30.RC522_RFID_Module

Introduction

Radio Frequency Identification (RFID) refers to technologies that use wireless communication between an object (or tag) and interrogating device (or reader) to automatically track and identify such objects.

Some of the most common applications for this technology include retail supply chains, military supply chains, automated payment methods, baggage tracking and management, document tracking and pharmaceutical management, to name a few.

In this lesson, we will use RFID for reading and writing.

Hardware Required

- ✓ 1 * Raspberry Pi
- ✓ 1 * T-Extension Board
- ✓ 1 * RFID RC522
- ✓ 1 * 40-pin Cable
- ✓ Several Jumper Wires
- ✓ 1 * Breadboard

Principle

RC522

The MFRC522 is a highly integrated reader/writer for contactless communication at 13.56 MHz. The MFRC522 reader supports ISO 14443A / MIFARE® mode.

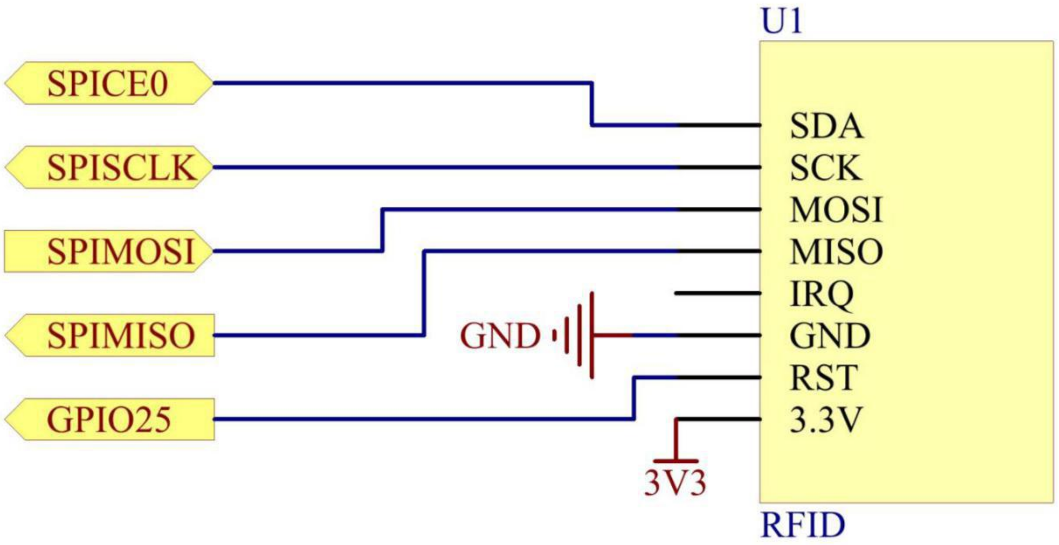
The MFRC522's internal transmitter part is able to drive a reader/writer antenna designed to communicate with ISO/IEC 14443A/MIFARE® cards and transponders without additional active circuitry. The receiver part provides a robust and efficient implementation of a demodulation and decoding circuitry for signals from ISO/IEC 14443A/MIFARE® compatible cards and transponders. The digital part handles the complete ISO/IEC 14443A framing and error detection (Parity & CRC).The MFRC522 supports MIFARE®Classic (e.g. MIFARE® Standard) products. The MFRC522 supports contactless communication using MIFARE® higher transfer speeds up to 848 kbit/s in both directions.

Various host interfaces are implemented:

- ✓ SPI interface
- ✓ Serial UART (similar to RS232 with voltage levels according pad voltage supply)
- ✓ I2C interface.
- ✓ The figure below shows a typical circuit diagram, using a complementary antenna connection to the MFRC522.

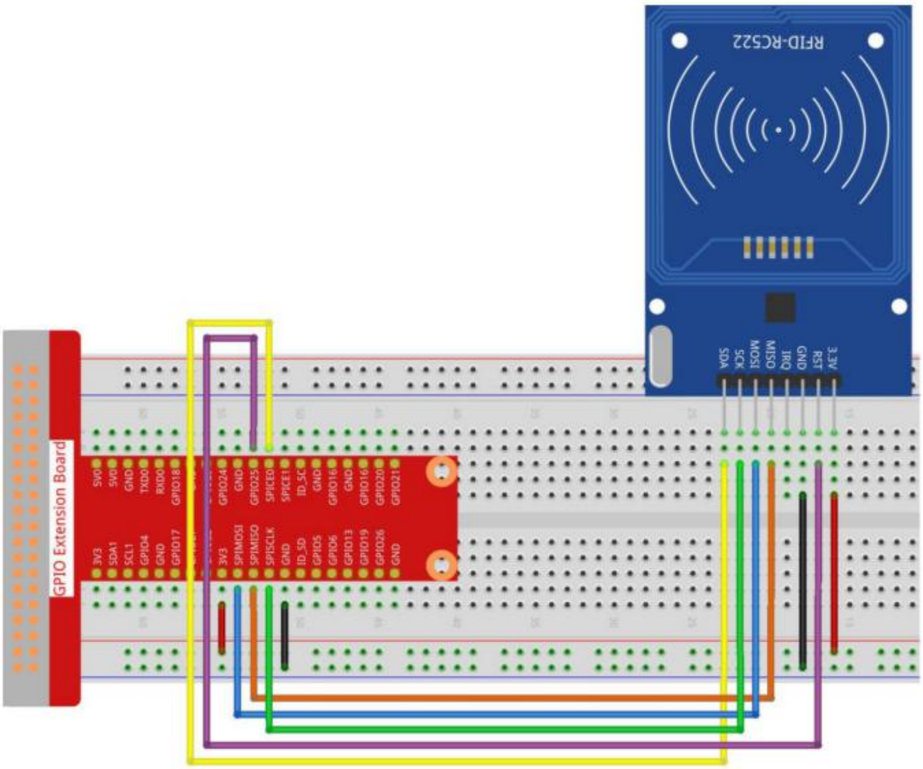
Schematic Diagram

T-Board Name	physical	wiringPi	BCM
SPICE0	Pin 24	10	9
SPISCLK	Pin 23	14	11
SPIMOSI	Pin 19	12	10
SPIMISO	Pin 21	13	9
GPIO25	Pin 22	6	25



Experimental Procedures

Step 1: Build the circuit.



Step 2: Set up SPI (refer to Appendix for more details. If you have set SPI, skip this step.)

For C Language Users

Step3: Go to the folder of the code.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/30.RC522_RFID_Module
```

Step 4: Compile the code.

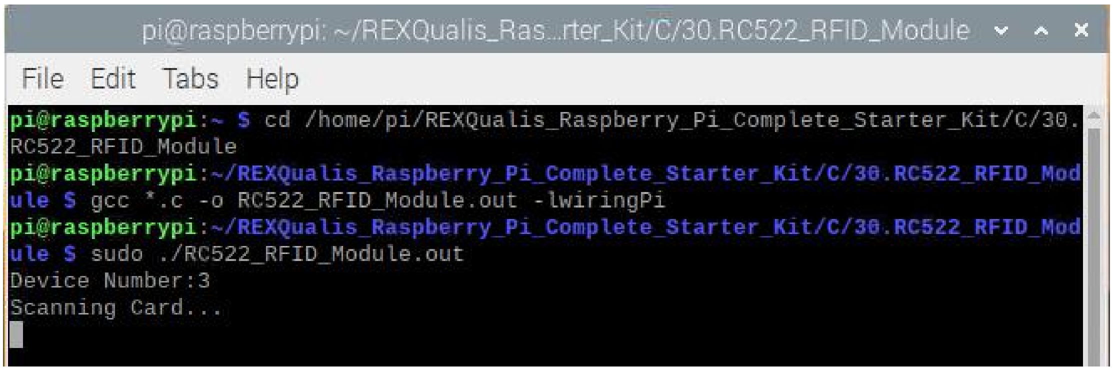
```
gcc *.c -o RC522_RFID_Module.out -lwiringPi
```

Note: The program includes mfc522.c, mfc522_debug.c, mfc522.c_hal_linux.c, dump.c, 30.RC522 RFID Module .c and so on. We use the wildcard (*), which causes all the files that conform to the format to be processed together.

Step 4: Run the executable file .

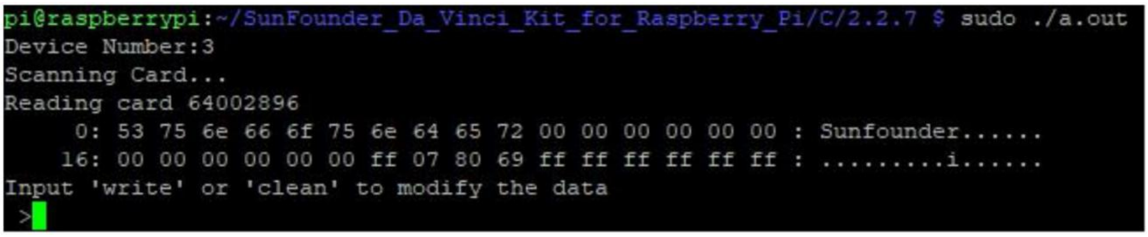
```
sudo ./RC522_RFID_Module.out
```

When the program runs, terminal will pop up to notice you to print “Scaning Card...”. At this time, we put the query object (matching tag or card) close to the reader (RFID-RC522) module, and then we can read the data inside.

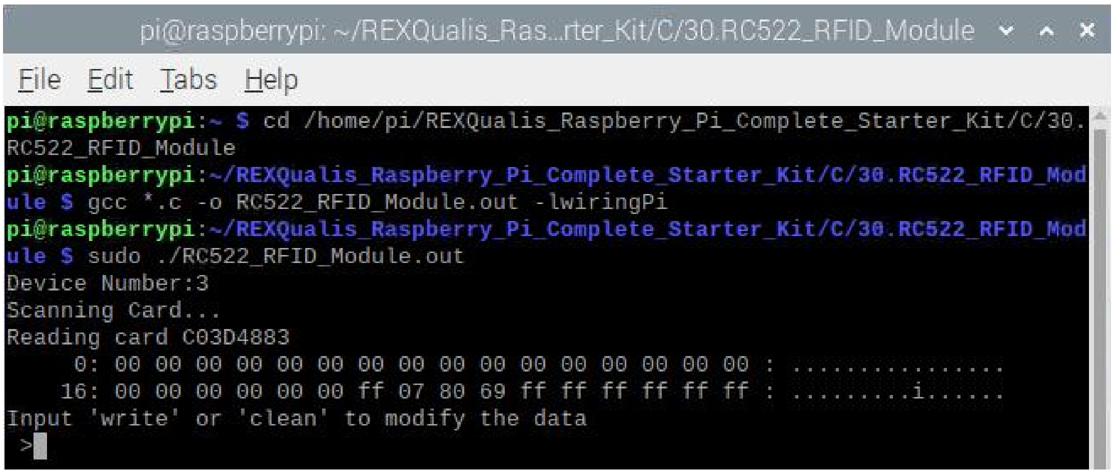

pi@raspberrypi: ~/REXQualis_Ras...rter_Kit/C/30.RC522_RFID_Module
File
Edit
Tabs
Help

```
pi@raspberrypi:~ $ cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/30.
RC522_RFID_Module
pi@raspberrypi:~/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/30.RC522_RFID_Mod
ule $ gcc *.c -o RC522_RFID_Module.out -lwiringPi
pi@raspberrypi:~/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/30.RC522_RFID_Mod
ule $ sudo ./RC522_RFID_Module.out
Device Number:3
Scanning Card...
```

When the query object is read, the following codes are printed.



```
pi@raspberrypi:~/SunFounder_Da_Vinci_Kit_for_Raspberry_Pi/C/2.2.7 $ sudo ./a.out
Device Number:3
Scanning Card...
Reading card 64002896
     0: 53 75 6e 66 6f 75 6e 64 65 72 00 00 00 00 00 00 : Sunfounder......
    16: 00 00 00 00 00 00 ff 07 80 69 ff ff ff ff ff ff : .........i......
Input 'write' or 'clean' to modify the data
 >
```


pi@raspberrypi: ~/REXQualis_Ras...rter_Kit/C/30.RC522_RFID_Module
File
Edit
Tabs
Help

```
pi@raspberrypi:~ $ cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/30.
RC522_RFID_Module
pi@raspberrypi:~/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/30.RC522_RFID_Mod
ule $ gcc *.c -o RC522_RFID_Module.out -lwiringPi
pi@raspberrypi:~/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/30.RC522_RFID_Mod
ule $ sudo ./RC522_RFID_Module.out
Device Number:3
Scanning Card...
Reading card C03D4883
     0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
    16: 00 00 00 00 00 00 ff 07 80 69 ff ff ff ff ff ff : .........i......
Input 'write' or 'clean' to modify the data
 >
```

Reading card C03D4883, here the number represents global unique identifier number of the query object(USN/UID).

The middle two lines of characters are the data stored in the query object. We can modify the data stored on the card by typing "write" or "clean".

Code

```
#include <stdio.h>

#include <stdint.h>

#include <unistd.h>

#include <string.h>

#include <getopt.h>
```

```
#include <stdlib.h>

#include "RC522.h"

int scan_loop(uint8_t *CardID);

int tag_select(uint8_t *CardID);

int main(int argc, char **argv) {

    MFRC522_Status_t ret;

    uint8_t CardID[5] = { 0x00, };

    uint8_t tagType[16] = {0x00,};

    static char command_buffer[1024];

    ret = MFRC522_Init('B');

    if (ret < 0) {

        perror("Failed to initialize");

        exit(-1);

    }

    while (1) {

            puts("Scanning Card... ");

            while (1) {

                ret = MFRC522_Request(PICC_REQIDL, tagType);

                if (ret == MI_OK) {

                    ret = MFRC522_Anticoll(CardID);

                    if(ret == MI_OK){

                        ret = tag_select(CardID);

                        if (ret == MI_OK) {

                            ret = scan_loop(CardID);

                            if (ret < 0) {

                                printf("Error...\r\n");

                                break;

                            }

                        }

                    }
```

```
                        else{
                            printf("Get Card ID failed!\r\n");
                        }
                    }
                    MFRC522_Halt();
                }
                MFRC522_Halt();
                MFRC522_Init('B');
    }
}
int scan_loop(uint8_t *CardID) {
    while (1) {
        char input[32];
        int block_start=2;
        printf("Reading card %02X%02X%02X%02X \n", CardID[0], CardID[1],
CardID[2],
CardID[3]);
        MFRC522_Debug_DumpSector(CardID, block_start);
        printf("Input 'write' or 'clean' to modify the data\n >");
        scanf("%s", input);
        printf("\n");
        if(strcmp(input, "clean") == 0){
            getchar();
            if (MFRC522_Debug_Clean(CardID, block_start)) {
                return -1;
            }
        } else if (strcmp(input, "write") == 0) {
            char write_buffer[256];
            size_t len = 0;
            getchar();
            printf("Input the Data (less than 16 character)\n>");
            scanf("%s",write_buffer);
                if (MFRC522_Debug_Write(CardID, block_start, write_buffer,
                        strlen(write_buffer)) < 0) {
                    return -1;
                }
        }
    }
    return 0;
}
int tag_select(uint8_t *CardID) {
    int ret_int;
    ret_int = MFRC522_SelectTag(CardID);
    if (ret_int == 0) {
        printf("Card Select Failed\r\n");
        return -1;
    }
    ret_int = 0;
    return ret_int;
}
```

For Python Language Users

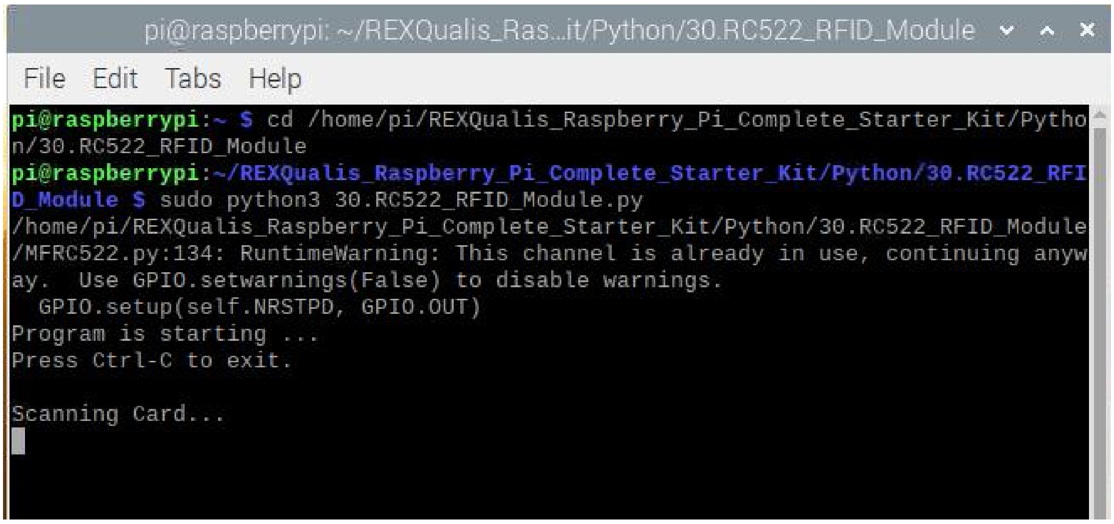
Step 3: Go to the folder of the code.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python/30.RC522_RFID_Module
```

Step 4: Run the executable file.

```
sudo python3 30.RC522_RFID_Module.py
```

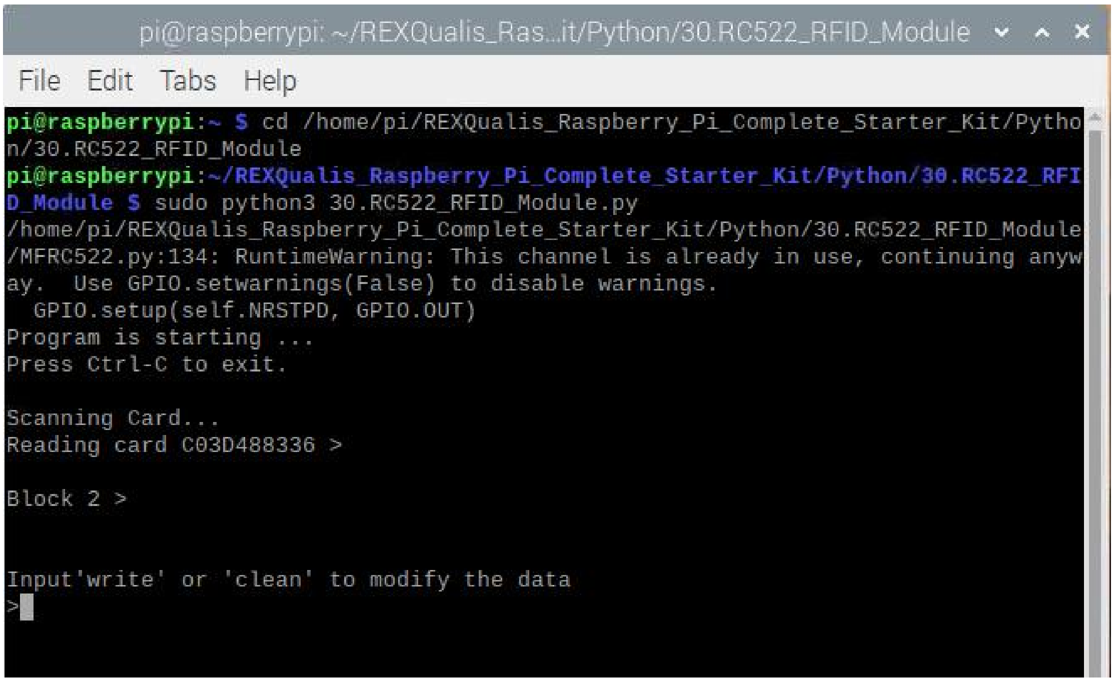
When the program runs, terminal will pop up to notice you to print “ Scaning Card... ” . At this time, we put the query object (matching tag or card) close to the reader (RFID-RC522) module, and then we can read the data inside.


pi@raspberrypi: ~/REXQualis_Ras...it/Python/30.RC522_RFID_Module
File Edit Tabs Help

```
pi@raspberrypi:~ $ cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Pytho
n/30.RC522_RFID_Module
pi@raspberrypi:~/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python/30.RC522_RFI
D_Module $ sudo python3 30.RC522_RFID_Module.py
/home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python/30.RC522_RFID_Module
/MFRC522.py:134: RuntimeWarning: This channel is already in use, continuing anyw
ay.  Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(self.NRSTPD, GPIO.OUT)
Program is starting ...
Press Ctrl-C to exit.

Scanning Card...
```

When the query object is read, the following codes are printed.


pi@raspberrypi: ~/REXQualis_Ras...it/Python/30.RC522_RFID_Module
File Edit Tabs Help

```
pi@raspberrypi:~ $ cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Pytho
n/30.RC522_RFID_Module
pi@raspberrypi:~/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python/30.RC522_RFI
D_Module $ sudo python3 30.RC522_RFID_Module.py
/home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python/30.RC522_RFID_Module
/MFRC522.py:134: RuntimeWarning: This channel is already in use, continuing anyw
ay.  Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(self.NRSTPD, GPIO.OUT)
Program is starting ...
Press Ctrl-C to exit.

Scanning Card...
Reading card C03D488336 >

Block 2 >


Input'write' or 'clean' to modify the data
>
```

The code “ Reading card C03D488336” represents global unique identifier number (USN/UID)of inquiry objects.We can modify the data stored on the card by typing "write" or "clean".

Code

```
import RPi.GPIO as GPIO

import MFRC522

import sys

import os
```

```
mfrc = MFRC522.MFRC522()
def setup():
    print ("Program is starting ... "     )
    print ("Press Ctrl-C to exit.\n")
    pass
def loop():
    global mfrc
    while(True):
        print ("Scanning Card... ")
        mfrc = MFRC522.MFRC522()
        isScan = True
        while isScan:
            (status,TagType) = mfrc.MFRC522_Request(mfrc.PICC_REQIDL)
            (status,uid) = mfrc.MFRC522_Anticoll()
            if status == mfrc.MI_OK:
                if mfrc.MFRC522_SelectTag(uid) == 0:
                    print ("MFRC522_SelectTag Failed!")
                if cmdloop(uid) < 1 :
                    isScan = False
def cmdloop(cardID):
    blockAddr = 2
    while(True):
        print ("Reading card %2X%2X%2X%2X%2X > \n"%(cardID[0],cardID[1],cardID[2],cardID[3],cardID[4]))
        key = [0xFF,0xFF,0xFF,0xFF,0xFF,0xFF]
        status = mfrc.MFRC522_Auth(mfrc.PICC_AUTHENT1A, blockAddr, key, cardID)
        if status == mfrc.MI_OK:
            mfrc.MFRC522_Read(blockAddr)
            print(" ")
        else:
    print ("Authentication error")
            return 0
        print ("\nInput'write' or 'clean' to modify the data\n>",end="")
        inCmd = input()
        cmd = inCmd.split(" ")
        if cmd[0] == "write":
            print ("\nInput the Data (less than 16 character)\n>",end="")
            inCmd = input()
            cmd = inCmd.split(" ")
            data = [0]*16
            data = cmd[0][0:17]
            data = map(ord,data)
            data = list(data)
            lenData = len(list(data))
            if lenData<16:
                data+=[0]*(16-lenData)
            key = [0xFF,0xFF,0xFF,0xFF,0xFF,0xFF]
            status = mfrc.MFRC522_Auth(mfrc.PICC_AUTHENT1A, blockAddr, key,
cardID)
            if status == mfrc.MI_OK:
                mfrc.MFRC522_Write(blockAddr, data)
            else:
                print ("Authentication error")
                return 0
        elif cmd[0] == "clean":
            data = [0]*16
            key = [0xFF,0xFF,0xFF,0xFF,0xFF,0xFF]
            status = mfrc.MFRC522_Auth(mfrc.PICC_AUTHENT1A, blockAddr, key,
cardID)
```

```
            if status == mfrc.MI_OK:
                mfrc.MFRC522_Write(blockAddr, data)
            else:
                print ("Authentication error")
                return 0
        else :
            return 0
def destroy():
    GPIO.cleanup()
if __name__ == "__main__":
    setup()
    try:
        loop()
    except KeyboardInterrupt:   # Ctrl+C captured, exit
        destroy()
```

Phenomenon Picture