RFID

Overview

In this project, we will use RC522 RFID card reader to read and write the M1-S50 card

Experimental Materials:

Raspberry Pi *1

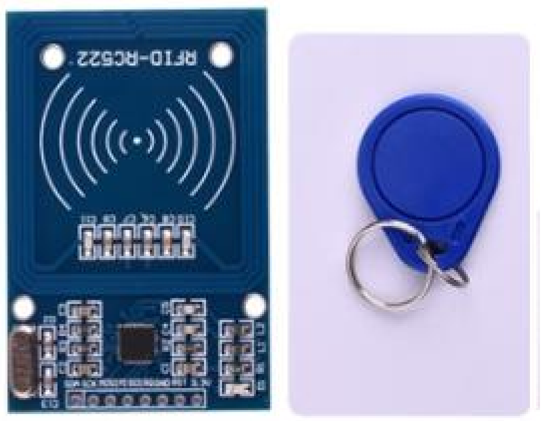
T-type expansion board *1

Breadboard*1

RFID *1

Some DuPont lines

Product description：



RFID

RFID（Radio Frequency Identification） is a wireless communication technology. A complete RFID system is generally composed of the responder and reader. Generally, we use tags as responders, and each tag has a unique code, which is attached to the object to identify the target object. The reader is a device for reading (or writing) tag information.

Products derived from RFID technology can be divided into three categories: passive RFID products, active RFID products and semi active RFID products. And Passive RFID products are the earliest, the most mature and most widely used products in the market among others. It can be seen everywhere in our daily life such as, the bus card, dining card, bank card, hotel access cards, etc., and all of these belong to close-range contact recognition. The main operating frequency of Passive RFID products are: 125KHZ (low frequency), 13.56MHZ (high frequency), 433MHZ (ultrahigh frequency), 915MHZ (ultrahigh frequency). Active and semi active RFID products work at higher frequencies.

The RFID module we use is a passive RFID product with the operating frequency of 13.56MHz.

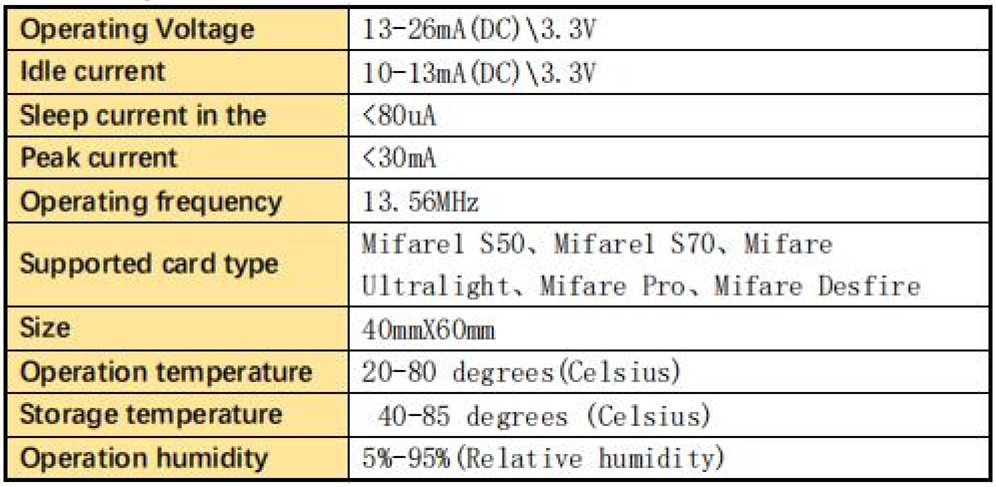
MFRC522

The MFRC522 is a highly integrated reader/writer IC for contactless communication at 13.56MHz.

The MFRC522’s internal transmitter is able to drive a reader/writer antenna designed to communicate with ISO/IEC 14443 A/MIFARE cards and transponders without additional active circuitry. The receiver module provides a robust and efficient implementation for demodulating and decoding signals from ISO/IEC 14443 A/MIFARE compatible cards and transponders. The digital module manages the complete ISO/IEC 14443A framing and error detection (parity and CRC) functionality

This RFID Module uses MFRC522 as the control chip, and SPI (Peripheral Interface Serial) as the reserved interface.

Technical Parameters:



Operating Voltage	13-26mA(DC)\3.3V
Idle current	10-13mA(DC)\3.3V
Sleep current in the	<80uA
Peak current	<30mA
Operating frequency	13.56MHz
Supported card type	Mifare1 S50、Mifare1 S70、Mifare Ultralight、Mifare Pro、Mifare Desfire
Size	40mmX60mm
Operation temperature	20-80 degrees(Celsius)
Storage temperature	40-85 degrees (Celsius)
Operation humidity	5%-95%(Relative humidity)

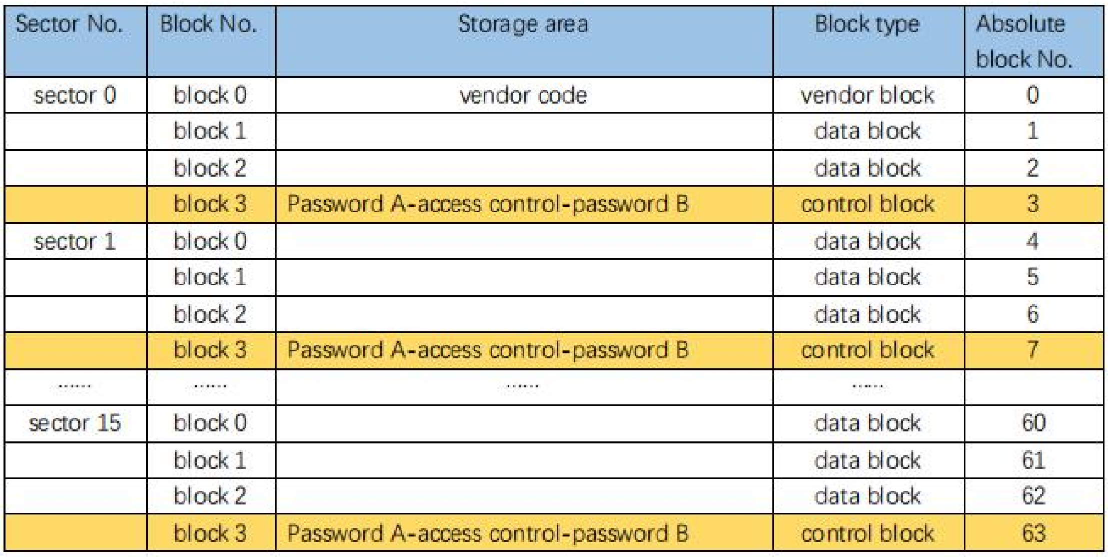
Mifare1 S50 Card

Mifare S50 is often called Mifare Standard with the capacity of 1K bytes. And each card has a 4-bytes global unique identifier number (USN/UID), which can be rewritten 100 thousand times and read infinite times. Its storage period can last for 10 years.

The Mifare S50 capacity (1K byte) is divided into 16 sectors (Sector0-Sector15). Each sector contains 4 data block (Block0-Block3. 64 blocks of 16 sectors will be numbered according absolute address, from

0 to 63).

And each block contains 16 bytes (Byte0-Byte15), 64*16=1024. As is shown in the following table:



Sector No.	Block No.	Storage area	Block type	Absolute block No.
sector 0	block 0	vendor code	vendor block	0
	block 1		data block	1
	block 2		data block	2
	block 3	Password A-access control-password B	control block	3
sector 1	block 0		data block	4
	block 1		data block	5
	block 2		data block	6
	block 3	Password A-access control-password B	control block	7
......	......	......	......	
sector 15	block 0		data block	60
	block 1		data block	61
	block 2		data block	62
	block 3	Password A-access control-password B	control block	63

Each sector has a set of independent password and access control which are put in the last block of each sector, and the block is also known as sector trailer, that is Block 3 in each sector. Sector 0, block 0 (namely absolute address 0) of S50 is used to store the vendor code, which has been solidified and can’ t be changed,and the card serial number is stored here. In addition to the manufacturer and the control block, the rest of the cards are data blocks, which can be used to store data. Data block can be used for two kinds of applications:

(1) used as general data storage and can be operated for reading and writing.

(2) used as data value, and can be operated for initializing the value, adding value, subtracting and reading the value.

The sector trailer block in each sector is the control block, including a 6-byte password A, 4-byte access control and 6-byte password B. For example, the control block of a brand new card is as follows:

A0 A1 A2 A3 A4 A5	FF 07 80 69	B0 B1 B2 B3 B4 B5
password A	access control	password B

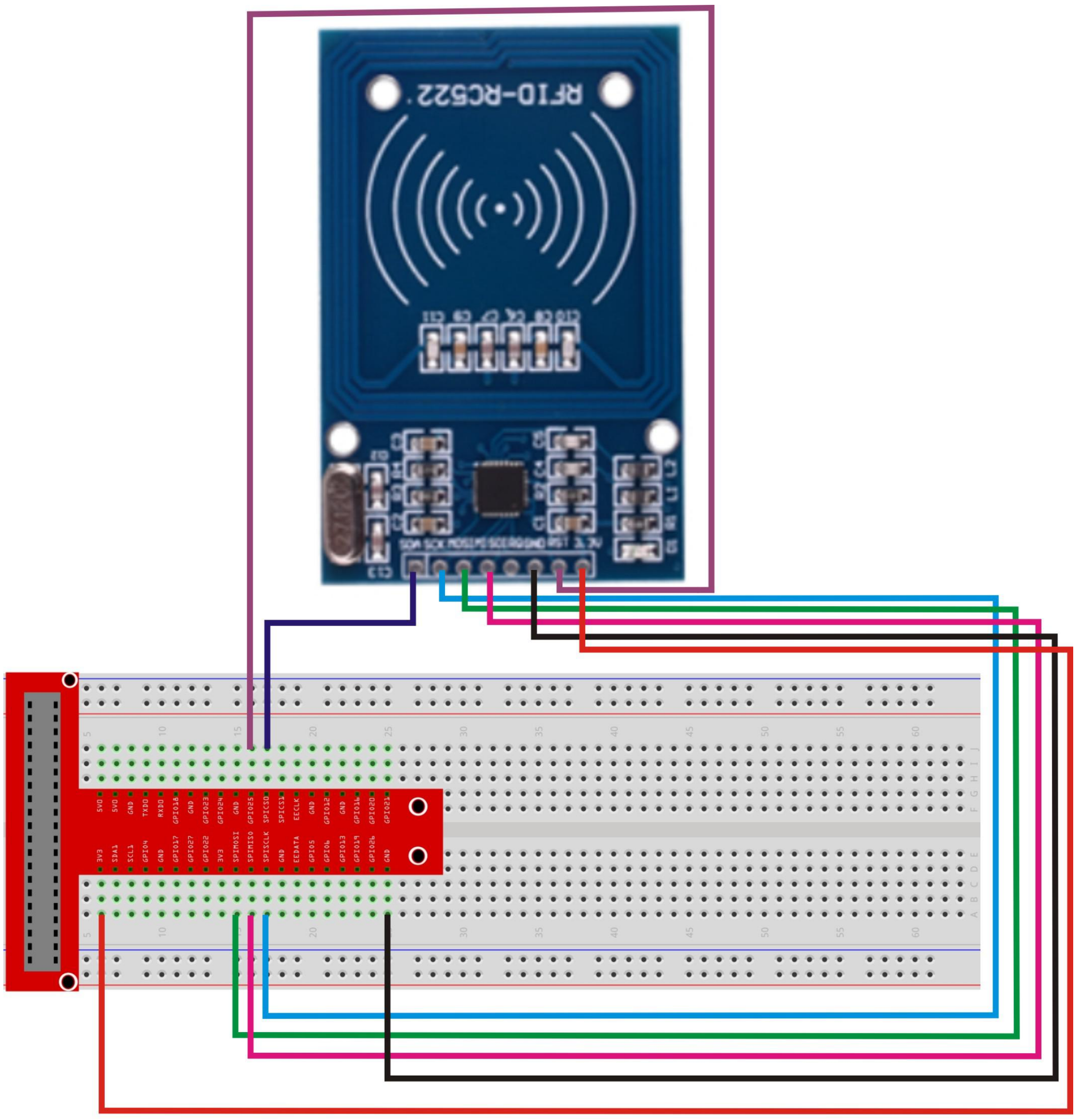
The default password of a brand new card is generally 0A1A2A3A4A5 for password A, B0B1B2B3B4B5 for password B, or both the password A and password B are 6 FF. Access control is used to set the access conditions for each block (including the control block itself) in a sector.

Blocks of S50 are divided into data blocks and control blocks. There are four operations, "read", "write", "add value", "subtract value (including transmission and storage)" for data blocks, and there are two operations, "read" and "write" for control blocks.

By default, after verifying password A or password B, we can do reading or writing operation to data blocks.And after verifying password A, we can do reading or writing operation to control blocks. But password A can never be read. If you choose to verify password A and then you forget the password A, the block will never be able to read again. It is highly recommended that beginners should not try to change the contents of control blocks.

the default password A and B is FFFFFFFFFFFF.

Wiring diagram：



Configure SPI

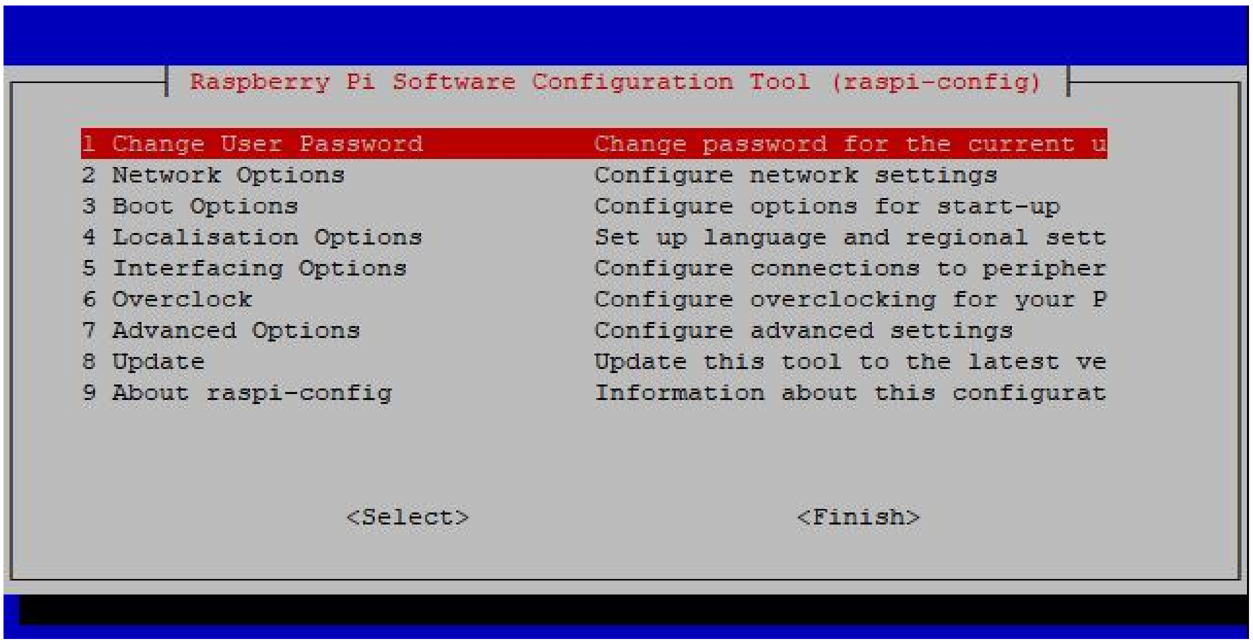
Enable SPI

The SPI interface raspberry pie is closed in default. You need to open it manually. You can enable the SPI interface in the following way.

Type command in the terminal:

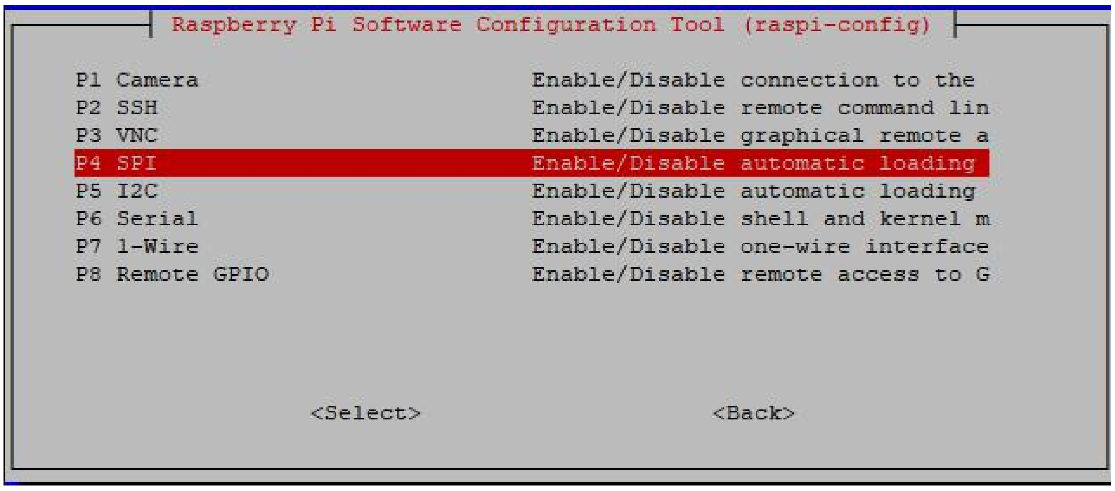
```
sudo raspi-config
```

Then open the following dialog box:

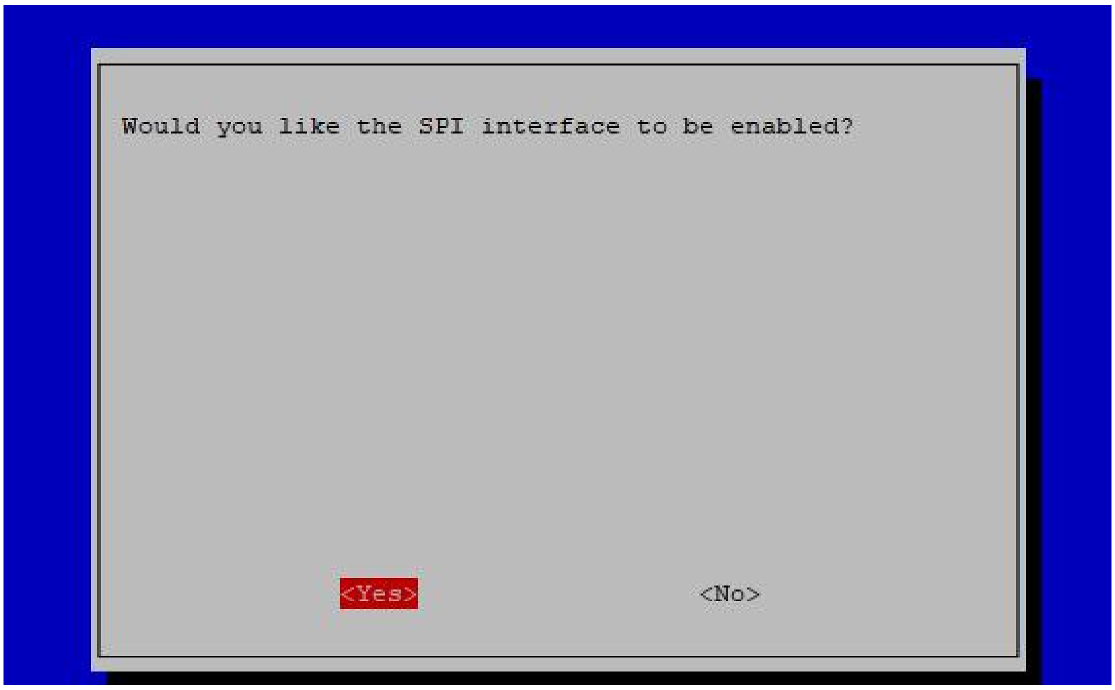


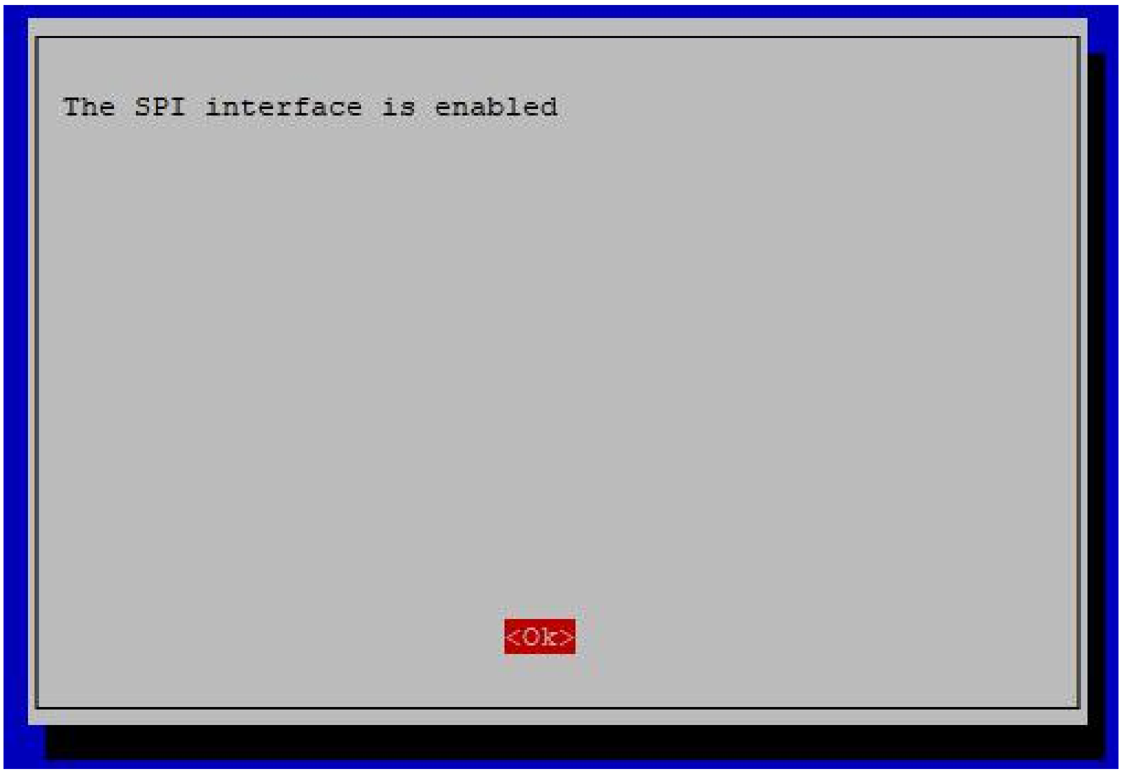
```
Raspberry Pi Software Configuration Tool (raspi-config)

1 Change User Password          Change password for the current u
2 Network Options               Configure network settings
3 Boot Options                  Configure options for start-up
4 Localisation Options          Set up language and regional sett
5 Interfacing Options           Configure connections to peripher
6 Overclock                     Configure overclocking for your P
7 Advanced Options              Configure advanced settings
8 Update                        Update this tool to the latest ve
9 About raspi-config            Information about this configurat

                <Select>                     <Finish>
```

Choose “5 Interfacing Options” “P4 SPI” “Yes” “Ok” in order and restart your RPi later. Then the SPI module is started.



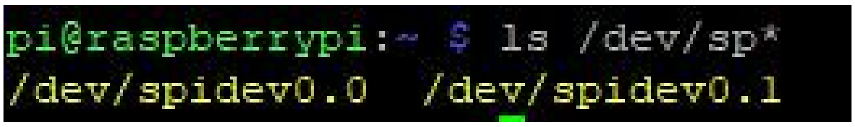
```
Raspberry Pi Software Configuration Tool (raspi-config)

P1 Camera                      Enable/Disable connection to the
P2 SSH                         Enable/Disable remote command lin
P3 VNC                         Enable/Disable graphical remote a
P4 SPI                         Enable/Disable automatic loading
P5 I2C                         Enable/Disable automatic loading
P6 Serial                      Enable/Disable shell and kernel m
P7 1-Wire                      Enable/Disable one-wire interface
P8 Remote GPIO                 Enable/Disable remote access to G

              <Select>                 <Back>
```


Would you like the SPI interface to be enabled?


The SPI interface is enabled

Type the following command to check whether the module SPI is loaded successfully:

```
ls /dev/sp*
```



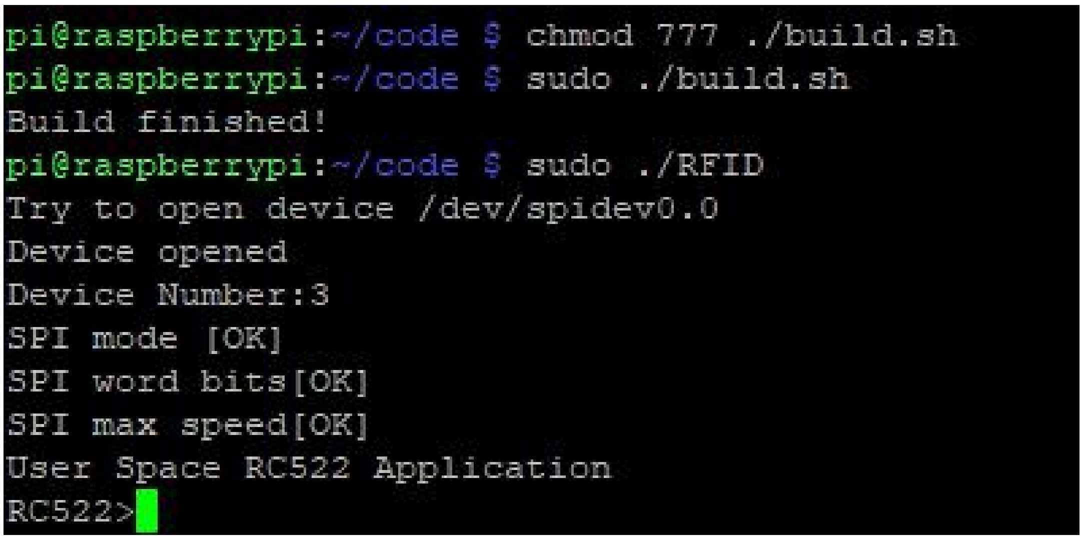
```
pi@raspberrypi:~ $ ls /dev/sp*
/dev/spidev0.0  /dev/spidev0.1
```

Experimental results:

In the directory where the code file is located, execute the following command

```
chmod 777 ./build.sh
sudo ./build.sh
sudo ./RFID
```

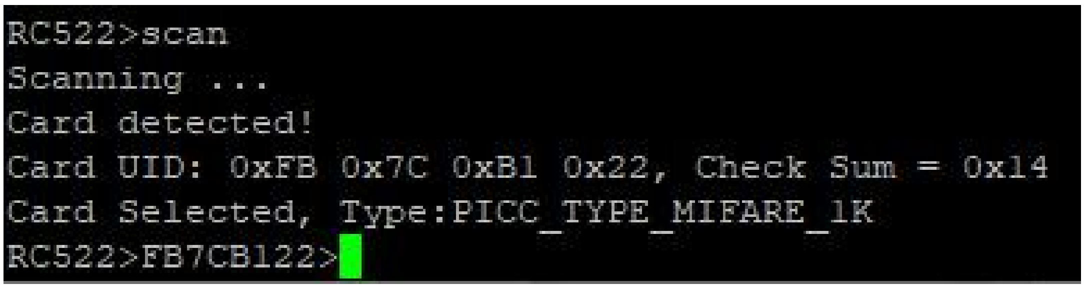
After the program is executed, the following contents will be displayed in the terminal:



```
pi@raspberrypi:~/code $ chmod 777 ./build.sh
pi@raspberrypi:~/code $ sudo ./build.sh
Build finished!
pi@raspberrypi:~/code $ sudo ./RFID
Try to open device /dev/spidev0.0
Device opened
Device Number:3
SPI mode [OK]
SPI word bits[OK]
SPI max speed[OK]
User Space RC522 Application
RC522>
```

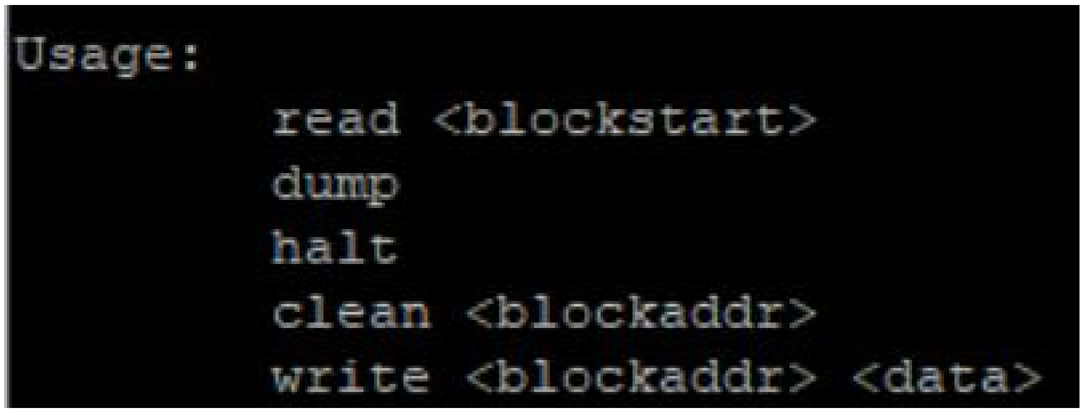
Here, type the command “quit” to exit the program.

Type command "scan", then the program begins to detect whether there is a card close to the sensing area of MFRC522 reader. Place an M1-S50 card in the sensing area. The following results indicate that the M1-S50 card has been detected, the UID of which is FB7CB12214 (HEX).



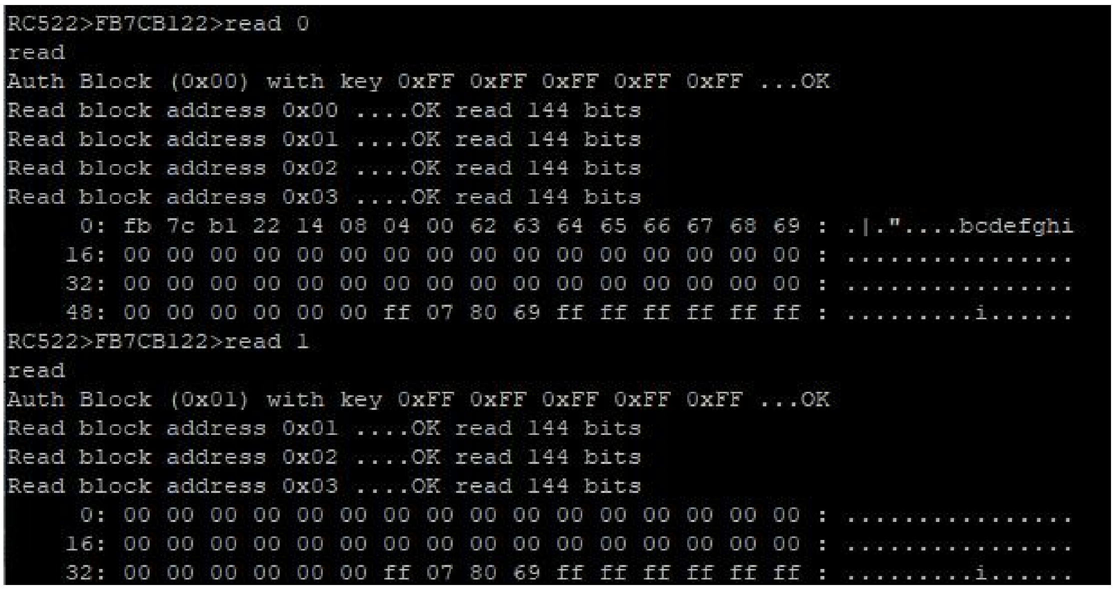
```
RC522>scan
Scanning ...
Card detected!
Card UID: 0xFB 0x7C 0xB1 0x22, Check Sum = 0x14
Card Selected, Type:PICC_TYPE_MIFARE_1K
RC522>FB7CB122>
```

When the Card is placed in the sensing area, you can read and write the card through the following command.



```
Usage:
        read <blockstart>
        dump
        halt
        clean <blockaddr>
        write <blockaddr> <data>
```

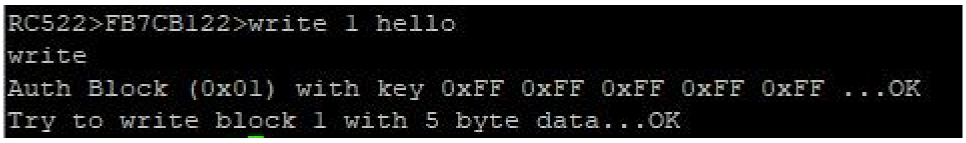
In the command read<blockstart>, the parameter blockstart is the address of the data block, and the range is 0-63. This command is used to display all the data from blockstart address to the end of the sector. For example, sector 0 contains data block 0,1,2,3. Using the command read 0 can display all contents of data block 0,1,2,3. Using the command read 1 can display all contents of data block 1,2,3. As is shown below:


```
RC522>FB7CB122>read 0
read
Auth Block (0x00) with key 0xFF 0xFF 0xFF 0xFF 0xFF ...OK
Read block address 0x00 ....OK read 144 bits
Read block address 0x01 ....OK read 144 bits
Read block address 0x02 ....OK read 144 bits
Read block address 0x03 ....OK read 144 bits
     0: fb 7c b1 22 14 08 04 00 62 63 64 65 66 67 68 69 : .|."....bcdefghi
    16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
    32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
    48: 00 00 00 00 00 00 ff 07 80 69 ff ff ff ff ff ff : .........i......
RC522>FB7CB122>read 1
read
Auth Block (0x01) with key 0xFF 0xFF 0xFF 0xFF 0xFF ...OK
Read block address 0x01 ....OK read 144 bits
Read block address 0x02 ....OK read 144 bits
Read block address 0x03 ....OK read 144 bits
     0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
    16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
    32: 00 00 00 00 00 00 ff 07 80 69 ff ff ff ff ff ff : .........i......
```

Command dump is used to display the content of all data blocks in all sectors.

write <address> <data> is used to write “data" to data block with address “address”. Where the address range is 0-63 and the data length is 0-16. For example, if you want to write the string "hello" to the data block with address “1”, you can type the following command.

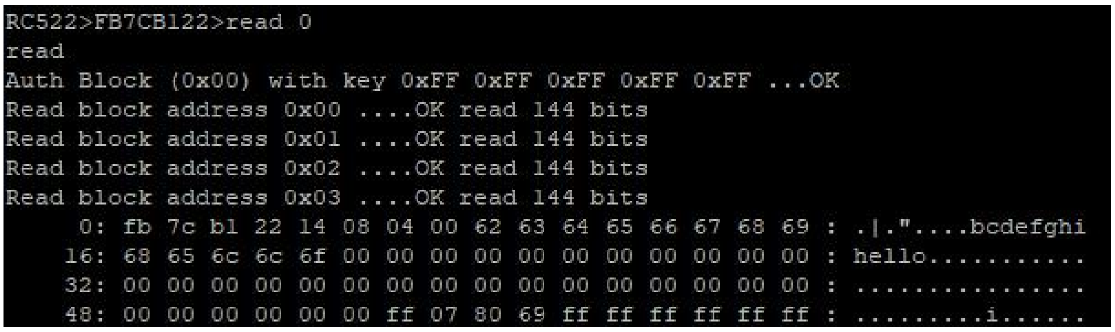
write 1 hello


```
RC522>FB7CB122>write 1 hello
write
Auth Block (0x01) with key 0xFF 0xFF 0xFF 0xFF 0xFF ...OK
Try to write block 1 with 5 byte data...OK
```

Read the contents of this sector and check the data just written.

read 0

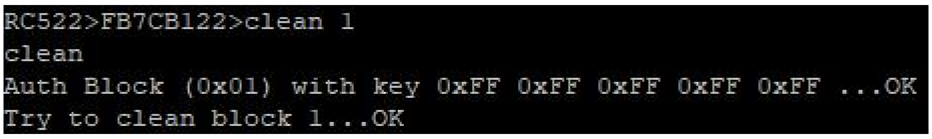
The following results indicate that the string "hello" has been written successfully into the data block 1.



```
RC522>FB7CB122>read 0
read
Auth Block (0x00) with key 0xFF 0xFF 0xFF 0xFF 0xFF ...OK
Read block address 0x00 ....OK read 144 bits
Read block address 0x01 ....OK read 144 bits
Read block address 0x02 ....OK read 144 bits
Read block address 0x03 ....OK read 144 bits
     0: fb 7c b1 22 14 08 04 00 62 63 64 65 66 67 68 69 : .|."....bcdefghi
    16: 68 65 6c 6c 6f 00 00 00 00 00 00 00 00 00 00 00 : hello...........
    32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
    48: 00 00 00 00 00 00 ff 07 80 69 ff ff ff ff ff ff : .........i......
```

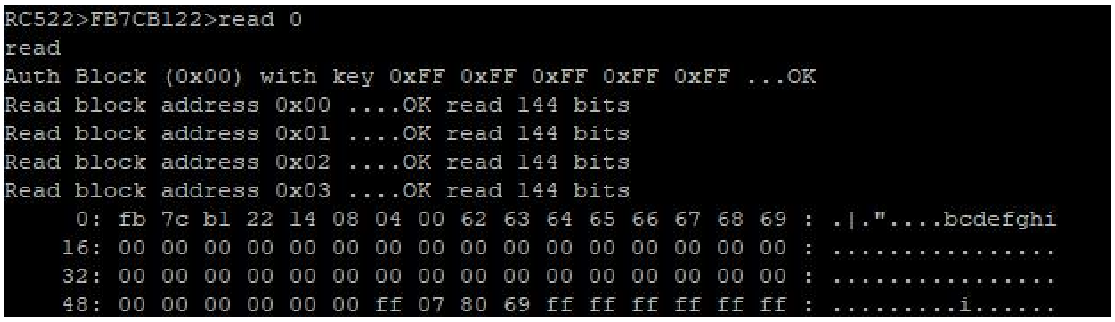
Command clean <address> is used remove the contents of the data block with address "address". For example, if you want to clear the contents of the data block 1 that has just been written, you can type the following command.

clean 1



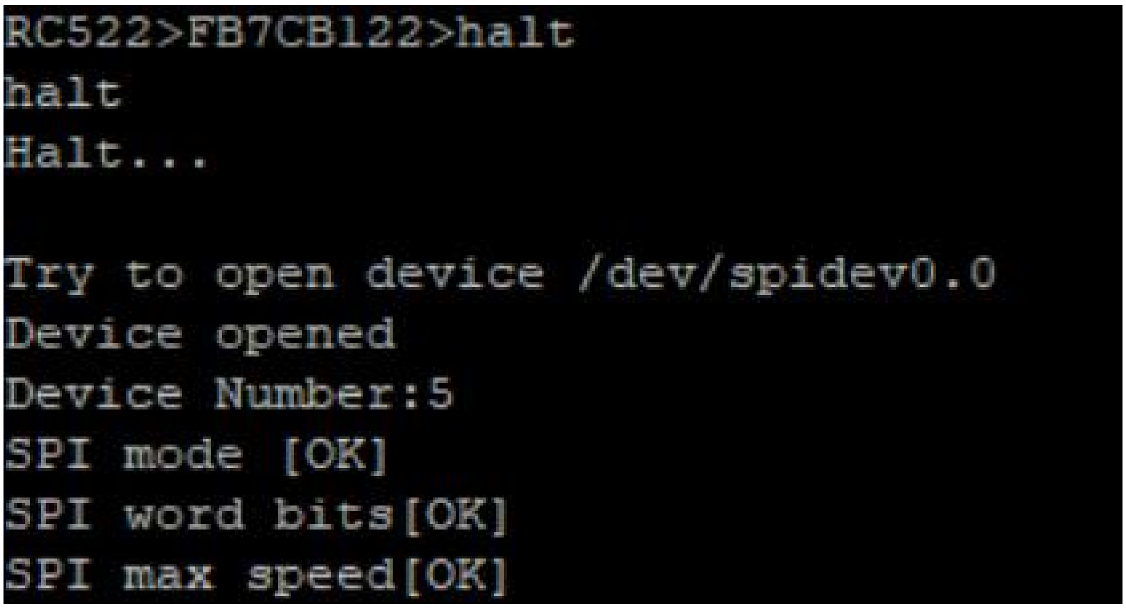
```
RC522>FB7CB122>clean 1
clean
Auth Block (0x01) with key 0xFF 0xFF 0xFF 0xFF 0xFF ...OK
Try to clean block 1...OK
```

Read the contents of data blocks in this sector again to test whether the data is erased. The following results indicate that the contents of data block 1 have been erased.



```
RC522>FB7CB122>read 0
read
Auth Block (0x00) with key 0xFF 0xFF 0xFF 0xFF 0xFF ...OK
Read block address 0x00 ....OK read 144 bits
Read block address 0x01 ....OK read 144 bits
Read block address 0x02 ....OK read 144 bits
Read block address 0x03 ....OK read 144 bits
     0: fb 7c b1 22 14 08 04 00 62 63 64 65 66 67 68 69 : .|."....bcdefghi
    16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
    32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
    48: 00 00 00 00 00 00 ff 07 80 69 ff ff ff ff ff ff : .........i......
```

Command halt is used to quit the selection state of the card.



```
RC522>FB7CB122>halt
halt
Halt...

Try to open device /dev/spidev0.0
Device opened
Device Number:5
SPI mode [OK]
SPI word bits[OK]
SPI max speed[OK]
```