NFC Application Interface for Smart Phones and Appliances

Jingyu Lin, Hamid Shahnasser
School of Engineering
San Francisco State University
San Francisco, United State of America
jingyu@mail.sfsu.edu

Abstract—**Near Field Communication (NFC) is an emerging short-range wireless communication protocol based on Radio Frequency Identification (RFID) technology. This paper demonstrates a design of NFC application interface for NFC-enabled smart phones and appliances. The object of this design is to provide an NFC interface for appliances and devices, simplifying the development of NFC-mobile application. The interface consists of two components. One is NXP PN532 module, which is a highly integrated transceiver module for NFC communication, and the second is the Arduino platform that is an open-source electronics prototyping platform based on ATmega328 Microcontroller. Arduino platform uses C++ language which makes the development work faster and easier.**

Keywords—**NFC, Android OS, Arduino, Smart phone, Appliances access control**

I. INTRODUCTION

Mobile phones have become a pervasive platform of communication, entertainment and computation. There are over 2 billion people who own mobile phones around the world [1]. With current wireless technologies, like WIFI and Bluetooth, mobile phones have the ability to interact with other devices to execute data exchange and access control. For example exchanging text message, business personal cards, and controlling appliances like air conditioner and TV sets.

Currently, a new short-range technology is introduced in mobile phones called Near Field Communication (NFC). It enables two devices to communicate with each other by touching each other. The establishment of connection is simple without device searching and pairing. NFC is based on Radio Frequency Identification (RFID) technology. One advantage of NFC is that it enables a device to emulate either a tag or a tag reader. NFC can be applied for applications such as: ticketing, payment, access control, smart poster, and peer-to-peer data exchange. NFC simplifies the communication and data transfer between wireless devices. Thus, NFC technology could be used for various applications. Considerable research has been carried out to explore the possible NFC applications and their implementation for example: NFC-enabled health monitoring system [2], measurement system for remote monitoring and control [3] and other mobile application. Besides the above,NFC is a promising technology, which can be applied to mobile payment as well [4].

In this paper, we present a design of a universal NFC interface module for appliances and smart phones. Designers and researchers can adopt this interface to connect to any specific appliances and devices for implementing NFC-enabled mobile applications.

II. INTERFACE ARCHITECTURE

The interface consists of two hardware components: an Arduino UNO board [5] and an NFC transmission module. This design uses NFC shield [6], which utilizes NXP PN532 integrated circuit [7], as the NFC transmission module. NFC shield connects to Arduino UNO board through Serial Peripheral Interface (SPI).

The Arduino UNO board acts as a host controller to indicate the NFC shield to receive or transmit data. It provides 14 digital I/O Pins and supports I2C and RS-232 interfaces for implementing access control, or data stream exchange to the appliances. By means of these interfaces, the communication protocol between the appliance and the NFC interface module can be customized by designers. These factors lead to the development of the NFC applications becoming easier. Fig. 1 illustrates the architecture of NFC interface module.

In this design, the NFC shield will be configured to emulate an ISO/IEC 14443-4 card, Proximity IC Card (PICC), and the NFC-enabled smart phone acts as a card ISO14443-4 reader/writer, Proximity Coupling Device (Contactless PCD). The communication between an NFC-enabled smart phone and the NFC interface module is the process of reading/writing messages from or to an RFID tag. The following sections will demonstrate the implementation of the interface and testing this design by using the smart phone to control and measure a fan's revolutions per minute (RPM) through this interface.

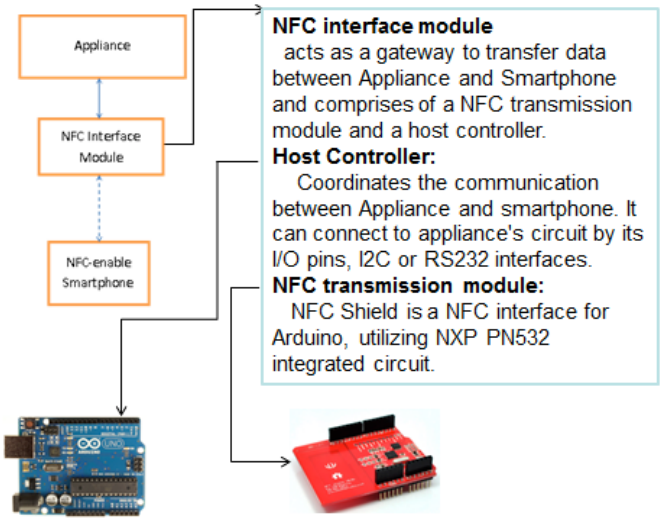

Fig. 1. The architecture of the NFC interface module

III. IMPLEMENTATION

The implementation is focused on designing a program in the host controller. The program commands the NFC shield to receive and transmit data when the interface establishes connection with the smart phone.

In the ISO/IEC14443-4 card emulation mode, all the commands (C-APDU) [8] coming from the ISO/IEC14443-4 PCD are transmitted through the NFC shield to the host controller [7]. The host controller gives a response by utilizing the command R-APDU which will be sent to the PCD through the NFC shield. The communication between Arduino platform and the NFC shield is performed by normal information frame. Its payload carries the APDU. The program calls APIs to invoke PN532 firmware commands to operate the interface. Three primary commands, which are used by the host controller, are TgInitAsTarget, TgGetData and TgSetData. Command TgInitAsTarget configures the NFC shield to act as a target; command TgGetData is used to read the data received by the NFC shield from the PCD, and command TgSetData is used to send the data to the NFC shield and forward it to the PCD. The program will initialize the NFC shield working mode by sending command TgInitAsTarget before indicating the NFC shield to execute command TgGetData or TgSetData. The program flowchart is shown in Fig. 2.

The method configurePeerAsTarget() is modified to implement the TgInitAsTarget command for configuring the NFC shield to emulate ISO/IEC 14443-4 card mode[9], Fig. 3 depicts the implementation of the configurePeerAsTarget() method and Fig 4 shows its pseudocode. The pseudocode of program and its implementation in C/C++ are shown in Fig. 5 and Fig.6 respectively. The interfaces targetRx() and targetTx(), which are shown in Fig. 4, implement the commands TgGetData and TgSetData respectively. The implementation of these two function are from function TargetTxRx(). The C-APDU is returned by targetRx(). The R-APDU which is sent to PCD is carried by targetTx().

IV. TESTING

A demo application is designed to test the implementation of communication between the NFC interface module and the smart phone. An application running on Android OS is used to measure and control a fan's RPM. Fig. 7 shows these devices. Google supports NFC application and its development in Android operating system. It provides sets of APIs to handle different kind of tag technologies I/O operation. The design adopted the class IsoDep[10] to develop ISO/IEC 14443-4 tag

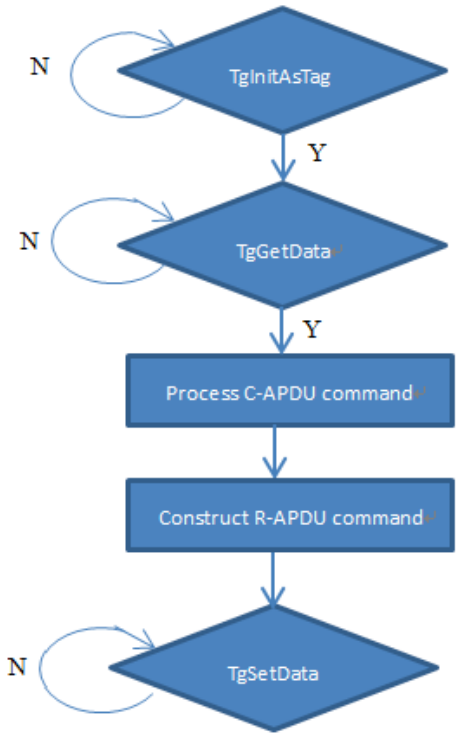

Fig.2. The host controller program flowchart

```
initlize a array with emulating target parameters
construct normal frame by using array
execute TgGetData command

if recevie acknowledgement
  return true
else
  return false

return data returned by TgGetData command
```

Fig.4. Pseudocode of function configurePeerAsTarget

```
send command TgInitAsTarget
if NFC interface can emulate a target
    send command TgGetData
    if received ack frame
       read command returned by TgGetData

      if command is READ
          assgin data to a array dataout with ack

      if command is WRITE
          read the data form array datain
          write ack to array dataout

      send command TgSetData
```

Fig. 5. Pseudocode of the program

```
uint32_t PN532::configurePeerAsTarget() {

    byte pbuffer[38] = { PN532_TGINITASTARGET, //0x8C
        0x01, /* 0x04, //Mode (PICC only) */
              //----------- MifareParams[] 6 bytes
        0x08, 0x00, //SENS_RES ATQA
        0x12, 0x34, 0x56, //NFCID1 (UID)
        // Data Exchange Protocol (DEP) or for ISO/IEC14443-4
        // protocol when PN532 is activated in ISO/IEC14443-4 PICC emulated
        0x20, //0x20 (for ISO/IEC14443-4 PICCemulation)
        // -------------FeliCaParams[] 18 bytes
        0x01, 0xFE, 0xA2, 0xA3, 0xA4, 0xA5, 0xA6, 0xA7, // POL_RES
        0xC0, 0xC1, 0xC2, 0xC3, 0xC4, 0xC5, 0xC6, 0xC7,
        0xFF, 0xFF,
        //----------NFCID3t(10 bytes)
        0xAA, 0x99, 0x88, 0x77, 0x66, 0x55, 0x44, 0x33, 0x22, 0x11
        0x00, 0x00 //Length of general and historical bytes
        };

    for(uint8_t iter=0;iter<38;iter++)
    {
        pn532_packetbuffer[iter] = pbuffer[iter];
    }

    if (! sendCommandCheckAck(pn532_packetbuffer, 38))
        return false;
    // read data packet
    readspidata(pn532_packetbuffer, 12);

    return true;
}
```

Fig.3. The implementation of function configurePeerAsTarget

```
void loop(void){
 if (configTarget){
  // Configure PN532 as Peer to Peer Target
  // If connection is error-free
  if(nfc.configurePeerAsTarget()){
   Serial.println("configurePeerAsTarget done");
  }
  int l = nfc.targetRx(&DataIn[0], (byte)lenIn);
  if (l > 0){
  //do not send configurePeerAsTarget
  //any more (after first data received)
  configTarget = false;
  //check received command
  int iCommand = checkCommand((char*)&DataIn[0]);
  byte lenOut;
  if (iCommand == READ){
   //process sending data to reader, 0x90 0x00 = OK
   memcpy(&DataOut[iLen], "\x90\x00", lenOut = 2);
   lenOut = iLen + lenOut;;
 }
 else if(iCommand == WRITE){
  //process the data and send ackonwledgement back
  memcpy(DataOut, "\x90\x00", lenOut = 2);
 }
 else{
  //send a R-PDU show receiving a unkown command
  memcpy(DataOut, "\x6a\x82", lenOut = 2);
 }
 if (nfc.targetTx(DataOut, lenOut)){ //send data to PCD
              Serial.print("Data sent: \r\n ");}
 }
 else{configTarget = true;}
}
```

Fig.6. The host controller program

access program. In the smart phone application, the command and data whose format is C-APDU.

Fig. 8 shows the GUI of the Android OS application. The seek bar is used to change the RPM. If the set button is pressed before the smart phone touches the NFC interface, it will invoke write command to send a new RPM value to the host controller. The read command is called to receive the current fan's RPM which will be displayed in the edit box. The test case indicated that the design of the NFC interface module for the smart phone and the appliance is implemented and the communication between host controller and smart phone is succeeded.

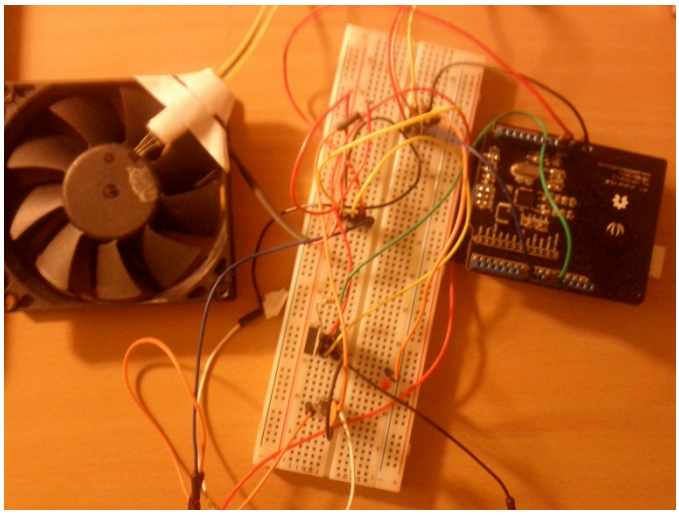

Fig. 7. Testing circuit

V. CONCLUSION

In this paper, we presented a NFC interface by using Arduino open-source platform and NXP commercial NFC module. Arduino UNO board is rich in I/O interfaces and wired communication serial buses. The program runs on the Arduino and it is easy to use for tailoring, developing and testing interfaces.

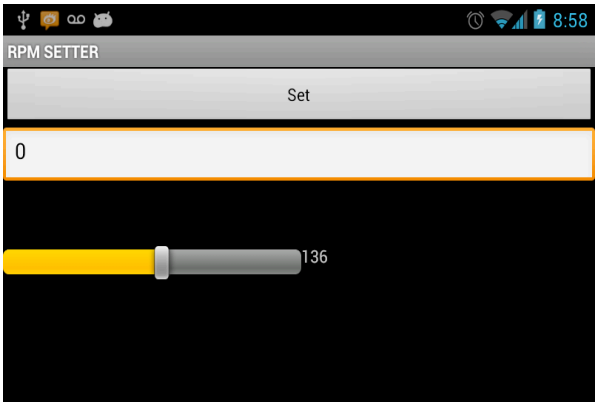

Fig. 8. Android OS application's GUI

REFERENCES

[1] Nokia: 2 billion cell phone users by 2006, http://news.com.com/2102-1039 3-5485543.html

[2] Esko Strömmer, Jouni Kaartinen, Juha Pärkkä, Arto Ylisaukko-oja, Ilkka Korhonen: Application of Near Field Communication for Health Monitoring in Daily Life, Proceedings of the 28th IEEE EMBS Annual International Conference, pp. 3246-3249, New York City(2006)

[3] Charl Opperman, Gerhard Hancke: A Generic NFC-enabled Measurement System for Remote Monitoring and Control of Client-side Equipment, 2011 Third International Workshop on Near Field Communication, pp. 44-49, Hagenberg(2011)

[4] Sunil Timalsina, Rabin Bhusal, and Sangman Moh: NFC and Its Application to Mobile Payment: Overview and Comparison. In: 2012 8th International Conference, Information Science and Digital Content Technology (ICIDT), vol. 1, pp. 203-206, Jeju(2012)

[5] Arduino UNO Board Feature, http://arduino.cc/en/Main/ArduinoBoardUno

[6] NFC Shield Feature and PN532 Library, http://www.seeedstudio.com/wiki/NFC_Shield

[7] PN532 User Manual, NXP semiconductor

[8] Type 4 Tag Operation Specification, NFC Forum

[9] PN532 C106 application note, NXP semiconductor

[10] IsoDep Class Overview, http://developer.android.com/reference/android/nfc/tech/IsoDep.html