

# CAJA FUERTE CON CONTROL DE ACCESO MEDIANTE KEYPAD Y RFID

## SAFE BOX WITH ACCESS CONTROL VIA KEYPAD AND RFID

Kevyn Medina<sup>1</sup>, Gael Pastrana<sup>2</sup>

### Resumen

En este proyecto se propone el desarrollo de una caja fuerte electrónica, la cual está controlada mediante un teclado 4x4 y una tarjeta RFID, utilizando el microcontrolador ATmega328p. El sistema permitirá el acceso seguro mediante la introducción de un código o identificación por RFID, controlando la apertura de la caja a través de un servomotor. Donde también se implementará una pantalla LCD 16x2 para visualizar información cómo el código mientras se introduce y una funcionalidad de historial de intentos almacenado en la memoria EEPROM, así como un buzzer que se activará en caso de que la caja permanezca abierta por un tiempo determinado. El proyecto se simulará en Proteus y se implementará físicamente con una estructura impresa en 3D. Donde como objetivo principal es diseñar un sistema seguro y de control de acceso que integre diferentes periféricos del ATmega328P de una manera eficiente.

**Palabras Clave:** Caja fuerte, RFID, Keypad, AtMega328p, Control de acceso.

### Abstract

This project proposes the development of an electronic safe controlled via a 4x4 keypad and an RFID card, using the ATmega328P microcontroller. The system will allow secure access through code entry or RFID identification, controlling the opening of the safe using a servo motor. A 16x2 LCD will also be implemented to display information such as the code while it is being entered, as well as a login attempt history feature stored in EEPROM memory. Additionally, a buzzer will be activated if the safe remains open for a certain period of time. The project will be simulated in Proteus and physically implemented with a 3D-printed structure. The main objective is to design a secure and efficient access control system that integrates various peripherals of the ATmega328P.

**Keywords:** Safe box, RFID, Keypad, ATmega328P, Access control.

---

<sup>1</sup> Ing. Computación, #1190823 (kevyn.medina@uabc.edu.mx).

<sup>2</sup> Ing. Computación, #1182425 (gael.pastrana@uabc.edu.mx)

## 1. Introducción

En un mundo donde la seguridad de nuestros bienes personales y documentos importantes es imperativa, los métodos tradicionales de almacenamiento seguro, como las cajas fuertes mecánicas con llaves o combinaciones físicas, presentan diversas limitaciones. Entre ellas se encuentran la vulneración al acceso no autorizado por duplicación de llaves e incluso la pérdida o el olvido de la llave y combinaciones. Además, dichos sistemas suelen carecer de registros de acceso o mecanismos de alerta ante intentos de intrusión o situaciones fuera de lo común, como dejar la puerta abierta por un tiempo prolongado, lo cual representa un riesgo de seguridad adicional. Estas limitaciones evidencian la necesidad de desarrollar sistemas de seguridad más inteligentes, versátiles y difíciles de vulnerar.

### 1.1 Fundamentos teóricos

#### 1.1.1 Microcontrolador

Una unidad de microcontrolador (MCU), como lo es el ATmega328P, es esencialmente una computadora pequeña en un solo chip. Está diseñado para administrar tareas específicas dentro de un sistema integrado sin requerir un sistema operativo complejo. Estos circuitos integrados compactos (IC) contienen un núcleo de procesador (o núcleos), memoria de acceso aleatorio (RAM) y memoria de solo lectura programable borrable eléctricamente (EEPROM) para almacenar los programas personalizados que se ejecutan en el microcontrolador, incluso cuando la unidad está desconectada de una fuente de alimentación. [1]

#### 1.1.2 LCD

LCD o pantalla de cristal líquido, está conformada por un número de píxeles en color o monocromos colocados delante de una fuente de luz o reflectora, es delgada y plana.

Es un dispositivo empleado para la visualización de contenidos o información de una forma gráfica, mediante caracteres, símbolos o pequeños dibujos, los cuales pueden variar todo

esto en función del modelo. Se encuentra gobernado por un microcontrolador que dirige todo su funcionamiento. Por ejemplo, un LCD de 16x2 dispone de 2 filas de 16 caracteres cada una. [2]

#### 1.1.3 Teclado Matricial 4x4

La interfaz de teclado (keypad) incluye una herramienta que se utiliza con frecuencia y es bastante fácil de usar. El teclado es uno de los medios de entrada más ampliamente utilizados en aplicaciones de interfaz con máquinas. La función principal del teclado es, en realidad, ahorrar el uso de puertos de entrada; por ejemplo, un teclado 4x4 puede utilizarse como entrada con hasta 16 teclas, pero solo requiere 8 puertos. En este sistema de seguridad para una caja fuerte, el teclado cumple la función de permitir la introducción de una contraseña para abrir la puerta de la caja. [2]

#### 1.1.4 RFID

La tecnología de Identificación por Radiofrecuencia (RFID) utiliza ondas de radio para identificar personas u objetos. Existe un dispositivo que lee la información contenida en un dispositivo inalámbrico o "etiqueta" a distancia, sin necesidad de contacto físico ni línea de visión. [3]

## 1.2 Objetivos del proyecto

El objetivo principal de este proyecto es diseñar e implementar una caja fuerte electrónica que integre un sistema de control de acceso multifactorial, utilizando un teclado 4x4 y un módulo RFID, gestionado por un microcontrolador ATmega328P. El sistema incluirá una pantalla LCD para interacción con el usuario, un servomotor para el mecanismo de apertura, un buzzer de alerta y una memoria EEPROM para el registro de accesos.

## 1.3 Hipótesis de trabajo

El sistema implementado será capaz de reducir los riesgos de acceso no autorizado y mejorar el control de uso de la caja fuerte, cumpliendo con métricas de funcionalidad como: tiempos de

respuesta menores a 2 segundos, precisión del 100% en la autenticación RFID, y registro de al menos 50 intentos en EEPROM sin pérdida de datos.

## 2. Materiales y Métodos

- Arduino (ATmega328p)
- Pantalla LCD 16x2
- Sensor RFID RC522
- Keypad 4x4
- Servomotor
- Buzzer
- Filamento 3D
- Tornillos

### 2.1 Diagrama de bloques.

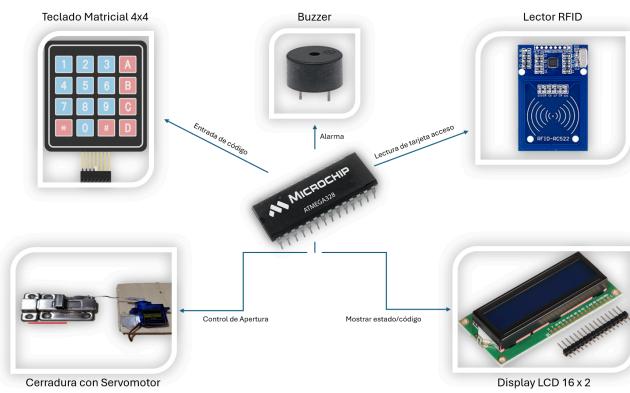


Figura 1. Diagrama a bloques del sistema de caja fuerte electrónica

### 2.2 Diagrama en Proteus.

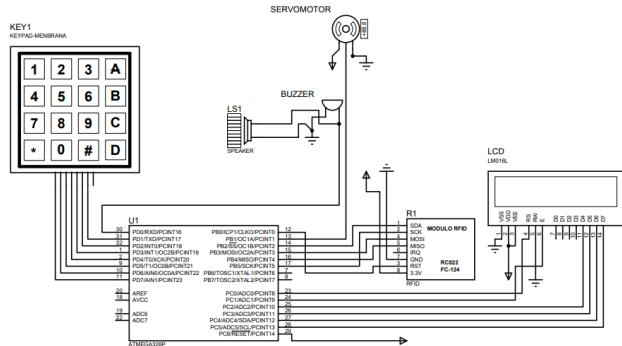


Figura 2. Simulación del sistema en Proteus.

### 2.3 Bosquejo de estructura física.



Figura 3. Diseño de estructura impresa en 3D (puerta cerrada).

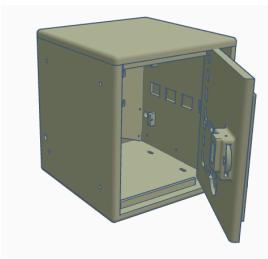


Figura 4. Diseño de estructura impresa en 3D (puerta abierta).

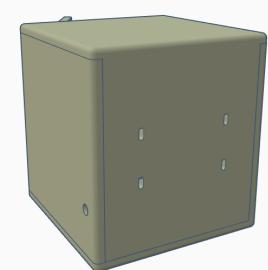


Figura 4. Diseño de estructura impresa en 3D (parte trasera).

### 2.4 Requisitos.

- Validación de acceso mediante código o tarjeta RFID.
- Visualización de clave en LCD.
- Visualización de historial en LCD.
- Apertura y cierre mediante servomotor controlado por ATmega328p.
- Registro de intentos en memoria EEPROM con posibilidad de consulta.
- Activación de buzzer al permanecer la puerta abierta durante un tiempo especificado.
- Estructura física realizada con impresión 3D.

## 2.5 Presupuesto.

Material	Precio
Arduino	100\$
Pantalla LCD	70\$
Sensor RFID	60\$
Keypad	60\$
Servomotor	100\$
Buzzer	20\$
Filamento	500\$
Tornillos	30\$

## 2.6 Periféricos utilizados.

- Timers.
- Keypad 4x4.
- LCD 16x2 (I2C).
- RFID RC522 (SPI).
- EEPROM.
- PWM.

## 2.7 Planeación del proyecto.

Actividad	Mayo							
	5 - 7	8 - 10	11 - 14	15 - 17	18 - 20	21 - 22	23	24 - 26
Planificación								
Verificación de componentes individualmente								
Conectar y programar Keypad y RFID								
Mostrar el estado en pantalla LCD								
Controlar el servomotor con Keypad y RFID								
Agregar buzzer con timer								
Probar todos los componentes conectados								
Impresión 3D								
Montaje de piezas								
Simulación de diferentes casos y corregir errores.								
Entrega de proyecto								
Entrega de reporte								

## 3. Resultados Esperados.

Al finalizar el desarrollo e implementación del proyecto se espera que el sistema cumple con los siguientes resultados:

- Funcionamiento confiable y consistente del sistema de control de acceso, validando solo usuarios autorizados.
- Visualización clara de información relevante, como ingreso de código y datos almacenados en la pantalla LCD.
- Accionamiento correcto del servomotor al abrir y cerrar la caja tras una verificación exitosa del acceso.
- Almacenamiento y consulta del historial de intentos de acceso en la memoria EEPROM, sin tener pérdidas de información.
- Activación del buzzer en caso de que la puerta permanezca abierta más tiempo del establecido.
- Integración de al menos cuatro periféricos del microcontrolador AtMega328P.
- Simulación funcional en Proteus, antes de implementarlo físicamente.
- Prototipo físico funcional con carcasa impresa en 3D, donde se representa de una manera realista una caja fuerte electrónica.
- Validación de fiabilidad del sistema, poniendo a prueba de manera repetitiva, evaluando el acceso ante claves correctas e incorrectas, teniendo respuestas ante fallos o accesos no autorizados.

## 4. Resultados

### 4.1 Implementación física.

Se realizó la fabricación de las paredes de la caja fuerte mediante impresión 3D, con material PLA. Donde se pudo obtener un buen diseño que se adaptara a las diferentes medidas de los componentes, así como pantalla LCD, Keypad, sensor RFID, servomotor y arduino (Figura 4). Cómo también tenemos la implementación del circuito en físico del sistema de acceso colocado detrás de la puerta (Figura 6).



Figura 4. Estructura impresa en 3D (frontal).



Figura 5. Estructura impresa en 3D (lateral).

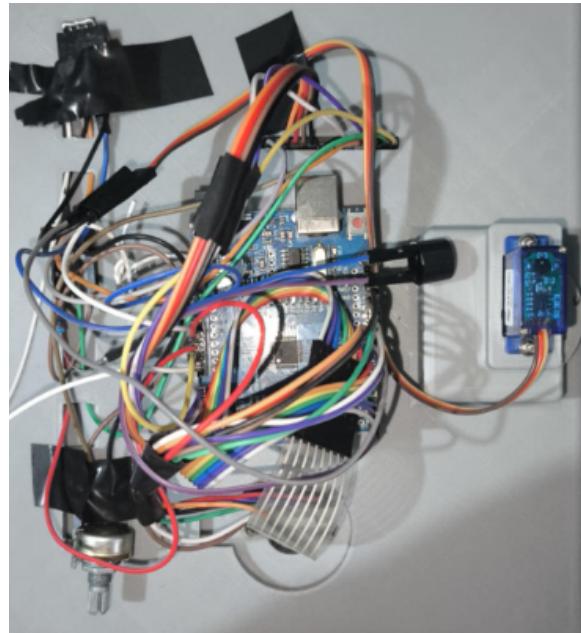


Figura 6. Circuito físico

### 4.2 Funcionamiento del sistema.

Interfaz de usuario.

- Ingreso de código (Figura 7): Interfaz donde el usuario introduce la clave mediante el keypad y se visualiza en la pantalla LCD en forma de asterisco, además de la hora.

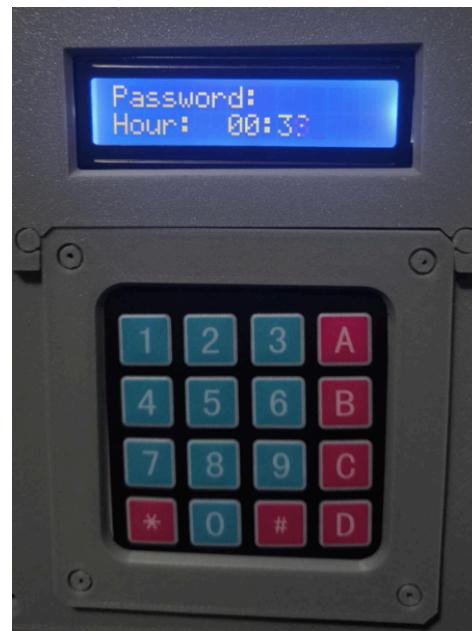


Figura 7. Interfaz para ingresar código.

- Historial de accesos (Figura 8): Al darle la tecla “#” se muestra un registro de datos almacenados en la EEPROM, donde se muestra la marca de tiempo de un acceso exitoso.



Figura 8. Interfaz para visualizar historial de accesos..

- Cambio de código (Figura 9): Al presionar doble vez en la tecla “\*” entras a la interfaz donde es posible actualizar la clave de acceso.



Figura 9. Interfaz para cambiar el código de acceso.

#### 4.3 Pruebas y validación.

Se realizaron diferentes pruebas para que el funcionamiento sea confiable y consistente. Donde se ingresan diferentes combinaciones de clave válidas y no válidas. También usando keypad y RFID de manera intercalada. A su vez verificando que en ambos casos, al momento de que pase un tiempo determinado, suena el buzzer hasta cerrar la puerta.

- Mensajes de validación: Se tienen dos mensajes en caso de que la clave de acceso sea correcta e incorrecta. En la Figura 8 aparece el mensaje cuando ingresas el código correcto y esto hace accionar el servomotor para abrir la puerta. En la Figura 9. Tenemos el mensaje si se ingresa una clave incorrecta y después de unos segundos vuelve a la interfaz de inicio (Figura 7).



Figura 10. Visualización de mensaje cuando ingresas la clave de acceso correcta.



**Figura 11.** Visualización de mensaje cuando ingresas la clave de acceso incorrecta.

- En la figura 8 tenemos la validación al momento de acercar el tag al sensor RFID detrás del keypad, lo que hace que también accione el servomotor para abrir la puerta.



**Figura 12.** Visualización de acceso al momento de acercar el tag al lector RFID.

- En las siguientes imágenes (Figura 13, 14, 15) se tienen diversos registros de acceso almacenados en la EEPROM.



**Figura 13.** Registro autorizado 1.



**Figura 14.** Registro autorizado 2.



**Figura 15.** Registro autorizado 3 .

## Video:

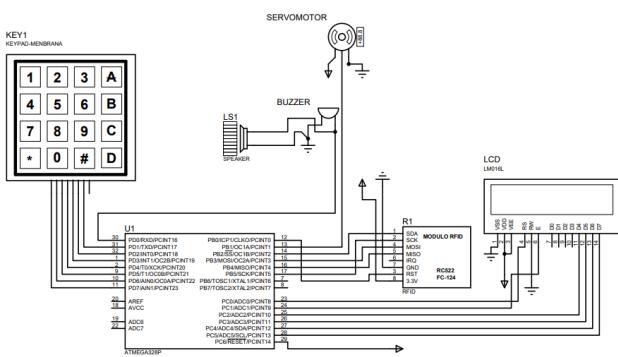
[https://youtu.be/p4\\_zt036yT8](https://youtu.be/p4_zt036yT8)

### Github:

<https://github.com/strben23/Proyecto-Final-SafeBox>

### 4.3 Simulación.

En el caso de la simulación, se realizó en el software de diseño electrónico Proteus y a continuación se muestran las conexiones realizadas con todos los componentes del sistema.



Se observa un teclado matricial 4x4 para el manejo del código de apertura, una pantalla LCD 16x2 que indica el estado de la caja fuerte y la hora, un módulo RFID como método extra de autentificación para la caja fuerte, un servomotor que controla el la apertura o cierre de la puerta, un buzzer que indica si la caja fuerte ha estado abierta por determinado tiempo y el microcontrolador ATmega328p para el manejo del sistema de caja fuerte.

**Video:** <https://youtu.be/QcetvpRtR0M>

### **Conclusiones.**

El proyecto de la caja fuerte electrónica con control de acceso mediante keypad y RFID cumplió con los objetivos que se plantearon, demostrando ser un sistema seguro, eficiente y funcional. Donde se integraron múltiples periféricos del microcontrolador ATmega328p, así como el keypad 4x4, módulo RFID RC522, pantalla LCD, servomotor y memoria EEPROM, lo que permitió crear un prototipo robusto y confiable. Se obtuvieron resultados deseados, tales como que el sistema respondiera de forma instantánea ante entradas válidas, la autenticación por RFID donde se muestra una precisión del 100%, además de que se logró almacenar los intentos de acceso en la memoria EEPROM sin perder datos.

Además la implementación física con una estructura impresa en 3D permite un diseño compacto y realista, mientras que la simulación en Proteus nos permite depurar y optimizar el sistema antes de construirlo. Las funciones como historial de accesos y la alerta por puerta abierta, añade un poco de valor debido a que mejora la utilidad en escenarios realistas.

En este proyecto no solo se aborda una implementación de una caja fuerte tradicional, sino que también es una base para futuras mejoras en la integración de otra tecnología como comunicación inalámbrica o uso de sensores biométricos, con esto la combinación de hardware

y software demuestra ser una solución efectiva para el control de acceso seguro.

## **Referencias**

- [1] Schneider, J. & Smalley, I. (2024). Microcontroller. IBM.
- [2] Uaeh. (s. f.). Actuadores | Arduino. [http://ceca.uaeh.edu.mx/informatica/oas\\_fina/l/red4\\_arduino/actuadores.html](http://ceca.uaeh.edu.mx/informatica/oas_fina/l/red4_arduino/actuadores.html)
- [3] Radio Frequency Identification (RFID): What is it? | Homeland Security. (s. f.). U.S. Department Of Homeland Security. <https://www.dhs.gov/archive/radio-frequency-identification-rfid-what-it>

## Anexos

C/C++

```
/*
 * PFinal.c
 *
 * Created: 5/18/2025 5:50:57 PM
 * Author : Gaelp
 */

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <time.h>
#include <stdbool.h>
#include "EEPROM.h"
#include "GPIODriver.h"
#include "LCD.h"
#include "mfrc522.h"
#include "UART.h"
#include "servo.h"

#define F_CPU 16000000UL
#define LCD_WIDTH 16 // Adjust for
    your LCD size (16x2 or 20x4)
#define RST_DDR  DDRB
#define RST_PORT PORTB
#define RST_PIN  PB1
#define MAX_HISTORIAL 20
#define HISTORIAL_START 10
#define HISTORIAL_INDICE 9

char PressedKey(void);
void timer1_init(void);
void MFRC522_resetPinInit(void);
uint8_t read_rfid_tag(void);
uint8_t compare_uids(uint8_t *uid1,
    uint8_t *uid2);
void validarPasswd(uint8_t passwd[]);
void cambiarPasswd(uint8_t
    newPasswd[]);
void mensajeHoraPasswd(void);
void abrirPuerta();
void cerrarPuerta();
void
    actualizarIndiceHistorial(uint8_t
    idx);
void actualizarHistorial();
void leerHistorial();
void sonarBuzzer();
void apagarBuzzer();
```

```
volatile uint8_t hora = 0;
volatile uint8_t minuto = 0;
volatile uint8_t segundo = 0;
volatile uint8_t contadorSegundosReal
    = 0; // Contador de segundos
    reales
volatile uint8_t tiempoPuertaAbierta
    = 0;
char hora_str[12];
volatile uint8_t actualizarHora = 0;
bool puertaAbierta = false;
bool leyendoHistorial = false;
bool buzzerActivado = false;
uint8_t saved_uid[4];
uint8_t known_uid[4] = {0x3E, 0x1B,
    0xED, 0x00};
uint8_t known_uid2[4] = {0x08, 0x13,
    0x24, 0x91};

int main(void){
    DDRD |= (1 << PD0); // PD0
    como salida
    PORTD &= ~(1 << PD0); //
    Inicialmente apagado

    // Inicializacion de puerto D para
    Teclado Matricial 4x4
    DDRD |= 0b00001111; //D0-D3 outputs
    DDRD &= 0b00001111; //D4-D7 inputs
    PORTD |= 0b11110000; //D4-D7 Pull
    ups
    PORTD |= 0b00001111; //D1-D3 outputs
    high
    //GPIO_ConfigPin(PORT_D, 0,
    OUTPUT);
    PORTD &= ~(1 << PORTD0);
    char pressedKey = 'A'; //
    Inicializa variable de lectura de
    tecla presionada

    lcd_init(); // Inicializacion de
    LCD 16x2
    timer1_init(); // Inicializacion de
    Timer para la hora
    MFRC522_resetPinInit();
    MFRC522_init();
    servo_init();
```

```

//UART_init(9600, UART_8BITS,
UART_PARITY_DISABLE,
UART_STOPBIT_1);
sei();

uint8_t passwd[4];
uint8_t newPasswd[4];
uint8_t teclasPresionadas = 0;
uint8_t cambioPasswd = 0;

mensajeHoraPasswd();

while(1){
    pressedKey = PressedKey();

    //if (read_rfid_tag()) {
        //if
    (compare_uids(saved_uid,
    known_uid)) {

//UART_string("Authorized tag
detected!\r\n");
    //} else {

//UART_string("Unknown tag.\r\n");
    //}
    //_delay_ms(1000); //
Debounce
    //}

    // Se actualiza el reloj si
    //paso un minuto

    if(puertaAbierta &&
tiempoPuertaAbierta>=60 ){
        sonarBuzzer();
    }

    if (actualizarHora &&
!leyendoHistorial) {
        actualizarHora = 0;

        sprintf(hora_str,
sizeof(hora_str), "%02u:%02u",
hora, minuto);
        lcd_goto_xy(1, 7);

lcd_write_word(hora_str);
    }
}

```

```

    // Cambio de pin - Usuario
    presiona dos veces seguidas '*'
    if (pressedKey == '*' &&
!puertaAbierta) {
        cambioPasswd++;
    } else if (pressedKey
!= 'A') {
        cambioPasswd = 0;
    }
    if (cambioPasswd == 2){
        cambioPasswd++;
        teclasPresionadas = 0;

        if (cambioPasswd == 3){
            lcd_clear();
            lcd_goto_xy(0, 0);
            lcd_write_word("-"
Changing Code");
            lcd_goto_xy(1,
0);

            lcd_write_word("New Code: ");

            while
(teclasPresionadas < 4){
                pressedKey
= PressedKey();
                if
((pressedKey >= '0' && pressedKey
<= '9')){

                lcd_goto_xy(1,
10+teclasPresionadas);

                lcd_write_character('*');

                newPasswd[teclasPresionadas] =
pressedKey - '0';

                teclasPresionadas++;
            }
        }

        cambiarPasswd(newPasswd);

        mensajeHoraPasswd();
        teclasPresionadas
= 0;
        pressedKey = 'A';
        cambioPasswd = 0;
    }
}

```

```

        }

        // Usuario ingresa código
        if (((pressedKey >= '0' &&
        pressedKey <= '9') &&
        teclasPresionadas < 4) &&
        !puertaAbierta){
            lcd_goto_xy(0,
            10+teclasPresionadas);

            lcd_write_character('*');

            passwd[teclasPresionadas] =
            pressedKey - '0';
            teclasPresionadas++;

            // Código valido
            if (teclasPresionadas
            >= 4){

                validarPasswd(passwd);
                teclasPresionadas
                = 0;
            }
        }

        // Usuario aproxima Tarjeta
        // RFID
        if (read_rfid_tag() &&
        !puertaAbierta){
            // Tag RFID valido
            if
            (compare_uids(saved_uid, known_uid)
            || compare_uids(saved_uid,
            known_uid2)) {

                //UART_string("Authorized tag
                detected!\r\n");
                abrirPuerta();
            } else{
                lcd_clear();
                lcd_goto_xy(0,0);

                lcd_write_word("Incorrect RFID!");
                lcd_goto_xy(1,0);

                lcd_write_word("Try again... ");
                _delay_ms(2000);

                mensajeHoraPasswd();
            }
        }
    }
}

```

```

        _delay_ms(1000);//

Debounce
}

// Usuario abre historial con
'#'
if (pressedKey == '#'){
    leerHistorial();
}

// Usuario cierra puerta con
'*'
if (puertaAbierta &&
pressedKey == '*')
    cerrarPuerta();
}

void timer1_init() {
    // Prescaler de 1024
    // Frecuencia del timer: 16MHz /
    1024 = 15625 Hz
    // Para 1s = 15625 ciclos
    TCCR1A = 0;
    TCCR1B = (1 << WGM12) | (1 << CS12)
    | (1 << CS10); // CTC + prescaler
    1024
    OCR1A = 15625; // 1Hz/1s
    TIMSK1 = (1 << OCIE1A); // Habilita
    interrupción por comparación
}

ISR(TIMER1_COMPA_vect) {

    contadorSegundosReal++;

    if (contadorSegundosReal >= 10) {
        contadorSegundosReal = 0;

        minuto++;
        segundo = 0;

        if(puertaAbierta){

            tiempoPuertaAbierta++;
        }else{

            tiempoPuertaAbierta=0;
        }
    }
}

```

```

        apagarBuzzer();
    }

    if (minuto >= 60) {
        minuto = 0;
        hora++;

        if (hora >= 24) {
            hora = 0;
        }
    }
}

actualizarHora = 1;
}

void MFRC522_resetPinInit() {
    RST_DDR |= (1 << RST_PIN);      // RST as output
    RST_PORT |= (1 << RST_PIN);     // RST high
}

uint8_t read_rfid_tag(void) {
    uint8_t uid[4];
    uint8_t tagType[2];

    if (MFRC522_request(0x26, tagType))
    {
        if (MFRC522_anticoll(uid)) {
            memcpy(saved_uid, uid,
4); // Save UID globally
            return 1; // Success
        }
    }
    return 0; // Failure
}

uint8_t compare_uids(uint8_t *uid1,
    uint8_t *uid2) {
    for (uint8_t i = 0; i < 4; i++) {
        if (uid1[i] != uid2[i])
            return 0; // Not equal
    }
    return 1; // Equal
}

char PressedKey(void) {
    static uint8_t keyReleased = 1; // Bandera que indica si se soltó la tecla anterior
    char key = 0;
}

```

```

// Escanea cada fila del teclado
PORTD = 0b11110110; _delay_ms(1);
if (!(PIND & (1 << PIND7))) key =
'1';
else if (!(PIND & (1 << PIND6))) key =
'4';
else if (!(PIND & (1 << PIND5))) key =
'7';
else if (!(PIND & (1 << PIND4))) key =
'*';

if (!key) {
    PORTD = 0b11111010;
    _delay_ms(1);
    if (!(PIND & (1 << PIND7))) key =
'2';
    else if (!(PIND & (1 << PIND6))) key =
'5';
    else if (!(PIND & (1 << PIND5))) key =
'8';
    else if (!(PIND & (1 << PIND4))) key =
'0';
}

if (!key) {
    PORTD = 0b11111100;
    _delay_ms(1);
    if (!(PIND & (1 << PIND7))) key =
'3';
    else if (!(PIND & (1 << PIND6))) key =
'6';
    else if (!(PIND & (1 << PIND5))) key =
'9';
    else if (!(PIND & (1 << PIND4))) key =
'#';
}

/*if (!key) {
    PORTD = 0b11111110;
    _delay_ms(1);
    if (!(PIND & (1 << PIND7))) key =
'A';
    else if (!(PIND & (1 << PIND6))) key =
'B';
    else if (!(PIND & (1 << PIND5))) key =
'C';
    else if (!(PIND & (1 << PIND4))) key =
'D';
}*/

```

```

// Lógica de detección de nueva
pulsación
if (key && keyReleased) {
    keyReleased = 0;
    return key;
}

// Si ya no se detecta ninguna
tecla, marcamos como liberada
if (!key) {
    keyReleased = 1;
}

return 'A';
}

void validarPasswd(uint8_t passwd[]){
    uint8_t contadorCorrecto = 0;

    for (uint8_t i=0; i<4; i++){
        if (EEPROM_read(i) ==
passwd[i])
            contadorCorrecto++;
    }

    if (contadorCorrecto == 4){
        lcd_clear();
        lcd_goto_xy(0,0);
        lcd_write_word("Successful
Code!");
        lcd_goto_xy(1,0);
        lcd_write_word("Opening
Door...");
        _delay_ms(15000);
        abrirPuerta();
    } else{
        lcd_clear();
        lcd_goto_xy(0,0);
        lcd_write_word("Incorrect
Code!");
        lcd_goto_xy(1,0);
        lcd_write_word("Try
again...");
        _delay_ms(15000);
        mensajeHoraPasswd();
    }
}

void cambiarPasswd(uint8_t
newPasswd[]){
    uint8_t contadorCorrecto = 0;
}

```

```

for (uint8_t i=0; i<4; i++){
    EEPROM_update(i,
newPasswd[i]);
}
}

void mensajeHoraPasswd(void){
    lcd_clear();
    lcd_goto_xy(1, 0);
    lcd_write_word("Hour: ");
    lcd_goto_xy(0,0);
    lcd_write_word("Password: ");
}

void abrirPuerta(void){
    puertaAbierta = true;
    //Logica abrir puerta
    lcd_clear();
    lcd_goto_xy(0,0);
    lcd_write_word("Status Door:");
    lcd_goto_xy(1,0);
    lcd_write_word("OPEN");
    //GPIO_EscribirPin(PORTD, 0, HIGH);

    servoAngulo(37);

    actualizarHistorial();

}

void cerrarPuerta(void){
    puertaAbierta = false;

    //Logica cerrar puerta
    servoAngulo(160);

    mensajeHoraPasswd();

}

void
actualizarIndiceHistorial(uint8_t
idx){
    if (idx < MAX_HISTORIAL){
        idx++;
    } else{
        idx = 0;
    }
    EEPROM_update(HISTORIAL_INDICE,
idx);
}

```

```
}

void actualizarHistorial(){
    uint8_t indice =
EEPROM_read(HISTORIAL_INDICE);
    uint16_t direccion =
HISTORIAL_START + (indice * 2);

EEPROM_update(direccion, hora);
EEPROM_update(direccion+1, minuto);

actualizarIndiceHistorial(indice);

}

void leerHistorial(){
    leyendoHistorial = true;
    uint8_t indiceLectura = 0;
    uint8_t c = 1;
    uint8_t hr, min;
    char key = 'A';
    char horaHistorial[12];

lcd_clear();
lcd_goto_xy(0,0);
lcd_write_word("Entry History");
lcd_goto_xy(1,0);
lcd_write_word("Press '#': Next");

while (1){
    key = PressedKey();

    if (key == '#'){
        hr =
EEPROM_read(HISTORIAL_START +
(indiceLectura * 2));
        min =
EEPROM_read((HISTORIAL_START +
(indiceLectura * 2))+1);

        if (hr <= 24 && min <=
60){
            lcd_goto_xy(1,0);

snprintf(horaHistorial,
sizeof(horaHistorial), "%u.
%02u:%02u      ", c, hr, min);

lcd_write_word(horaHistorial);
        indiceLectura++;
        c++;
    } else{

```

```

    lcd_goto_xy(1,0);

    lcd_write_word("END          ");
    _delay_ms(2000);
    break;
}
key = 'A';
}
}

mensajeHoraPasswd();
leyendoHistorial = false;
}

void UART_print_dec(uint8_t num) {
    char buf[4]; // máximo 3 dígitos +
    '\0'
    sprintf(buf, sizeof(buf), "%u",
    num);
    UART_string(buf);
}

void sonarBuzzer(){
    PORTD |= (1 << PD0);
}

void apagarBuzzer(){
    PORTD &= ~(1 << PD0);
}

```