Xi'an Jiaotong-Liverpool University
西交利物浦大学

DEPARTMENT OF MECHATRONICS AND ROBOTICS

MEC104 2023-24

Final Group Project Report

Group Number: A40

Name	Student ID	Contribution Description for ALL Group Projects	Percentage (%)	Signature
Sihang.Zhao	2253127	LED soldering, testing; smart car soldering, code writing, testing; digital clock circuit construction, code writing, testing; open project topic selection, circuit construction, soldering, code writing, testing, report writing	30	赵思航
Chenyu.lu	2252676	LED soldering, testing; smart car soldering, code writing, testing; digital clock circuit construction, code writing, testing; open project topic selection, circuit construction, soldering, code writing, testing, report writing	30	[illegible]
Yilun.Zhang	2251482	Team leader, LED soldering, testing; smart car soldering, code writing, testing; digital clock circuit construction, code writing, testing; open project topic selection, circuit construction, soldering, code writing, testing, report writing	30	张逸伦
Xinan.Yan	2252676		10	

Please record the **overall** contribution percentage of each member, including **ALL** the group works on **LED soldering, Smart Car, Digital Clock and Open Project**. Examples can be found on the slides of Open Project & Group Report.

Leave the following marking form blank.

Group Marks	
Feedback Comments	

Do not exceed this cover page!

Part I – Smart Car Racing

1.1 Abstract

A smart car typically incorporates several key functions, including IR tracking sensors, collision avoidance, fall prevention, and speed control. By integrating these features, we can enhance the car's ability to track a black line at high speeds. Additionally, we have made significant efforts and conducted numerous tests to improve speed and control. These include reducing the weight of the battery, refining the code, and experimenting with different motors to optimize performance.

1.2 Design and Experiments

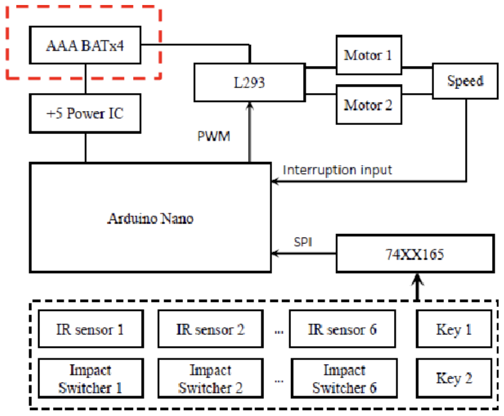
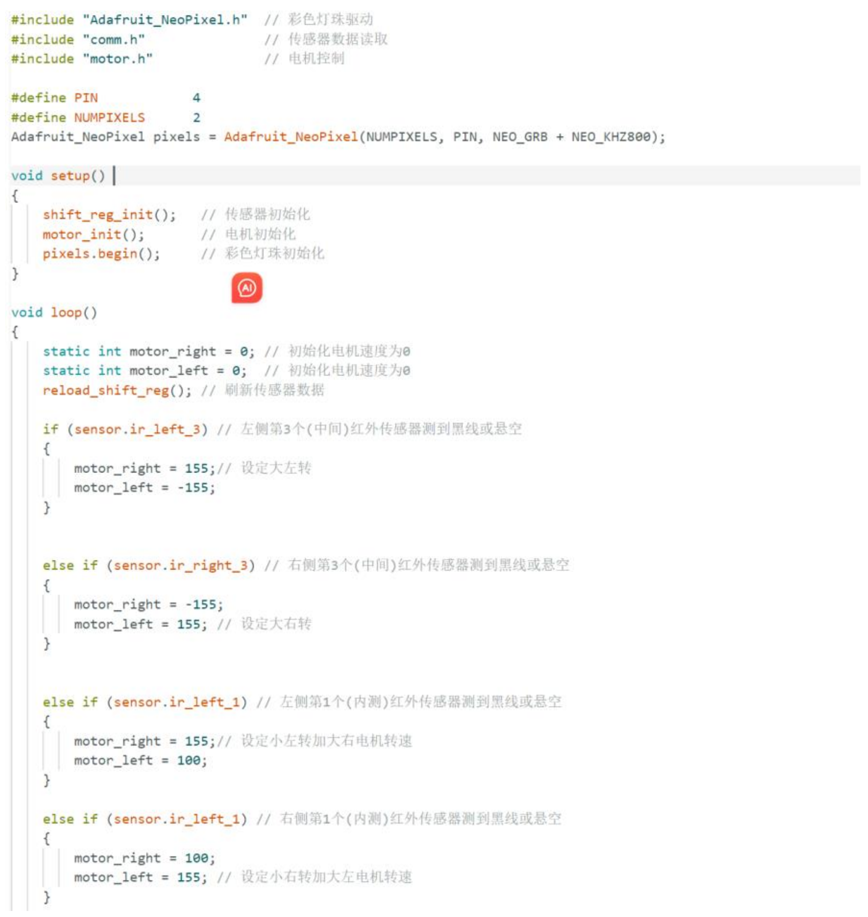


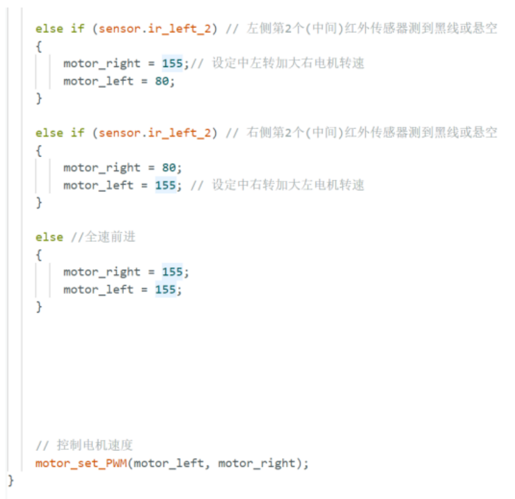
Fig. 1. Smart Car system [1]

Figure 1 illustrates the fundamental logic behind the smart car, highlighting key components such as IR sensors, the Nano IC, and motors. The IR sensors consist of an IR LED and an IR receiver, crucial for enabling the car to follow a set trajectory. Installation of these sensors, along with trajectory detection and subsequent testing and optimization, are essential steps. The MCU plays a vital role by reading signals and utilizing a shift register to convert parallel signals into serial format. For obstacle detection, impact switches identify barriers, and IR sensors detect the edges of surfaces.

1.3 Coding



```
#include "Adafruit_NeoPixel.h"  // 彩色灯珠驱动
#include "comm.h"               // 传感器数据读取
#include "motor.h"              // 电机控制

#define PIN            4
#define NUMPIXELS      2
Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

void setup()
{
    shift_reg_init();   // 传感器初始化
    motor_init();       // 电机初始化
    pixels.begin();     // 彩色灯珠初始化
}

void loop()
{
    static int motor_right = 0; // 初始化电机速度为0
    static int motor_left = 0;  // 初始化电机速度为0
    reload_shift_reg(); // 刷新传感器数据

    if (sensor.ir_left_3) // 左侧第3个(中间)红外传感器测到黑线或悬空
    {
        motor_right = 155;// 设定大左转
        motor_left = -155;
    }


    else if (sensor.ir_right_3) // 右侧第3个(中间)红外传感器测到黑线或悬空
    {
        motor_right = -155;
        motor_left = 155; // 设定大右转
    }


    else if (sensor.ir_left_1) // 左侧第1个(内测)红外传感器测到黑线或悬空
    {
        motor_right = 155;// 设定小左转加大右电机转速
        motor_left = 100;
    }

    else if (sensor.ir_left_1) // 右侧第1个(内测)红外传感器测到黑线或悬空
    {
        motor_right = 100;
        motor_left = 155; // 设定小右转加大左电机转速
    }
```



```
    else if (sensor.ir_left_2) // 左侧第2个(中间)红外传感器测到黑线或悬空
    {
        motor_right = 155;// 设定中左转加大右电机转速
        motor_left = 80;
    }

    else if (sensor.ir_left_2) // 右侧第2个(中间)红外传感器测到黑线或悬空
    {
        motor_right = 80;
        motor_left = 155; // 设定中右转加大左电机转速
    }

    else //全速前进
    {
        motor_right = 155;
        motor_left = 155;
    }






    // 控制电机速度
    motor_set_PWM(motor_left, motor_right);
}
```

This Arduino-based program controls a car by managing its motor movements for activities like line following or obstacle avoidance.

1. The **setup()** function initializes serial communications and motor configurations.
2. The **loop()** function continuously executes several operations:

- It updates the sensor readings by invoking the **reload_shift_reg()** function, which pulls in the current sensor data.
- It evaluates the car's position by processing these sensor readings through the **read_sensor_values()** function, mainly using infrared sensors for detection.

1.4 Conclusion

1.4.1 Problems

Due to the different angles of each straight and corner, too fast chassis speed will cause the car to lose control and run off the track. However, we found that different chassis speed will lead to different feedback from the infrared sensor in front. At this time, we decided to adjust the attitude of the car by giving different speeds to the tire at different corners to ensure that the car will not run out of the corner in the process of traveling. To ensure that the car in the fastest time to run the track

1.4.2Achievements

In the end, we succeeded in making the car finish the whole course in 32 seconds with different turning speeds. In this experiment, we learned to think and complete problems from different directions, and this experience made us better analyze and solve problems

Part II – Digital Clock

2.1 Abstract

In this section, we will design and build two-digit digital clock. It has main functions to started, stopped and reset. This lab will mainly use two 7-segment LED displays and build circuit in bread broads and Arduino. Finally, our clock could count up to a certain number.

2.2 Design and Experiments

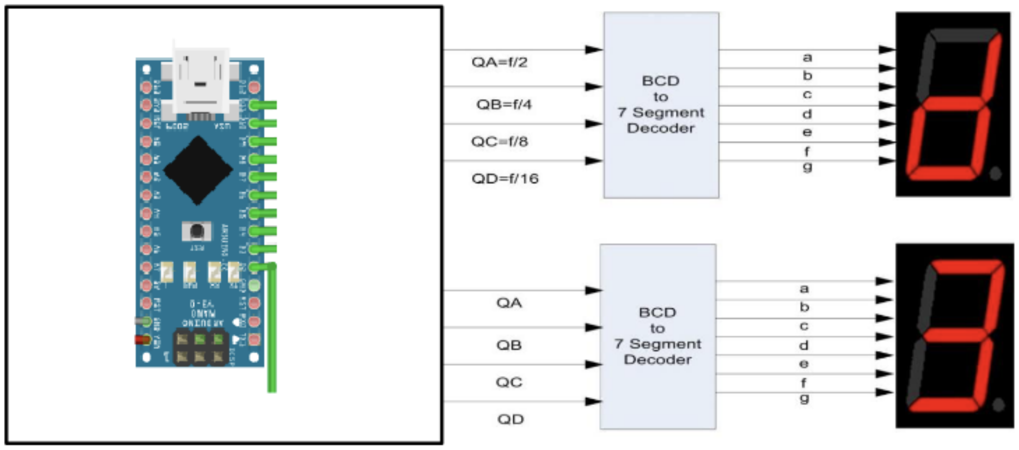


Fig. 2 Basic Structure [2]

This diagram presents the fundamental principles of a digital clock. At the heart of the system, an Arduino microcontroller orchestrates the entire circuit, issuing the necessary signals. Following this, a decoder, capable of converting Binary-Coded Decimal (BCD) into the format required for a 7-segment display, processes these signals. The 7-segment LED display then receives these processed signals and displays the output. Essentially, the BCD-to-7-segment decoder is a digital circuit that translates BCD numbers into control signals specifically for a 7-segment LED display.

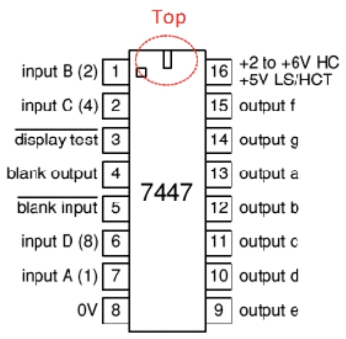


Fig.3 7447[2]

The 7-segment indicator is a digital display device composed of 7 LEDs (Light Emitting Diodes), used to display numbers and some letters.

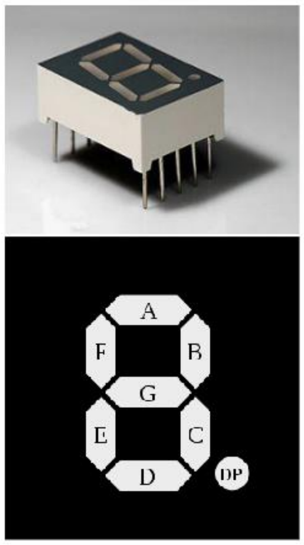
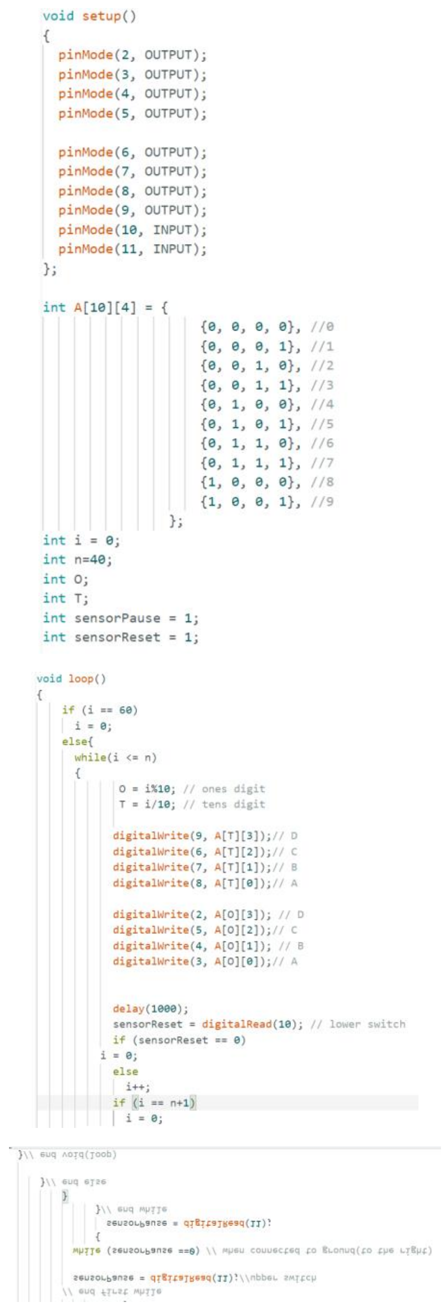


Fig.4 7-Segment Indicator [2]

This diagram illustrates the input and output relationships that are crucial for future programming efforts. To utilize the SN7447 chip for mapping inputs A-D to outputs a-g:

1. Attach a 330-ohm resistor to each output to regulate current flow.
2. Ensure correct chip orientation by aligning it according to the notch or printed dot on the chip.
3. Provide power to each integrated circuit: connect Pin 16 (Vcc) to a 5V source and Pin 8 (GND) to the ground.

Coding



```
void setup()
{
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);

  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(10, INPUT);
  pinMode(11, INPUT);
};

int A[10][4] = {
                    {0, 0, 0, 0}, //0
                    {0, 0, 0, 1}, //1
                    {0, 0, 1, 0}, //2
                    {0, 0, 1, 1}, //3
                    {0, 1, 0, 0}, //4
                    {0, 1, 0, 1}, //5
                    {0, 1, 1, 0}, //6
                    {0, 1, 1, 1}, //7
                    {1, 0, 0, 0}, //8
                    {1, 0, 0, 1}, //9
                };
int i = 0;
int n=40;
int O;
int T;
int sensorPause = 1;
int sensorReset = 1;

void loop()
{
    if (i == 60)
      i = 0;
    else{
      while(i <= n)
      {
          O = i%10; // ones digit
          T = i/10; // tens digit

         digitalWrite(9, A[T][3]);// D
         digitalWrite(6, A[T][2]);// C
         digitalWrite(7, A[T][1]);// B
         digitalWrite(8, A[T][0]);// A

         digitalWrite(2, A[O][3]); // D
         digitalWrite(5, A[O][2]);// C
         digitalWrite(4, A[O][1]); // B
         digitalWrite(3, A[O][0]);// A


         delay(1000);
         sensorReset = digitalRead(10); // lower switch
         if (sensorReset == 0)
        i = 0;
         else
           i++;
         if (i == n+1)
           i = 0;
      \\ end first while
      sensorPause = digitalRead(11);\\upper switch

      while (sensorPause ==0) \\ when connected to ground(to the right)
          {
           sensorPause = digitalRead(11);
          }\\ end while
      }
    }\\ end else
}\\ end void(loop)
```

1. Initialization Section:

- The `inputs` array contains the pin numbers used to connect the digital outputs. Four pins are used, labeled as A, B, C, and D.
- The `BCD` array defines the BCD representation of numbers from 0 to 19. Each number is represented by an array of four elements corresponding to the states (high or low) of the four digital output pins.

2. Setup:

- In the `setup()` function, the digital pins are set as outputs using the `pinMode()` function.

3. Loop:

- The **loop()** function serves as the primary loop, executing several actions repeatedly:
- A static variable named **num** tracks the current number in the sequence.
- Inside a **for** loop, the function converts the current value of **num** into its Binary-Coded Decimal (BCD) representation. It then outputs this value to the digital output pins via the **digitalWrite()** function.
- After outputting the number, **num** is incremented. The modulus operator **%** is used to ensure that **num** resets to 0 after reaching 19, allowing the sequence to cycle from 0 to 19 continuously.
- The **delay()** function pauses the loop for 500 milliseconds between each number displayed, regulating the speed of the number cycling.

The purpose of this code is to cyclically display the BCD representation of numbers 0 to 19, with each number displayed for 500 milliseconds before switching to the next one.

2.3 Conclusion

2.3.1 Problems

After assembling the circuit, we noticed that the LED did not light up as expected. Initially, we examined the circuit and found no issues with its configuration. Suspecting a malfunction in the chip, we replaced it, but the problem persisted. Ultimately, we discovered that the issue was due to poor contact in the switch. Replacing the switch resolved the problem, allowing the clock to function normally.

2.3.2 Achievements

We successfully configured the clock to cycle its count from 0 to 19 and added functionality for manual start and stop control. Through this experiment, we have honed our skills in constructing circuits on a breadboard and gained a basic understanding of the operational principles of the relevant chips and electronic components.

Part III – Open Project

3.1 Abstract

An NFC device key card typically consists of a compact smart card equipped with an NFC chip, enabling it to interact with NFC-compatible access control systems. Users authenticate themselves by either touching the key card to or holding it close to an NFC card reader, which then allows them entry into secured areas or access to restricted devices. These key cards are encased in a sturdy shell, making them durable and easy to transport. Additionally, managing user permissions—such as granting or revoking access—is straightforward and user-friendly.

3.2 Introduction

Near Field Communication (NFC) is a short-range wireless technology that allows devices to exchange information over distances of just a few centimeters. Derived from RFID (Radio Frequency Identification) technology, NFC offers enhanced security and is applicable in a broader range of scenarios. It is commonly used for tasks such as making payments, controlling access, smart labeling, and transmitting data. Operating at 13.56 MHz, NFC supports three modes: reader/writer, point-to-point, and card emulation. This technology is extensively integrated into smartphones, smart cards, and labels, among other devices.

In laboratory settings where the security and management of sensitive equipment are paramount, we have developed a device that utilizes NFC technology. This device, equipped with both LED and OLED displays and a control relay system, enables secure access and efficient management of laboratory instruments. Access to operate or open the device is restricted to individuals with a specifically authorized NFC card, thereby enhancing the security and operational efficiency of the equipment.

3.3 Theory

Equipment Setup:

- Development Board and NFC Module: Incorporate an NFC-capable card reader module, connected to the master Arduino board. Develop code to read NFC card data and verify it against pre-registered information in the system.

LED and OLED Display Management:

- Utilize LED and OLED modules to showcase pertinent data.

- Connect these displays to the main control board and program them to manage their activation and the content displayed according to specific requirements.

Relay Module Integration:

- Attach the relay module to the main control board.
- Write software to toggle the relay, thereby controlling the device's power supply based on command inputs.

Equipment Management System:

- Link each registered NFC card within the system to a designated device, ensuring that only the cardholder can activate the device.
- Use relay controls to manage device power, while the LED and OLED displays provide real-time status and information.

3.4 Experiment procedure

The following is an explanation of the code function of the NFC device key card

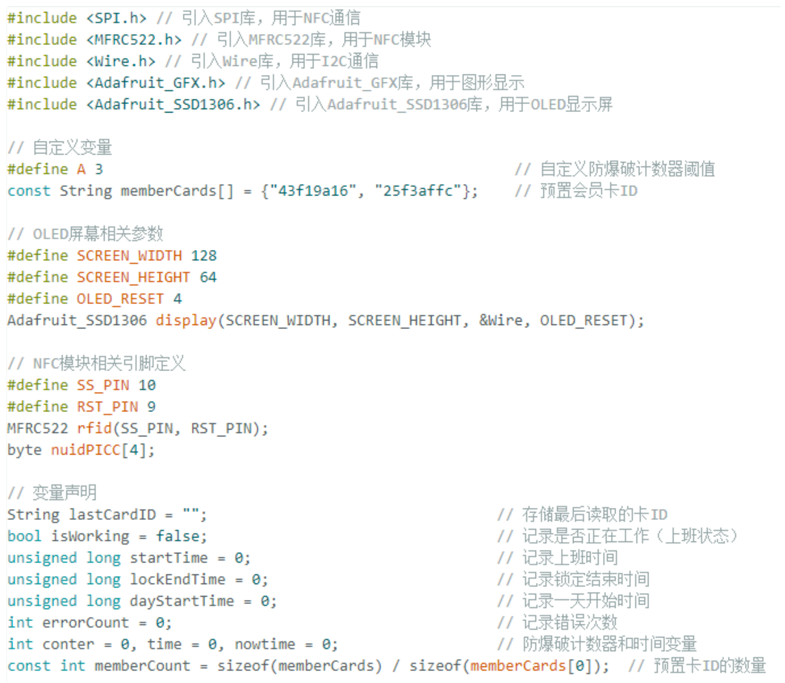
Define the width and height of the OLED display and reset the pins.

Create an instance of Adafruit_SSD1306 to control the OLED display.

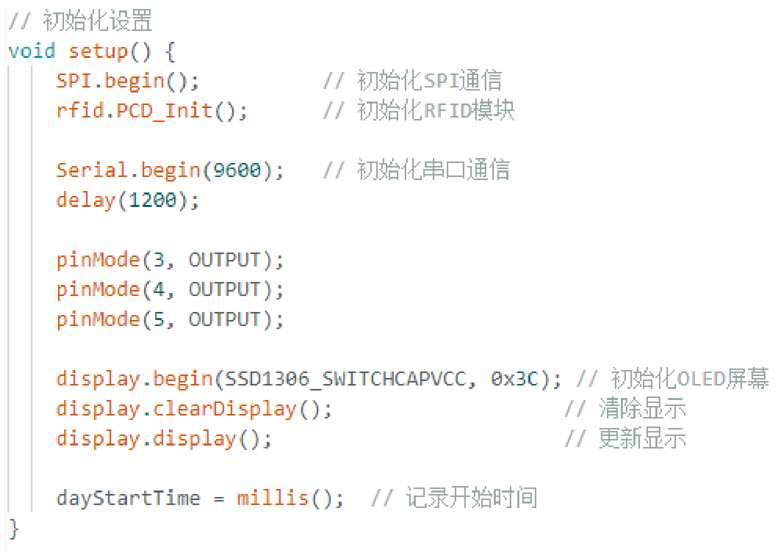
Defines pins for the NFC module.

Create an instance of MFRC522 to communicate with the NFC module.

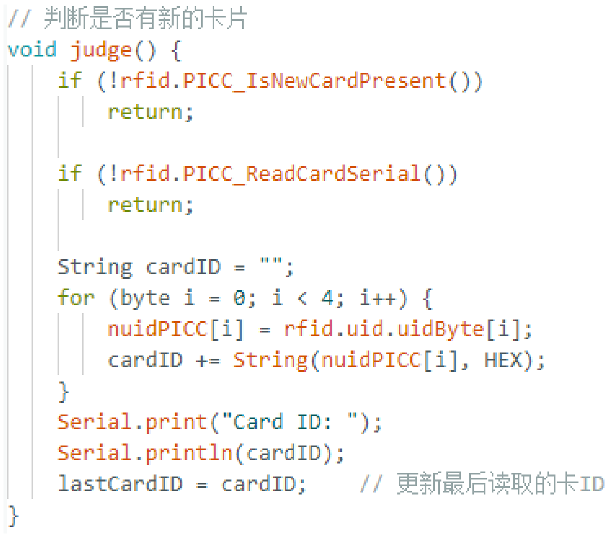
Declare variables such as an array of memory card ids, a burst counter and time variable, a person status variable, and so on.


```
#include <SPI.h> // 引入SPI库，用于NFC通信
#include <MFRC522.h> // 引入MFRC522库，用于NFC模块
#include <Wire.h> // 引入Wire库，用于I2C通信
#include <Adafruit_GFX.h> // 引入Adafruit_GFX库，用于图形显示
#include <Adafruit_SSD1306.h> // 引入Adafruit_SSD1306库，用于OLED显示屏

// 自定义变量
#define A 3                                                  // 自定义防爆破计数器阈值
const String memberCards[] = {"43f19a16", "25f3affc"};    // 预置会员卡ID

// OLED屏幕相关参数
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET 4
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

// NFC模块相关引脚定义
#define SS_PIN 10
#define RST_PIN 9
MFRC522 rfid(SS_PIN, RST_PIN);
byte nuidPICC[4];

// 变量声明
String lastCardID = "";                                    // 存储最后读取的卡ID
bool isWorking = false;                                    // 记录是否正在工作（上班状态）
unsigned long startTime = 0;                               // 记录上班时间
unsigned long lockEndTime = 0;                             // 记录锁定结束时间
unsigned long dayStartTime = 0;                            // 记录一天开始时间
int errorCount = 0;                                        // 记录错误次数
int conter = 0, time = 0, nowtime = 0;                     // 防爆破计数器和时间变量
const int memberCount = sizeof(memberCards) / sizeof(memberCards[0]);  // 预置卡ID的数量
```

setup() function: runs at Arduino startup and is used to initialize the setup. It initializes the SPI and RFID modules, sets up serial communication, defines pin patterns, and initializes the OLED display. And the initial time will be recorded.

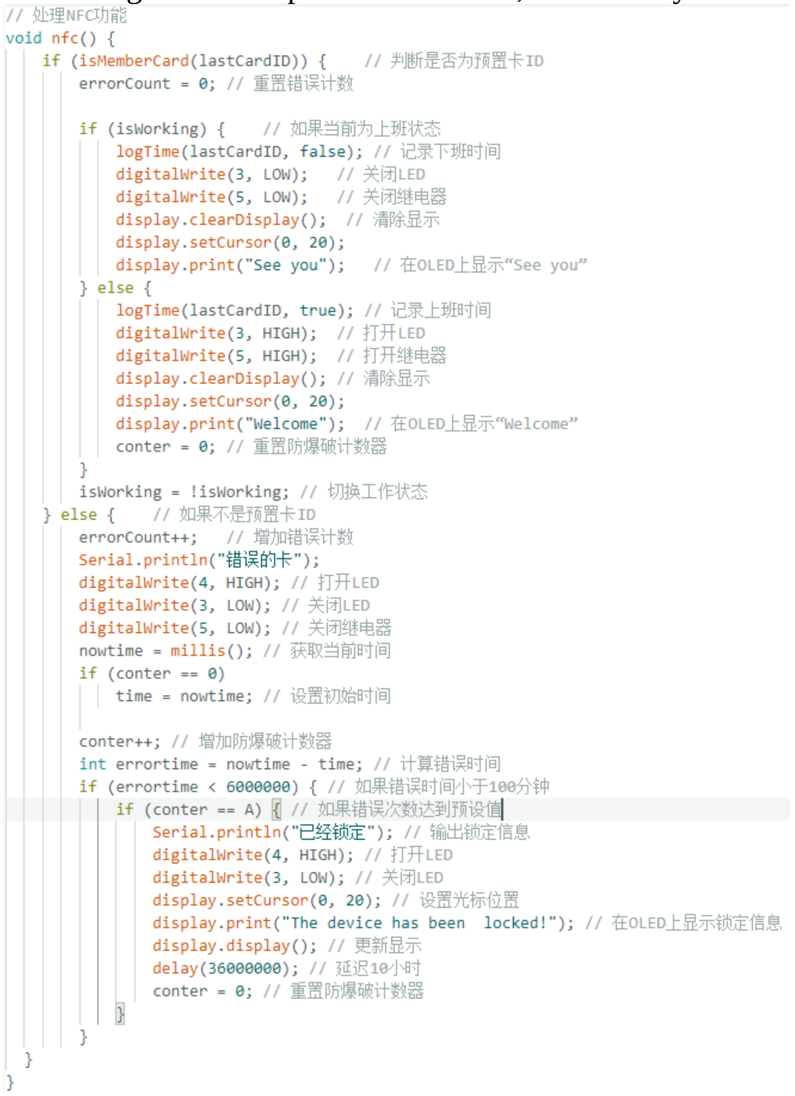

```
// 初始化设置
void setup() {
    SPI.begin();          // 初始化SPI通信
    rfid.PCD_Init();      // 初始化RFID模块

    Serial.begin(9600);   // 初始化串口通信
    delay(1200);

    pinMode(3, OUTPUT);
    pinMode(4, OUTPUT);
    pinMode(5, OUTPUT);

    display.begin(SSD1306_SWITCHCAPVCC, 0x3C); // 初始化OLED屏幕
    display.clearDisplay();                    // 清除显示
    display.display();                         // 更新显示

    dayStartTime = millis();  // 记录开始时间
}
```

The Judge() function determines whether a new card is detected and reads the card serial number. This code is mainly written with reference to the example code of the MFRC522 module.



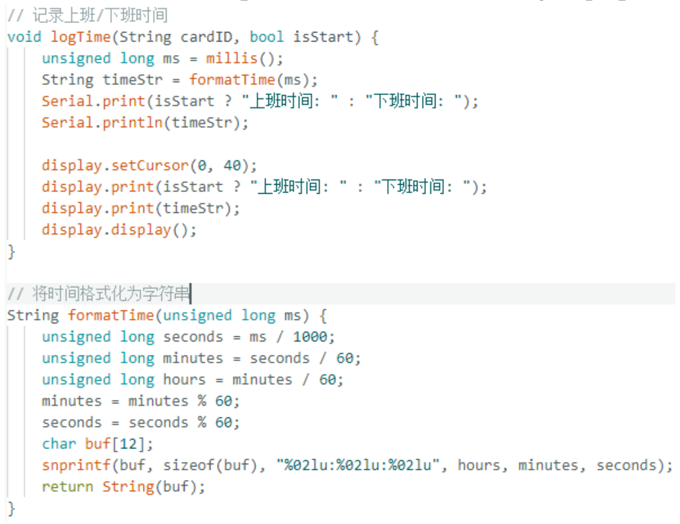
```
// 判断是否有新的卡片
void judge() {
    if (!rfid.PICC_IsNewCardPresent())
        return;

    if (!rfid.PICC_ReadCardSerial())
        return;

    String cardID = "";
    for (byte i = 0; i < 4; i++) {
        nuidPICC[i] = rfid.uid.uidByte[i];
        cardID += String(nuidPICC[i], HEX);
    }
    Serial.print("Card ID: ");
    Serial.println(cardID);
    lastCardID = cardID;    // 更新最后读取的卡ID
}
```

The NFC () function handles tasks related to NFC (Near Field Communication) recognition. This function is responsible for performing operations related to reading the NFC card, such as validating the card, processing its data, and triggering appropriate actions based on the card information. It also has a blasting function. If the wrong card ID is recognized multiple times in a row, the entire system will be locked.



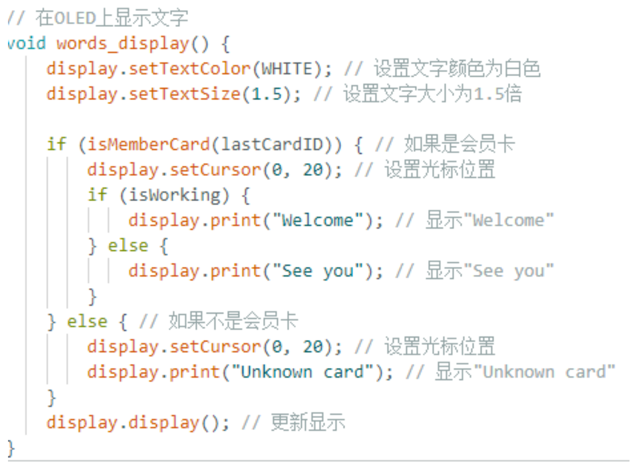
```
// 处理NFC功能
void nfc() {
    if (isMemberCard(lastCardID)) {    // 判断是否为预置卡ID
        errorCount = 0; // 重置错误计数

        if (isWorking) {    // 如果当前为上班状态
            logTime(lastCardID, false); // 记录下班时间
            digitalWrite(3, LOW);   // 关闭LED
            digitalWrite(5, LOW);   // 关闭继电器
            display.clearDisplay();  // 清除显示
            display.setCursor(0, 20);
            display.print("See you");   // 在OLED上显示“See you”
        } else {
            logTime(lastCardID, true); // 记录上班时间
            digitalWrite(3, HIGH);  // 打开LED
            digitalWrite(5, HIGH);  // 打开继电器
            display.clearDisplay(); // 清除显示
            display.setCursor(0, 20);
            display.print("Welcome");  // 在OLED上显示“Welcome”
            conter = 0; // 重置防爆破计数器
        }
        isWorking = !isWorking; // 切换工作状态
    } else {    // 如果不是预置卡ID
        errorCount++;   // 增加错误计数
        Serial.println("错误的卡");
        digitalWrite(4, HIGH); // 打开LED
        digitalWrite(3, LOW); // 关闭LED
        digitalWrite(5, LOW); // 关闭继电器
        nowtime = millis(); // 获取当前时间
        if (conter == 0)
            time = nowtime; // 设置初始时间

        conter++; // 增加防爆破计数器
        int errortime = nowtime - time; // 计算错误时间
        if (errortime < 6000000) { // 如果错误时间小于100分钟
            if (conter == A) { // 如果错误次数达到预设值
                Serial.println("已经锁定"); // 输出锁定信息
                digitalWrite(4, HIGH); // 打开LED
                digitalWrite(3, LOW); // 关闭LED
                display.setCursor(0, 20); // 设置光标位置
                display.print("The device has been  locked!"); // 在OLED上显示锁定信息
                display.display(); // 更新显示
                delay(36000000); // 延迟10小时
                conter = 0; // 重置防爆破计数器
            }
        }
    }
}
```

The formatTime() function and logTime() function can record the time when the card is swiped, so as to facilitate the analysis of the time when relevant personnel use laboratory equipment.



```
// 记录上班/下班时间
void logTime(String cardID, bool isStart) {
    unsigned long ms = millis();
    String timeStr = formatTime(ms);
    Serial.print(isStart ? "上班时间: " : "下班时间: ");
    Serial.println(timeStr);

    display.setCursor(0, 40);
    display.print(isStart ? "上班时间: " : "下班时间: ");
    display.print(timeStr);
    display.display();
}

// 将时间格式化为字符串
String formatTime(unsigned long ms) {
    unsigned long seconds = ms / 1000;
    unsigned long minutes = seconds / 60;
    unsigned long hours = minutes / 60;
    minutes = minutes % 60;
    seconds = seconds % 60;
    char buf[12];
    snprintf(buf, sizeof(buf), "%02lu:%02lu:%02lu", hours, minutes, seconds);
    return String(buf);
}
```

The void words_display() function is responsible for visualizing the state of the system or other important information on the OLED screen for the user to view or manipulate.



```
// 在OLED上显示文字
void words_display() {
  display.setTextColor(WHITE); // 设置文字颜色为白色
  display.setTextSize(1.5); // 设置文字大小为1.5倍

  if (isMemberCard(lastCardID)) { // 如果是会员卡
    display.setCursor(0, 20); // 设置光标位置
    if (isWorking) {
      display.print("Welcome"); // 显示"Welcome"
    } else {
      display.print("See you"); // 显示"See you"
    }
  } else { // 如果不是会员卡
    display.setCursor(0, 20); // 设置光标位置
    display.print("Unknown card"); // 显示"Unknown card"
  }
  display.display(); // 更新显示
}
```

3.5 Conclusion

3.5.1 Problems

During the initial phase of assembling the circuit, the RFID module consistently failed to function properly. Upon investigation, it was determined that the issue stemmed from the inability of the rst pin on the module to maintain a consistent potential. This inconsistency occasionally triggered resets of the RFID module. To remedy this, we decided to connect the rst pin directly to 5V to keep it consistently high, ensuring the module remained operational. While programming the OLED screen, I encountered a limitation with the library function that did not accommodate newline characters (\n) for breaking text across lines. Initially, I managed this by distributing a long sentence across multiple lines starting at different points. However, I later found that adjusting the positioning of text by adding or subtracting spaces could more effectively control where text appeared on the screen. This technique explains the presence of extra 'k's before the word 'locked' in the display message “The device has been locked!”

3.5.2 Achievements

In this open project, we developed a sophisticated laboratory equipment management system that utilizes NFC cards for user identification. This system manages the power of the equipment, logs the usage time by personnel, and displays the current status on LED and OLED screens. It also includes a feature to thwart brute force attacks. The system has potential for further optimization before it can be implemented. For instance, the current relay module could be upgraded to a DC-controlled AC solid-state relay module, allowing for direct control of mains power. Additionally, integrating a WiFi module would enable the automatic upload of data. This enhancement would allow laboratory staff to conveniently monitor the status and usage records of the equipment via smartphones, computers, and other digital devices.

Reference lists:

[1] Xjtlu.edu.cn, “Smart Car Project Tutorial,” extension://bfdogplmndidlpjfhoijckpakkdjkkil/pdf/viewer.html?file=https%3A%2F%2Fcore.xjtlu.edu.cn%2Fpluginfile.php%2F176056%2Fmod_resource%2Fcontent%2F1%2FSmart%2520Car.pdf(accessed at May 18 2024)

[2] Xjtlu.edu.cn, “Digital Clock Project,” extension://bfdogplmndidlpjfhoijckpakkdjkkil/pdf/viewer.html?file=https%3A%2F%2Fcore.xjtlu.edu.cn%2Fpluginfile.php%2F181720%2Fmod_resource%2Fcontent%2F1%2FDigital%2520Clock.pdf (accessed at May 19 2024)