RFID Introduction

RFID, also known as radio frequency identification, is a kind of communication technology. It can identify specific targets through radio signals and read and write data without establishing a mechanical or optical contact between recognition system and the specific goal. Radio frequency generally is microwave, 1-100GHz, suitable for short range identification communication.

RFID reader-writer is also divided into mobile and fixed types. The current RFID technology is widely used, such as: library, the entrance guard system, food safety traceability, etc.

Components:

Transponder: it is composed of an antenna, a coupling element and a chip. Generally, transponders are made looking like tags, each tag has a unique electronic code, identifying objects by attaching it on them.

Reader: it is a device that consists of an antenna, a coupling element and a chip, which can read (and sometimes write) tag information and can be designed as a hand-held RFID reader or a fixed reader.

Working Principle

The basic principle of RFID technology is not complicated: the tag will receive RF signals sent by the reader when it is into the magnetic field, then send the information stored in the chips (passive tags) with the obtained energy out of the induced current, or active tags send a specific frequency, which will be read and decoded the information by the reader, and sent to be processed by the central information system.

A complete set of RFID system is composed of three parts: the reader, the electronic tag (namely, the transponder) and the application system. The working principle is that the reader launches a specific frequency radio wave energy to drive the circuit sending out the internal data, then the reader orderly receives the interrupted data and sends to application doing the corresponding processing.

Judging from the communication and energy induction mode between electronic tags, there exists inductive coupling and back scattering coupling. The low frequency RFID generally uses the first type, while the high frequency uses the second.

Depending on the structure and technology, the reader can be a read or read / write device, it is the control and process center of RFID system information. The reader usually consists of coupling module, transceiver module, control module and interface unit. Generally, the half duplex communication mode is used to exchange information between reader and transponder, and the reader provides the energy and timing for the passive transponder by coupling.

Experiment Purpose

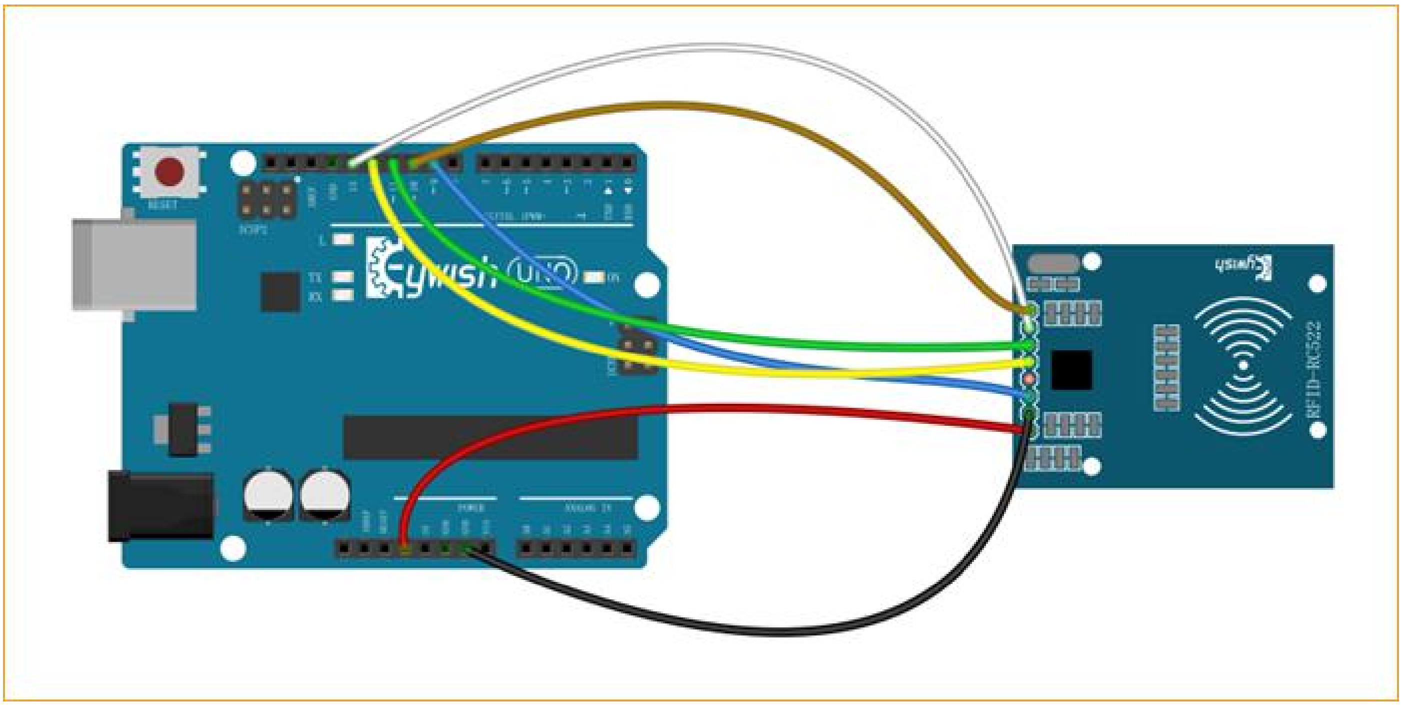
Reading through the card reader, if the tag matches what we use, then the buzzer beeps with a low frequency. Otherwise, a hasty alarm is issued.

Component List

- Keywish Arduino UNO R3 mainboard
- Breadboard
- USB cable
- RFID suite *1
- Passive buzzer *1
- Some wires

Wiring of Circuit

RFID	Arduino Uno R3
SDA	10
SCK	13
MOSI	11
MISO	12
IRQ	NALL
GND	GND
RST	9
VCC	3.3V



Co

de

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

#define SS_PIN 10
#define RST_PIN 9


MFRC522 rfid(SS_PIN, RST_PIN); // Instance of the class

MFRC522::MIFARE_Key key;

// Init array that will store new NUID
byte nuidPICC[4];

void setup() {
  Serial.begin(9600);

  SPI.begin(); // Init SPI bus
  rfid.PCD_Init(); // Init MFRC522

  for (byte i = 0; i < 6; i++) {
    key.keyByte[i] = 0xFF;
  }

  Serial.println(F("This code scan the MIFARE Classsic NUID."));
  Serial.print(F("Using the following key:"));
  printHex(key.keyByte, MFRC522::MF_KEY_SIZE);
}

void loop() {

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

  // Verify if the NUID has been readed
  if ( ! rfid.PICC_ReadCardSerial())
    return;
```

```
Serial.print(F("PICC type: "));
  MFRC522::PICC_Type piccType = rfid.PICC_GetType(rfid.uid.sak);
  Serial.println(rfid.PICC_GetTypeName(piccType));

  // Check is the PICC of Classic MIFARE type
  if (piccType != MFRC522::PICC_TYPE_MIFARE_MINI &&
    piccType != MFRC522::PICC_TYPE_MIFARE_1K &&
    piccType != MFRC522::PICC_TYPE_MIFARE_4K) {
    Serial.println(F("Your tag is not of type MIFARE Classic."));
    return;
  }

  if (rfid.uid.uidByte[0] != nuidPICC[0] ||
    rfid.uid.uidByte[1] != nuidPICC[1] ||
    rfid.uid.uidByte[2] != nuidPICC[2] ||
    rfid.uid.uidByte[3] != nuidPICC[3] ) {
    Serial.println(F("A new card has been detected."));

    // Store NUID into nuidPICC array
    for (byte i = 0; i < 4; i++) {
      nuidPICC[i] = rfid.uid.uidByte[i];
    }

    Serial.println(F("The NUID tag is:"));
    Serial.print(F("In hex: "));
    printHex(rfid.uid.uidByte, rfid.uid.size);
    Serial.println();
    Serial.print(F("In dec: "));
    printDec(rfid.uid.uidByte, rfid.uid.size);
    Serial.println();
  }
  else Serial.println(F("Card read previously."));

  // Halt PICC
  rfid.PICC_HaltA();

  // Stop encryption on PCD
  rfid.PCD_StopCrypto1();
}


/**
 * Helper routine to dump a byte array as hex values to Serial.
```

```
*/
void printHex(byte *buffer, byte bufferSize) {
  for (byte i = 0; i < bufferSize; i++) {
    Serial.print(buffer[i] < 0x10 ? " 0" : " ");
    Serial.print(buffer[i], HEX);
  }
}

/**
 * Helper routine to dump a byte array as dec values to Serial.
 */
void printDec(byte *buffer, byte bufferSize) {
  for (byte i = 0; i < bufferSize; i++) {
    Serial.print(buffer[i] < 0x10 ? " 0" : " ");
    Serial.print(buffer[i], DEC);
  }
}
```

Experiment Result