Proyecto Arduino

Lenguaje de Programación II

**Integrantes**

Benjamín Olguín

Matias Vilches

Gustavo Plaza

Tabla de contenido

[Introducción 3](#_Toc12374056)

[Usos y aplicaciones del RFID 4](#_Toc12374057)

[Aplicabilidad 4](#_Toc12374058)

[Sensores Utilizados y/o Dispositivos Externos 4](#_Toc12374059)

[LCD 1602 I2C 4](#_Toc12374060)

[Lector RFID RC522: 4](#_Toc12374061)

[Arduino nano 328P: 5](#_Toc12374062)

[Potenciómetro 1K: 5](#_Toc12374063)

[Servomotor SG90: 5](#_Toc12374064)

[Piezo Buzzer: 5](#_Toc12374065)

[Motivo Elección 6](#_Toc12374066)

[Desarrollo. 6](#_Toc12374067)

[Esquema 6](#_Toc12374068)

[Visualización 7](#_Toc12374069)

[Pruebas realizadas 7](#_Toc12374070)

[Conclusión 8](#_Toc12374071)

[Código 9](#_Toc12374072)

# Introducción

En el siguiente informe presentamos el proyecto de arduino de Lenguaje de programación II.

Este proyecyo será un control de accesos que usará un Arduino Nano quién controlara el acceso de las personas autorizadas ya registradas en su memoria.

# Usos y aplicaciones del RFID

En este apatado veremos que usos y aplicaciones se pueden realizar con un sensor RFID. Por otra parte, que sensores utilizaremos para nuestro proyecto.

## Aplicabilidad

La tecnología de identificación por radiofrecuencia (RFID), permite realizar lecturas de tarjetas, llaveros u otros artefactos como pulseras que contengan en su interior chips o estampillas con datos registrados, y hacerlo de manera cómoda y segura para los usuarios.

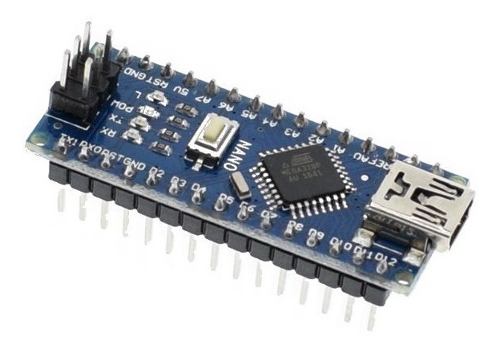
Para este ejemplo contamos con un proyecto que funcionando es capaz de ofrecer las siguientes características.

* Solo los usuarios que posean un llavero u otro chip registrado en el programa puede ingresar.
* Se puede realizar una lista de registros para habilitar el ingreso a múltiples usuarios.
* Habilitar y deshabilitar los registros en la memoria.

## Sensores Utilizados y/o Dispositivos Externos

LCD 1602 I2C: Se utilizan solo cuatro pines, La pantalla LCD es capaz de mostrar un máximo de 32 caracteres, en 16 columnas por 2 filas (16X2).

Lector RFID RC522: El módulo lector RFID-RC522 RF utiliza 3.3V como voltaje de alimentación, funciona a la frecuencia de 13.56 MHz y se controla a través del protocolo SPI.

Arduino nano 328P: es una placa de desarrollo de tamaño compacto, completo y compatible con protoboards, basada en el microcontrolador ATmega328P.

Potenciómetro 1K: Un potenciómetro y Arduino son una pareja muy útil en muchos skecth como por ejemplo, controlar la luminosidad de una pantalla **LCD**.

Servomotor SG90: Es un motor eléctrico pero con dos características especiales. Por un lado, nos permite **mantener una posición** que indiquemos, siempre que esté dentro del rango de operación del propio dispositivo. Por otro lado nos permite **controlar la velocidad de giro**, podemos hacer que antes de que se mueva a la siguiente posición espere un tiempo.

Piezo Buzzer: Un buzzer pasivo o un altavoz son dispositivos que permiten convertir una señal eléctrica en una onda de sonido.

## Motivo Elección

* Permiten soluciones de identificación y seguridad más cómodas y duraderas.
* Permite crear soluciones para el control de acceso físico a locales, clubs, fábricas, recintos, etc.

# Desarrollo.

En este apartado veremos el esquema que usaremos, las pruebas realizadas y el codigo utilizado para la correcta operación de nuestro proyecto.

## Esquema

Diseño del esquema que se usará en el proyecto.

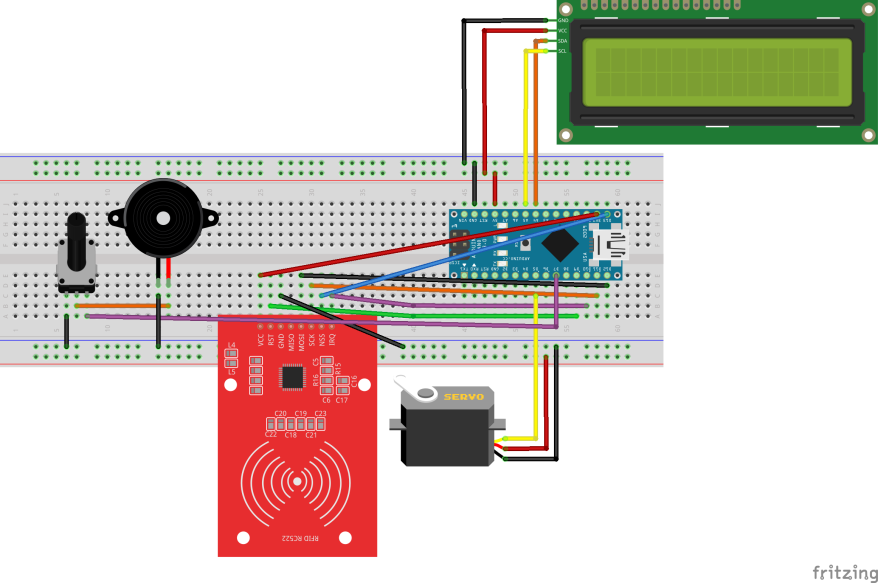


Ilustración 1

## Visualización

En la siguiente imagen podemos ver el sistema de control de acceso operativo.

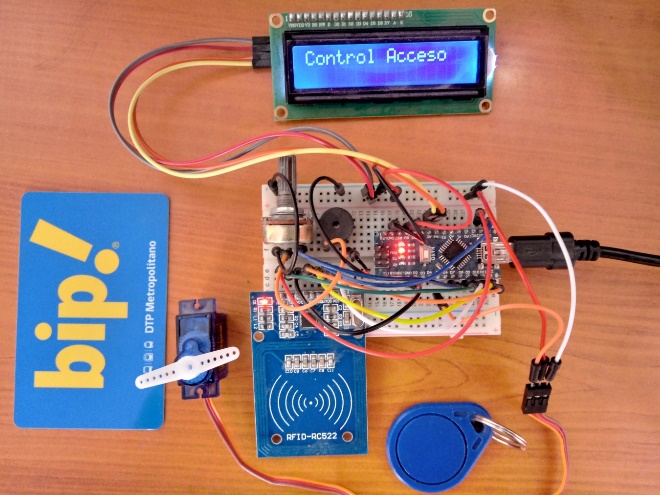


Ilustración 2

## Pruebas realizadas

Para este proyecto se contaba con 4 tarjetas y/o llaveros RFID:

Tarjetas Probadas

* Llavero Azul UID: 90 B2 1A 83
* Tarjeta Blanca UID: 50 A3 01 74
* Tarjeta Bip UID: C2 5E 05 76
* TNE UID: 02 A1 4F 5C

De los anteriores nombrados se registraron las 2 siguientes:

* TNE UID: 02 A1 4F 5C
* Llavero Azul UID: 90 B2 1A 83

Estos están registrados para permitir el acceso el cual emite un sonido característico y enciende el movimiento del servomotor las otras tarjetas muestran un mensaje y sonido correspondiente a un número desconocido.

# Conclusión

Una mejora para este proyecto sería el registro de acceso en algún archivo de texto o plantilla de cálculos, pero idealmente en algún sistema que gestione el acceso a través de una base de datos, y de esta forma llevar un registro actualizado de quien ingresa y el horario de ingreso y/o salida.

Además una mejor forma de realizar el cierre del mecanismo seria la implementación de un sensor en la puerta o torniquete para que al posicionar la puerta o torniquete como cerrado este active de forma automática el seguro (servomotor del proyecto) ya que actualmente este se cierra en un tiempo fijo (establecido en este proyecto como 5 segundos) restringiendo a la persona a ingresar es este tiempo establecido o reintentar el ingreso.

# Código

\* Uses MIFARE RFID card using RFID-RC522 reader

\* Uses MFRC522 - Library

\* -----------------------------------------------------------------------------------------

\* MFRC522 Arduino Arduino Arduino Arduino Arduino

\* Reader/PCD Uno/101 Mega Nano v3 Leonardo/Micro Pro Micro

\* Signal Pin Pin Pin Pin Pin Pin

\* -----------------------------------------------------------------------------------------

\* RST/Reset RST 9 5 D9 RESET/ICSP-5 RST

\* SPI SS SDA(SS) 10 53 D10 10 10

\* SPI MOSI MOSI 11 / ICSP-4 51 D11 ICSP-4 16

\* SPI MISO MISO 12 / ICSP-1 50 D12 ICSP-1 14

\* SPI SCK SCK 13 / ICSP-3 52 D13 ICSP-3 15

\*

\* --------------------------

\* Conexiones I2C Arduino Uno

\* --------------------------

\* SDA to A4

\* SCL to A5

\* VCC to 5volt

\* GND to GND

\*

\*/

// Librerias RC522

#include <SPI.h>

#include <MFRC522.h>

#include <Wire.h>

// Libreria I2C LCD

#include <LiquidCrystal\_I2C.h>

// Libreria Servo

#include <Servo.h>

// Objeto servomotor

Servo servomotor;

//Crear el objeto lcd dirección 0x27 y 16 columnas x 2 filas

LiquidCrystal\_I2C lcd(0x27,16,2);

// Pines Lector RC522

#define RST\_PIN 9

#define SS\_PIN 10

MFRC522 mfrc522(SS\_PIN, RST\_PIN);

// Tiempo de acceso (Milisegundo)

const int tiempo\_acceso = 4500;

// Pin Piezo Buzzer

const int buzzerPin = 7;

// Pin Servomotor

const int servoPin = 5;

/\* Registra el UID aqui! \*/

// Primer UID NEW\_UID que se completara con datos de tarjeta a leer

#define NEW\_UID {0x00, 0x00, 0x00, 0x00}

// UID de Tarjetas registradas

byte registro[2][4] = {

{0x90, 0xB2, 0x1A, 0x83} , // Numerico : 90B21A83

{0x02, 0xA1, 0x4F, 0x5C}

};

/\*

|--------------

| VOIDs ANEXOS

|--------------

\*/

// Sonido de acceso denegado

void sonidoError(){

// Encender buzzer

digitalWrite(buzzerPin,HIGH);

// Esperar para producir sonido

delay(100);

// Epagar buzzer

digitalWrite(buzzerPin,LOW);

// Esperar para producir silencio

delay(100);

digitalWrite(buzzerPin,HIGH);

delay(100);

digitalWrite(buzzerPin,LOW);

delay(100);

digitalWrite(buzzerPin,HIGH);

delay(100);

digitalWrite(buzzerPin,LOW);

}

// Sonido de acceso concedido

void sonidoCorrecto(){

// Producir sonido

digitalWrite(buzzerPin,HIGH);

delay(300);

digitalWrite(buzzerPin,LOW);

}

// Dejar pantalla LCD en blanco luego de mostrar mensaje

// Con mensaje "Control Acceso"

void limpiarLCD(){

// Limpiar LCD

lcd.clear();

// Primero cuadrado primera linea de la pantalla

lcd.setCursor(0,0);

// Mostrar mensaje Control Acceso

lcd.print("Control Acceso");

}

long int concatHex(byte uid[]){

long unsigned int uid\_num = uid[0];

uid\_num = uid\_num\*256 + uid[1];

uid\_num = uid\_num\*256 + uid[2];

uid\_num = uid\_num\*256 + uid[3];

return uid\_num;

}

/\*

|--------------

| VOID SETUP

|--------------

\*/

// Iniciacion de componentes y variables (al encender arduino)

void setup() {

/\* Monitor Serial \*/

Serial.begin(9600);

/\* Piezo Buzzer \*/

pinMode(buzzerPin, OUTPUT);

/\* RFID RC522 \*/

SPI.begin();

mfrc522.PCD\_Init();

// Mostrar mensaje en Serial monitor

Serial.println(F("Lector Datos MIFARE PICC:"));

Serial.print("Cantidad registros: ");

Serial.println(sizeof(registro)/4);

// Inicializar el LCD I2C

lcd.init();

//Encender la luz de fondo.

lcd.backlight();

lcd.print("Control Acceso");

// Inicializar Servo

servomotor.attach(servoPin);

servomotor.write(0);

}

/\*

|--------------

| VOID LOOP

|--------------

\*/

void loop() {

// Cambiar cursor primer caracter segunda linea

lcd.setCursor(0,1);

/\* Preparar Clave Sector, por defecto en fabrica FFFFFFFFFFFF \*/

MFRC522::MIFARE\_Key key;

// LLave defecto FFFFFFFFFFFF

for (byte i = 0; i < 6; i++) {

key.keyByte[i] = 0xFF;

}

// Primer UID (Se completara con datos de tarjeta a leer)

byte uid[] = NEW\_UID;

// Segundo UID (Registrada al inico para dar acceso - llave correcta)

//byte com\_uid[] = COM\_UID;

// Si se presenta una nueva tarjeta

if ( ! mfrc522.PICC\_IsNewCardPresent() || ! mfrc522.PICC\_ReadCardSerial() ) {

delay(50);

return;

}

// Mostrar UID tarjeta que se presento

Serial.print(F("Card UID:"));

// Recorrer el largo del uid para asignarlo en variable uid[]

for (byte i = 0; i < mfrc522.uid.size; i++) {

// Imprimir en el serial monitor el UID de la tarjeta leida

Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");

Serial.print(mfrc522.uid.uidByte[i], HEX);

// Almacenar UID de tarjeta leida en variable uid[]

uid[i] = mfrc522.uid.uidByte[i];

}

// Salto de linea

Serial.println();

// Compactar valor uid bytes en numerico tarjeta leida

long unsigned int leido = concatHex(uid);

// Variables para lectura de tarjetas

byte card[4]; // Tarjeta para recorrer registros (Hexadecimal)

long unsigned int number; // Tarjeta para recorrer registros (Decimal)

bool encontrado = false; // Determina si se encontro tarjeta

// Mostrar tarjeta leida

Serial.print("Tarjeta leida: ");

Serial.println(leido, HEX);

for(int r = 0; r < (sizeof(registro)/4); r++){

encontrado = false;

card[0] = registro[r][0];

card[1] = registro[r][1];

card[2] = registro[r][2];

card[3] = registro[r][3];

number = concatHex(card);

Serial.print("Comparando: ");

Serial.print(number);

Serial.print(" : ");

Serial.println(leido);

// Compactar valor uid bytes en entero tarjeta registrada

if (number == leido){

encontrado = true;

Serial.print("Encontrado en ");

Serial.println(r);

// Sonido de acceso correcto

sonidoCorrecto();

// Mostrar mensaje de acceso concedido

lcd.print("Correcta!");

// Conceder acceso

servomotor.write(180);

// Tiempo del acceso

delay(tiempo\_acceso);

limpiarLCD();

// Cerrar acceso

lcd.setCursor(0,1);

lcd.print("Cerrando...");

delay(500);

servomotor.write(0);

limpiarLCD();

break;

}

}

// Emitir mensaje al no encontrar UID leido

if(encontrado == false){

// Sonido de error

sonidoError();

// Mensaje de error

lcd.print("No encontrada!");

}

// Pausar programa

delay(1000);

// Limpiar LCD

limpiarLCD();

}