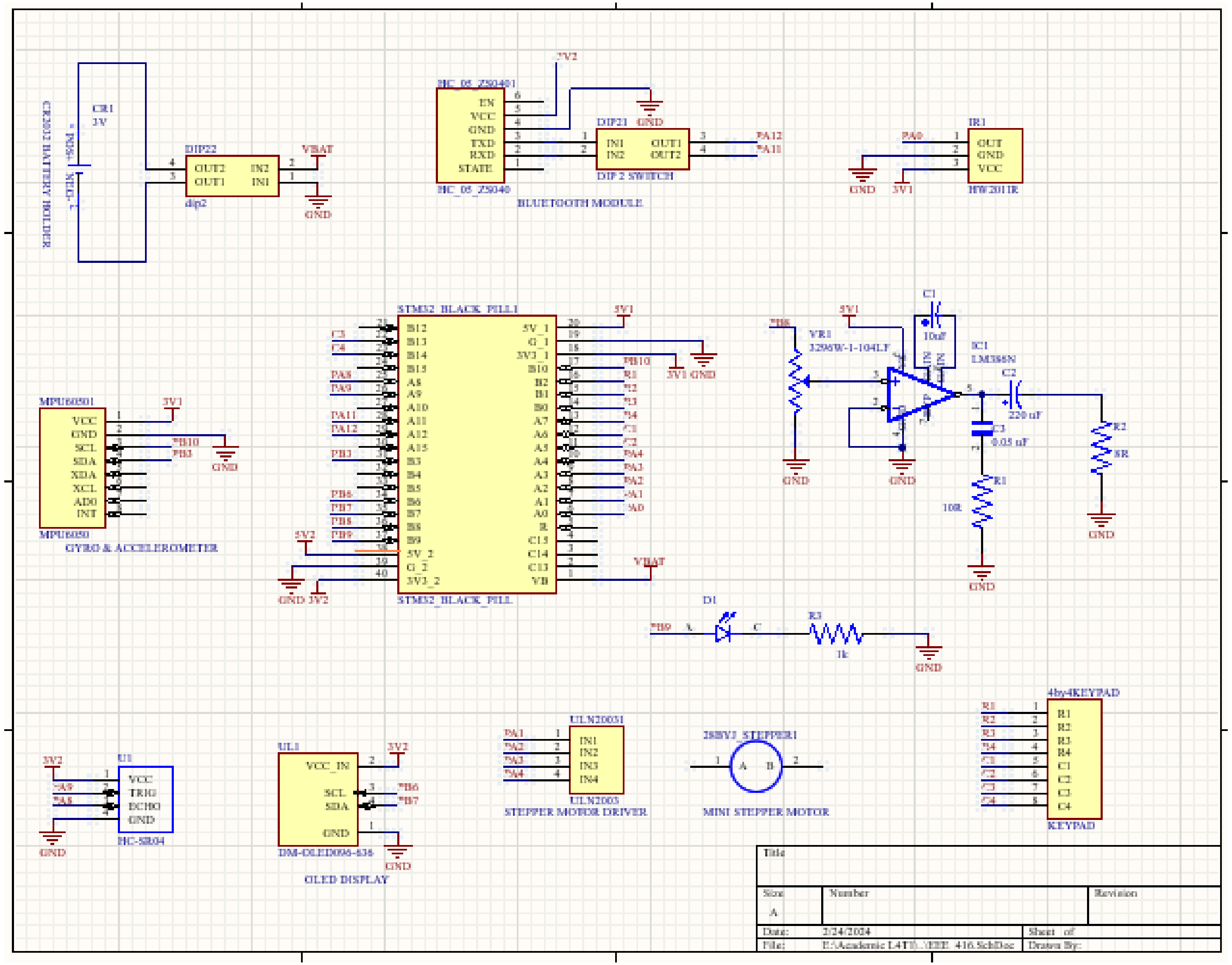
❑ Testing Final Circuit

3.2 Design: Components

- STM32 Black Pill
- Ultrasonic Sensor
- Bluetooth (HC- SR04)
- 4*4 Keypad
- Battery Holder
- Stepper Motor Driver
- Stepper Motor

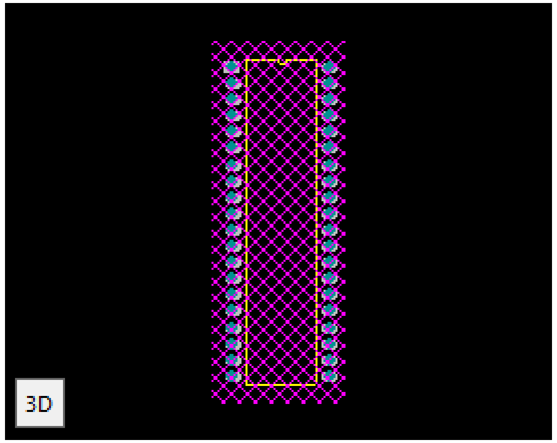
- MPU6050
- IR Sensor(HW201)
- Bluetooth
- Mini Speaker
- LM386 Amplifier
- Capacitors
- Resistors, Potentiometer

3.2 Design: Circuit Diagram(Schematics)

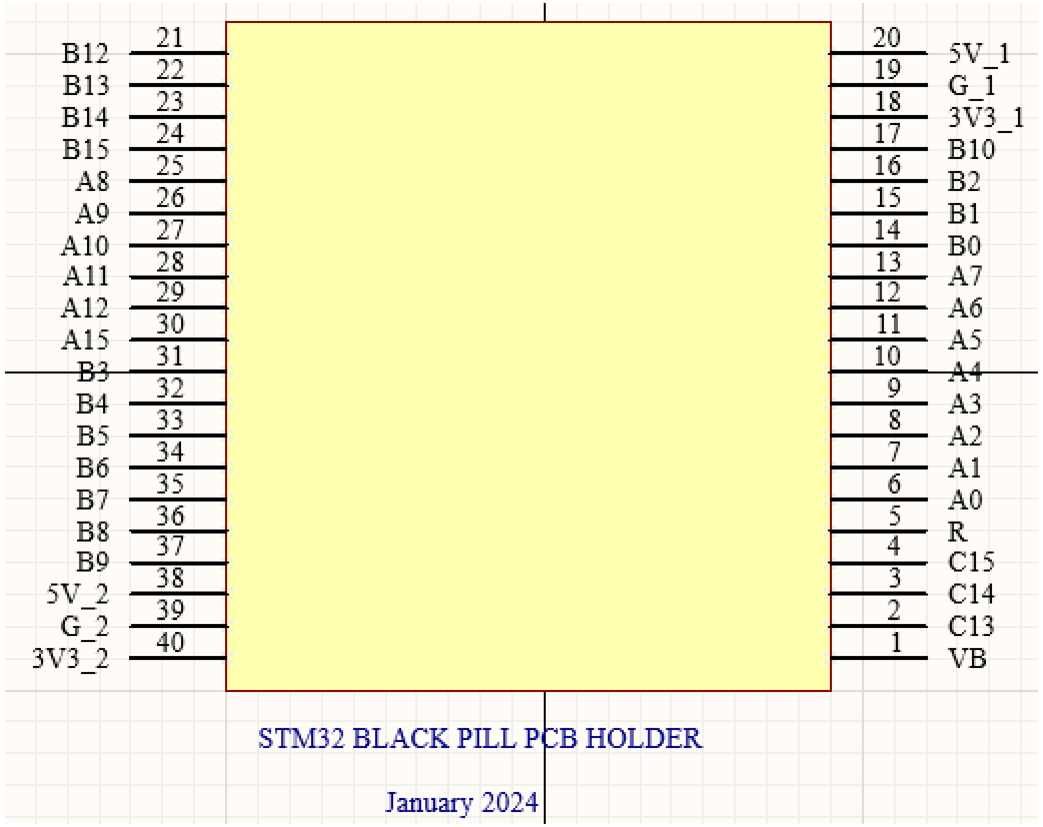


3.4 Design: PCB Layout and 3D Rendering

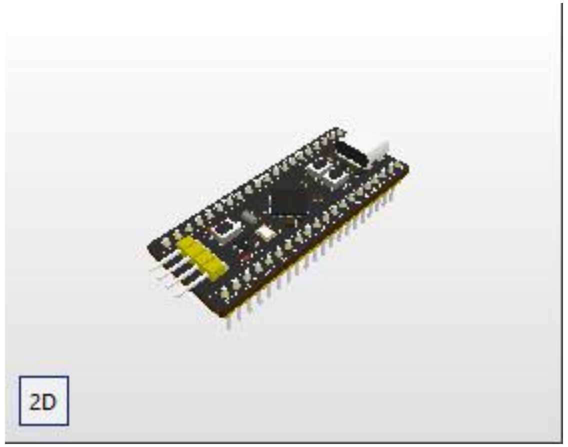
Additional Work: designing a library for STM32 Black Pill



Footprint



Schematics

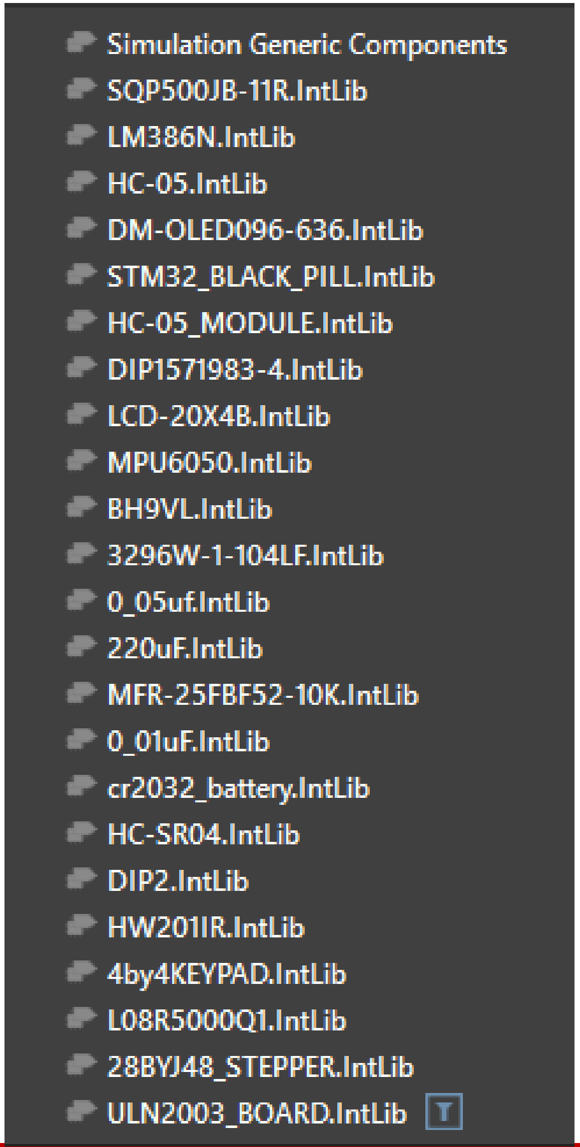


3D rendering(collected)

3.4 Design: Designing Custom Libraries

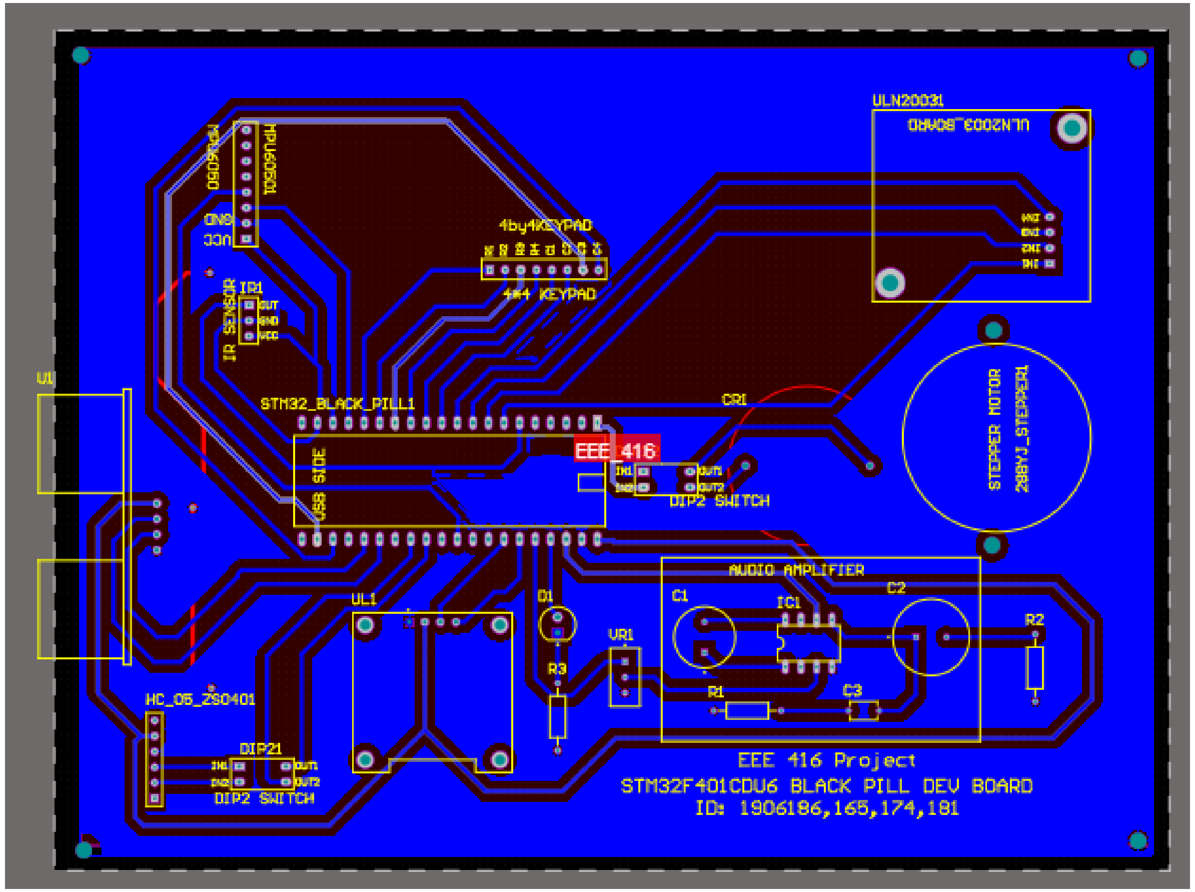
Additional Custom Libraries

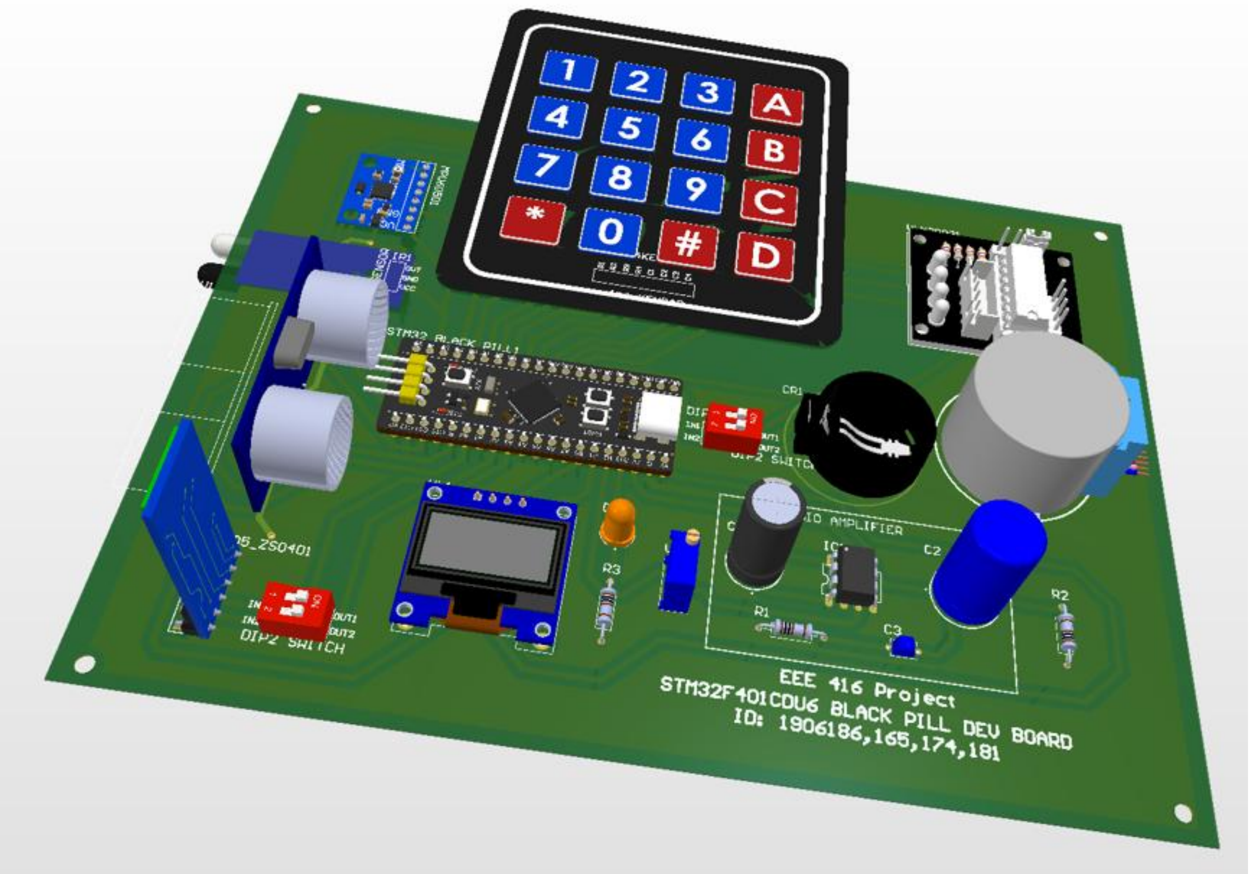
- Ultrasonic Sensor
- Bluetooth
- Keypad
- Battery Holder
- Stepper Motor Driver
- Stepper Motor



Simulation Generic Components
SQP500JB-11R.IntLib
LM386N.IntLib
HC-05.IntLib
DM-OLED096-636.IntLib
STM32_BLACK_PILL.IntLib
HC-05_MODULE.IntLib
DIP1571983-4.IntLib
LCD-20X4B.IntLib
MPU6050.IntLib
BH9VL.IntLib
3296W-1-104LF.IntLib
0_05uf.IntLib
220uF.IntLib
MFR-25FBF52-10K.IntLib
0_01uF.IntLib
cr2032_battery.IntLib
HC-SR04.IntLib
DIP2.IntLib
HW201IR.IntLib
4by4KEYPAD.IntLib
L08R5000Q1.IntLib
28BYJ48_STEPPER.IntLib
ULN2003_BOARD.IntLib

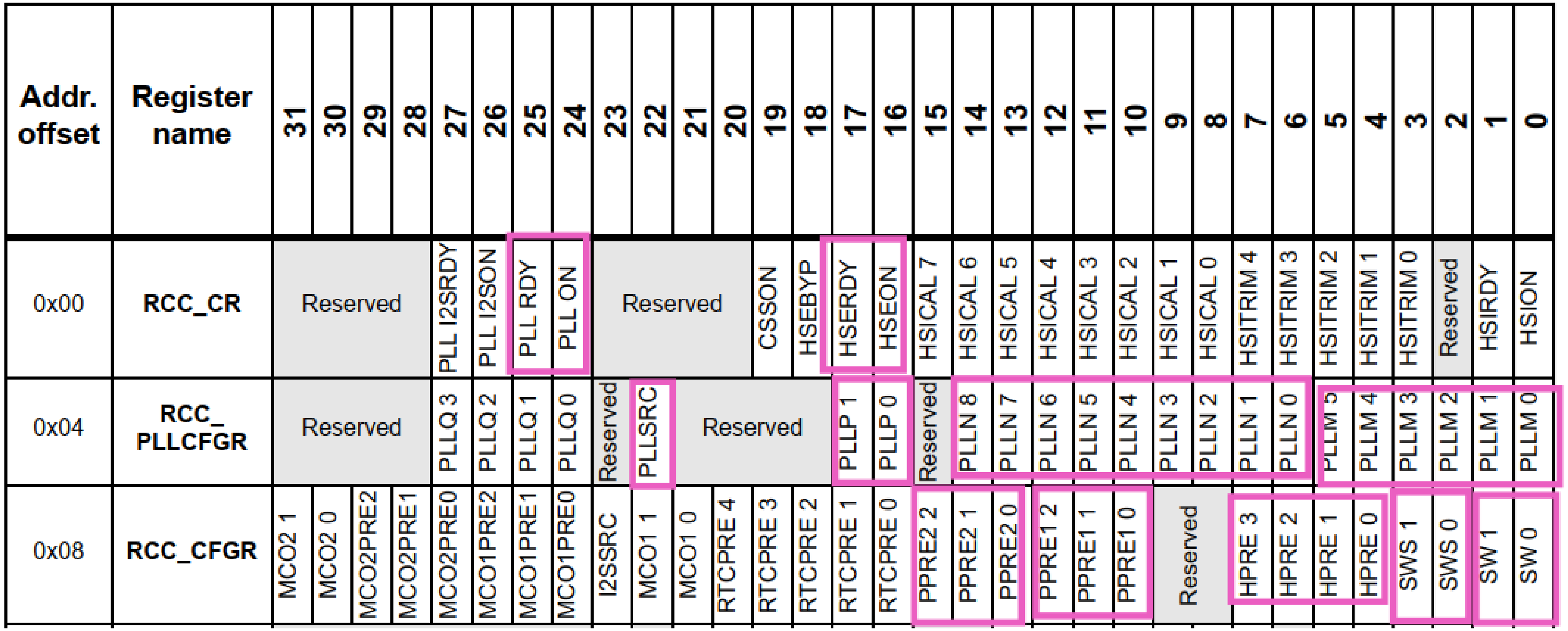
3.4 Design: PCB Layout and 3D Rendering





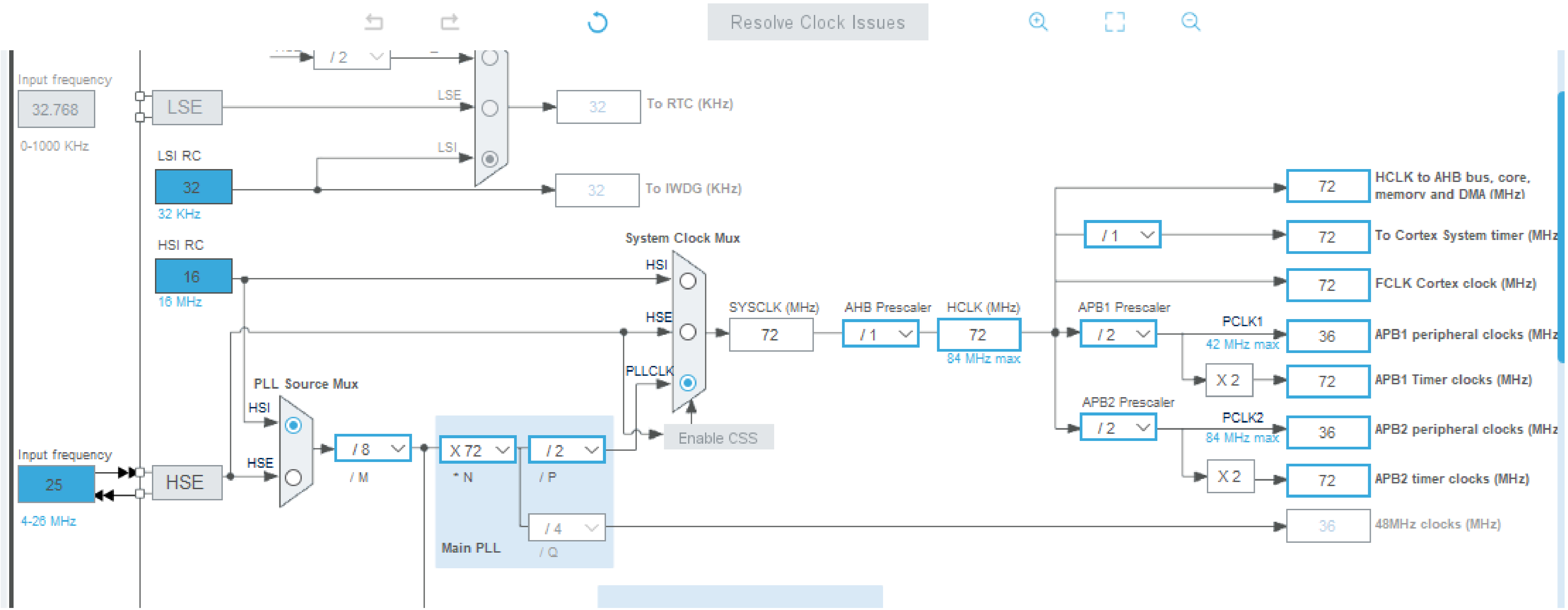
4 Implementation: LED Blinking
Clock SetUp using CubeIDE Interface

Table 22. RCC register map and reset values for STM32F401xB/C and STM32F401xD/E

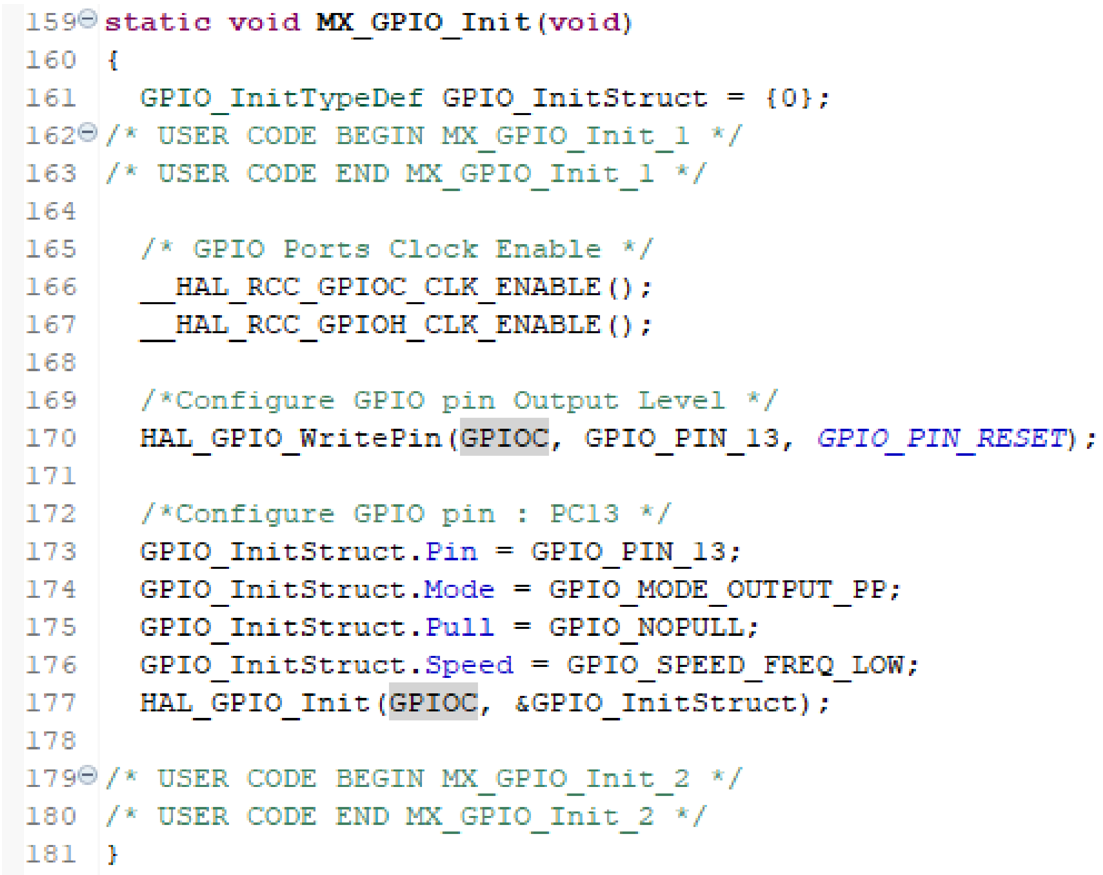


Addr. offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	RCC_CR	Reserved				PLL I2SRDY	PLL I2SON	PLL RDY	PLL ON	Reserved				CSSON	HSEBYP	HSERDY	HSEON	HSICAL 7	HSICAL 6	HSICAL 5	HSICAL 4	HSICAL 3	HSICAL 2	HSICAL 1	HSICAL 0	HSITRIM 4	HSITRIM 3	HSITRIM 2	HSITRIM 1	HSITRIM 0	Reserved	HSIRDY	HSION
0x04	RCC_ PLLCFGR	Reserved				PLLQ 3	PLLQ 2	PLLQ 1	PLLQ 0	Reserved	PLLSRC	Reserved				PLLP 1	PLLP 0	Reserved	PLLN 8	PLLN 7	PLLN 6	PLLN 5	PLLN 4	PLLN 3	PLLN 2	PLLN 1	PLLN 0	PLLM 5	PLLM 4	PLLM 3	PLLM 2	PLLM 1	PLLM 0
0x08	RCC_CFGR	MCO2 1	MCO2 0	MCO2PRE2	MCO2PRE1	MCO2PRE0	MCO1PRE2	MCO1PRE1	MCO1PRE0	I2SSRC	MCO1 1	MCO1 0	RTCPRE 4	RTCPRE 3	RTCPRE 2	RTCPRE 1	RTCPRE 0	PPRE2 2	PPRE2 1	PPRE2 0	PPRE1 2	PPRE1 1	PPRE1 0	Reserved		HPRE 3	HPRE 2	HPRE 1	HPRE 0	SWS 1	SWS 0	SW 1	SW 0

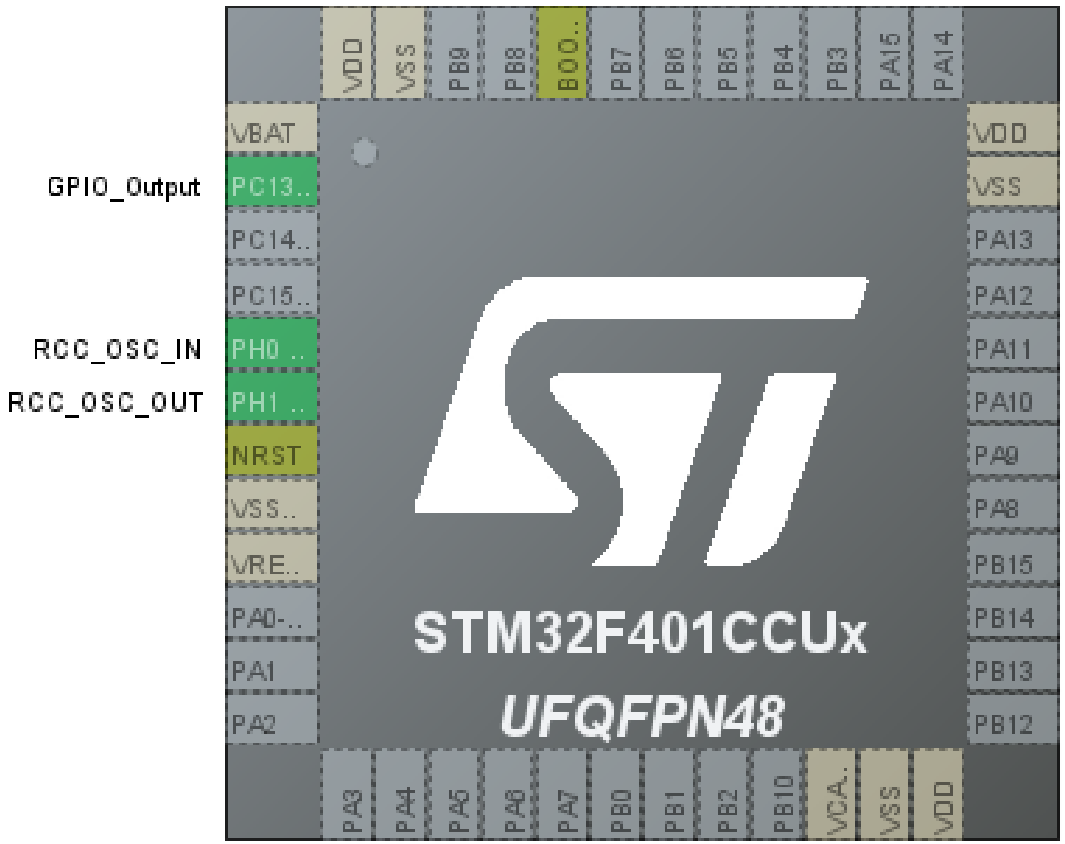
4 Implementation: LED Blinking Clock SetUp



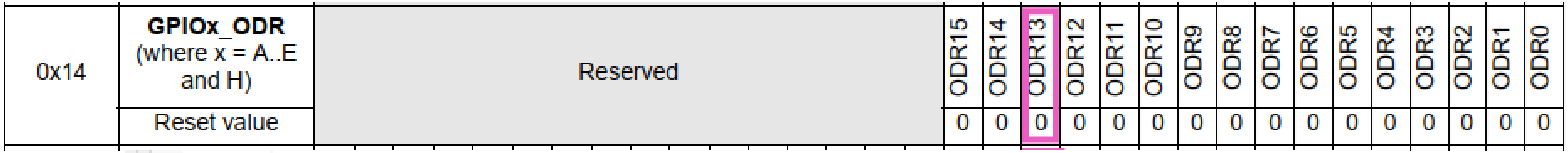
4 Implementation: LED Blinking GPIO Configuartion:(PC13)



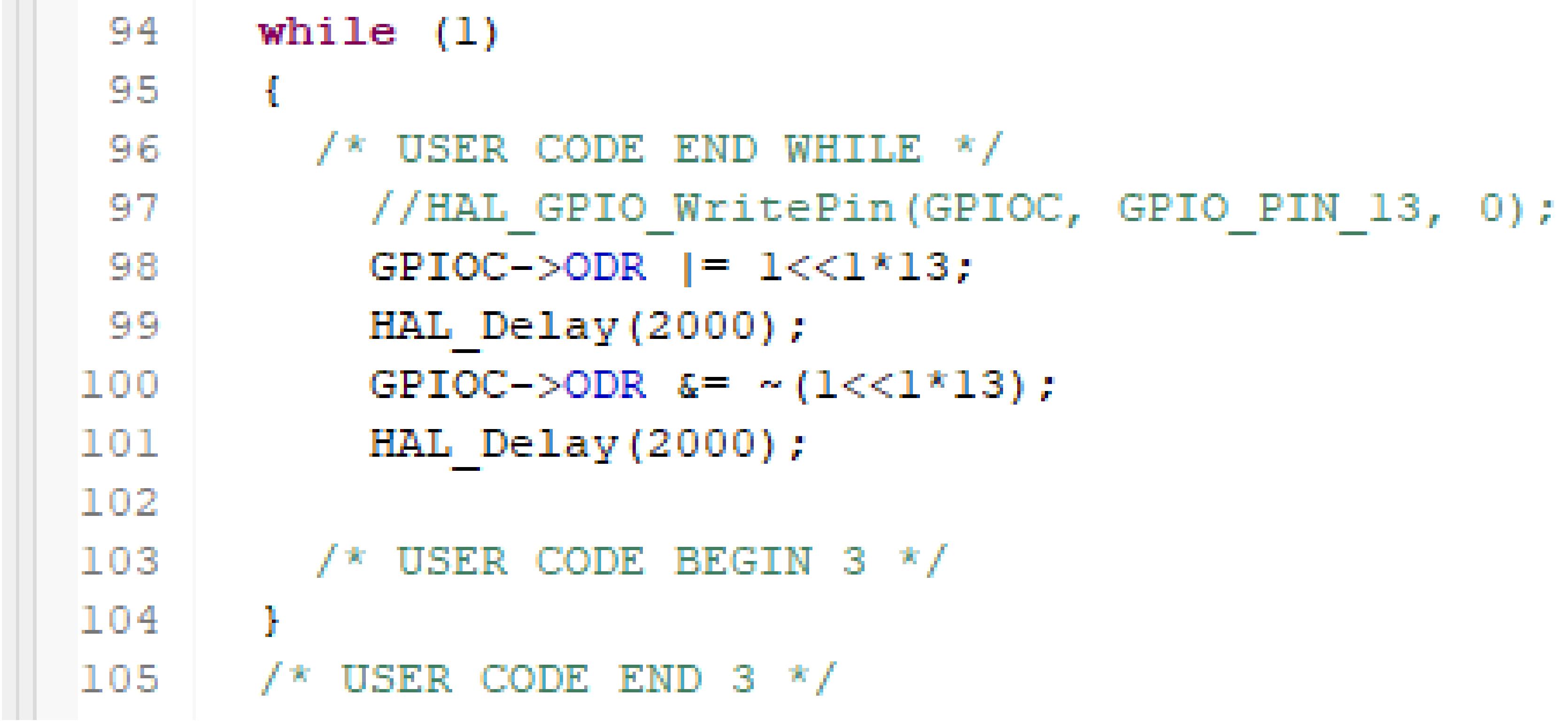
```c
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};
/* USER CODE BEGIN MX_GPIO_Init_1 */
/* USER CODE END MX_GPIO_Init_1 */

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOH_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, GPIO_PIN_RESET);

  /*Configure GPIO pin : PC13 */
  GPIO_InitStruct.Pin = GPIO_PIN_13;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);

/* USER CODE BEGIN MX_GPIO_Init_2 */
/* USER CODE END MX_GPIO_Init_2 */
}
```



4. Implementation: LED Blinking

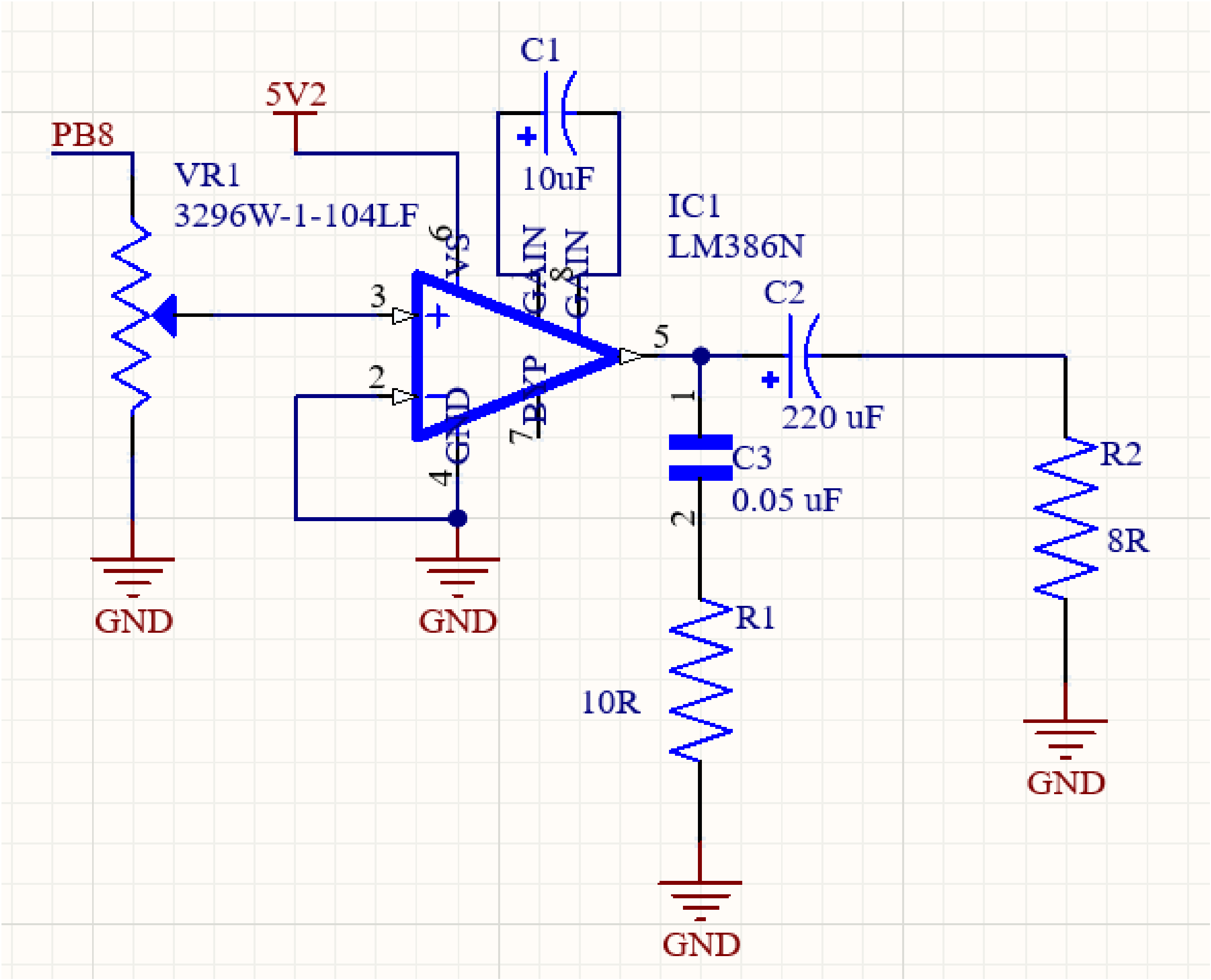


0x14	GPIOx_ODR (where x = A..E and H)	Reserved	ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

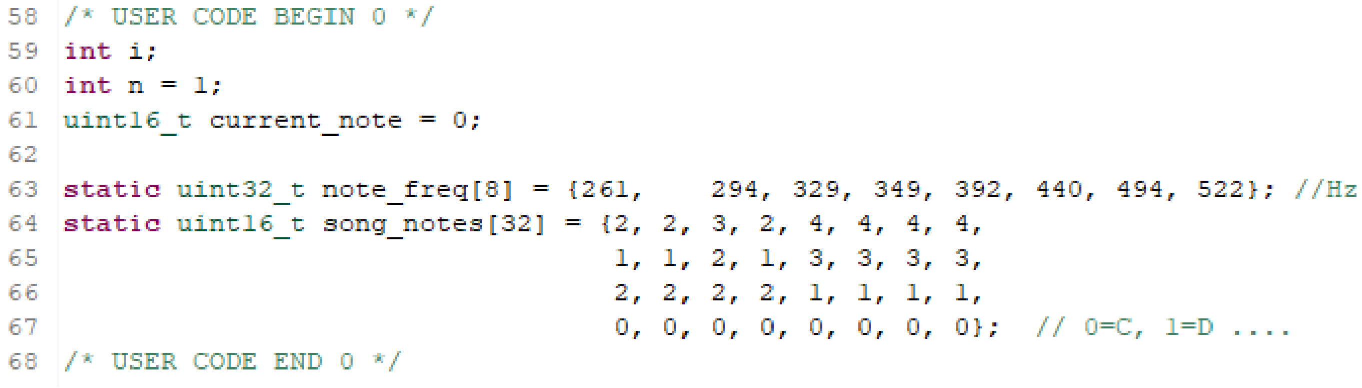


```
while (1)
{
  /* USER CODE END WHILE */
    //HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, 0);
    GPIOC->ODR |= 1<<1*13;
    HAL_Delay(2000);
    GPIOC->ODR &= ~(1<<1*13);
    HAL_Delay(2000);

  /* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
```

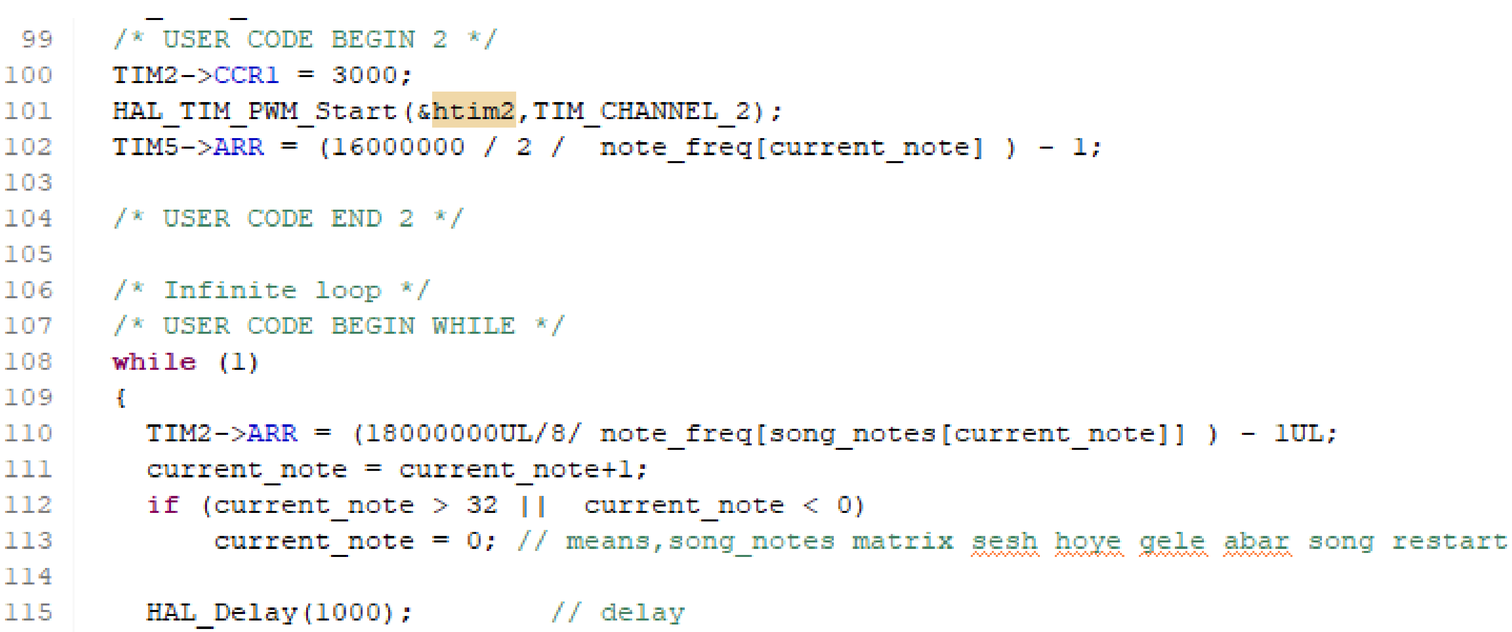
4.Implementation: Speaker



4.Implementation: Speaker Port:A3 (TIM2_CH2)



```c
/* USER CODE BEGIN 0 */
int i;
int n = 1;
uint16_t current_note = 0;

static uint32_t note_freq[8] = {261,    294, 329, 349, 392, 440, 494, 522}; //Hz
static uint16_t song_notes[32] = {2, 2, 3, 2, 4, 4, 4, 4,
                                  1, 1, 2, 1, 3, 3, 3, 3,
                                  2, 2, 2, 2, 1, 1, 1, 1,
                                  0, 0, 0, 0, 0, 0, 0, 0};  // 0=C, 1=D ....
/* USER CODE END 0 */
```



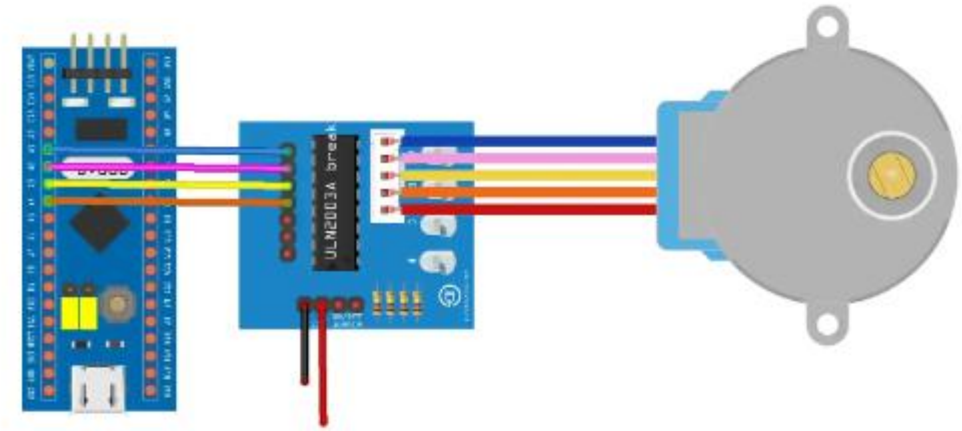
```c
  /* USER CODE BEGIN 2 */
  TIM2->CCR1 = 3000;
  HAL_TIM_PWM_Start(&htim2,TIM_CHANNEL_2);
  TIM5->ARR = (16000000 / 2 /  note_freq[current_note] ) - 1;

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    TIM2->ARR = (18000000UL/8/ note_freq[song_notes[current_note]] ) - 1UL;
    current_note = current_note+1;
    if (current_note > 32 ||  current_note < 0)
        current_note = 0; // means,song_notes matrix sesh hoye gele abar song restart

    HAL_Delay(1000);        // delay
```

4.Implementation: Demonstration

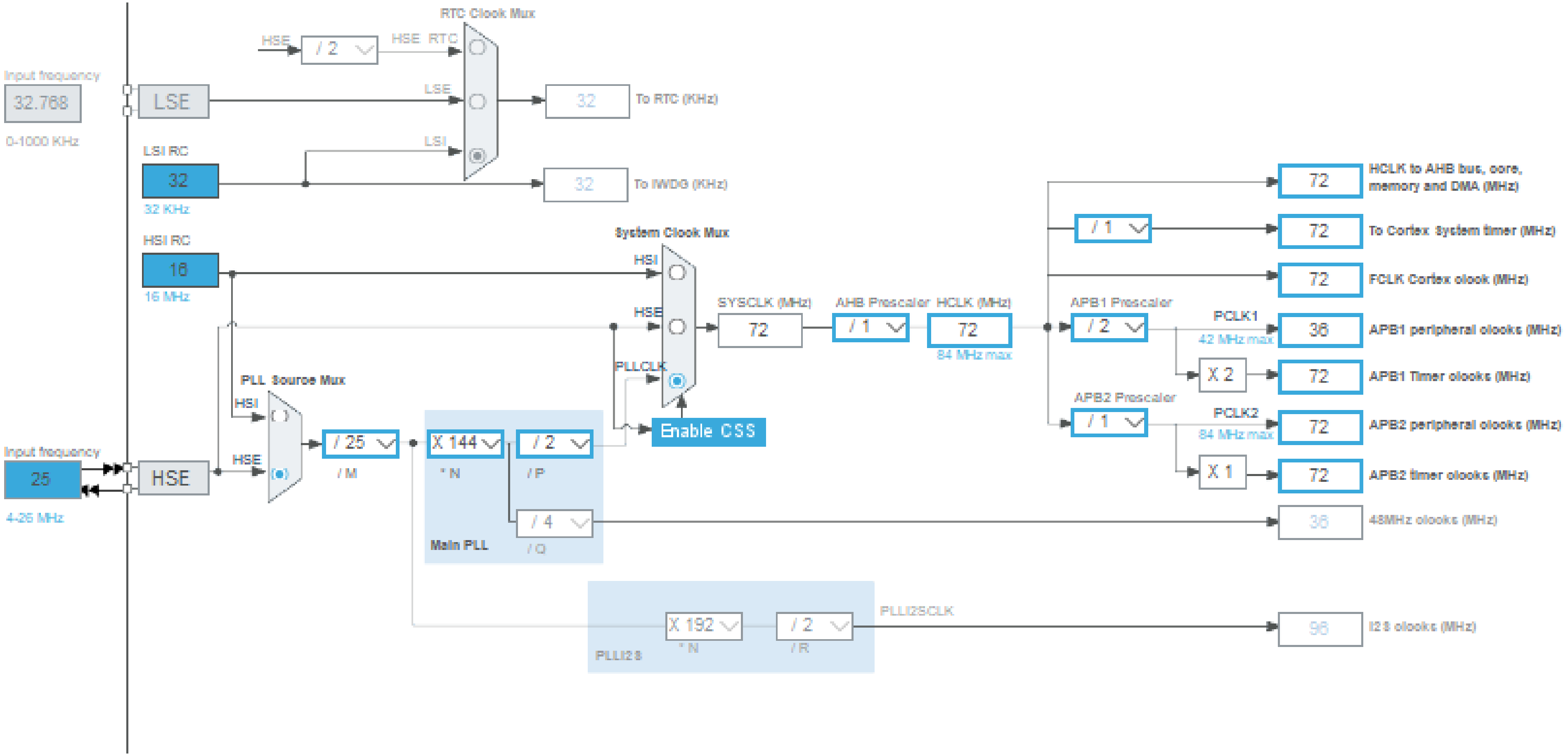
Stepper Motor

4.Implementation: Photo Gallery

Connection of stepper motor and motor driver with microprocessor



4.Implementation:Clock Configuration



RTC Clock Mux

HSE / 2 HSE_RTC

Input frequency 32.768 LSE 0-1000 KHz

LSE LSI

32 To RTC (KHz)

LSI RC 32 32 KHz

32 To IWDG (KHz)

System Clock Mux

HSI RC 16 16 MHz

HSI HSE PLLCLK

SYSCLK (MHz) 72

AHB Prescaler / 1

HCLK (MHz) 72 84 MHz max

PLL Source Mux HSI HSE

Input frequency 25 HSE 4-26 MHz

/ 25 / M

X 144 * N

/ 2 / P

/ 4 / Q

Main PLL

Enable CSS

72 HCLK to AHB bus, core, memory and DMA (MHz)

/ 1 72 To Cortex System timer (MHz)

72 FCLK Cortex clock (MHz)

APB1 Prescaler / 2 PCLK1 42 MHz max 36 APB1 peripheral clocks (MHz)

X 2 72 APB1 Timer clocks (MHz)

APB2 Prescaler / 1 PCLK2 84 MHz max 72 APB2 peripheral clocks (MHz)

X 1 72 APB2 timer clocks (MHz)

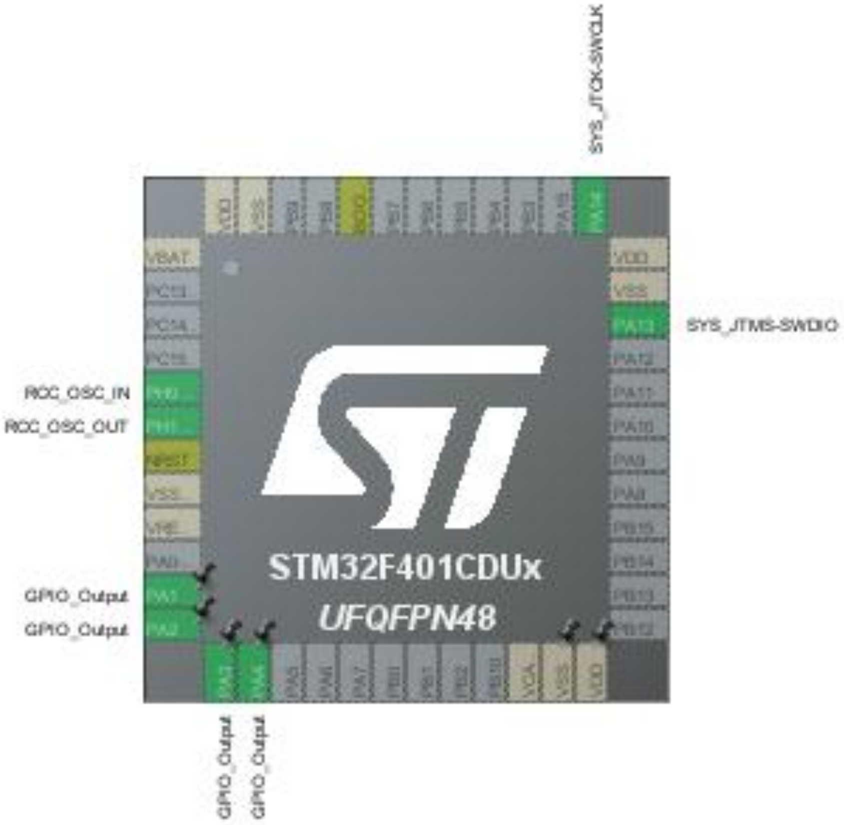
36 48MHz clocks (MHz)

PLLI2S X 192 * N / 2 / R PLLI2SCLK

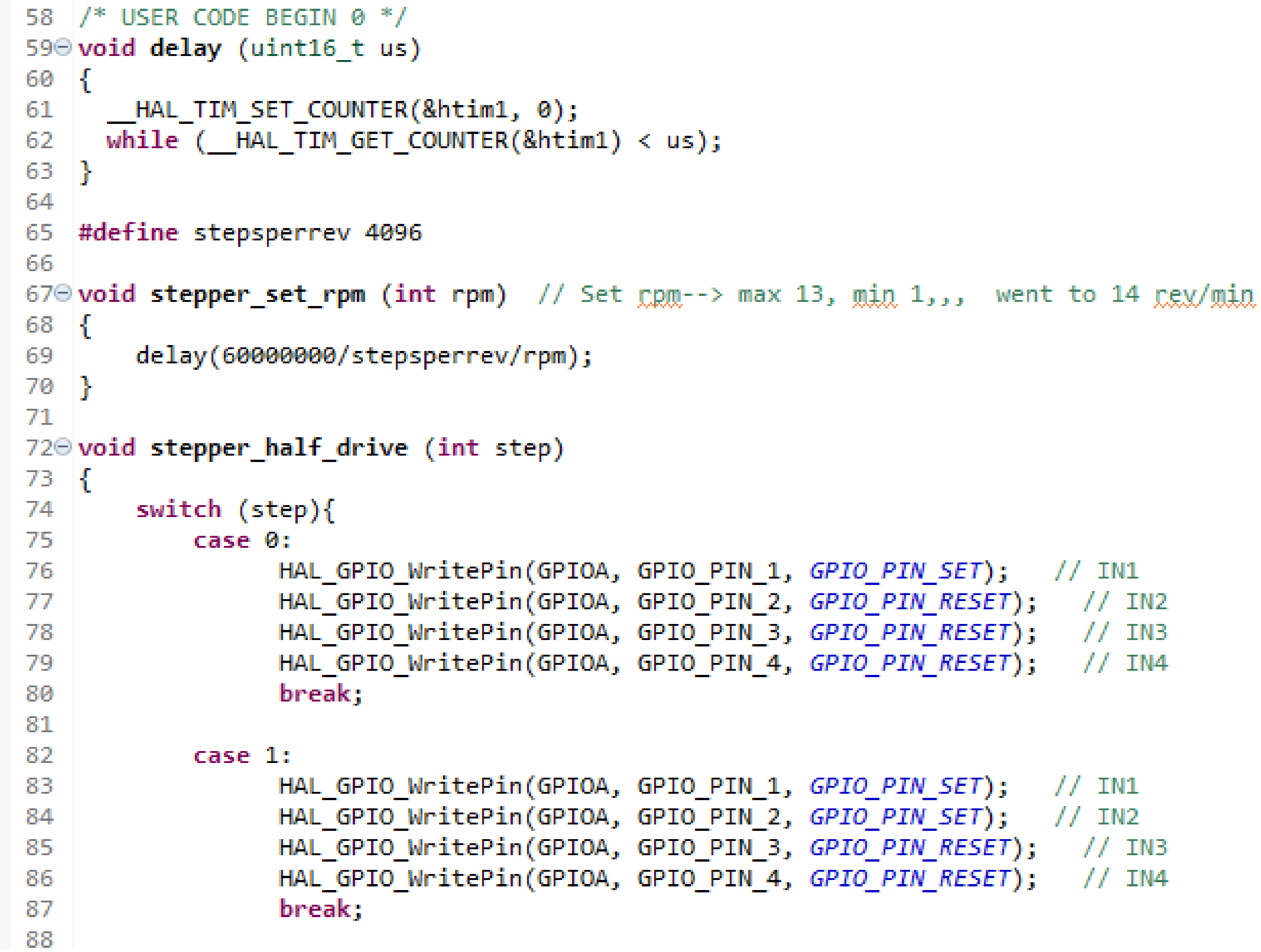
96 I2S clocks (MHz)

4.Implementation: Pinout and Configuration

GPIO pins from PA1 to PA4 are connected to 4 coils of the stepper motor

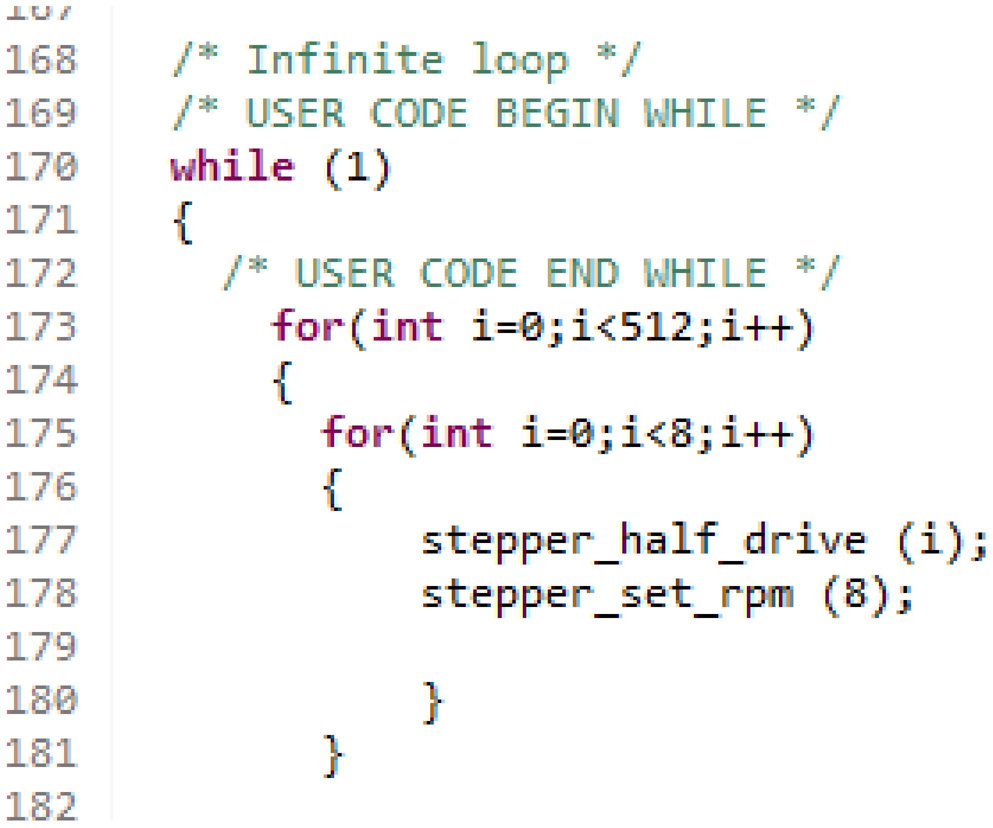


4.Implementation: Code Snipped



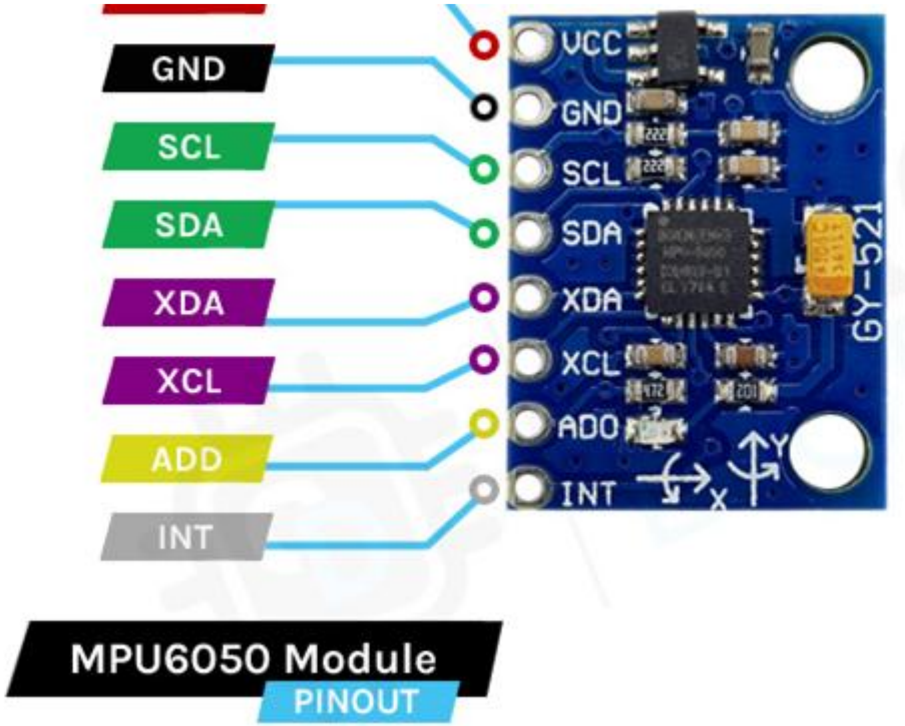
```c
/* USER CODE BEGIN 0 */
void delay (uint16_t us)
{
  __HAL_TIM_SET_COUNTER(&htim1, 0);
  while (__HAL_TIM_GET_COUNTER(&htim1) < us);
}

#define stepsperrev 4096

void stepper_set_rpm (int rpm)  // Set rpm--> max 13, min 1,,,  went to 14 rev/min
{
    delay(60000000/stepsperrev/rpm);
}

void stepper_half_drive (int step)
{
    switch (step){
        case 0:
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_SET);   // IN1
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, GPIO_PIN_RESET);   // IN2
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, GPIO_PIN_RESET);   // IN3
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_RESET);   // IN4
              break;

        case 1:
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_SET);   // IN1
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, GPIO_PIN_SET);   // IN2
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, GPIO_PIN_RESET);   // IN3
              HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_RESET);   // IN4
              break;

```

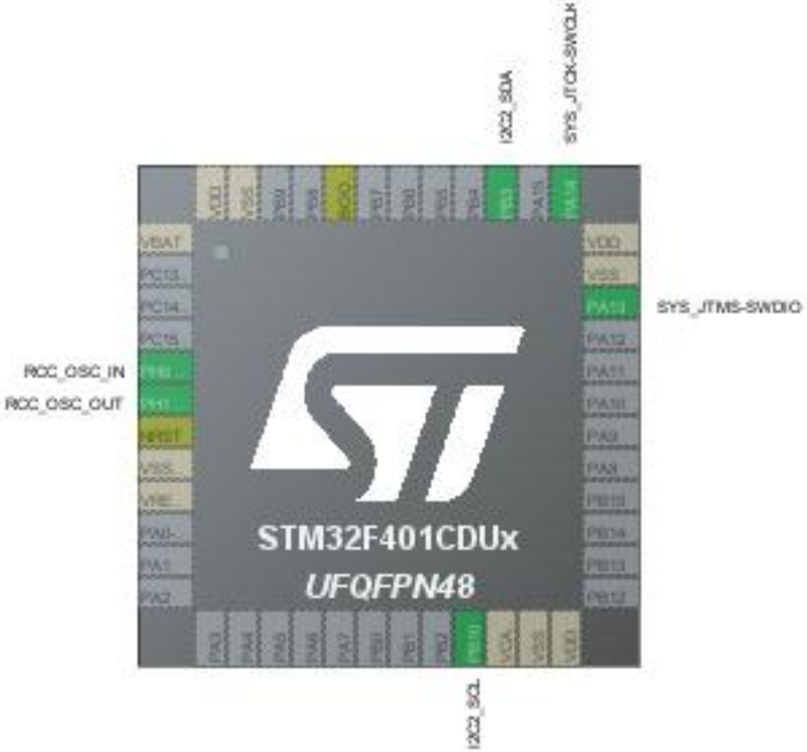
4.Implementation: Main Code



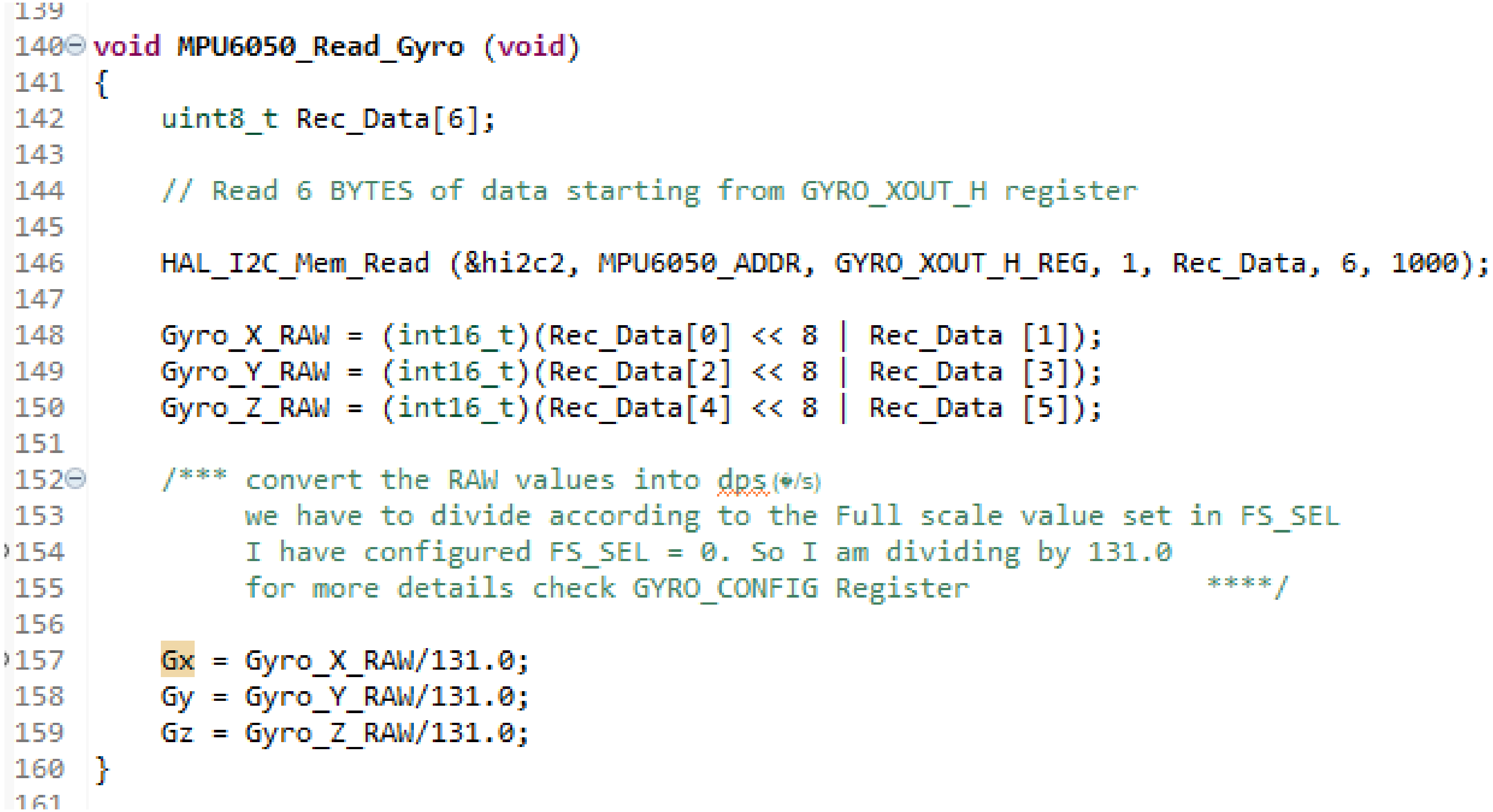
```c
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
  /* USER CODE END WHILE */
    for(int i=0;i<512;i++)
    {
      for(int i=0;i<8;i++)
      {
          stepper_half_drive (i);
          stepper_set_rpm (8);

          }
      }

```

4.Implementation: Gyroscope





4.Implementation: Code

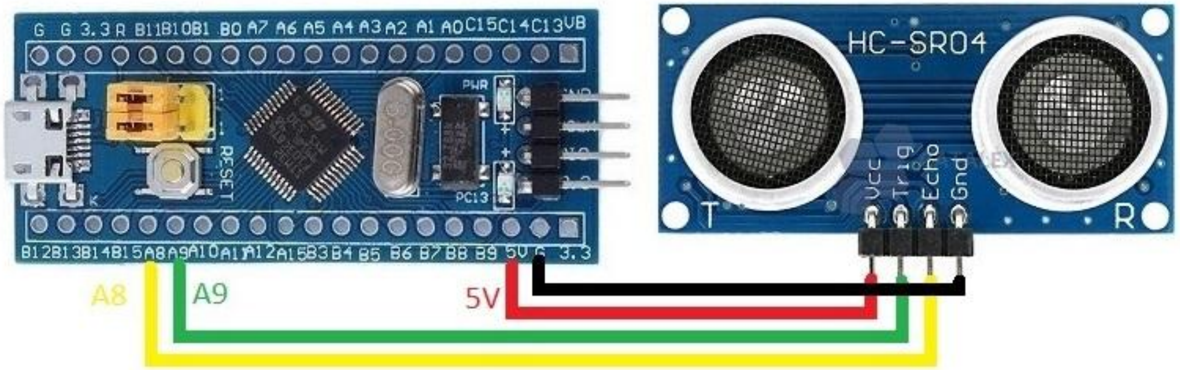


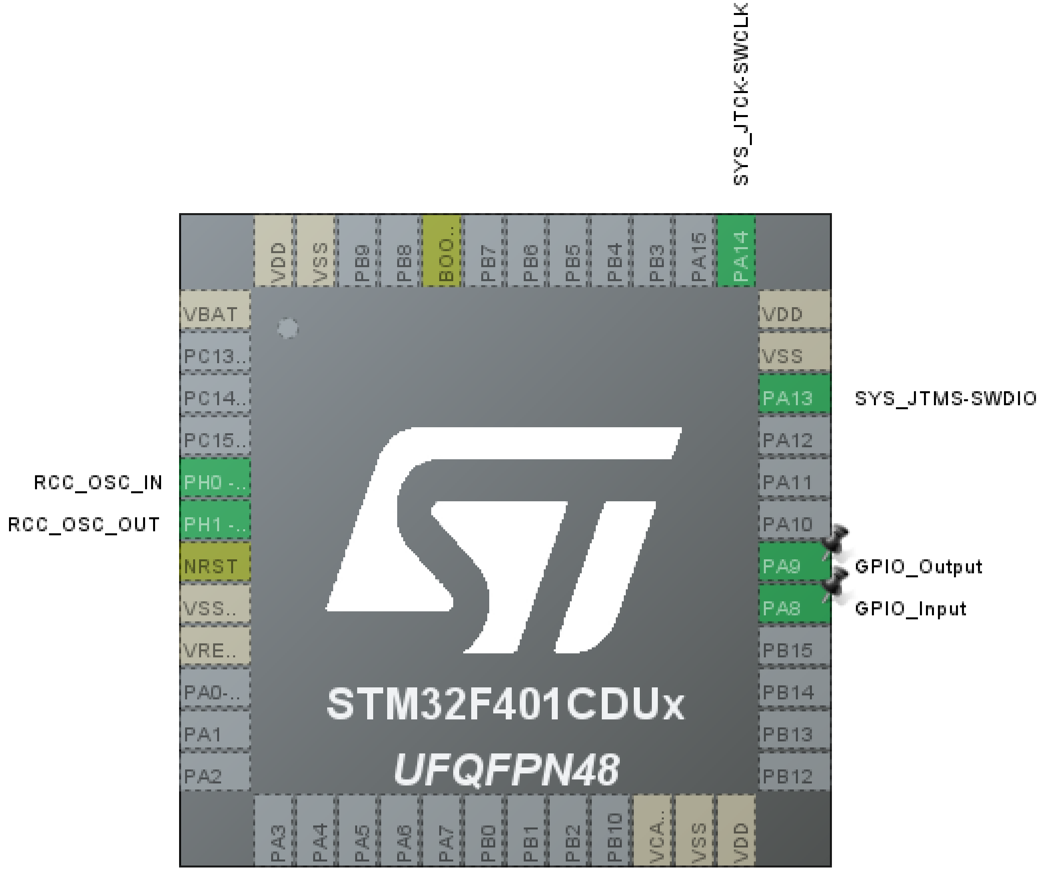
```c
void MPU6050_Read_Gyro (void)
{
    uint8_t Rec_Data[6];

    // Read 6 BYTES of data starting from GYRO_XOUT_H register

    HAL_I2C_Mem_Read (&hi2c2, MPU6050_ADDR, GYRO_XOUT_H_REG, 1, Rec_Data, 6, 1000);

    Gyro_X_RAW = (int16_t)(Rec_Data[0] << 8 | Rec_Data [1]);
    Gyro_Y_RAW = (int16_t)(Rec_Data[2] << 8 | Rec_Data [3]);
    Gyro_Z_RAW = (int16_t)(Rec_Data[4] << 8 | Rec_Data [5]);

    /*** convert the RAW values into dps(°/s)
         we have to divide according to the Full scale value set in FS_SEL
         I have configured FS_SEL = 0. So I am dividing by 131.0
         for more details check GYRO_CONFIG Register              ****/

    Gx = Gyro_X_RAW/131.0;
    Gy = Gyro_Y_RAW/131.0;
    Gz = Gyro_Z_RAW/131.0;
}
```

4.Implementation: Demonstration

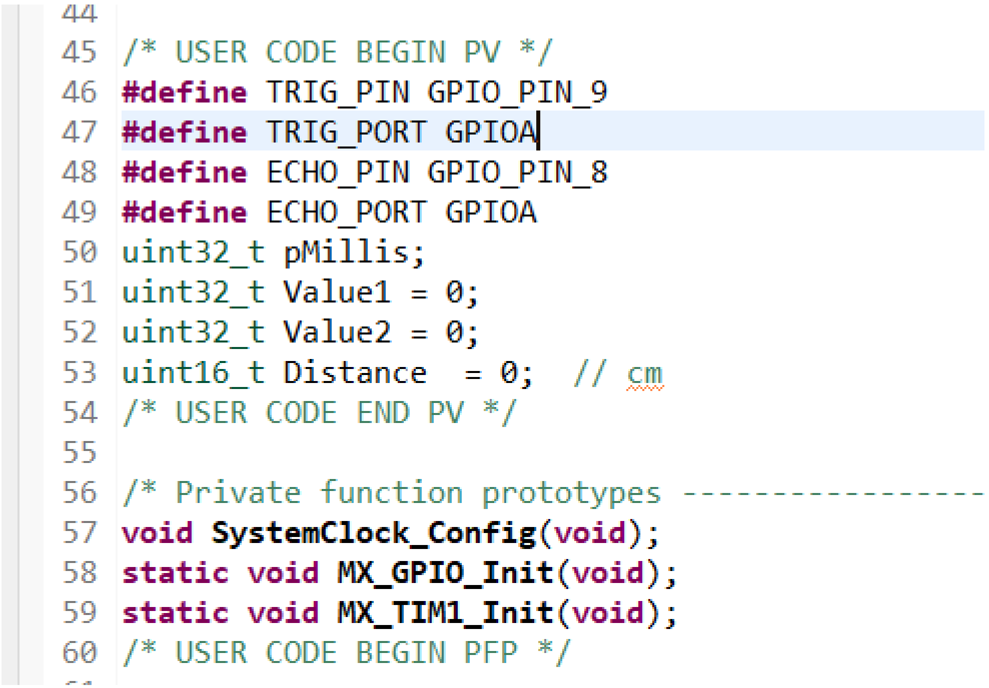
Ultrasonic Sonar Sensor

4. Implementation: Photo Gallery

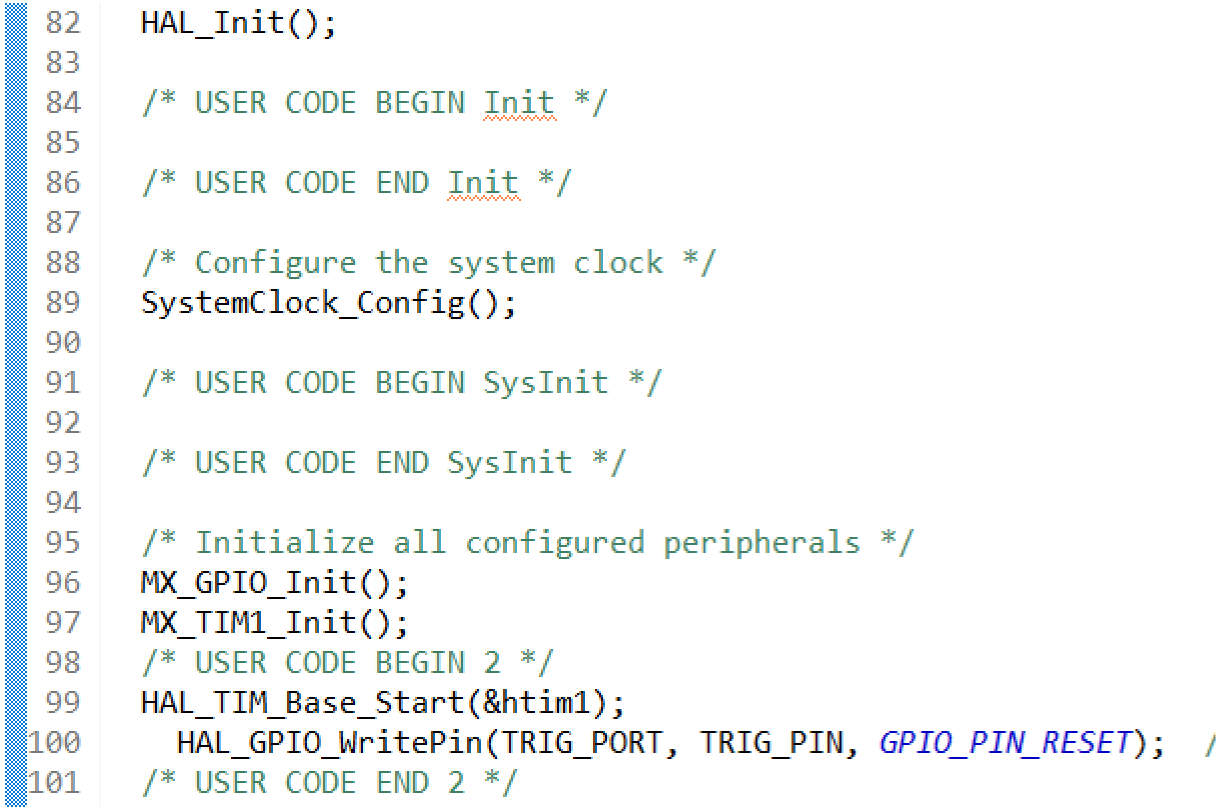

G G 3.3 R B11 B10 B1 B0 A7 A6 A5 A4 A3 A2 A1 A0 C15 C14 C13 VB
PWR
RESET
PC13
B12 B13 B14 B15 A8 A9 A10 A11 A12 A15 B3 B4 B5 B6 B7 B8 B9 5V G 3.3
HC-SR04
Vcc
Trig
Echo
Gnd
T
R
A8
A9
5V



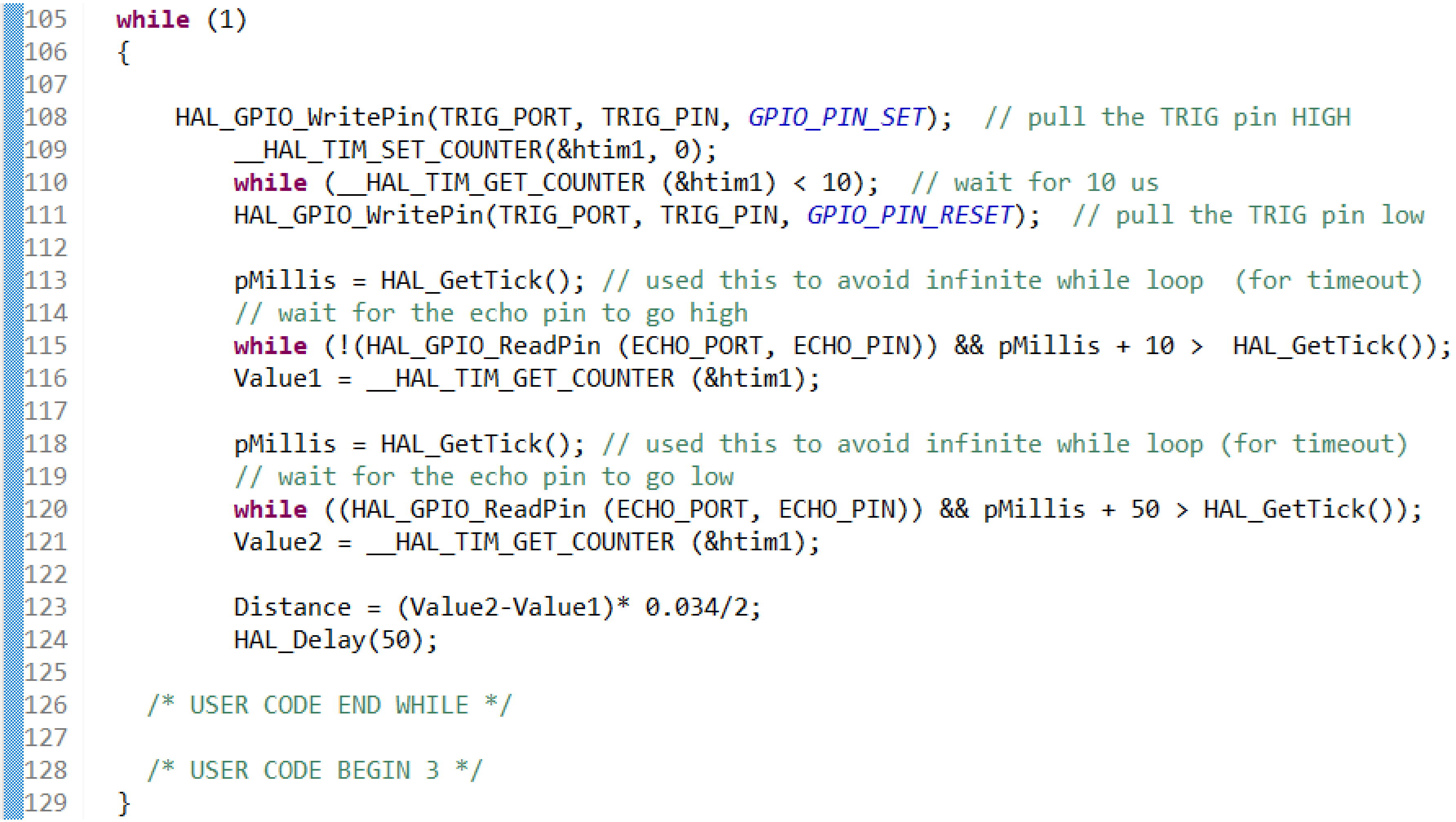
Code



```c

/* USER CODE BEGIN PV */
#define TRIG_PIN GPIO_PIN_9
#define TRIG_PORT GPIOA
#define ECHO_PIN GPIO_PIN_8
#define ECHO_PORT GPIOA
uint32_t pMillis;
uint32_t Value1 = 0;
uint32_t Value2 = 0;
uint16_t Distance  = 0;  // cm
/* USER CODE END PV */

/* Private function prototypes -----------------
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM1_Init(void);
/* USER CODE BEGIN PFP */
```



```c
HAL_Init();

/* USER CODE BEGIN Init */

/* USER CODE END Init */

/* Configure the system clock */
SystemClock_Config();

/* USER CODE BEGIN SysInit */

/* USER CODE END SysInit */

/* Initialize all configured peripherals */
MX_GPIO_Init();
MX_TIM1_Init();
/* USER CODE BEGIN 2 */
HAL_TIM_Base_Start(&htim1);
  HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_RESET);  /
/* USER CODE END 2 */
```



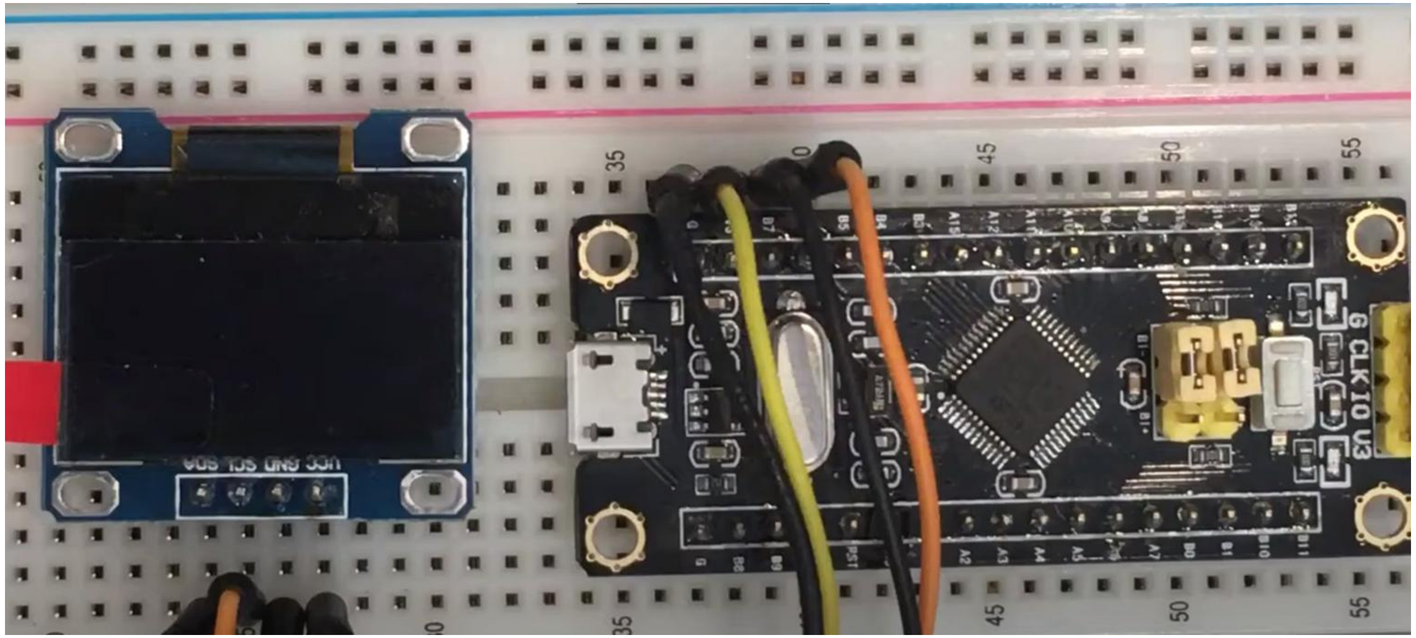
```
while (1)
{

  HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_SET);  // pull the TRIG pin HIGH
    __HAL_TIM_SET_COUNTER(&htim1, 0);
    while (__HAL_TIM_GET_COUNTER (&htim1) < 10);  // wait for 10 us
    HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_RESET);  // pull the TRIG pin low

    pMillis = HAL_GetTick(); // used this to avoid infinite while loop  (for timeout)
    // wait for the echo pin to go high
    while (!(HAL_GPIO_ReadPin (ECHO_PORT, ECHO_PIN)) && pMillis + 10 >  HAL_GetTick());
    Value1 = __HAL_TIM_GET_COUNTER (&htim1);

    pMillis = HAL_GetTick(); // used this to avoid infinite while loop (for timeout)
    // wait for the echo pin to go low
    while ((HAL_GPIO_ReadPin (ECHO_PORT, ECHO_PIN)) && pMillis + 50 > HAL_GetTick());
    Value2 = __HAL_TIM_GET_COUNTER (&htim1);

    Distance = (Value2-Value1)* 0.034/2;
    HAL_Delay(50);

 /* USER CODE END WHILE */

 /* USER CODE BEGIN 3 */
}
```

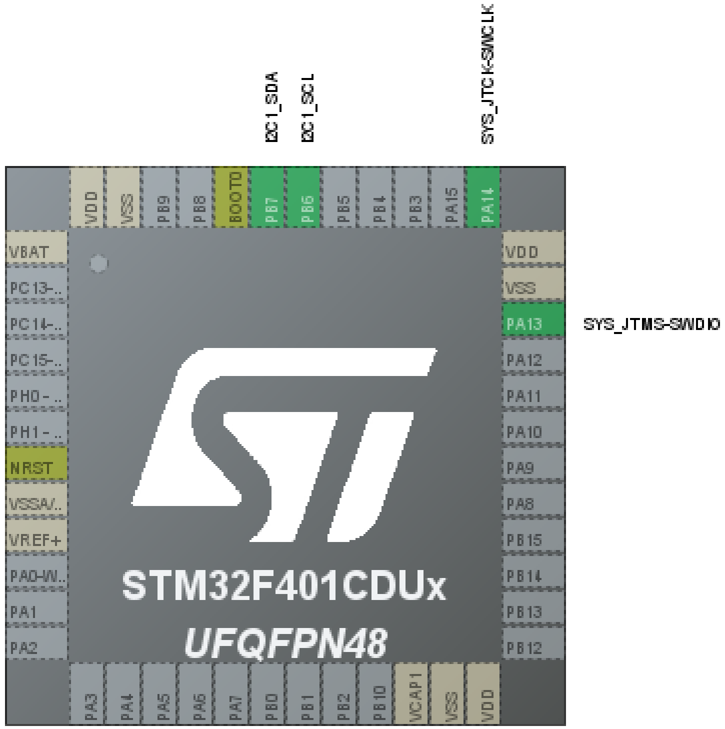
We are setting the trigger pin high for 10us and then we wait for the echo pin to get high. Measuring the difference between time gap and distance is calculated

4. Implementation: Demonstration

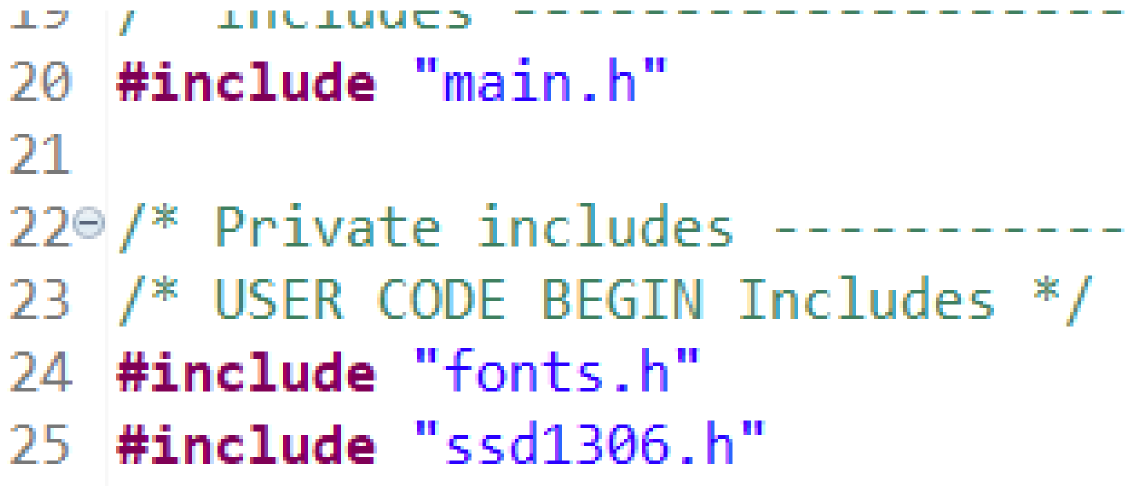
Oled Display

4.Implementation: Photo Gallery

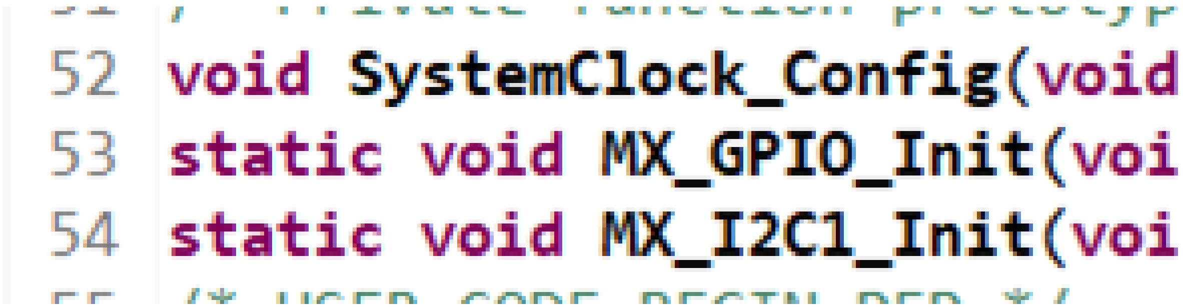




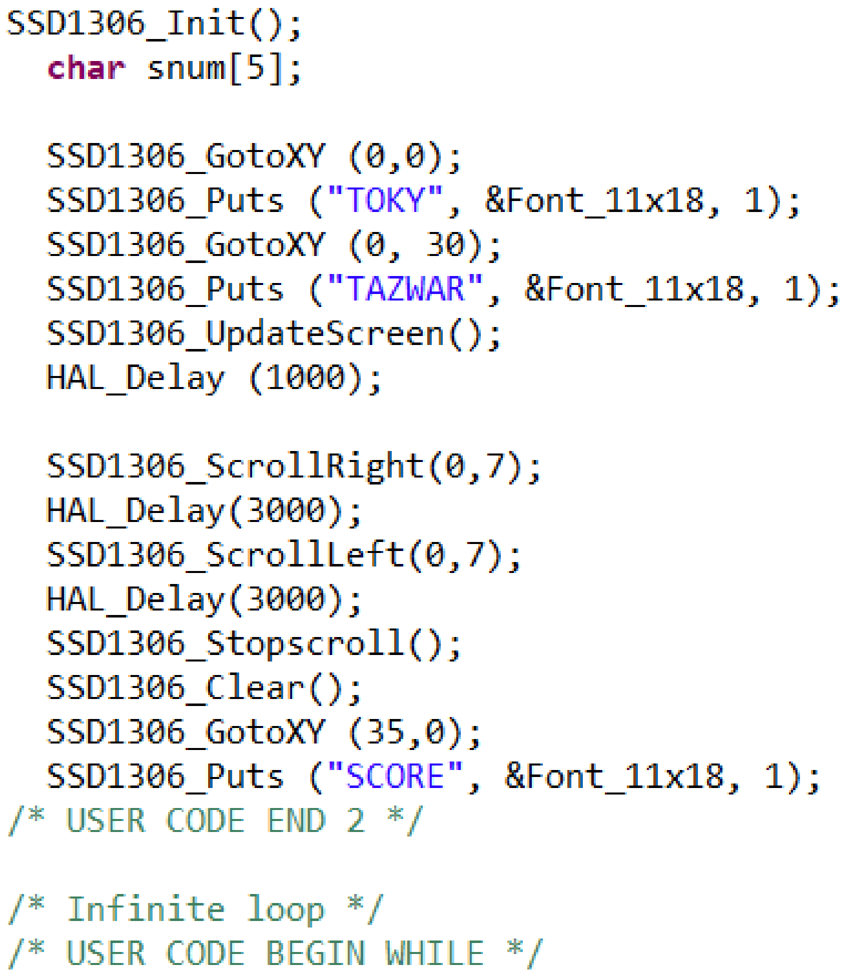
Code



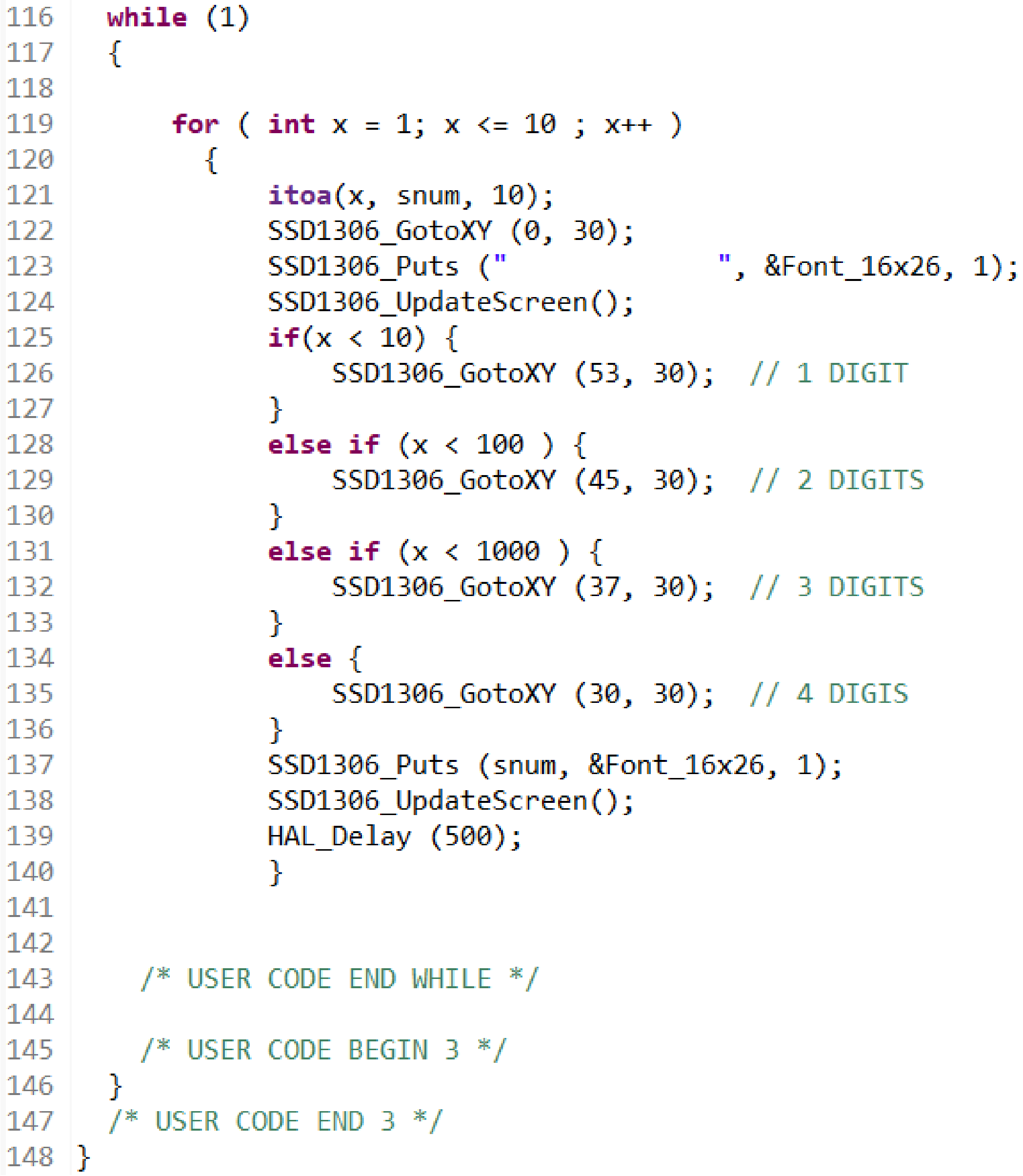
```
/* Includes ----------------------
#include "main.h"

/* Private includes ------------
/* USER CODE BEGIN Includes */
#include "fonts.h"
#include "ssd1306.h"
```



```
/* Private function prototyp
void SystemClock_Config(void
static void MX_GPIO_Init(voi
static void MX_I2C1_Init(voi
/* USER CODE BEGIN PFP */
```



```
SSD1306_Init();
 char snum[5];

 SSD1306_GotoXY (0,0);
 SSD1306_Puts ("TOKY", &Font_11x18, 1);
 SSD1306_GotoXY (0, 30);
 SSD1306_Puts ("TAZWAR", &Font_11x18, 1);
 SSD1306_UpdateScreen();
 HAL_Delay (1000);

 SSD1306_ScrollRight(0,7);
 HAL_Delay(3000);
 SSD1306_ScrollLeft(0,7);
 HAL_Delay(3000);
 SSD1306_Stopscroll();
 SSD1306_Clear();
 SSD1306_GotoXY (35,0);
 SSD1306_Puts ("SCORE", &Font_11x18, 1);
/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
```



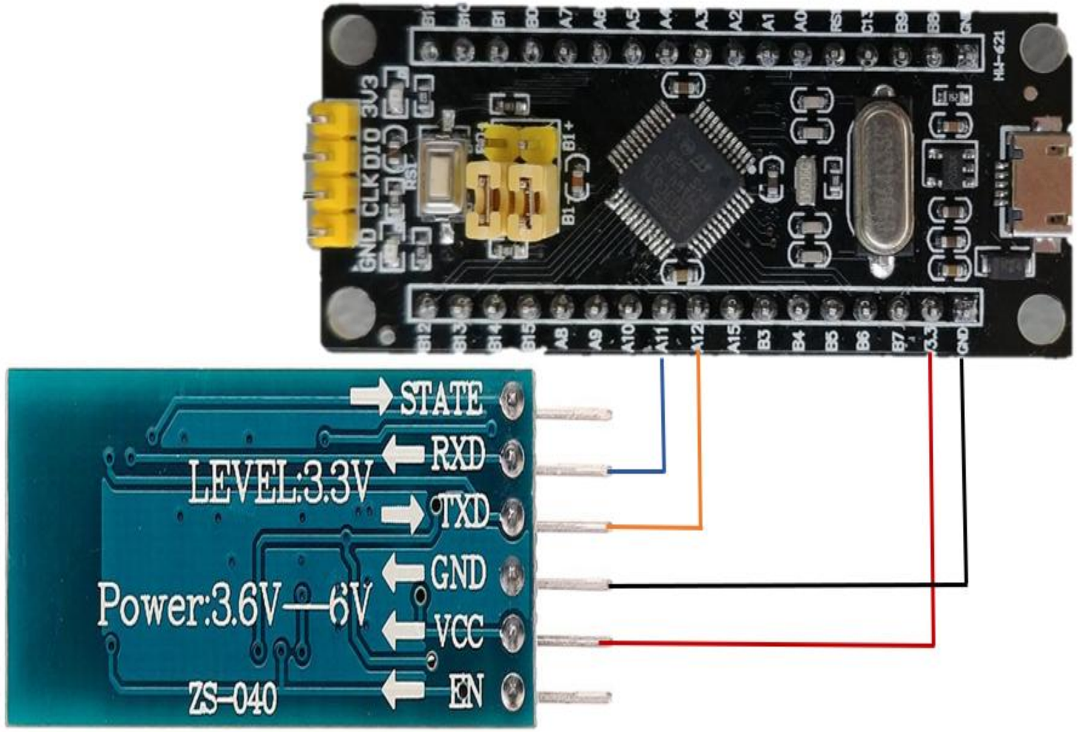
```
 while (1)
 {

   for ( int x = 1; x <= 10 ; x++ )
    {
      itoa(x, snum, 10);
      SSD1306_GotoXY (0, 30);
      SSD1306_Puts ("      ", &Font_16x26, 1);
      SSD1306_UpdateScreen();
      if(x < 10) {
        SSD1306_GotoXY (53, 30);  // 1 DIGIT
      }
      else if (x < 100 ) {
        SSD1306_GotoXY (45, 30);  // 2 DIGITS
      }
      else if (x < 1000 ) {
        SSD1306_GotoXY (37, 30);  // 3 DIGITS
      }
      else {
        SSD1306_GotoXY (30, 30);  // 4 DIGIS
      }
      SSD1306_Puts (snum, &Font_16x26, 1);
      SSD1306_UpdateScreen();
      HAL_Delay (500);
      }


  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
 }
 /* USER CODE END 3 */
}
```

4. Implementation: Demonstration

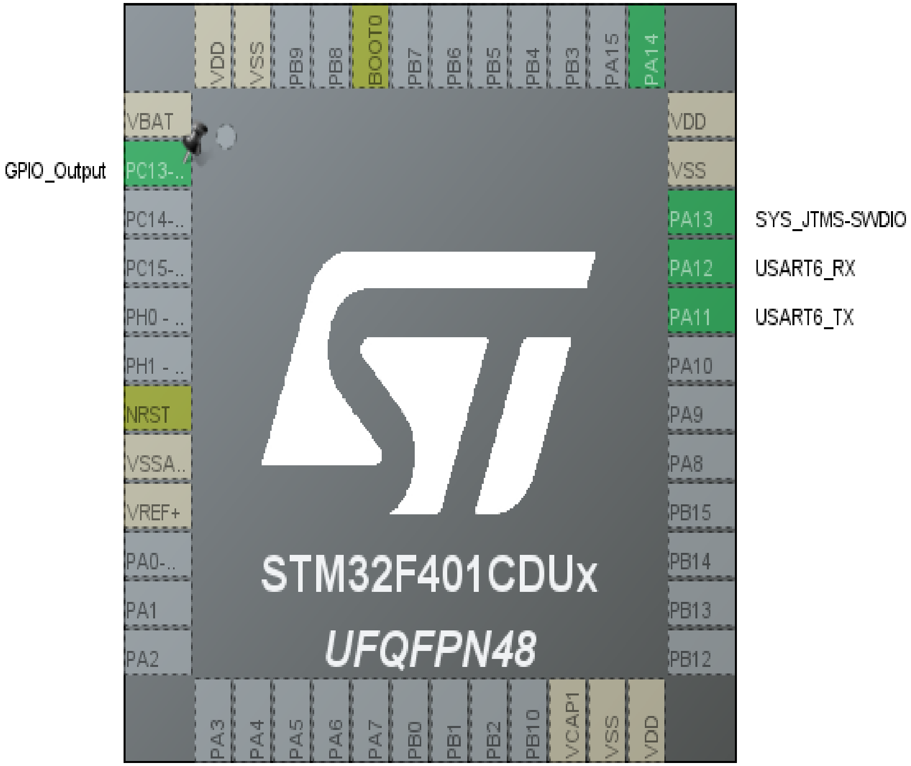
Bluetooth Module

4. Implementation: Bluetooth Module

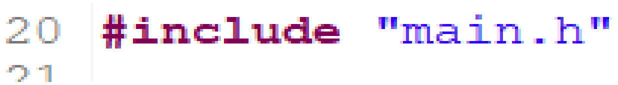
Photo Gallery



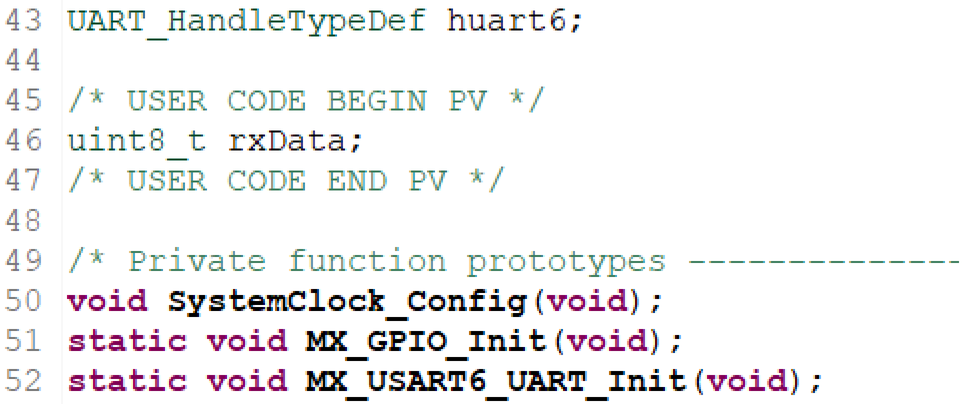
Pinout Diagram



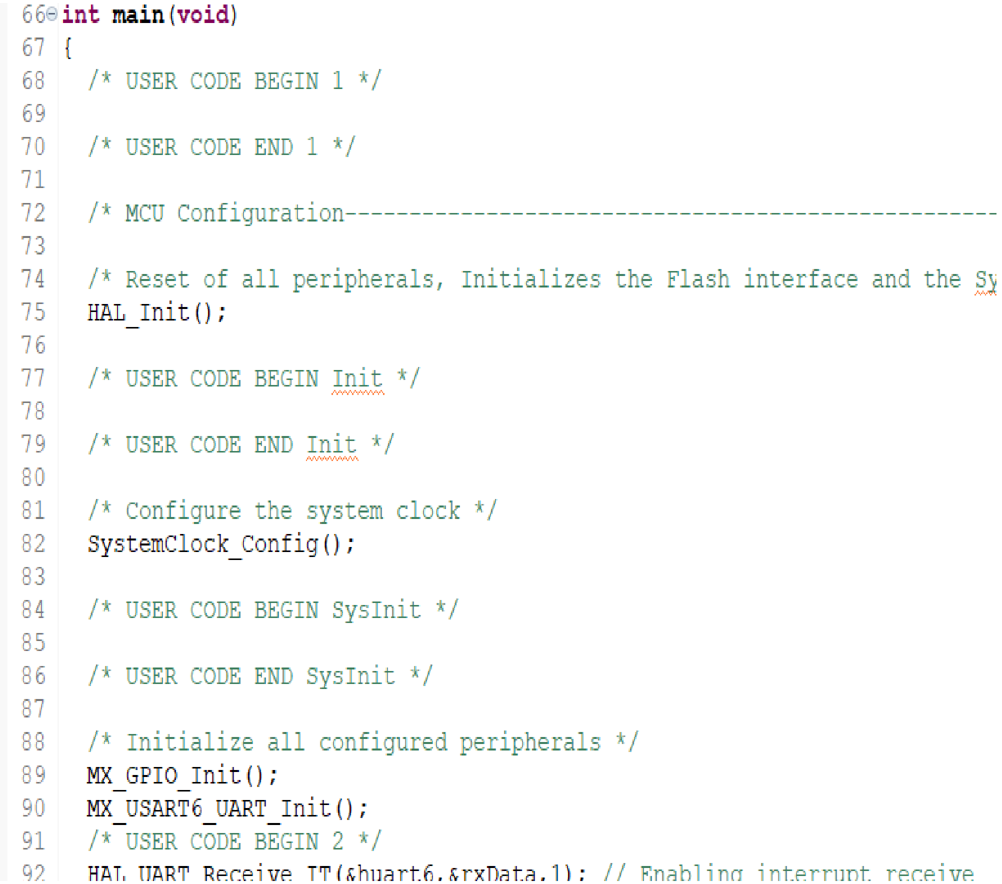
Code (Main Part)



```c
#include "main.h"
```

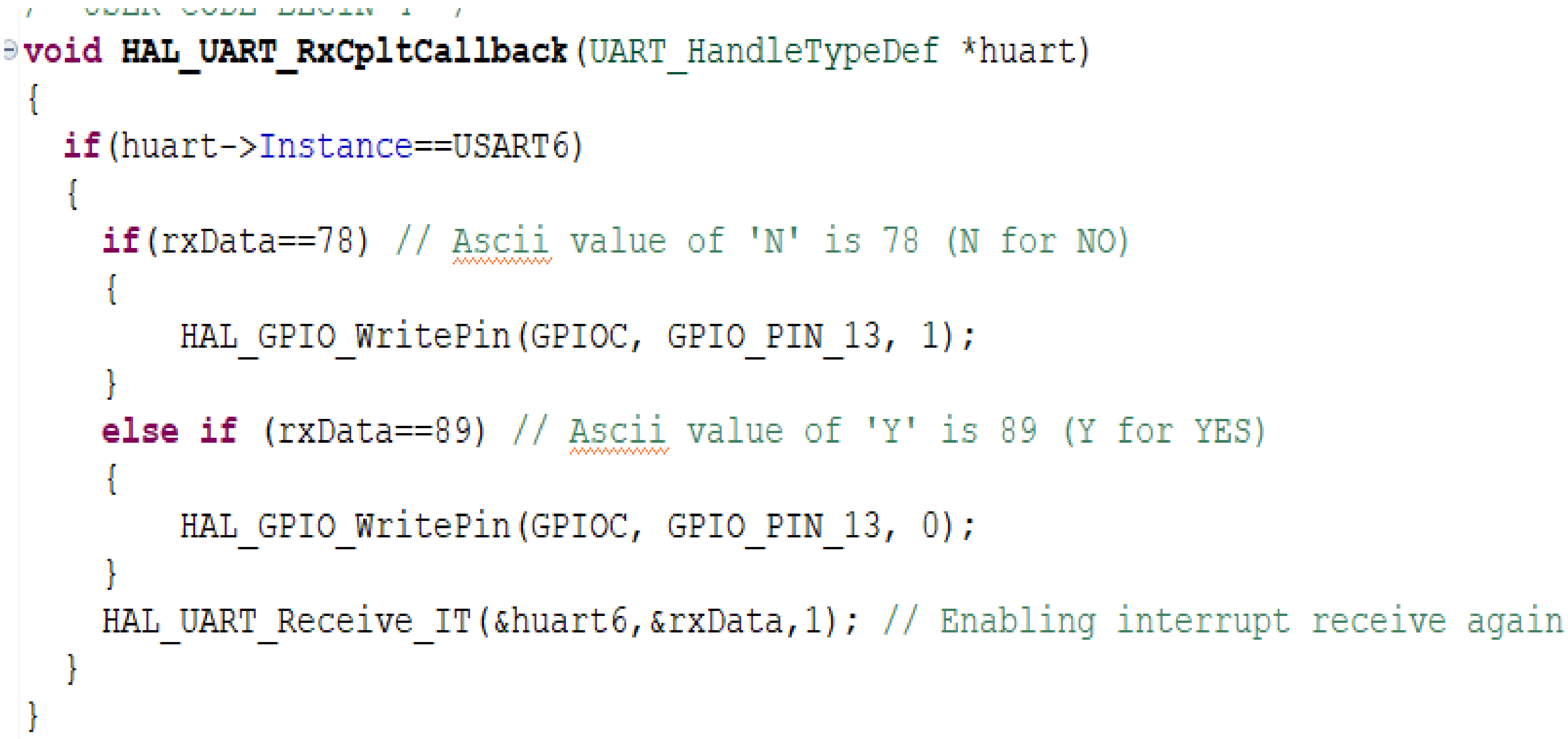


```c
UART_HandleTypeDef huart6;

/* USER CODE BEGIN PV */
uint8_t rxData;
/* USER CODE END PV */

/* Private function prototypes ---------------
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_USART6_UART_Init(void);
```



```c
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------

  /* Reset of all peripherals, Initializes the Flash interface and the Sy
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_USART6_UART_Init();
  /* USER CODE BEGIN 2 */
  HAL_UART_Receive_IT(&huart6,&rxData,1); // Enabling interrupt receive
```

Code (Main Part)



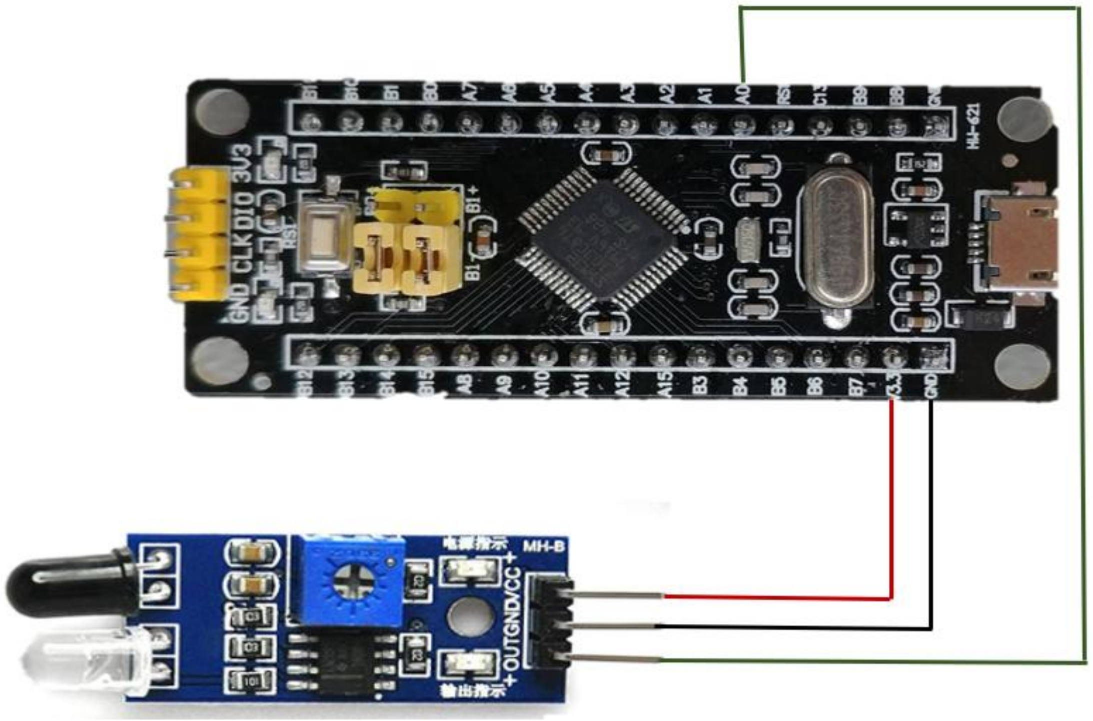
```
/* USER CODE BEGIN 4 */
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
  if(huart->Instance==USART6)
  {
    if(rxData==78) // Ascii value of 'N' is 78 (N for NO)
    {
        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, 1);
    }
    else if (rxData==89) // Ascii value of 'Y' is 89 (Y for YES)
    {
        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, 0);
    }
    HAL_UART_Receive_IT(&huart6,&rxData,1); // Enabling interrupt receive again
  }
}
```

4. Implementation: Demonstration

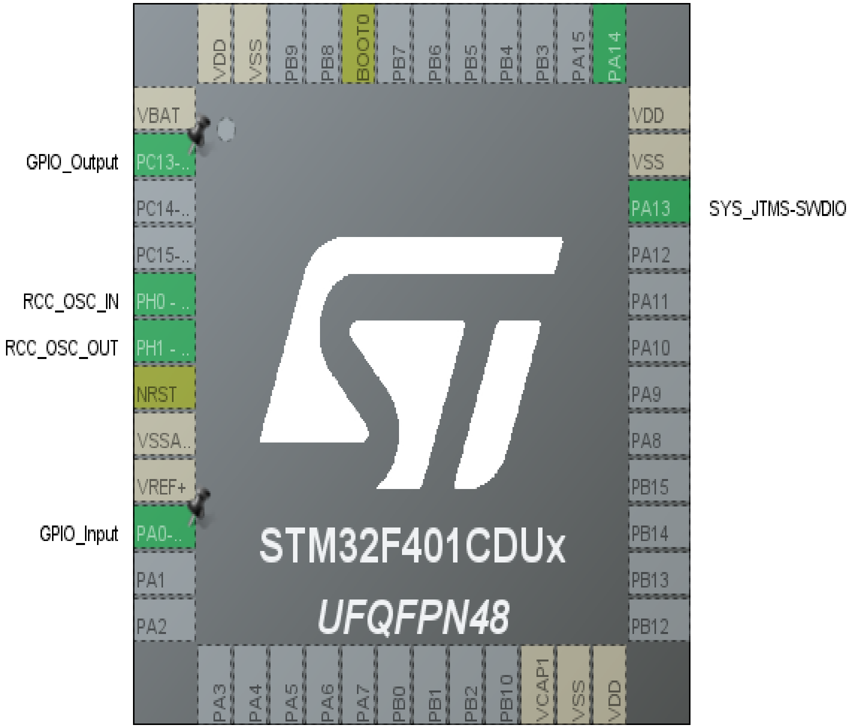
IR Sensor Module

4. Implementation: IR Sensor Module

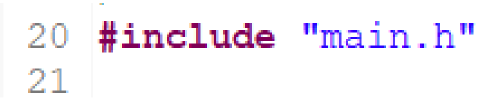
Photo Gallery



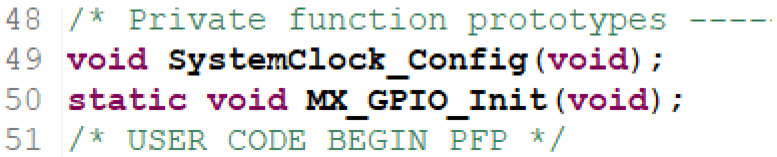
Pinout Diagram



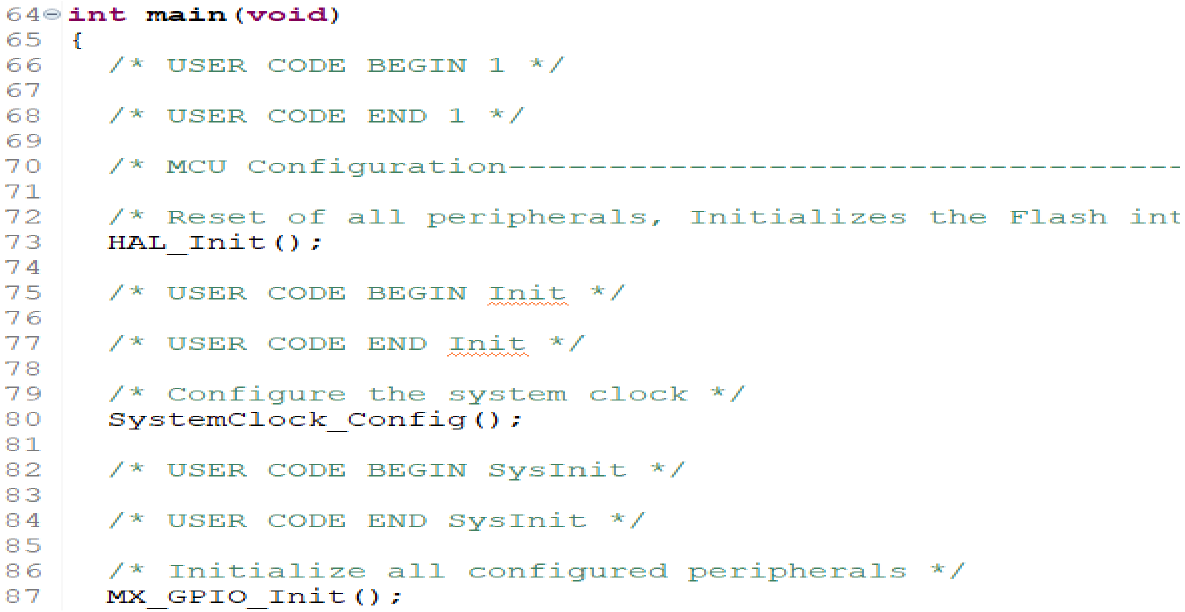
Code (Main Part)



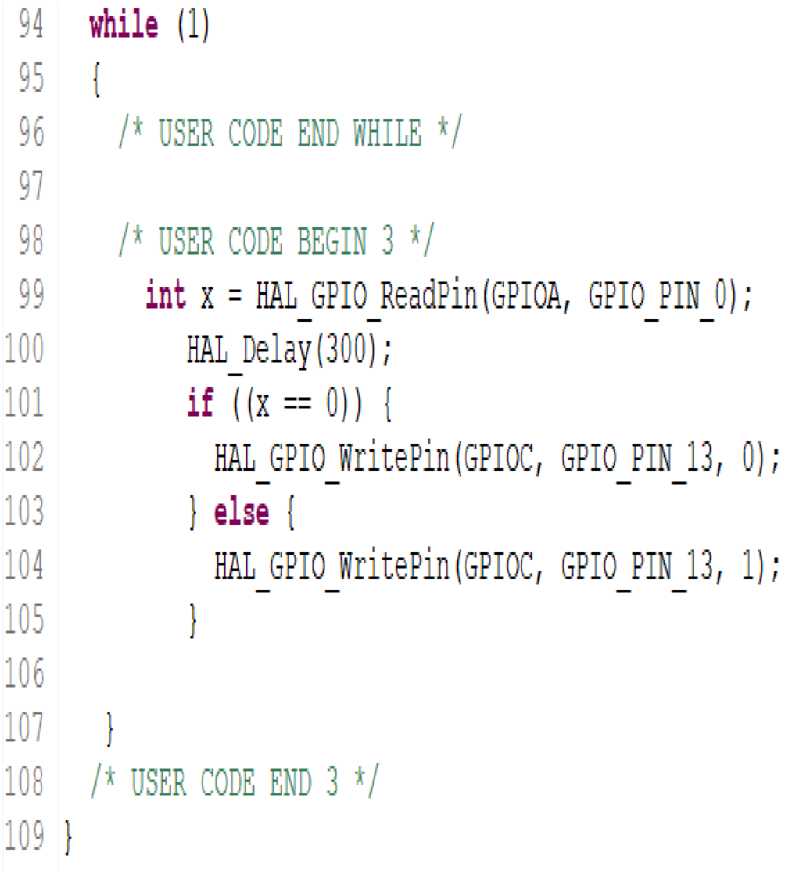
```c
#include "main.h"

```



```c
/* Private function prototypes ----
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
/* USER CODE BEGIN PFP */
```



```c
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration----------------------------------

  /* Reset of all peripherals, Initializes the Flash int
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
```



```c
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
      int x = HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0);
        HAL_Delay(300);
        if ((x == 0)) {
          HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, 0);
        } else {
          HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13, 1);
        }


  }
  /* USER CODE END 3 */
}
```

4. Implementation: External Links

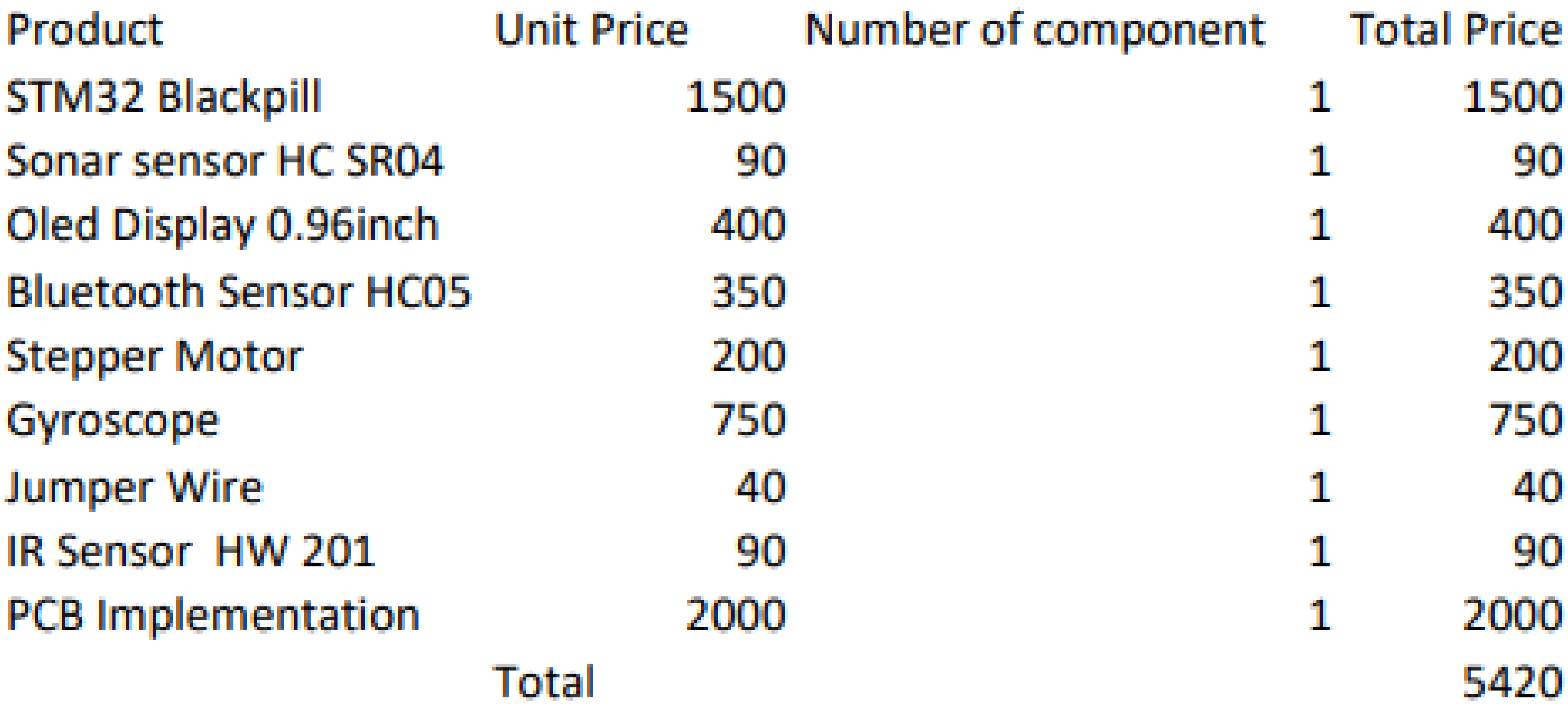
- Github Link: https://github.com/RaihanAminRana/EEE_416_Project/

- Youtube Link: https://youtu.be/1EUnkGZ2P24?si=Wa-tKn1xe4DiXcg2

5.1 Novelty

Our model is Blackpill that had been launched a year ago. So, there is almost no resource for this microcontroller, like codes, websites and videos. We used the resources available for bluepill and then wrote our code, set the clock and other library functions. Nobody in the EEE 416 lab currently working with blackpill except us and our seniors had not done this before.

5.2 Project Management and Cost Analysis



Product	Unit Price	Number of component	Total Price
STM32 Blackpill	1500	1	1500
Sonar sensor HC SR04	90	1	90
Oled Display 0.96inch	400	1	400
Bluetooth Sensor HC05	350	1	350
Stepper Motor	200	1	200
Gyroscope	750	1	750
Jumper Wire	40	1	40
IR Sensor HW 201	90	1	90
PCB Implementation	2000	1	2000
	Total		5420

5.3 Practical Considerations of the Design

Real word problems comes with numerous constraints

- Space Optimization
- Cost Reduction
- Energy Efficiency
- Mobility and Portability

5.4 Assessment of the Impact of the Project

Our compact design using the STM32 Black Pill board can be practically applied across various domains and industries.

- Smart Home Automation Systems
- Industrial Control and Monitoring Systems:
- IoT Environmental Monitoring Stations
- Wearable Health Monitoring Devices:

5.5 Evaluation of the Sustainability

Sustainability comes with different aspects

- Resource Efficiency
- Energy Efficiency
- Longevity and Durability
- End-of-Life Considerations
- Lifecycle Assessment

6.1 Individual Contribution of Each Member

- ID 1906186 - Led Blinking, Speaker, PCB Layout
- ID 1906181 - Stepper Motor, Gyroscope (I2C & SPI)
- ID 1906165 - Bluetooth, IR
- ID 1906174 - Sonar Sensor, Display (LCD & LED)

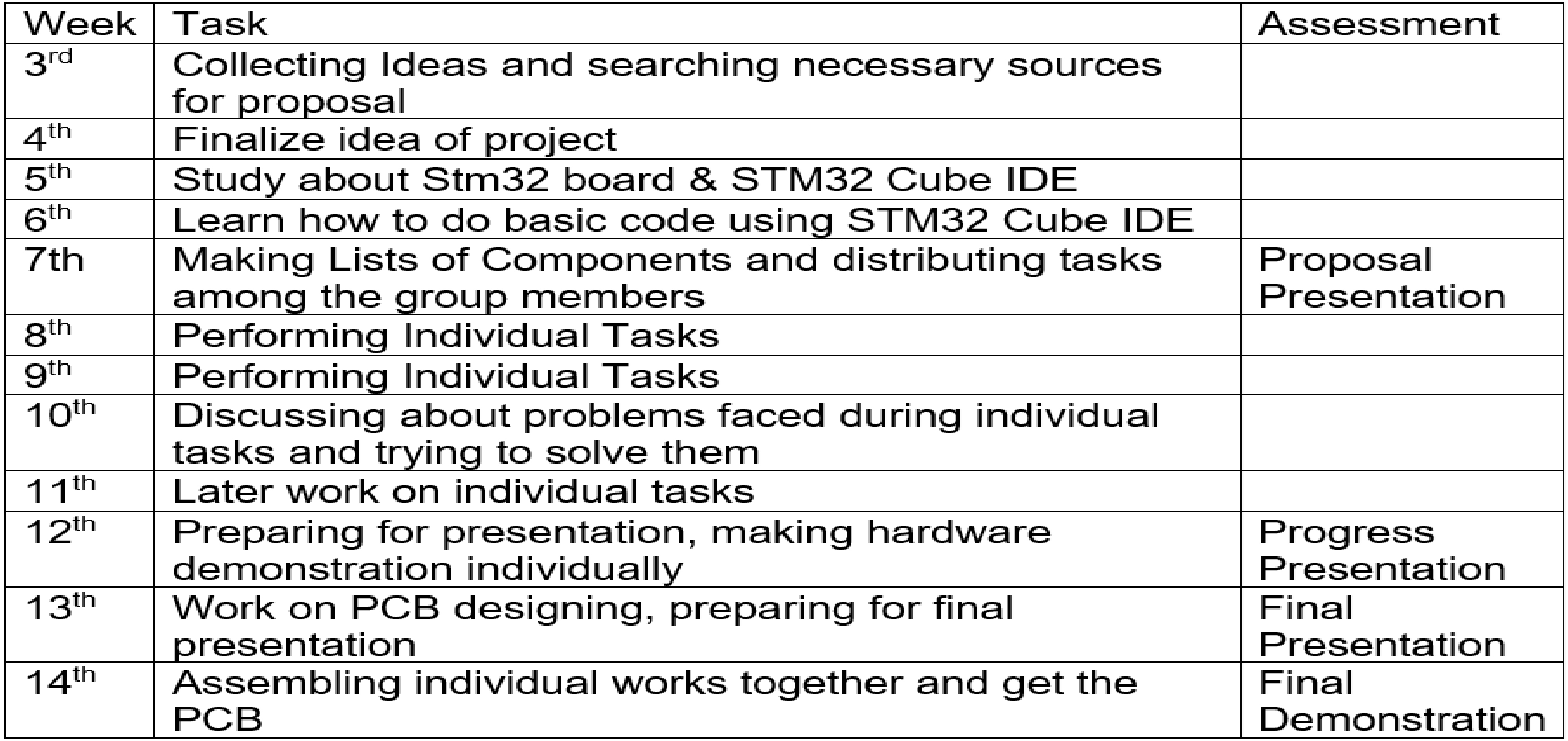
6.2 Mode of Teamwork and Diversity

- Overall planning through a meeting
- Individual task assignment
- Helping each other on technical problems
- Motivating one another
- Integrating the individual tasks and completing the project

6.3 Diversity Statement of Team

As a team, we believe that diversity fuels our innovation and drives our success. We recognize that diversity encompasses a broad spectrum of identities, including but not limited to race, ethnicity, gender, sexual orientation, age, religion, disability, and socio-economic background. We value and celebrate the unique talents, perspectives, and contributions of each team member, understanding that diversity strengthens our team and enhances our ability to innovate.

6.4 Log Book of Project Implementation



Week	Task	Assessment
3rd	Collecting Ideas and searching necessary sources for proposal	
4th	Finalize idea of project	
5th	Study about Stm32 board & STM32 Cube IDE	
6th	Learn how to do basic code using STM32 Cube IDE	
7th	Making Lists of Components and distributing tasks among the group members	Proposal Presentation
8th	Performing Individual Tasks	
9th	Performing Individual Tasks	
10th	Discussing about problems faced during individual tasks and trying to solve them	
11th	Later work on individual tasks	
12th	Preparing for presentation, making hardware demonstration individually	Progress Presentation
13th	Work on PCB designing, preparing for final presentation	Final Presentation
14th	Assembling individual works together and get the PCB	Final Demonstration

thank you