

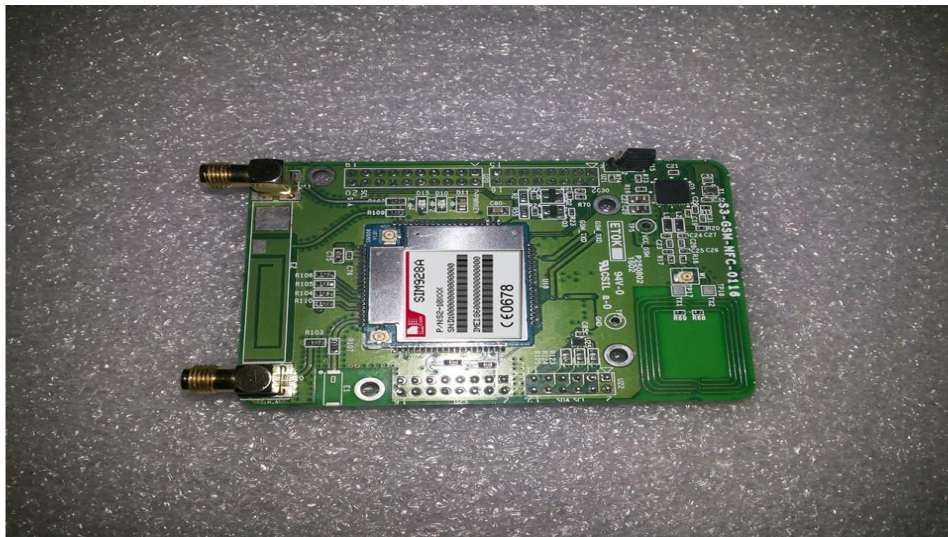
S3-GSM-NFC-0116 Revision 1

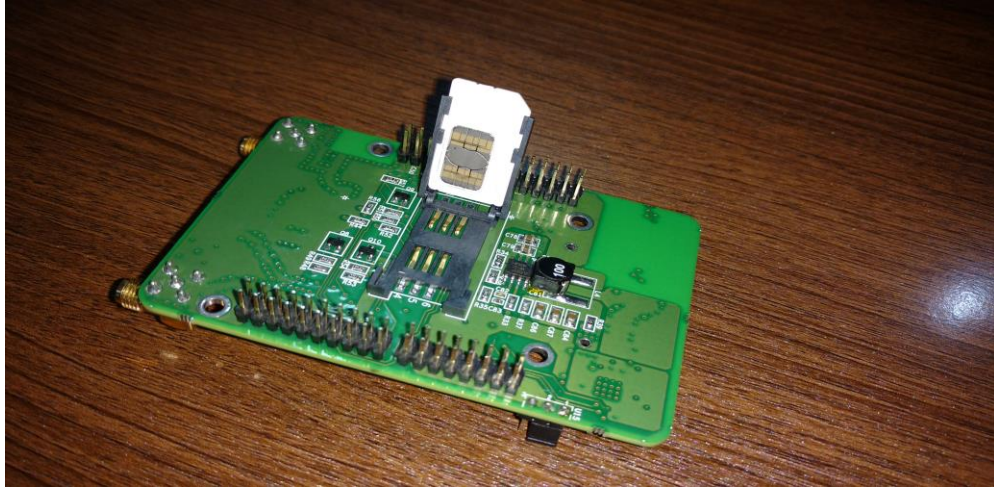
ADD ON SHIELD FOR UDOO, ARDUINO AND RASPBERRY PI.

1. Over view
2. Specification
3. Get started

Overview

It is GSM/GPRS and NFC Shield. You can use your UDOO, Arduino and Raspberry pi boards. GSM/GPRS is a cell phone network with GPRS shield to dial a phone number or send a text to your friend via easy to use AT commands now. This is quad band low power consumption GSM/GPRS and GPS module SIM928A as well as a compact PCB antenna. You can directly interface with your main board to communicate. It is more concise and reliable. We ported together in this module as Near Field Communication (NFC) is a set of short-range wireless technologies. It's behind daily application such as access the control system and mobile payment system. It's a highly integrated transceiver module PN532 which handles contactless communication at 13.56MHz. You can read and write a 13.56MHz tag with this module. To implement point to point data exchange with two NFCs.





Specification

GSM/GPRS:

Working voltage: 5V via Vin pin for UDOO, 5V pin for Arduino and RPi

Compatible: Arduino, UDOO and RPi

Support Interface: UART (HSU)

Communication support: SIMCOM AT commands

Operation temperature: -40C to +80C

NFC:

Working voltage: 3.3V to 5V

Support Interface: I2C and UART

Communication: P2P

Contactless Communication: 13.56MHz.

Operation temperature: -30C to +85C

Get Started:

We are following the steps to test the device with main boards.

In this module can support for Arduino, UDOO and Raspberry pi.

1. Tested to power up the board.

Power up the board from Manual for UDOO:



Pin configuration:

Vin pin of UDOO => Vin pin of Shield

GND pin of UDOO => GND pin of Shield

+5v pin of UDOO => 8th pin of shield

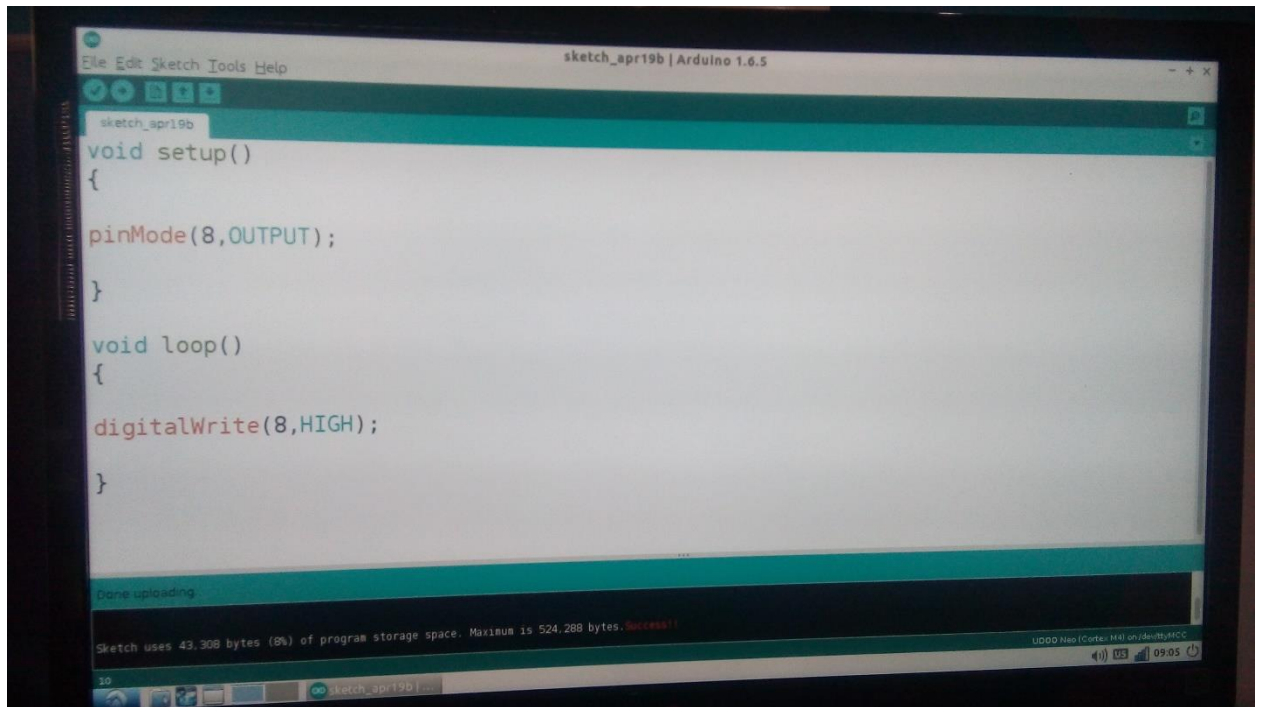
Power up the board from the program for UDOO:



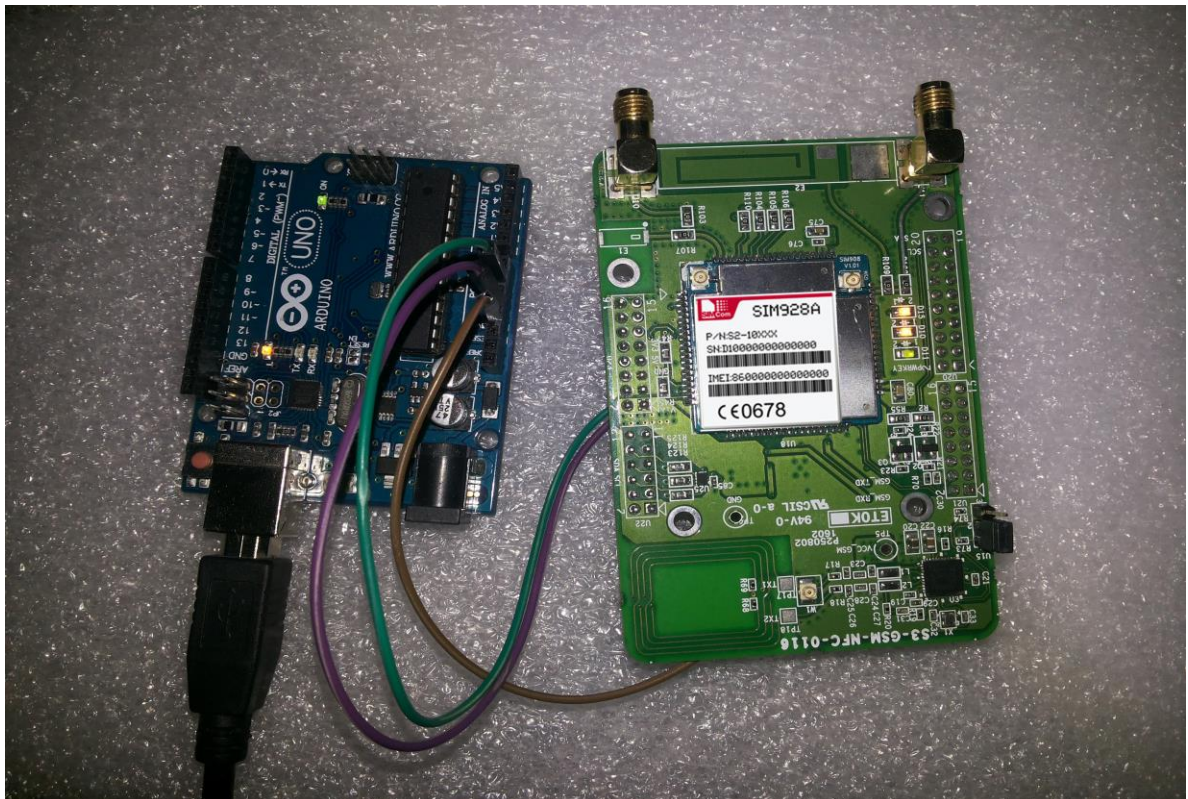
Explanation:

One unit of UDOO board and top of it connect the shield. Then Put the power of UDOO board and connected with HDMI monitor. In this UDOO has Arduino IDE. To open that IDE to Write the code for power up the board. Once the code is successfully uploaded your board will be power up. Given below the code for power up the board.

To open the Arduino IDE -> File -> New once you write the code then compile the and Upload the code. If the code is successful the board will be power up automatically.



Power up the board from Program for Arduino:



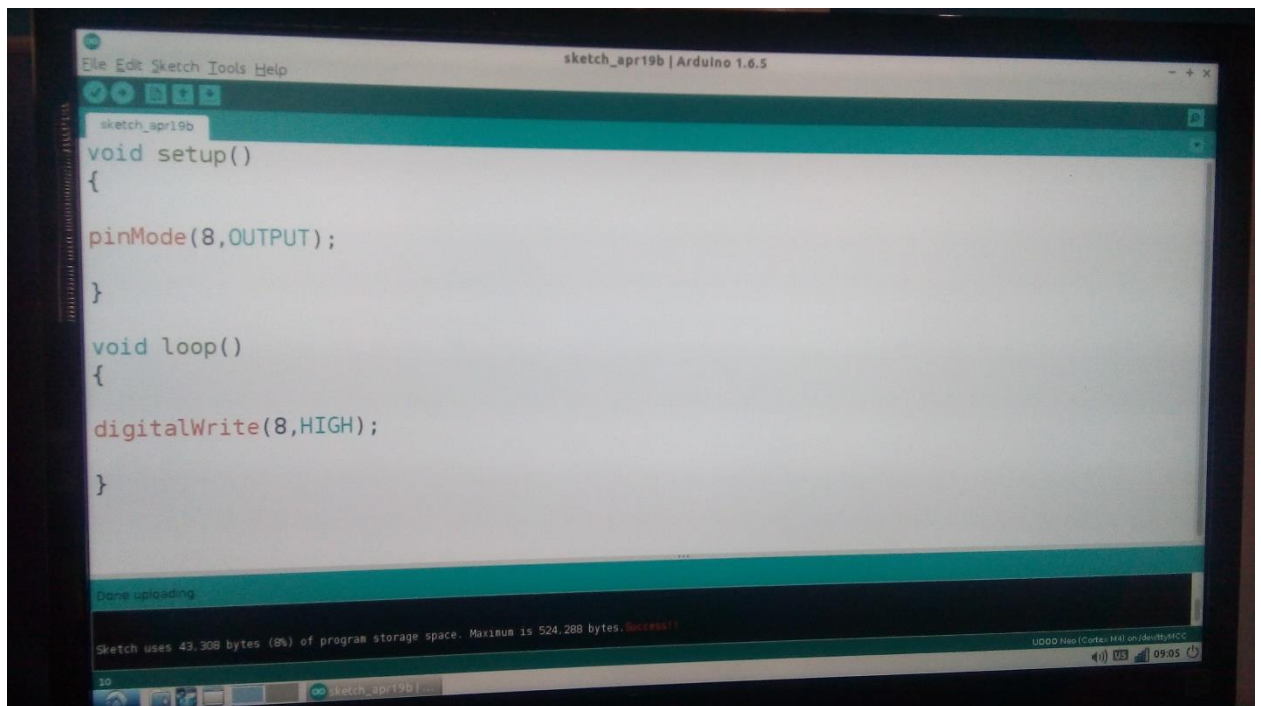
Pin configuration:

+5v pin of Arduino -> Vin pin of shield

GND pin of Arduino -> GND pin of GND

Digital Pin no 8th of Arduino -> 8th pin number of shield

Given below program to power up the board for Arduino:



```
sketch_apr19b | Arduino 1.6.5
File Edit Sketch Tools Help
sketch_apr19b
void setup()
{
  pinMode(8, OUTPUT);
}

void loop()
{
  digitalWrite(8, HIGH);
}

Done uploading
Sketch uses 43,308 bytes (8%) of program storage space. Maximum is 524,288 bytes. Success!
UDOO Neo I/Cortex-M4 on devkitM4C
10 09:05
```

2. Testing for GSM program from shield through UART:

Requirements for testing AT command from shield:

- FDDI chip
- GSM shield
- Minicom for Linux or Tera term for Windows
- Jumper wires



Pin configuration:

Caution: In that FDDI has two type of voltage. We use the jumper as 5v.

Vcc pin of FDDI -> Vin pin of shield

GND pin of FDDI -> GND pin of shield

Rx pin of FDDI -> Rx pin of shield (Manually Rx pin of shield is configured as TX)

Tx pin of FDDI -> Tx pin of shield (Manually Tx pin of shield is configured as Rx)

8th pin of Arduino or UDOO board -> 8th pin of shield (Once the led will glow the board is powered up after that remove the 8th pin and tested with AT command)

You need to install the Minicom software if you are using Linux.

For Linux:

Sudo su

Apt-get install minicom

For windows:

Download the Tera term software.

Once you download the software just connect the serial port cable of FDDI. Then make the connection to set the serial port baud rate as 9600. Once the console window appear U check with AT Command.

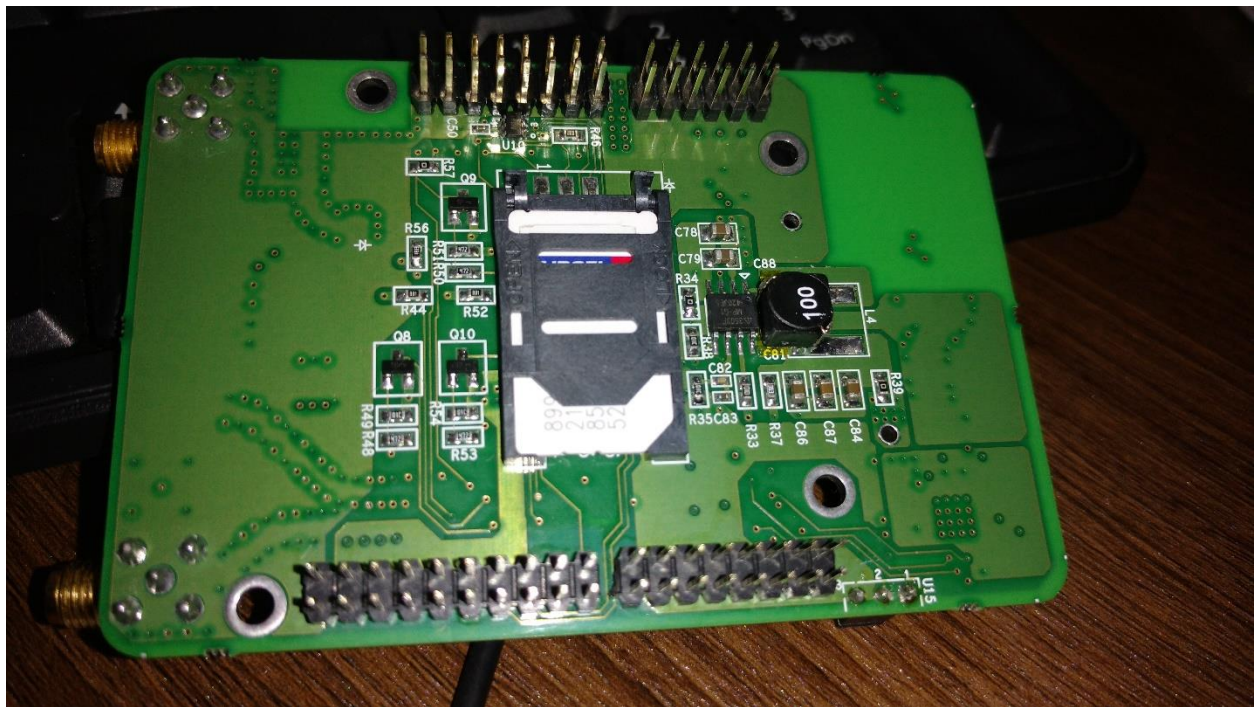
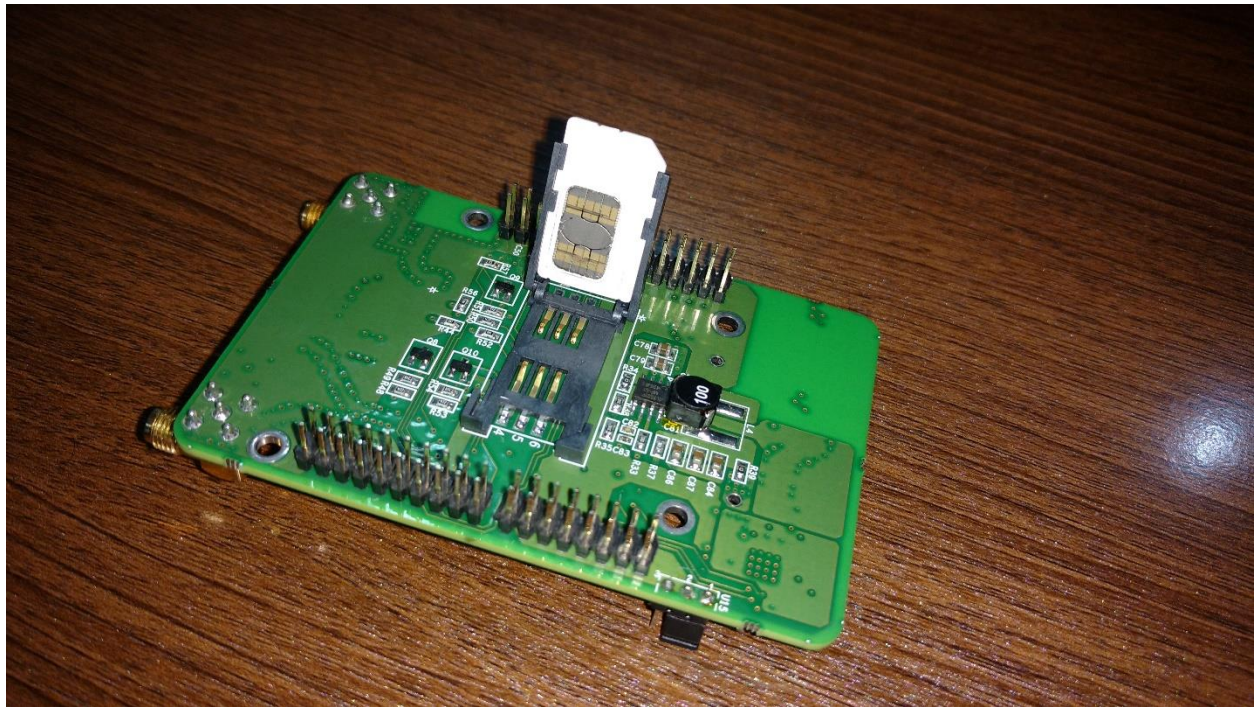
When you type

AT

Ok (response) your board is working fine. After that you proceed with your GSM SMS and Call program.

Way to do the SMS and Call program:

Before you doing program you need to register the sim via AT command with the help of Basic At commands as well as serial debugger of FDDI. After that You can add the shield on the main board. After tested with AT commands. Now you can able to do the SMS and call program. First of all you have to put the sim card in that shield.





Following are the sample program. Program can access through UART In your UDOO UART connection as Serial0.

```
void setup ()
{
  Serial0.begin (9600); //Baud rate of GSM module
  Serial0.print (“\r”);
  delay (1000);
  Serial0.print (“ATD +91*****;\r”); // Which number you want to call
  delay (1000);
  Serial0.write (0x1A);
  delay (5000);
}

Void loop() {
}
```


You will get the output as like this.



```
void setup ()  
{  
  Serial0.begin (9600); //Baud rate of GSM module  
  Serial0.print ("\r");  
  delay (1000);  
  Serial0.print ("AT+CMGF=1\r"); //Enter into text mode  
  delay (1000);  
  Serial0.print ("AT+CMGS=\"+91*****\" \r"); //Number to which you want  
  to send the sms  
  delay (1000);  
  Serial0.print ("Text from UDOO GSM shield to you "); //The text of the message  
  to be sent  
  delay (1000);
```

```
Serial0.write (0x1A);
```

```
delay (5000);
```

```
}
```

```
void loop ()
```

```
{
```

```
}
```

You will get the output as like this.



3. Testing NFC program from i2c protocol.

NFC shield working on UART, I2C as well as SPI based device. PN532 can support for all protocol access. In our shield we have done with i2c protocol. The shield has jumper connection. You need to make the connection of i2c device the pin number 1 and 2 of header u15. Mentioned that pin header as yellow box.



Once you make the header connection then you play the NFC shield with your main board. In this shield can support for Udo0, Arduino as well as Rpi.

- You need to download the PN532 library from github.com in the file of PN532 library. Once you download the library put into the 4 folders are (PN532, PN532_HSU, PN532_I2C, PN532_SPI) into the Arduino's libraries.
- And then download the Don's NDEF library and put it into Arduino's libraries and rename it as NDEF.
- Then Open the Arduino IDE, Click File->Example->Readtag
- We used I2C libraries so make sure that you need to make the header U15 in 1 and 2.
- Then run the code you will get the output.

Notes:

- When you are making connection as UDOO, you use the protocol communication as serial0.
- When you are making connection as Arduino, you use the protocol communication as serial.

Sample program for reading NFC shield through I2C.

UDOO NFC:

```
#include <Wire.h>
```

```
#include <PN532_I2C.h>
```

```
#include <PN532.h>
```

```
#include <NfcAdapter.h>
```



```

PN532_I2C pn532_i2c(Wire);
NfcAdapter nfc = NfcAdapter(pn532_i2c);
void setup(void) {
  Serial0.begin(115200);
  Serial0.println("NDEF Reader");
  nfc.begin();
}
void loop(void) {
  Serial0.println("\nScan a NFC tag\n");
  if (nfc.tagPresent())
  {
    NfcTag tag = nfc.read();
    tag.print();
  }
  delay(5000);
}

```

NFC tag data will be display in Serial monitor.

Arduino NFC:

#include <Wire.h>

#include <PN532_I2C.h>

#include <PN532.h>

#include <NfcAdapter.h>

PN532_I2C pn532_i2c(Wire);

NfcAdapter nfc = NfcAdapter(pn532_i2c);

```
void setup(void) {  
Serial.begin(115200);  
Serial.println("NDEF Reader");  
nfc.begin();  
}  
void loop(void) {  
Serial.println("\nScan a NFC tag\n");  
if (nfc.tagPresent())  
{  
NfcTag tag = nfc.read();  
tag.print();  
}  
delay(5000);  
}
```

You will get the output from serial monitor.