PRÁCTICA 6

Ejercicio 1

En este apartado utilizaremos el lector de tarjetas microSD para leer un archivo que se encuentra en la raíz de la microSD y nos muestra su contenido de texto por el puerto serie.

Tenemos que incluir las librerias SPI.h y SD.h. Declaramos un objeto File.

En el setup(), lo primero que nos encontramos es un if() para que nos indique si se ha inicialiado bien el lector de tarjetas. SD.begin(), aquí dentro tenemos que introducir un entero que corresponde al PIN SS de la micro donde esta el lector conectado.

Si se ha inicializado bien el lector, procedemos a abrir el archivo indicando el path correcto: myFile = SD.open("/archivo.txt"). La barra "/" es necesaria para indicar que el archivo se encuentra en la raíz.

El siguiente If-else nos devuelve error si no encuentra el archivo, pero en caso de que lo encuentre el while() escribe todo el contenido del archivo de texto por el puerto serie.

A continuación cerramos el archivo ya que solo podemos tener uno abierto a la vez. Y acaba el programa ya que en el loop() no tenemos nada.

Ejercicio 2

En este ejercicio incluimos y añadimos las librerías SPI.h y MFRC522.h.

Primero de todo hay que definir los pines de reset y el SS de nuestro lector RFID. A continuación declaramos el objeto pasándole estos parámetros.

En el setup() lo único nuevo es inicializar el Bus SPI y el MFRC522, todo lo demás va en loop().

En el loop() usaremos un if que detecte cuando haya una tarjeta rfid cerca del lector. La función booleana que utilizamos es la siguiente: "mfrc522.PICC_IsNewCardPresent()". Después otro if para confirmar que lo que este rfid es válido para lectura, lo sabremos con la expresión: "mfrc522.PICC_ReadCardSerial()". Si el UID es válido, solo queda leerlo y mostrarlo por el puerto serie. Eso lo haremos con un for() que recorra byte por byte el UID leído, lo pase a un valor hexadecimal y lo vaya mostrando por pantalla.

Ejercicio 3

En este ejercicio haremos uso de los buses SPI para conectar y hacer funcionar dos esclavos a la vez con nuestro maestro. El objetivo es utilizar el lector de microSD y el lector RFID a la vez. Cada vez que el lector RFID detecte un RFID, escribirá su UID en hexadecimal junto la fecha y la hora en un archivo .log que se encuentra en la raíz de la microSD que contiene el lector de microSD.

Para controlar el tiempo hemos añadido además de las librerías de los apartados anteriores, la Time.h y la TimeLib.h.

Lo primero es definir los pines SS de los dos módulos y además el reset de la RC522. Creamos un objeto time_h, File, y MFRC522. Además hemos declarado un par de variables String que necesitaremos para el

desarrollo de nuestro código.

En el void setup() simplemente es juntar el código de los apartados anteriores, pero añadiremos dos líneas de código para etablecer una fecha y hora fictícia, ya que como no utilizaremos un real time clock, establecemos la fecha y hora arbitrárialmente. A la práctica, lo que nos interesa es poder escribir bien en el registro.log los UID que pasen y que el intervalo de tiempo entre cada evento de escritura sea real. Por lo tanto si esto funciona y quisieramos pasarlo a una fecha y hora real, sería fácil. Establecemos la fecha y hora en el momento justo que se carga el programa con el comando: "setTime(15, 32, 00, 07, 06, 2023);"

Luego en el loop(), el objeto time_h llamado fecha, lo vamos actualizando todo el rato con el tiempo que haya pasado desde que se ha iniciado el programa, con la expresion "fecha = now();"

Los Ifs para leer una tarjeta RFID son iguales que antes, pero la diferencia en este apartado esta dentro del for() que lee el UID. Aquí necesitaremos una variable string temporal para ir pasando a mayúscula cada letra si hiciera falta. Además el UID se va añadiendo a la otra variable string declarada al principio del programa "rfid".

Después de terminar la lectura del UID, queremos escribir en el archivo dentro de la microSD. Para esto utilizaremos dos funciones void(). La primera se llama writeRFID y le pasamos como parámetro el string rfid y la fecha justo en el momento de la lectura. Esta función se encarga de coger esa información y ordenarla en otro string, para al final convertirla en un "const char*" que es el tipo de data que necesita arduino para escribir en el fichero de la microSD.

Una vez convertido el string que contiene toda la información que queremos es escribir, lo pasamos a la función WriteFile(), la cual se encarga de abrir el archivo /registros.log, escribir el mensaje que le hemos pasado en tipo const char* y cerrar el archivo.

Cabe destacar que cada vez que escribimos en el archivo, borra lo anterior, por tanto nuestro string "alltxt" es acumulativo y va sumando todo el rato cada string de cada evento de lectura. Para que cuando velva a escribir, esten en orden todas las lecturas.

De esta manera conseguimos que el programa escriba todos los eventos de lectura de UIDs en nuestro archivo registros.log que se encuentra en la raíz de la microSD. Además los intervalos de tiempo que hay entre cada evento son reales.

Código:

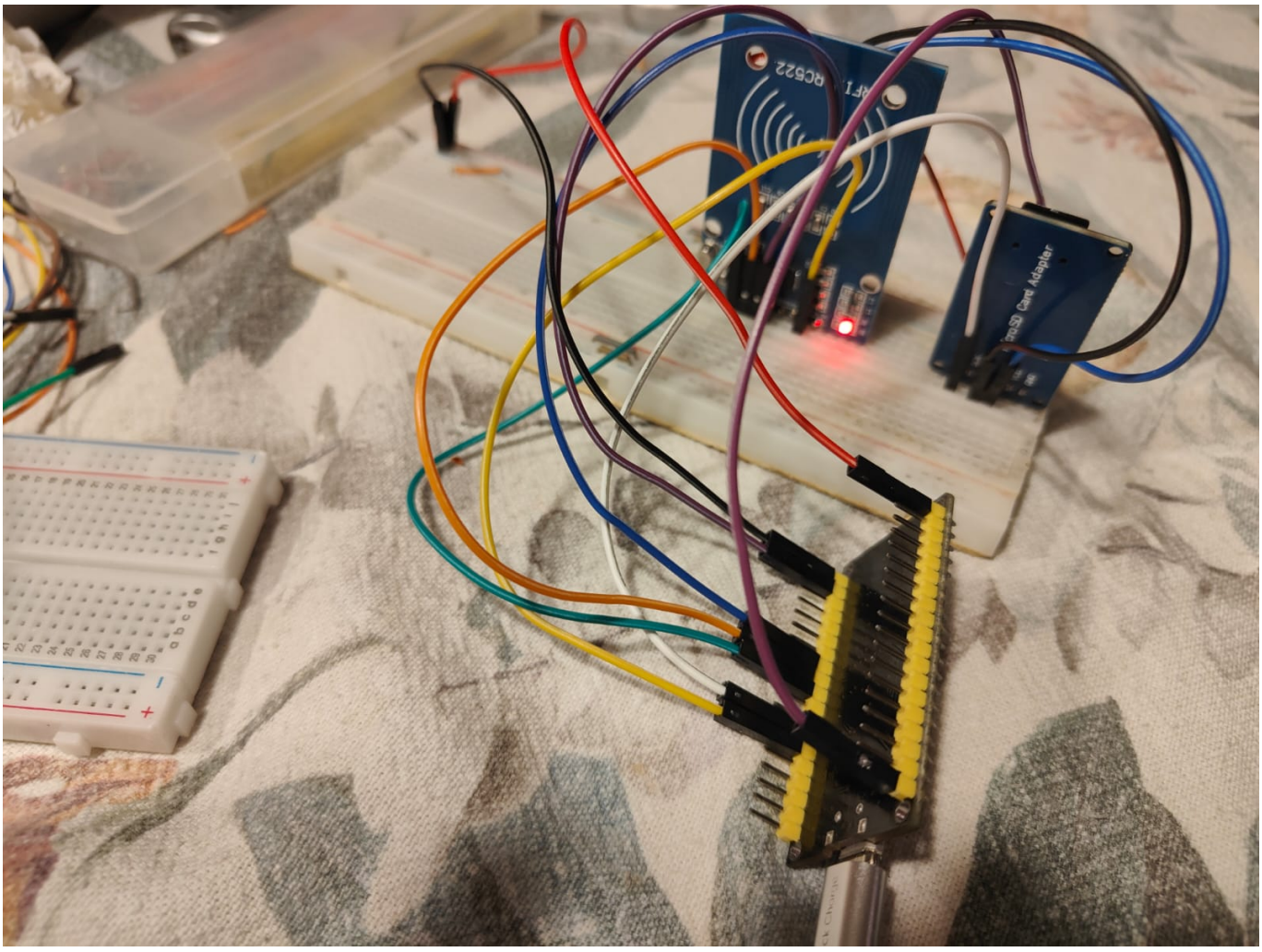
```
#include <SPI.h>
#include <SD.h>
#include <MFRC522.h>
#include <Time.h>
#include <TimeLib.h>

#define RST_PIN 0 //Pin 0 para el reset del RC522
#define SS_PIN 5 //Pin 5 para el SS (SDA) del RC522
#define CS 4 //Pin 4 para el SS del lector de SD
time_t fecha;
String rfid;
```

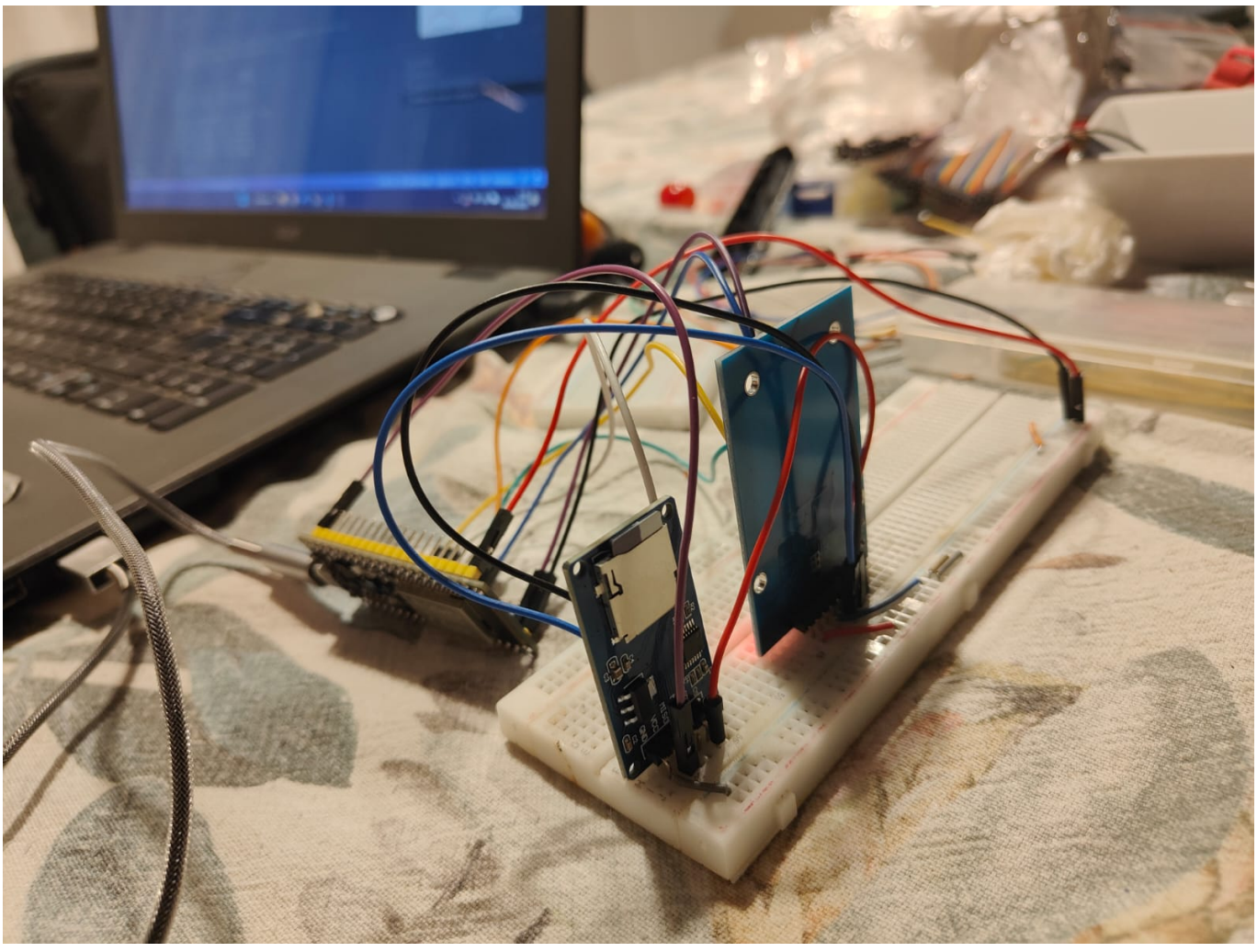
```
MFRC522 mfrc522(SS_PIN, RST_PIN); //Creamos el objeto para el RC522

File myFile;
String alltxt = "";

void WriteFile(const char * path, const char * message){

  myFile = SD.open(path, FILE_WRITE);

  if (myFile) {
    Serial.printf("Writing to %s ", path);
    //Serial.println(message);
    myFile.println(message);
    myFile.close();
    Serial.println("completed.");
  }

  else {
    Serial.println("error opening file ");
    Serial.println(path);
  }
}

void writeRFID (String rfid, time_t fecha)
{
    String date;
    const char* message;

        //Serial.println("open file.... registros.log");
        date += "Hora: ";
        date += hour(fecha);
        date += ":";
        date += minute(fecha);
        date += ":";
        date += second(fecha);
        date += "  ";
        date += day(fecha);
        date += "/";
        date += month(fecha);
        date += "/";
        date += year(fecha);
        date += "      RFID: ";
        date += rfid;
        date += "\n";

        //Serial.println(date);

        alltxt += date;

        message = alltxt.c_str();

        //Serial.print("msg = ");
        Serial.println(message);
```

```
        WriteFile("/registros.log", message);

}

void setup()
{
    Serial.begin(115200);
    setTime(15, 32, 00, 07, 06, 2023);
    fecha = now();
    SPI.begin(); //Iniciamos el Bus SPI
    mfrc522.PCD_Init(); // Iniciamos el MFRC522
    Serial.println("Lectura del UID");
    Serial.print("Iniciando SD ...");
  if (!SD.begin(CS))
  {
    Serial.println("No se pudo inicializar");
    return;
  }
  Serial.println("inicializacion exitosa");

}


void loop()
{
// Revisamos si hay nuevas tarjetas presentes
    fecha = now();
    if ( mfrc522.PICC_IsNewCardPresent())
    {
//Seleccionamos una tarjeta
        if ( mfrc522.PICC_ReadCardSerial())
        {
// Enviamos serialmente su UID
            Serial.print("Card UID:");
            rfid = "";
            String temp;
            for (byte i = 0; i < mfrc522.uid.size; i++)
            {
                Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
                Serial.print(mfrc522.uid.uidByte[i], HEX);
                temp = String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
                temp.toUpperCase();
                temp += String(mfrc522.uid.uidByte[i], HEX);
                temp.toUpperCase();
                rfid += temp;
            }
            Serial.println("\nStringcheck: " + rfid);
// Terminamos la lectura de la tarjeta actual
            mfrc522.PICC_HaltA();
        }
// Escribimos en el archivo log el RFID de la tarjeta y la fecha y hora "actuales"

        writeRFID(rfid, fecha);
```

```
        }
    }
```

A continuación adjuntamos cuatro imágenes donde se ve el correcto funcionamiento del sistema.


RC522
Micro SD Card Adapter



```
PROBLEMAS   SALIDA   CONSOLA DE DEPURACIÓN   TERMINAL

Hora: 15:32:55  7/6/2023      RFID:  96 EC C5 14
Hora: 15:33:25  7/6/2023      RFID:  2A DF 89 16

Writing to /registros.log Hora: 15:32:3  7/6/2023      RFID:
Hora: 15:32:3  7/6/2023      RFID:  03 8A 12 B7
Hora: 15:32:8  7/6/2023      RFID:  2A DF 89 16
Hora: 15:32:28  7/6/2023      RFID:  96 EC C5 14
Hora: 15:32:55  7/6/2023      RFID:  96 EC C5 14
Hora: 15:33:25  7/6/2023      RFID:  2A DF 89 16

completed.
```