Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

NFC Module for Arduino (SKU:DFR0231)

NFC Module for Arduino

Introduction

Your credit cards: gone. Bus pass and train tickets: vanished.Welcome to Near Field Communications (NFC), a contactless, Wi-Fi-lite style tech that could already be in your smartphone, and could soon be a regular feature of your commute.

Near Field Communications(NFC) is a set of short-range wireless technologies, typically requiring a distance of 10cm or less, for two devices such as smartphones or the similar things very close to each other to establish communication. Communication is also possible between a NFC device and unpowered NFC chips such as tags, stickers, key fobs and cards which do not require batteries.

The NFC Module for Arduino is designed to extend this powerful feature for your project or application based on Arduino.It integrates a PN532 NFC controller from Philips.The driver interface for this product is UART interface of the microcontroller. So it's possible for you to test it via a USB to UART converter directly. On the other hand, for the applications with microcontroller, the module provides an event for your processor when detects the NFC tags, stickers, key fobs, or cards via high speed serial uart.

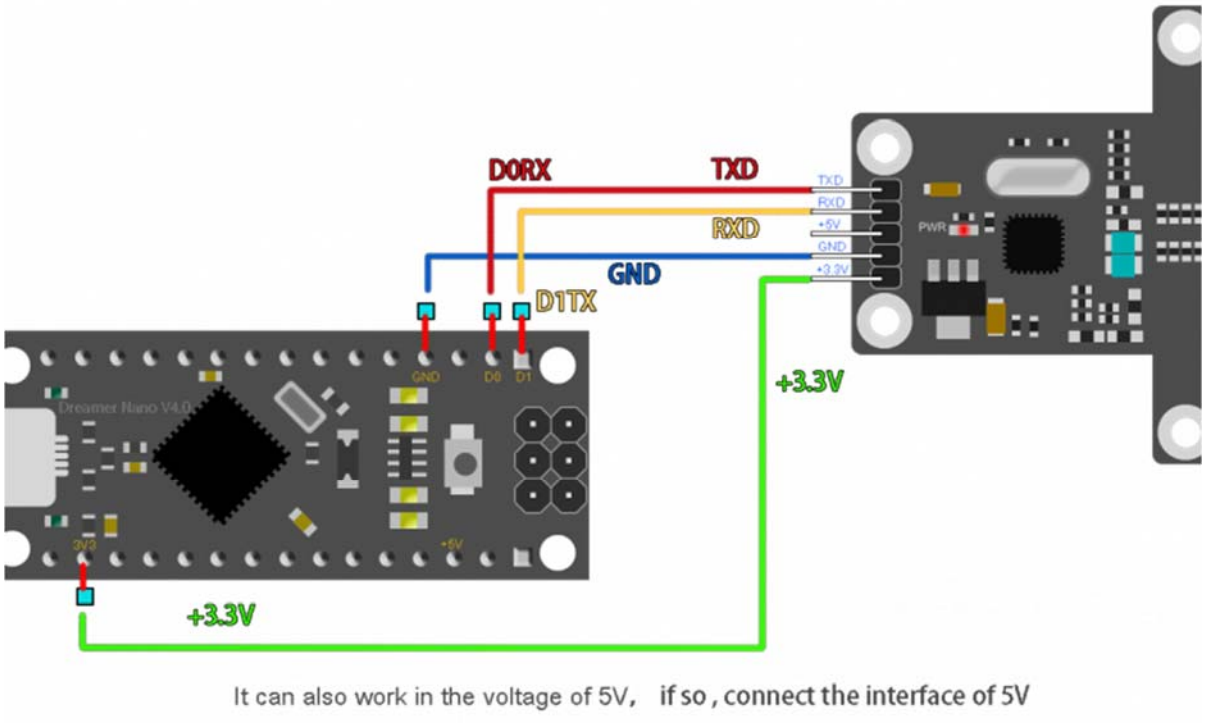
Applications

- Contactless payment systems
- Bluetooth and Wi-Fi connections
- Social networking situations, such as sharing contacts, photos, videos or files
- Electronic identity documents and keycards
- Smartphone automation and NFC tags

Specification

- Working Voltage: 3.3v or 5v
- Host interface: UART
- Integrate PN532 NFC controller
- Supports ISO 14443A/MIFARE
- Supports ISO 14443B in reader/writer mode only
- Typical max operating distance for detecting NFC tags from 20 to 50mm depending on the antenna *size of the tag
- Serve for contactless communication at 13.56MHz
- Size: 11cm x 5cm

Connect Diagram



NFC Module connect diagram

Sample Code

```
/*


 # Editor : Adrian

 # Date   : 2013.04.18

 # Ver    : 0.1

 # Product: NFC Module for Arduino

 # SKU    : DFR0231


 # Description:

 # When the a card close to the device , the PC will receive the data

 # Connect the NFC Card's TXD, RXD, GND, +3.3V to Nano's D0RX, D1TX, GND, +3.
3V

 # Or connect the NFC Card's TXD, RXD, GND, +5V to Nano's D0RX, D1TX, GND, +5
V



 PN532 reads the tag by Arduino mega/Leonardo

 command list:


 #wake up reader

 send: 55 55 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ff 03 fd d4 14 01 17 0
0

 return: 00 00 FF 00 FF 00 00 00 FF 02 FE D5 15 16 00


 #get firmware

 send: 00 00 FF 02 FE D4 02 2A 00

 return: 00 00 FF 00 FF 00 00 00 FF 06 FA D5 03 32 01 06 07 E8 00


 #read the tag

 send: 00 00 FF 04 FC D4 4A 01 00 E1 00

 return: 00 00 FF 00 FF 00 00 00 FF 0C F4 D5 4B 01 01 00 04 08 04 XX XX XX XX
5A 00

 XX is tag.
```

```
 */


const unsigned char wake[24]={
  0x55, 0x55, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, \
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0x03, 0xfd, 0xd4, 0x14, 0x01, 0x17,
0x00};//wake up NFC module
const unsigned char firmware[9]={
  0x00, 0x00, 0xFF, 0x02, 0xFE, 0xD4, 0x02, 0x2A, 0x00};//
const unsigned char tag[11]={
  0x00, 0x00, 0xFF, 0x04, 0xFC, 0xD4, 0x4A, 0x01, 0x00, 0xE1, 0x00};//detecti
ng tag command
const unsigned char std_ACK[25] = {
  0x00, 0x00, 0xFF, 0x00, 0xFF, 0x00, 0x00, 0x00, 0xFF, 0x0C, \
0xF4, 0xD5, 0x4B, 0x01, 0x01, 0x00, 0x04, 0x08, 0x04, 0x00, 0x00, 0x00, 0x00,
0x4b, 0x00};
unsigned char old_id[5];


unsigned char receive_ACK[25];//Command receiving buffer
//int inByte = 0;               //incoming serial byte buffer


#if defined(ARDUINO) && ARDUINO >= 100
#include "Arduino.h"
#define print1Byte(args) Serial1.write(args)
#define print1lnByte(args)  Serial1.write(args),Serial1.println()
#else
#include "WProgram.h"
#define print1Byte(args) Serial1.print(args,BYTE)
#define print1lnByte(args)  Serial1.println(args,BYTE)
#endif


void setup(){
  Serial.begin(9600);  // open serial with PC
  Serial1.begin(115200);       //open serial1 with device
  //Serial2.begin(115200);
  wake_card();
  delay(100);
```

```
  read_ACK(15);
  delay(100);
  display(15);
}

void loop(){
  send_tag();
  read_ACK(25);
  delay(100);
  if (!cmp_id ()) {
    if (test_ACK ()) {
      display (25);
      delay (100);
    }
  }
  copy_id ();
}

void copy_id (void) {//save old id
  int ai, oi;
  for (oi=0, ai=19; oi<5; oi++,ai++) {
    old_id[oi] = receive_ACK[ai];
  }
}

char cmp_id (void){//return true if find id is old
  int ai, oi;
  for (oi=0,ai=19; oi<5; oi++,ai++) {
    if (old_id[oi] != receive_ACK[ai])
      return 0;
  }
  return 1;
}
```

```
int test_ACK (void) {// return true if receive_ACK accord with std_ACK
  int i;
  for (i=0; i<19; i++) {
    if (receive_ACK[i] != std_ACK[i])
      return 0;
  }
  return 1;
}

void send_id (void) {//send id to PC
  int i;
  Serial.print ("ID: ");
  for (i=19; i<= 23; i++) {
    Serial.print (receive_ACK[i], HEX);
    Serial.print (" ");
  }
  Serial.println ();
}

void UART1_Send_Byte(unsigned char command_data){//send byte to device
  print1Byte(command_data);
#if defined(ARDUINO) && ARDUINO >= 100
  Serial1.flush();// complete the transmission of outgoing serial data
#endif
}

void UART_Send_Byte(unsigned char command_data){//send byte to PC
  Serial.print(command_data,HEX);
  Serial.print(" ");
}

void read_ACK(unsigned char temp){//read ACK into reveive_ACK[]
  unsigned char i;
  for(i=0;i<temp;i++) {
```

```
    receive_ACK[i]= Serial1.read();
  }
}

void wake_card(void){//send wake[] to device
  unsigned char i;
  for(i=0;i<24;i++) //send command
    UART1_Send_Byte(wake[i]);
}

void firmware_version(void){//send fireware[] to device
  unsigned char i;
  for(i=0;i<9;i++) //send command
    UART1_Send_Byte(firmware[i]);
}

void send_tag(void){//send tag[] to device
  unsigned char i;
  for(i=0;i<11;i++) //send command
    UART1_Send_Byte(tag[i]);
}

void display(unsigned char tem){//send receive_ACK[] to PC
  unsigned char i;
  for(i=0;i<tem;i++) //send command
    UART_Send_Byte(receive_ACK[i]);
  Serial.println();
}
```