

Uso de tags RFID con Arduino



UNIVERSIDAD
DE GRANADA

Marta Díaz Artigot

Índice

Índice.....	1
1. ¿Qué es la tecnología RFID?.....	3
1.2. Cómo funciona.....	3
1.3. Clasificación.....	3
2. Implementación con Arduino.....	4
2.1. Módulo RFID RC522.....	4
2.2. Librería RFID para MFRC522.....	6
2.3. Programa lector de etiquetas RFID.....	6
2.4. Estructura de de la memoria de una etiqueta RFID.....	8
2.5. Programa lector de etiquetas RFID con tarjetas encriptadas.....	9
2.6. Programa lector de etiquetas RFID restringidas.....	10

1. ¿Qué es la tecnología RFID?

La tecnología RFID o identificación por radiofrecuencia (Radio Frequency Identification) es un sistema de almacenamiento y recuperación de datos remotos capaz de identificar objetos a través de un identificador único gracias a ondas de radio.

El sistema RFID tiene dos componentes principales: una etiqueta RFID (transpondedor) y un lector RFID (transceptor).

Las etiquetas (tags) son unos dispositivos pequeños, similares a una pegatina, que contienen antenas para recibir y responder a peticiones por radiofrecuencia de un emisor-receptor RFID.

Hay etiquetas de solo lectura, una vez generadas ya no se puede modificar el UID, y etiquetas de lectura y escritura, que pueden cambiar su información gracias a los lectores RFID.

Las etiquetas pasivas no necesitan alimentación eléctrica interna, funcionan con inducción y tienen poco alcance, mientras que las activas sí lo requieren, haciendo que tengan un alcance de varios metros de distancia.

Una de las ventajas de esta tecnología frente a otras más extendidas, como los códigos de barras, es que no se requiere visión directa entre emisor y receptor.

1.2. Cómo funciona

Para leer la información codificada en una etiqueta pasiva RFID, se coloca cerca del lector RFID que genera un campo electromagnético que hace que los electrones se muevan a través de la antena de la etiqueta y posteriormente alimenten el chip.

El chip alimentado es capaz de enviar la información almacenada en la etiqueta RFID a través de la radiofrecuencia (retrodispersión).

La retrodispersión es detectada e interpretada por el lector RFID que luego envía los datos a un ordenador o un microcontrolador como el que tiene Arduino.

1.3. Clasificación

Los lectores y etiquetas RFID deben estar sintonizados a la misma frecuencia para poder comunicarse. Dependiendo de la frecuencia que utilicen hay cuatro tipos de sistemas:

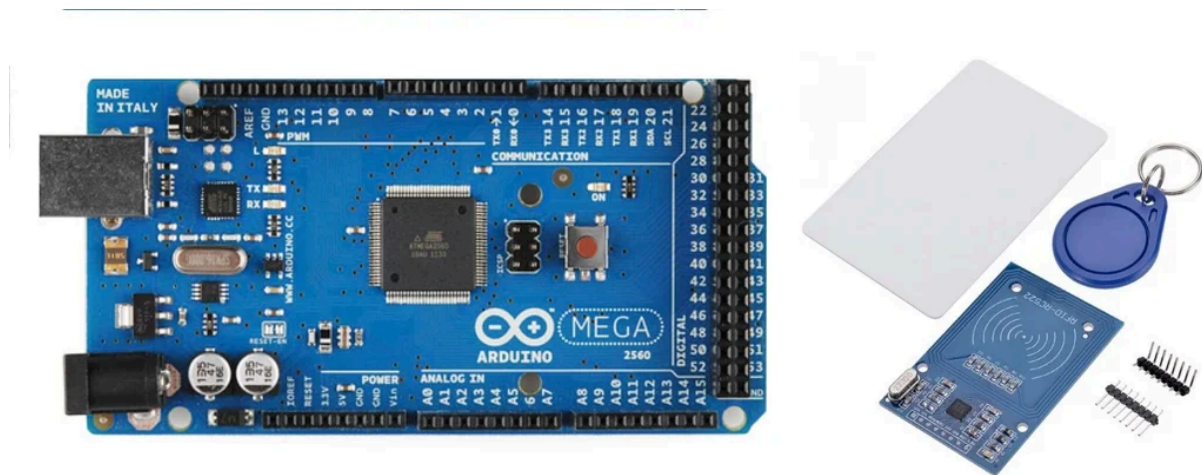
- Baja frecuencia (LF): 125-134 KHz. Alcance 10 cm. Ej: control de acceso.
- Alta frecuencia (HF): 13,56 MHz. Alcance 30 cm. Ej: pagos con protocolo NFC.
- Frecuencia ultra alta (UHF): 433, 860, 960 MHz. Alcance más de 100 m. Ej: peajes.
- Microondas: 2.45 GHz.

La frecuencia UHF no puede ser utilizada de forma global, ya que no hay un único estándar global, mientras que las etiquetas de LF y HF se pueden utilizar de forma global sin necesidad de licencia.

2. Implementación con Arduino

Para implementar un lector de etiquetas RFID con arduino necesitaremos: un Arduino, un lector de etiquetas RFID con alguna etiqueta RFID, cables y la librería para manejar RFID.

En mi caso utilizaré un Arduino Mega2560, un módulo RFID RC522 como lector y como etiquetas una en formato tarjeta y otra en formato llavero.

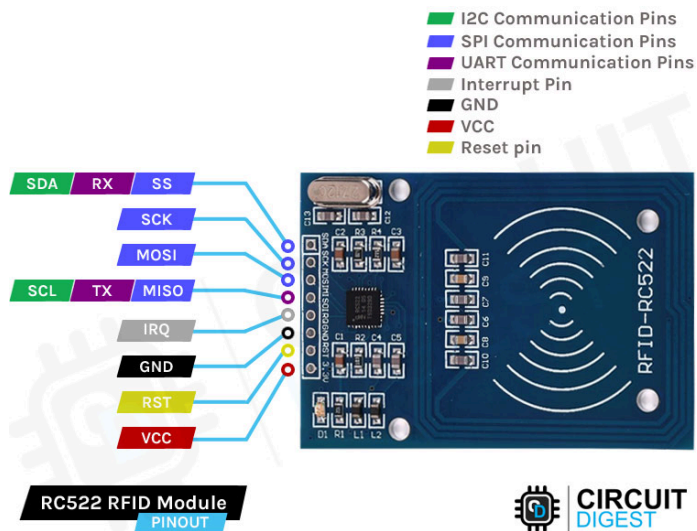


2.1. Módulo RFID RC522

Este módulo sirve tanto para leer como para escribir etiquetas RFID. En teoría tiene un alcance máximo de 35 cm pero por encima de 5 cm puede que el sistema no funcione correctamente.

A pesar de ser alimentado con 3,3V, los niveles lógicos son compatibles con 5V por lo que es compatible con cualquier Arduino o microcontrolador que trabaje a 5V.

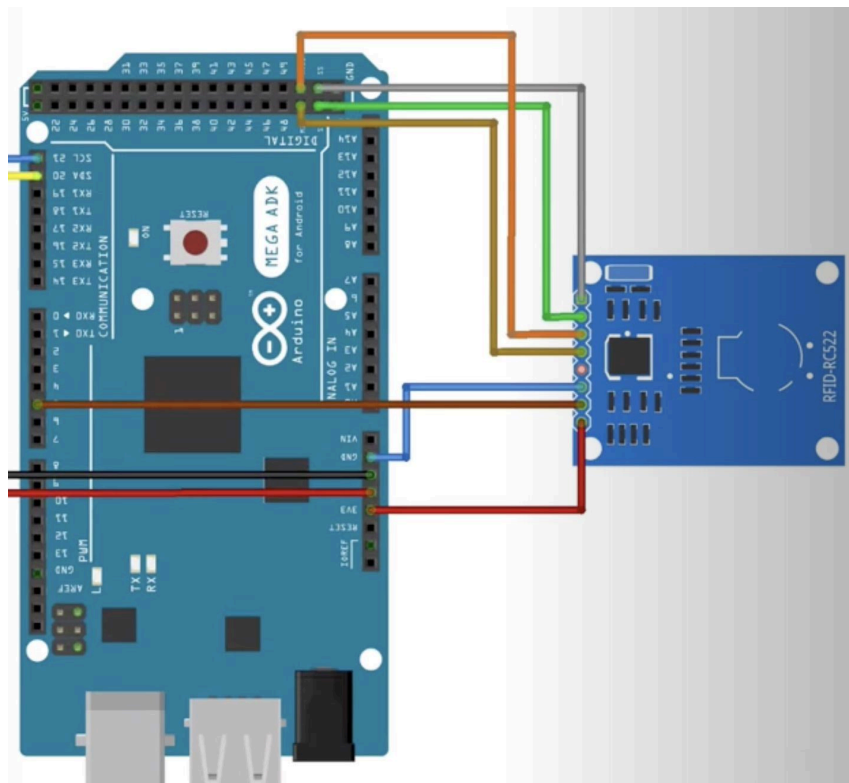
En las siguientes imágenes podemos ver un esquema en el que aparecen los pines que tiene el lector, el layout que se suele utilizar para conectarlo y como se haría esta conexión a un Arduino Mega.



* Typical pin layout used:

	MFR522	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

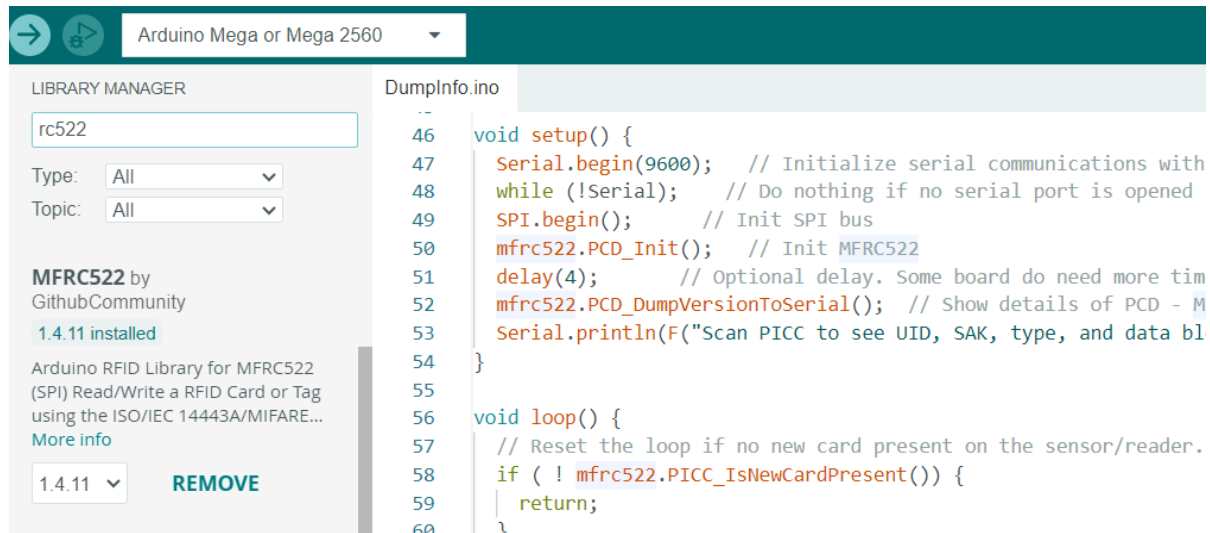
* More pin layouts for other boards can be found here: <https://github.com/miguelbalboa/rfid#pin-layout>



2.2. Librería RFID para MFRC522

Para poder utilizar el lector en Arduino tenemos que descargar una librería que nos facilite el trabajo, en este caso, la librería de Miguel Balboa que podemos encontrar en GitHub.

Para instalarla, la busqué en el buscador de librerías del IDE de Arduino.



2.3. Programa lector de etiquetas RFID

Para leer las etiquetas utilizamos este programa sencillo de ejemplo:

```
#include <SPI.h>
#include <MFRC522.h>

#define RST_PIN      5           // Configurable, see typical pin layout above
#define SS_PIN       53          // Configurable, see typical pin layout above

MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance

void setup() {
  Serial.begin(9600); // Initialize serial communications with the PC
  while (!Serial);    // Do nothing if no serial port is opened (added for Arduinos based on ATMEGA32U4)
  SPI.begin();        // Init SPI bus
  mfrc522.PCD_Init(); // Init MFRC522
  delay(4);           // Optional delay. Some board do need more time after init to be ready, see Readme
  mfrc522.PCD_DumpVersionToSerial(); // Show details of PCD - MFRC522 Card Reader details
  Serial.println(F("Scan PICC to see UID, SAK, type, and data blocks..."));
}

void loop() {
  // Reset the loop if no new card present on the sensor/reader. This saves the entire process when idle.
  if ( ! mfrc522.PICC_IsNewCardPresent()) {
    return;
  }

  // Select one of the cards
  if ( ! mfrc522.PICC_ReadCardSerial()) {
    return;
  }

  // Dump debug info about the card; PICC_HaltA() is automatically called
  mfrc522.PICC_DumpToSerial(&(mfrc522.uid));
}
```

Al acercar las etiquetas al lector, vemos que aparece la siguiente información por pantalla:

```

Output  Serial Monitor ×
Message (Enter to send message to 'Arduino Mega or Mega 2560' on 'COM3')

Firmware Version: 0x88 = (clone)
Scan PICC to see UID, SAK, type, and data blocks...
Card UID: A9 75 AD 99
Card SAK: 08
PICC type: MIFARE 1KB
Sector Block  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15  AccessBits
15      63  00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
      62  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      61  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      60  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
14      59  00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
      58  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      57  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      56  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
13      55  00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
      54  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      53  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      52  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
12      51  00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
      50  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      49  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      48  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]

.
.
.

0      3   00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
      2   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      1   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      0   A9 75 AD 99 E8 08 04 00 62 63 64 65 66 67 68 69 [ 0 0 0 ]

```

De esta información podemos obtener los siguientes datos:

- Todos los bloques tienen las mismas claves de cifrado.
- El identificador de la tarjeta es: A9 75 AD 99.
- La clave A es: 00 00 00 00 00 00.
- La clave B es: FF FF FF FF FF FF.
- Los permisos para acceder al sector son: FF 07 80.

Para ver cómo los hemos obtenido, veamos cómo se estructura la memoria en una etiqueta RFID.

2.4. Estructura de de la memoria de una etiqueta RFID

Las tarjetas o etiquetas RFID que utiliza el lector RC522 tienen una memoria de 1K organizada en 16 sectores (del 0 al 15). Cada sector se divide en 4 bloques (del 0 al 3). En cada bloque se pueden almacenar 16 bytes de datos (del 0 al 15).

COM4

Enviar

Firmware Version: 0x91 = v1.0

Scan PICC to see UID, SAK, type, and data blocks...

Card UID: 61 08 19 34

Card SAK: 08

PICC type: MIFARE 1KB

Sector	Block	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	AccessBits
15	63	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	62	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	61	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
14	59	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	58	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	57	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	56	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
13	55	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]

☒ Autoscroll

☐ Mostrar marca temporal

Sin ajuste de línea

9600 baudio

Limpiar salida

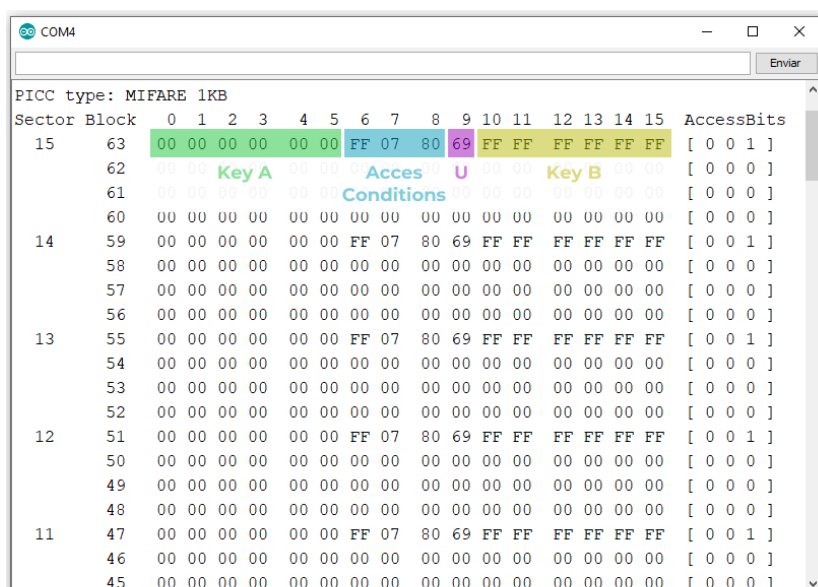
Sectores

Bloques

Datos

De estos 4 bloques de cada sector, solo se pueden utilizar 3 bloques para almacenar memoria, los 16 bytes del último bloque se utilizan para guardar las claves de cifrado y los permisos para acceder a ese sector. Este bloque se conoce como trailer y coincide con el último bloque de cada sector.

Los 16 bytes que pertenecen al trailer se dividen en dos claves para cifrar el contenido (Key A y Key B) que ocupan 6 bytes cada una. 3 bytes para establecer las condiciones y permisos de acceso al sector (Access Conditions). Y por último un byte restante (U) que no se utiliza para nada.



COM4

PICC type: MIFARE 1KB

Sector	Block	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	AccessBits
15	63	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	62	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	61	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
14	59	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	58	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	57	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	56	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
13	55	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	54	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	53	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	52	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
12	51	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	49	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	48	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
11	47	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	46	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	45	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]

☒ Autoscroll ☐ Mostrar marca temporal Sin ajuste de línea 9600 baudio Limpiar salida

El programa anterior es capaz de leer las etiquetas RFID de las que conoce la clave de encriptación, sin embargo, cuando acercamos una tarjeta encriptada de la que desconocemos la clave, como puede ser la tarjeta del autobús, ocurre lo siguiente:

```

Card UID: 7A ED C8 B8
Card SAK: 08
PICC type: MIFARE 1KB
Sector Block  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15  AccessBits
15      63  PCD_Authenticate() failed: Timeout in communication.
14      59  PCD_Authenticate() failed: Timeout in communication.
13      55  PCD_Authenticate() failed: Timeout in communication.
12      51  PCD_Authenticate() failed: Timeout in communication.
11      47  PCD_Authenticate() failed: Timeout in communication.
10      43  PCD_Authenticate() failed: Timeout in communication.
9       39  PCD_Authenticate() failed: Timeout in communication.
8       35  PCD_Authenticate() failed: Timeout in communication.
7       31  PCD_Authenticate() failed: Timeout in communication.
6       27  PCD_Authenticate() failed: Timeout in communication.
5       23  PCD_Authenticate() failed: Timeout in communication.
4       19  PCD_Authenticate() failed: Timeout in communication.
3       15  PCD_Authenticate() failed: Timeout in communication.
2       11  PCD_Authenticate() failed: Timeout in communication.
1        7  PCD_Authenticate() failed: Timeout in communication.
0        3  PCD_Authenticate() failed: Timeout in communication.

```

2.6. Programa lector de etiquetas RFID restringidas

También podemos crear un programa que solo lea información de etiquetas permitidas. Para ello en el código creamos una lista con los ID permitidos, en el bucle comprobamos si el ID de la tarjeta leída está en la lista. En caso de estarlo, se muestra la información por pantalla, de lo contrario, muestra un mensaje de acceso denegado.

```

1  #include <SPI.h>
2  #include <MFRC522.h>
3
4  #define RST_PIN 5
5  #define SS_PIN 53
6
7  MFRC522 mfrc522(SS_PIN, RST_PIN); // Crear instancia MFRC522
8
9  // Lista de UID permitidos
10 byte allowedUIDs[][4] = {
11     {0x27, 0xA4, 0x7F, 0x60}, // UID 1: 27 A4 7F 60
12     //{0xA9, 0x75, 0xAD, 0x99}, // UID 2: A9 75 AD 99 (para acceso denegado)
13 };
14
15 void setup() {
16     Serial.begin(9600); // Inicializar comunicación serial con el PC
17     while (!Serial);    // No hacer nada si no se abre el puerto serie
18
19     SPI.begin();        // Iniciar bus SPI
20     mfrc522.PCD_Init(); // Inicializar MFRC522
21
22     delay(4);           // Retardo opcional para algunas placas
23     mfrc522.PCD_DumpVersionToSerial(); // (Opcional) Mostrar detalles del lector
24
25     Serial.println(F("Escanee PICC para verificar el acceso..."));
26 }
27
28 void loop() {
29     // Restablecer el bucle si no hay una nueva tarjeta presente
30     if (!mfrc522.PICC_IsNewCardPresent()) {
31         return;
32     }

```

```

34 // Seleccionar una tarjeta
35 if (!mfrc522.PICC_ReadCardSerial()) {
36     return;
37 }
38
39 // Verificar si el UID de la tarjeta está en la lista de permitidos
40 bool isAllowed = false;
41 for (int i = 0; i < sizeof(allowedUIDs) / sizeof(allowedUIDs[0]); i++) {
42     if (memcmp(mfrc522.uid.uidByte, allowedUIDs[i], mfrc522.uid.size) == 0) {
43         isAllowed = true;
44         break;
45     }
46 }
47
48 // Procesamiento según el acceso permitido
49 if (isAllowed) {
50     // UID permitido: proceder a la lectura o autenticación
51     mfrc522.PICC_DumpToSerial(&(mfrc522.uid));
52 } else {
53     // UID no permitido: denegar acceso
54     Serial.print(F("Acceso denegado para UID: "));
55     for (byte i = 0; i < mfrc522.uid.size; i++) {
56         Serial.print(mfrc522.uid.uidByte[i], HEX);
57         if (i < mfrc522.uid.size - 1) {
58             Serial.print(" ");
59         }
60     }
61     Serial.println();
62 }
63 }
64 }

```

Resultados:

Escanee PICC para verificar el acceso...

Card UID: 27 A4 7F 60

Card SAK: 08

PICC type: MIFARE 1KB

Sector	Block	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	AccessBits
15	63	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	62	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	61	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
14	59	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF	[0 0 1]
	58	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	57	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]
	56	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	[0 0 0]

Acceso denegado para UID: A9 75 AD 99

Acceso denegado para UID: A9 75 AD 99

Acceso denegado para UID: A9 75 AD 99

Acceso denegado para UID: A9 75 AD 99