Near Field Communication

by Prof.Dr. Raphael Ruf at
Hochschule Ravensburg-Weingarten

Task 1: Data Write NFC Card

Authors:

Sai Charith Sathyanarayana (13945724)
Arvind Chinni Ravi (22145673)

Date of Submission:
November 20th, 2025

Table of Contents:

1. Introduction

The NFC Module V3 (PN532) is a versatile, highly integrated transceiver operating at 13.56 MHz that supports read/write operation, card emulation, and peer-to-peer modes. It delivers high throughput up to 424 kbit/s in both directions, while handling framing, parity, CRC, and modulation internally for robust, error-resilient communication. Its flexible host interfaces (SPI, I²C, and high-speed UART) simplify integration with microcontrollers such as the Arduino Nano. These features make the NFC V3 (PN532) ideal for embedded systems requiring secure, low-latency contactless data exchange and easy hardware interfacing for rapid prototyping and deployment. We designed the system to sequentially scan each of the 16 sectors and every block, ensuring comprehensive data recovery.

2. Methodology

The NFC card reading workflow comprises the following technical steps:

- **Inductive Power Transfer:** Passive NFC transponders are energised via inductive coupling to the reader's alternating magnetic field (13.56 MHz). The card's antenna coil rectifies and supplies power to the onboard IC, eliminating the need for a local battery.
- **Transponder Initialisation(IC):** The harvested energy triggers the NFC IC's poweron sequence and internal clock, bringing the protocol stack and security logic online.
- **RF Link Establishment:** The reader and tag negotiate an RF link using HF modulation schemes; carrier coupling and subcarrier modulation form the physical-layer connection.
- **Frame-level Communication:** The tag and reader exchange protocol frames (including UID and payload) with built-in parity and CRC checking to ensure data integrity.
- **Data Demodulation & Parsing:** The PN532-based receiver demodulates incoming signals and decodes frame payloads, extracting UID and memory block data.
- **Host Interface Transfer (SPI):** Decoded data is forwarded to the Arduino Nano over the Serial Peripheral Interface (SPI), using master/slave clocked transfers for deterministic throughput.
- **Application Processing & Visualisation:** The Arduino validates and parses sector/block contents, performs authentication as required, and outputs human-readable results to the serial monitor for logging and debugging.

3. Algorithm

3.1 Setup phase

- Start Serial communication at 115200 baud.
- Print welcome messages and instructions to the user.
- Initialize the PN532 NFC reader (nfcReader.begin()).
- Request the PN532 firmware version.
- If firmware is not detected: Print an error message and stop program execution.
- If firmware is detected: Print the PN532 version information.
- Configure PN532 in SAM (Secure Access Module) mode.
- Display a message telling the user to enter a message to write**.**

3.2 Loop phase

- Check for user input.
- If Serial input is available: Read the message.
- If the message is empty, prompt the user to enter again.
- Otherwise, display the message and ask the user to place the card.
- Detect NFC card.
- Attempt to read a passive ISO14443A card.
- If no card is detected, continue looping.
- If a card is detected and a message is stored: Display "Card detected".
- Move to the Write Phase.

3.3 Write phase

- Load the default MIFARE Classic Key A: 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF.
- Attempt authentication of block 4 using this key.
- If authentication fails:
    - Display "Authentication failed".
    - Clear the message and return to the Loop Phase.
- If authentication succeeds:
    - Create a 16-byte buffer (dataBlock).
    - Copy user message into the buffer (truncate if >16 chars).
    - Fill remaining bytes with zeros.
- Write the dataBlock to block 4.
- If write succeeds: Display "Message written to Card".
- If write fails: Display "Write failed".
- Inform the user that they can type a new message.
- Clear the stored message and return to Loop Phase.

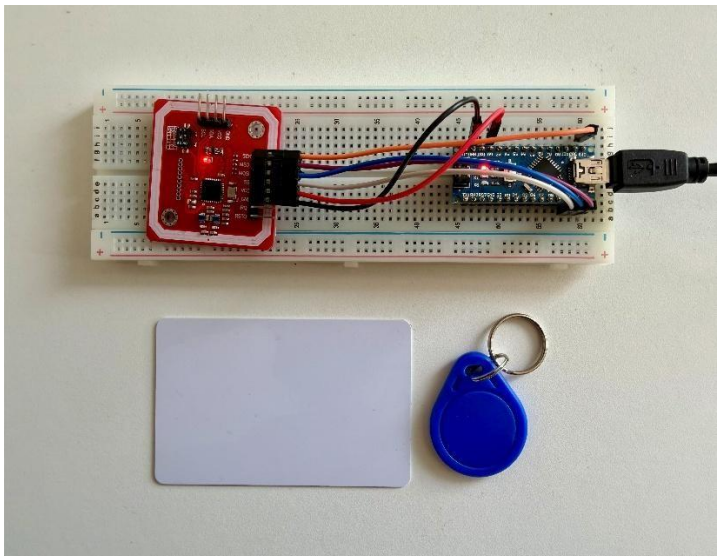
4. Program

```
#include <SPI.h>
#include <Adafruit_PN532.h>

// SPI pins for Arduino
#define PN532_SCK  13
#define PN532_MOSI 11
#define PN532_MISO 12
#define PN532_SS   10

Adafruit_PN532 nfcReader(PN532_SS);

String messageToWrite = "";

void setup() {
  Serial.begin(115200);
  Serial.println("NFC Writer");
  Serial.println("Type your message and press ENTER:");

  nfcReader.begin();

  uint32_t firmware = nfcReader.getFirmwareVersion();
  if (!firmware) {
    Serial.println("PN532 board not found.");
    while (1);
  }

  Serial.print("Found PN5"); Serial.println((firmware >> 24) & 0xFF, HEX);
  Serial.print("Firmware version: ");
  Serial.print((firmware >> 16) & 0xFF, DEC);
  Serial.print('.');
  Serial.println((firmware >> 8) & 0xFF, DEC);

  nfcReader.SAMConfig();
  Serial.println("Enter a message to write to a tag:");
}

void loop() {
  // Read user input from Serial Monitor
  if (Serial.available()) {
    messageToWrite = Serial.readStringUntil('\n');
    messageToWrite.trim();

    if (messageToWrite.length() == 0) {
      Serial.println("Empty message. Try again.");
      return;
    }

    Serial.print("Message received: ");
    Serial.println(messageToWrite);
    Serial.println("Place your Mifare Classic card near the reader...");
  }
```

```
  // Detect MIFARE Classic card
  uint8_t tagUID[7];
  uint8_t uidLength;
  bool tagDetected = nfcReader.readPassiveTargetID(PN532_MIFARE_ISO14443A, tagUID,
&uidLength);

  if (tagDetected && messageToWrite.length() > 0) {
    Serial.println("Tag detected.");

    // Authenticate block 4 with default key
    uint8_t defaultKey[6] = { 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF };
    bool authSuccess = nfcReader.mifareclassic_AuthenticateBlock(tagUID,
uidLength, 8, 0, defaultKey);

    if (authSuccess) {
      Serial.println("Authentication successful.");

      uint8_t dataBlock[16] = {0};
      for (int i = 0; i < messageToWrite.length() && i < 16; i++) {
        dataBlock[i] = messageToWrite[i];
      }

      if (nfcReader.mifareclassic_WriteDataBlock(8, dataBlock)) {
        Serial.println("Message written to Card.");
      } else {
        Serial.println("Write failed.");
      }
    } else {
      Serial.println("Authentication failed.");
    }

    Serial.println("\nYou can type a new message to write another Card.");
    messageToWrite = ""; // reset for next input
  }
}
```

5. Result

5.1 Connection



5.2 Output

```
NFC Writer
Type your message and press ENTER:
Found PN532
Firmware version: 1.6
Enter a message to write to a tag:
Message received: Hallo, Danke!
Place your Mifare Classic card near the reader...
Tag detected.
Authentication successful.
Message written to Card.

You can type a new message to write another Card.

Searching for PN532...
Found PN532
Firmware version: 1.6
Waiting for a MIFARE Classic card...

Found NFC Chip PN532
```

UID Length: 4 bytes
UID Value: 0x53 0x6C 0xA1 0x15

This is a Mifare classic card.

Sector 0:
Block 0: 0x53 0x6C 0xA1 0x15 0x8B 0x08 0x04 0x00 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69
Block 1: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Block 2: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Block 3: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 1:
Block 4: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Block 5: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Block 6: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Block 7: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 2:
Block 8: 0x48 0x61 0x6C 0x6C 0x6F 0x2C 0x20 0x44 0x61 0x6E 0x6B 0x65 0x21 0x00 0x00 0x00
Block 9: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Block 10: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
Block 11: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 3:

Block 12: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 13: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 14: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 15: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 4:

Block 16: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 17: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 18: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 19: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 5:

Block 20: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 21: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 22: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 23: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 6:

Block 24: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 25: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 26: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 27: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 7:

Block 28: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 29: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 30: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 31: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 8:

Block 32: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 33: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 34: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 35: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 9:

Block 36: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 37: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 38: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 39: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 10:

Block 40: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 41: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 42: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 43: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 11:

Block 44: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 45: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 46: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 47: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 12:

Block 48: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 49: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 50: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 51: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0Xff

Sector 13:

Block 52: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 53: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 54: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Block 55: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

Sector 14:

Block 56: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00

Block 57: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00

Block 58: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00

Block 59: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF
0xFF 0xFF

Sector 15:

Block 60: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00

Block 61: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00

Block 62: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00

Block 63: 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0x07 0x80 0x69 0xFF 0xFF 0xFF 0xFF
0xFF 0xFF

6. Conclusion

This project successfully showcased the integration of an Arduino Nano with the PN532 NFC module to read and extract data from NFC tags and cards. By leveraging the Adafruit PN532 library and following a systematic approach, the system efficiently accessed all 16 sectors and their respective blocks, demonstrating accuracy and reliability. This implementation highlights the potential of NFC in embedded systems and opens avenues for applications like secure entry, automated attendance systems, and intelligent data management.