

RFID Montageanleitung

Mit diesem Projekt wird ein RFID-Kartenleser (RFID=radio frequency identification) für eine Zugangskontrolle eingesetzt. Zunächst muss der RFID-Transponder oder die RFID-Karte ausgelesen werden. Der ausgelesene "Code" wird im Hauptprogramm für den Zugang verwendet und z.B. ein Türschloss geöffnet.

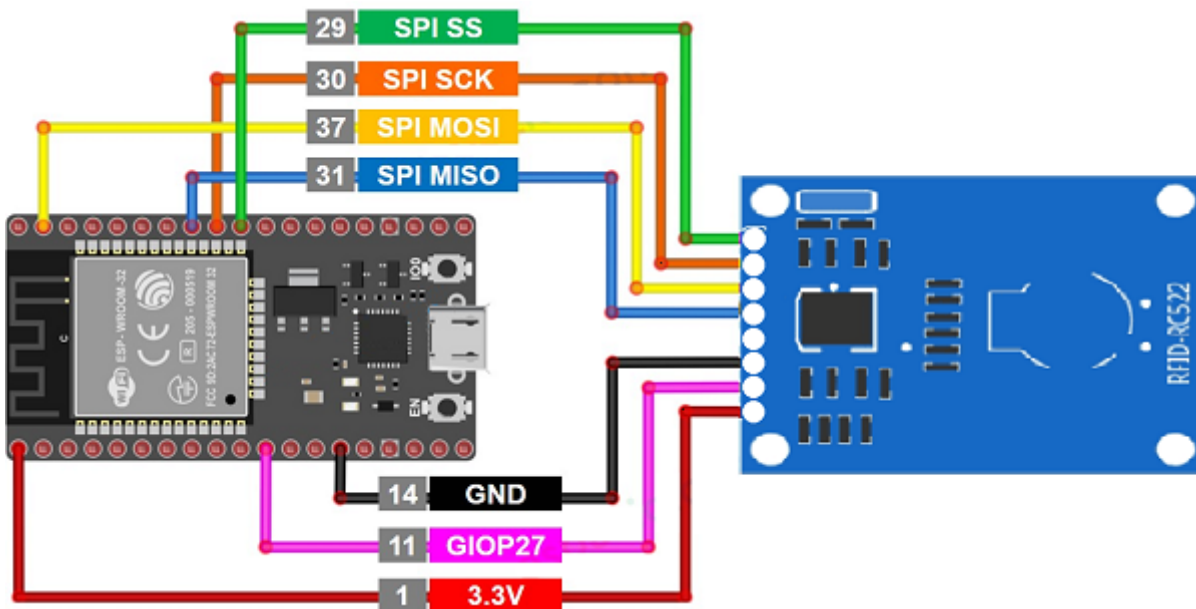
Die Grüne LED zeigt blinkend die Verwendung des korrekten Transponder an, die rote LED zeigt blinkend einen ungültigen Transponder an. Ein Ausgang ist für einen Relais-Anschluss für ein Türschloss vorgesehen.

Hardware

Die Stückliste für die RFID-Zugangskontrolle:

- 1 x ESP32 oder Arduino NANO
- 1 x RFID-RC522 Modul
- 1 x Transponder mit Schlüsselring
- 1 x Transponder im Kartenformat
- 1 x Relais für den Türöffner (optional)
- 1 x rote LED mit 100R Vorwiderstand
- 1 x grüne LED mit 100R Vorwiderstand
- 1 x Relais für den Türöffner (optional)
- Kleinmaterial, Schaltdraht

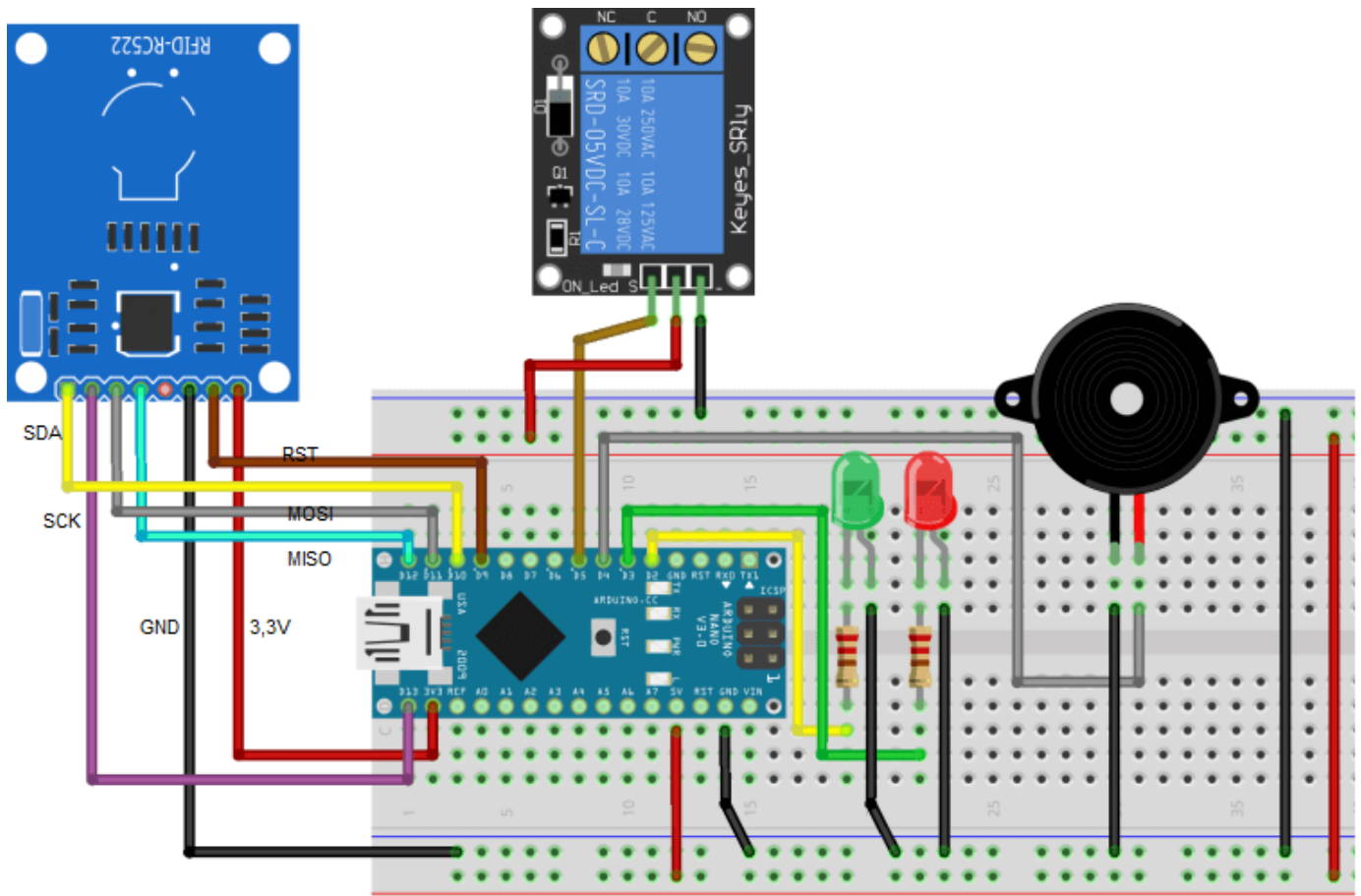
Verdrahtung ESP32



ESP32

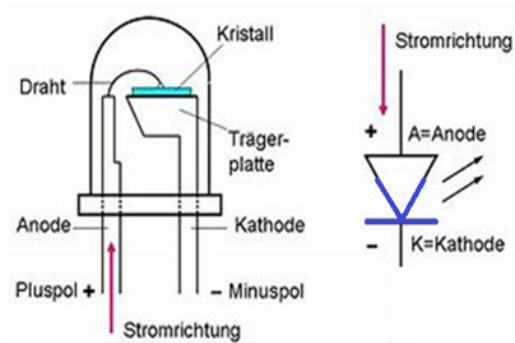
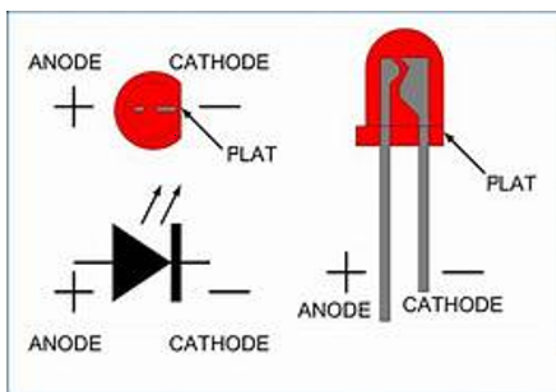
- ESP GPIO 32 -> rote LED
- ESP GPIO 33 -> grüne LED
- ESP GPIO 25 -> Relais

Verdrahtung NANO

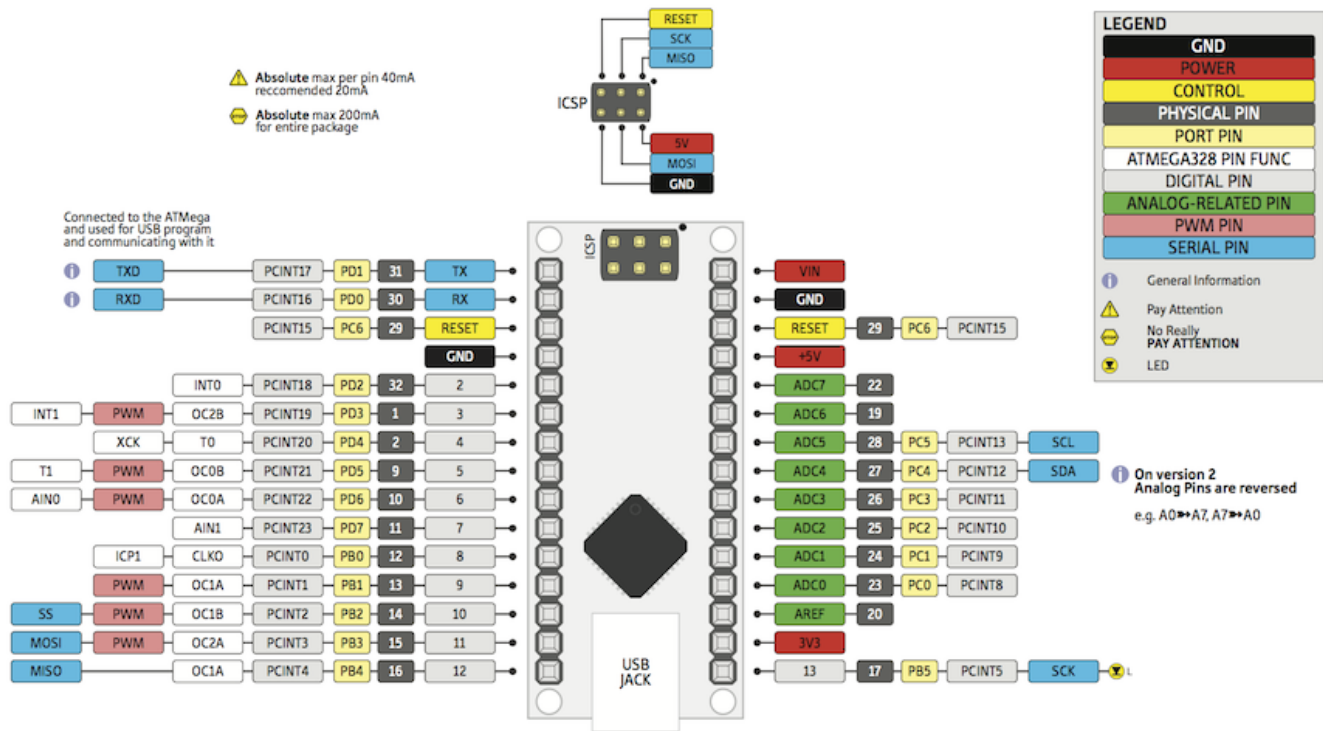


Die LED

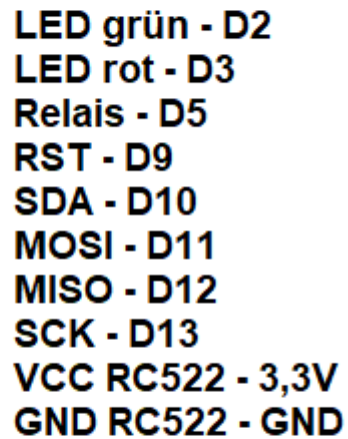
Die LED:



PINout NANO



NANO Shield



```

/*****
PROGRAMMINFO
*****/

```

Funktion: RFID-Transponder auslesen

Version: 21.06.2022

Board: ESP32 Dev

Libraries:

<https://github.com/espressif/arduino-esp32/tree/master/libraries>

V3

C++ Arduino IDE V1.8.19

Einstellungen:

https://dl.espressif.com/dl/package_esp32_index.json

http://dan.drown.org/stm32duino/package_STM32duino_index.json

http://arduino.esp8266.com/stable/package_esp8266com_index.json

*****/

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

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

```
#define SS_PIN 5 // ESP32 pin GIOP5
```

```
#define RST_PIN 27 // ESP32 pin GIOP27
```

```
MFRC522 rfid(SS_PIN, RST_PIN);
```

```
void setup() {
```

```
  Serial.begin(9600);
```

```
  SPI.begin(); // init SPI bus
```

```
  rfid.PCD_Init(); // init MFRC522
```

```
  Serial.println("Tap an RFID/NFC tag on the RFID-RC522 reader");
```

```
}
```

```
void loop() {
```

```
  if (rfid.PICC_IsNewCardPresent()) { // new tag is available
```

```
    if (rfid.PICC_ReadCardSerial()) { // NUID has been readed
```

```
      MFRC522::PICC_Type piccType = rfid.PICC_GetType(rfid.uid.sak);
```

```
      Serial.print("RFID/NFC Tag Type: ");
```

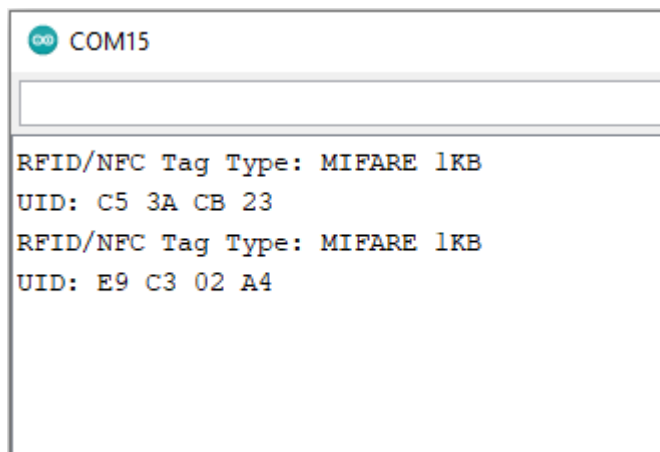
```
      Serial.println(rfid.PICC_GetTypeName(piccType));
```

```
// print UID in Serial Monitor in the hex format
Serial.print("UID:");
for (int i = 0; i < rfid.uid.size; i++) {
    Serial.print(rfid.uid.uidByte[i] < 0x10 ? " 0" : " ");
    Serial.print(rfid.uid.uidByte[i], HEX);
}
Serial.println();

rfid.PICC_HaltA(); // halt PICC
rfid.PCD_StopCrypto1(); // stop encryption on PCD
}
}
}
```

RAW code

RFID-Code mit dem Seriellen Monitor auslesen



C++ Code: Hauptprogramm RFID-Zugangskontrolle

```

/*****
PROGRAMMINFO
*****/

Funktion: RFID Zugangskontrolle
*****/

Version: 21.06.2022
*****/

Board: ESP32 Dev
*****/

Libraries:
https://github.com/miguelbalboa/rfid

*****/

C++ Arduino IDE V1.8.19
*****/

Einstellungen:
https://dl.espressif.com/dl/package\_esp32\_index.json
http://dan.drown.org/stm32duino/package\_STM32duino\_index.json
http://arduino.esp8266.com/stable/package\_esp8266com\_index.json
*****/

//einbinden der Bibliotheken fuer das
//ansteuern des MFRC522 Moduls
#include <SPI.h>
#include <MFRC522.h>

//definieren der Pins RST & SDA fuer den ESP32
#define RST_PIN 27
#define SS_PIN 5

#define ledRot 32
#define ledGruen 33
#define oc_PIN 25 //Open Close PIN

//erzeugen einer Objektinstanz
MFRC522 mfrc522(SS_PIN, RST_PIN);

//Variable zum speichern der bereits gelesenen RFID-ID
String lastRfid = "";

//Anzahl der zulässigen RFID-IDs im Array

```



```

const int NUM_RFIDS = 1;
//Array mit RFID-IDs welche zulässig sind
String rfids[NUM_RFIDS] = {" E9 C3 02 A4"};

void setup() {
  //beginn der seriellen Kommunikation mit 115200 Baud
  Serial.begin(115200);
  //eine kleine Pause von 50ms.
  delay(50);

  pinMode(ledRot, OUTPUT);
  pinMode(ledGruen, OUTPUT);
  pinMode(oc_PIN, OUTPUT);
  digitalWrite(oc_PIN, LOW);

  //begin der SPI Kommunikation
  SPI.begin();
  //initialisieren der Kommunikation mit dem RFID Modul
  mfrc522.PCD_Init();
}

void loop() {

  //Wenn keine neue Karte vorgehalten wurde oder die serielle Kommunikation
  //nicht gegeben ist, dann...
  if ( !mfrc522.PICC_IsNewCardPresent()) {
    //Serial.println("!PICC_IsNewCardPresent");
    return;
  }

  if (!mfrc522.PICC_ReadCardSerial()) {
    //Serial.println("!PICC_ReadCardSerial");
    return;
  }

  String newRfidId = "";
  for (byte i = 0; i < mfrc522.uid.size; i++) {
    newRfidId.concat(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
    newRfidId.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
}

```

```

}

//alle Buchstaben in Großbuchstaben umwandeln
newRfidId.toUpperCase();

//Wenn die neue gelesene RFID-ID ungleich der bereits zuvor gelesenen ist,
//dann soll diese auf der seriellen Schnittstelle ausgegeben werden.
if (!newRfidId.equals(lastRfid)) {
    //überschreiben der alten ID mit der neuen
    lastRfid = newRfidId;
}

bool zugangOk = false;
//prüfen ob die gelesene RFID-ID im Array mit bereits gespeicherten IDs vorhanden ist
for (int i = 0; i < NUM_RFIDS; i++) {
    //wenn die ID an der Stelle "i" im Array "rfids" mit dem gelesenen übereinstimmt, dann
    if (rfids[i].equals(newRfidId)) {
        zugangOk = true;
        //Schleife verlassen
        break;
    }
}

//Wenn die Variable "zugangOk" auf True ist, dann...
if (zugangOk) {
    digitalWrite(oc_PIN, HIGH); //Türschloss öffnen
    Serial.println("Türschloss oeffnen");
    delay(500);
    digitalWrite(oc_PIN, LOW);
    Serial.println("Türschloss schliessen");

    blinkLed(ledGruen);
    Serial.println("RFID-ID [" + newRfidId + "] OK!, Zugang wird gewährt");

} else {
    //Wenn nicht dann...
    blinkLed(ledRot);
    Serial.println("RFID-ID [" + newRfidId + "] nicht OK!, Zugang wird nicht gewährt");
}

```

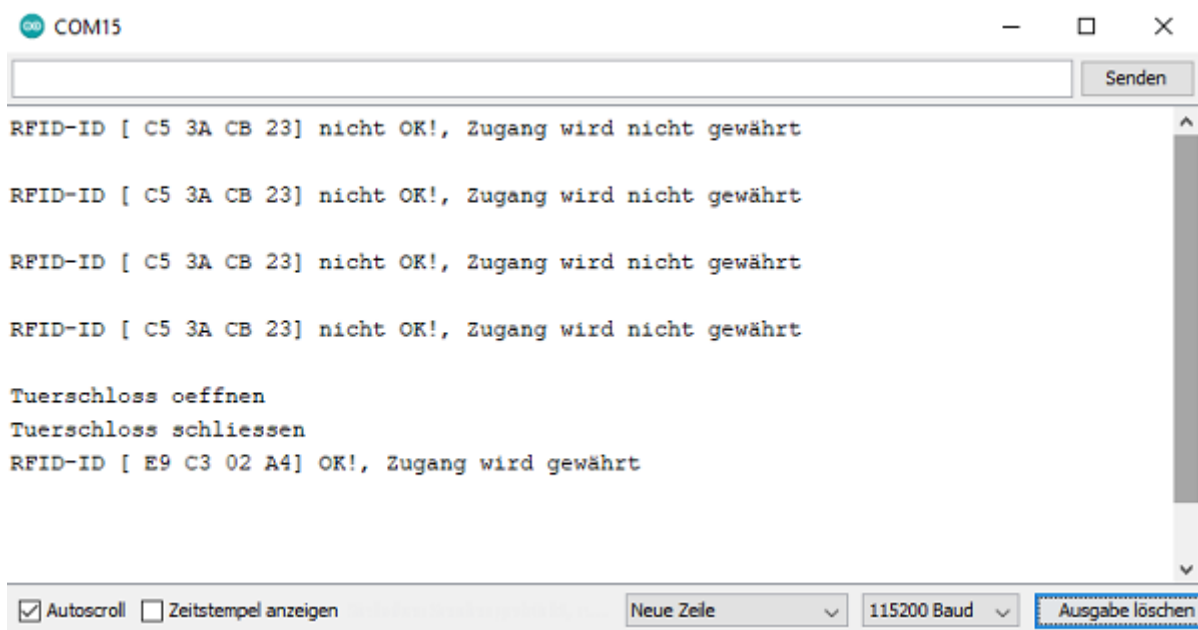
```
}

Serial.println();
}

//Blinken einer LED am Pin "pin"
void blinkLed(int pin) {
  //Schleife von 0 bis 5
  for (int a = 0; a < 5; a++) {
    //LED aktivieren
    digitalWrite(pin, HIGH);
    //eine Pause von 125 ms.
    delay(125);
    //LED deaktivieren
    digitalWrite(pin, LOW);
    //eine Pause von 125 ms.
    delay(125);
  }
}
```

RAW code

RFID-Zugansinformationen mit dem Seriellen Monitor auslesen



C++ Code: Hauptprogramm RFID-Zugangskontrolle für drei Transponder

```
/******  
  
                PROGRAMMINFO  
  
*****  
  
Funktion: RFID 3 Transponder mit LED-Anzeige  
  
*****  
  
Version: 21.06.2022  
  
*****  
  
Board: ESP32 Dev Module  
  
*****  
  
Libraries:  
  
https://github.com/espressif/arduino-esp32/tree/master/libraries  
  
*****  
  
C++ Arduino IDE V1.8.19  
  
*****  
  
Einstellungen:  
  
https://dl.espressif.com/dl/package\_esp32\_index.json  
http://dan.drown.org/stm32duino/package\_STM32duino\_index.json  
http://arduino.esp8266.com/stable/package\_esp8266com\_index.json  
*****/  
  
#include <SPI.h>  
  
#include <MFRC522.h>  
  
  
//definieren der Pins RST & SDA für den ESP32  
#define RST_PIN    27  
#define SS_PIN     5  
  
  
#define led1 32  
#define led2 33  
#define led3 25  
  
  
bool led1_status = false;  
bool led2_status = false;
```

```
bool led3_status = false;

MFRC522 rfid(SS_PIN, RST_PIN);

String rfid_nuids[] = {"e9c3 02a4", "c53acb23", "f73f9339"};

void setup() {
  Serial.begin(115200);
  SPI.begin();
  rfid.PCD_Init();

  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);

  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  digitalWrite(led3, LOW);
}

void loop() {
  if ( !rfid.PICC_IsNewCardPresent() || !rfid.PICC_ReadCardSerial()) {
    return;
  }

  String nuid = readHex(rfid.uid.uidByte, rfid.uid.size);
  Serial.println(nuid);

  if (nuid == rfid_nuids[0]) {
    Serial.println("Zugang Karte 1");
    led1_status = !led1_status;
    digitalWrite(led1, led1_status ? HIGH : LOW);
  } else if (nuid == rfid_nuids[1]) {
    Serial.println("Zugang Karte 2");
    led2_status = !led2_status;
    digitalWrite(led2, led2_status ? HIGH : LOW);
  } else if (nuid == rfid_nuids[2]) {
    Serial.println("Zugang Karte 3");
    led3_status = !led3_status;
    digitalWrite(led3, led3_status ? HIGH : LOW);
  }
}
```

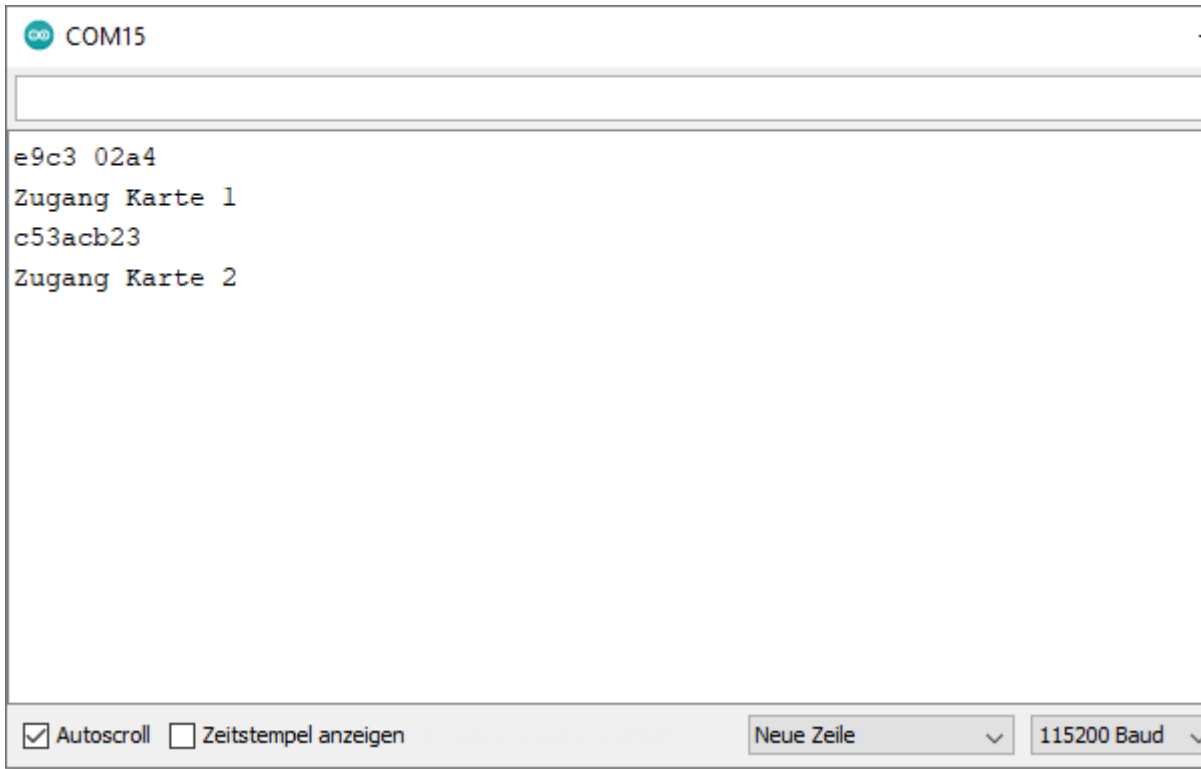
```
}

rfid.PICC_HaltA();
rfid.PCD_StopCrypto1();
}

String readHex(byte *buffer, byte bufferSize) {
    String result = "";
    for (byte i = 0; i < bufferSize; i++) {
        String* hex[] = {};
        result = result + String(buffer[i] < 0x10 ? " 0" : "");
        result = result + String(buffer[i], HEX);
    }
    return result;
}
```

RAW code

RFID-Zugansinformationen für drei RFID-Transponder mit dem Seriellen Monitor auslesen



NANO Programm RFID auslesen

```
/*****
                                     *****/

                        PROGRAMMINFO

*****/

Funktion: RFID auslesen

*****/

Version: 21.06.2022

*****/

Board: NANO

*****/

Libraries:
https://github.com/espressif/arduino-esp32/tree/master/libraries
C:\Users\User\Documents\Arduino
D:\gittemp\Arduino II\A156_Wetterdaten_V3
*****/

C++ Arduino IDE V1.8.19

*****/

Einstellungen:
https://dl.espressif.com/dl/package_esp32_index.json
```

http://dan.drown.org/stm32duino/package_STM32duino_index.json

http://arduino.esp8266.com/stable/package_esp8266com_index.json

*****/

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

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

```
#define SS_PIN 10 // ESP32 pin GPIO5
```

```
#define RST_PIN 9 // ESP32 pin GPIO27
```

```
MFRC522 rfid(SS_PIN, RST_PIN);
```

```
void setup() {
```

```
  Serial.begin(115200);
```

```
  SPI.begin(); // init SPI bus
```

```
  rfid.PCD_Init(); // init MFRC522
```

```
  Serial.println("Lege den RFID-Chip auf das RFID-Lesegerät");
```

```
}
```

```
void loop() {
```

```
  if (rfid.PICC_IsNewCardPresent()) { // new tag is available
```

```
    if (rfid.PICC_ReadCardSerial()) { // NUID has been readed
```

```
      MFRC522::PICC_Type piccType = rfid.PICC_GetType(rfid.uid.sak);
```

```
      Serial.print("RFID/NFC Typ: ");
```

```
      Serial.println(rfid.PICC_GetTypeName(piccType));
```

```
      // drucke das UID auf den Seriellen Monitor im Hex-Format
```

```
      Serial.print("UID:");
```

```
      for (int i = 0; i < rfid.uid.size; i++) {
```

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

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

```
      }
```

```
      Serial.println();
```

```
      rfid.PICC_HaltA(); // Halt PICC
```

```
      rfid.PCD_StopCrypto1(); // Stopp Verschlüsselung
```

```
}
```



```
}
```

```
}
```

NANO Hauptprogramm LED-Anzeige und Türe öffnen

```
/******  
PROGRAMMINFO  
*****  
Funktion: RFID LED ansteuern  
*****  
Version: 21.06.2022  
*****  
Board: NANO  
*****  
Libraries:  
https://github.com/miguelbalboa/rfid  
https://github.com/espressif/arduino-esp32/tree/master/libraries  
*****  
C++ Arduino IDE V1.8.19  
*****  
Einstellungen:  
https://dl.espressif.com/dl/package\_esp32\_index.json  
http://dan.drown.org/stm32duino/package\_STM32duino\_index.json  
http://arduino.esp8266.com/stable/package\_esp8266com\_index.json  
*****/  
//einbinden der Bibliotheken für das  
//ansteuern des MFRC522 Moduls  
#include <SPI.h>  
#include <MFRC522.h>  
  
//definieren der Pins RST & SDA für den NANO  
#define RST_PIN 9  
#define SS_PIN 10  
  
#define ledRot 3  
#define ledGruen 2  
#define oc_PIN 5 //Open Close PIN
```

```
//erzeugen einer Objektinstanz
MFRC522 mfrc522(SS_PIN, RST_PIN);

//Variable zum speichern der bereits gelesenen RFID-ID
String lastRfid = "";

//Anzahl der zulässigen RFID-IDs im Array
const int NUM_RFIDS = 1;
//Array mit RFID-IDs welche zulässig sind
String rfids[NUM_RFIDS] = {" E9 C3 02 A4"};

void setup() {
  //beginn der seriellen Kommunikation mit 115200 Baud
  Serial.begin(115200);
  //eine kleine Pause von 50ms.
  delay(50);

  pinMode(ledRot, OUTPUT);
  pinMode(ledGruen, OUTPUT);
  pinMode(oc_PIN, OUTPUT);
  digitalWrite(oc_PIN, LOW);

  //begin der SPI Kommunikation
  SPI.begin();
  //initialisieren der Kommunikation mit dem RFID Modul
  mfrc522.PCD_Init();
}

void loop() {

  //Wenn keine neue Karte vorgehalten wurde oder die serielle Kommunikation
  //nicht gegeben ist, dann...
  if ( !mfrc522.PICC_IsNewCardPresent()) {
    //Serial.println("!PICC_IsNewCardPresent");
    return;
  }

  if (!mfrc522.PICC_ReadCardSerial()) {
    //Serial.println("!PICC_ReadCardSerial");
  }
}
```

```
return;
```

```
}
```

```
String newRfidId = "";
```

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

```
    newRfidId.concat(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
```

```
    newRfidId.concat(String(mfrc522.uid.uidByte[i], HEX));
```

```
}
```

```
//alle Buchstaben in Großbuchstaben umwandeln
```

```
newRfidId.toUpperCase();
```

```
//Wenn die neue gelesene RFID-ID ungleich der bereits zuvor gelesenen ist,
```

```
//dann soll diese auf der seriellen Schnittstelle ausgegeben werden.
```

```
if (!newRfidId.equals(lastRfid)) {
```

```
    //überschreiben der alten ID mit der neuen
```

```
    lastRfid = newRfidId;
```

```
}
```

```
bool zugangOk = false;
```

```
//prüfen ob die gelesene RFID-ID im Array mit bereits gespeicherten IDs vorhanden ist
```

```
for (int i = 0; i < NUM_RFIDS; i++) {
```

```
    //wenn die ID an der Stelle "i" im Array "rfids" mit dem gelesenen übereinstimmt, dann
```

```
    if (rfids[i].equals(newRfidId)) {
```

```
        zugangOk = true;
```

```
        //Schleife verlassen
```

```
        break;
```

```
    }
```

```
}
```

```
//Wenn die Variable "zugangOk" auf True ist, dann...
```

```
if (zugangOk) {
```

```
    digitalWrite(oc_PIN, HIGH); //Türschloss öffnen
```

```
    Serial.println("Tuerschloss oeffnen");
```

```
    delay(500);
```

```
    digitalWrite(oc_PIN, LOW);
```

```
    Serial.println("Tuerschloss schliessen");
```

```
    blinkLed(ledGruen);
    Serial.println("RFID-ID [" + newRfidId + "] OK!, Zugang wird gewährt");

} else {
    //Wenn nicht dann...
    blinkLed(ledRot);
    Serial.println("RFID-ID [" + newRfidId + "] nicht OK!, Zugang wird nicht gewährt");
}

Serial.println();
}

//Blinken einer LED am Pin "pin"
void blinkLed(int pin) {
    //Schleife von 0 bis 5
    for (int a = 0; a < 5; a++) {
        //LED aktivieren
        digitalWrite(pin, HIGH);
        //eine Pause von 125 ms.
        delay(125);
        //LED deaktivieren
        digitalWrite(pin, LOW);
        //eine Pause von 125 ms.
        delay(125);
    }
}
```

Version #2

Erstellt: 31 März 2025 09:12:18 von Joel Hatsch

Zuletzt aktualisiert: 31 März 2025 09:22:57 von Joel Hatsch