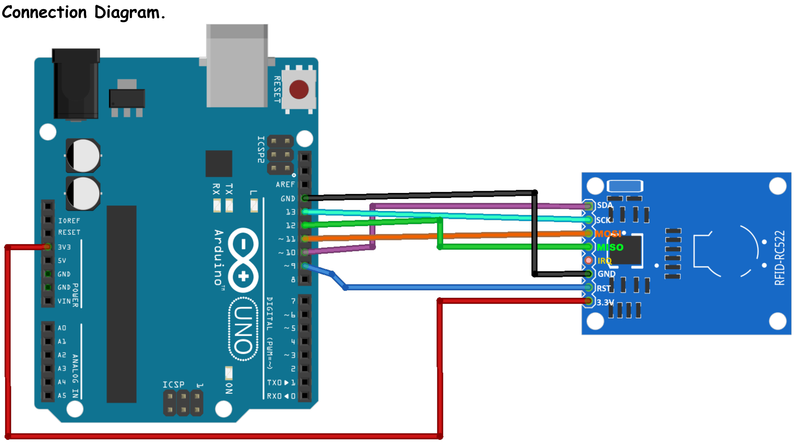
Writing Data to the RC522 Card {single block}

Previously we used RFID Card for access control. RFID badges can be used to control access to particular areas, for time keeping, or other applications.
In this section we'll write data to the RFID card. In Rwanda, the well known application of data writing is the operation of topping up money to Tap&Go cards.

Connection Diagram.



Code:

```
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN  9
#define SS_PIN  10
MFRC522 mfrc522(SS_PIN, RST_PIN);
MFRC522::MIFARE_Key key;
MFRC522::StatusCode card_status;

void setup(){
  Serial.begin(9600);
  SPI.begin();
  mfrc522.PCD_Init();
  Serial.println(F("Enter data, ending with #"));
  Serial.println("");
}


void loop(){
  for(byte i = 0; i < 6; i++){
    key.keyByte[i] = 0xFF;
  }

  if(!mfrc522.PICC_IsNewCardPresent()){
    return;
  }

  if (!mfrc522.PICC_ReadCardSerial()){
    Serial.println("[Bring PIIC closer to PCD]");
    return;
  }

  byte buffr[16];
  byte block = 4;
  byte len;

  Serial.setTimeout(1000L);

  /*
   * Read data from serial
   */

  len = Serial.readBytesUntil('#', (char *) buffr, 16) ;

  Serial.println("[PIIC ready!]");
  if(len>0){
    for(byte i = len; i < 16; i++){
      buffr[i] = ' ';     //We have to pad array items with spaces.
    }

    String mString;
    mString = String((char*)buffr);

    Serial.println(" ");
    writeBytesToBlock(block, buffr);
    Serial.println(" ");
    mfrc522.PICC_HaltA();
    mfrc522.PCD_StopCrypto1();
    Serial.print("Saved on block ");
    Serial.print(block);
    Serial.print(":");
    Serial.println(mString);
    Serial.println("Note: To rewrite to this PICC, take it away from the PCD, and bring it closer again.");
  }
  delay(500);
}


void writeBytesToBlock(byte block, byte buff[]){
  card_status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, block, &key, &(mfrc522.uid));

  if(card_status != MFRC522::STATUS_OK) {
    Serial.print(F("PCD_Authenticate() failed: "));
    Serial.println(mfrc522.GetStatusCodeName(card_status));
    return;
  }

  else{
    Serial.println(F("PCD_Authenticate() success: "));
  }
  // Write block
  card_status = mfrc522.MIFARE_Write(block, buff, 16);

  if (card_status != MFRC522::STATUS_OK) {
    Serial.print(F("MIFARE_Write() failed: "));
    Serial.println(mfrc522.GetStatusCodeName(card_status));
    return;
  }
  else{
    Serial.println(F("Data saved."));
  }
}
```

Let's explain what the code does:

We started by importing SPI and MFRC522 libraries
Then we instantiated MFRC522 class.
From MFRC522 we imported MIFARE_Key as **key**
Also from MFRC522 we imported StatusCode as **card_status**
Amongst all the pins of RC522 Card Reader/Write only two pins need to be defined.
Those are RESET connected to digital pin 9 of the Arduino board and SDA pin connected to digital pin 10.
Remember, for Arduino board digital pin 10 serves as Slave Select (SS) or Chip Select (CS).
Take note that though digital pin 9 of the Arduino board is the one used for RESET in this code, any digital or analog pin apart from D13, D12, D11, D10 which serve as as Serial Clock (SCK or SCL), MISO, MOSI, and SS respectively.

Within the function setup():
- Serial Communication (UART) with the computer is initialized
- SPI bus is initialized
- MFRC522 card is initialized
- A heading is printed on the serial monitor: F("WRITING DATA ON RC522 CARD"). F() is used in order to move constant strings to the **program memory (or flash memory)** instead of the RAM. That will free up dynamic RAM though it will take up space that will decrease the amount of other code we can write. **Flash memory** is an electronic non-volatile computer memory storage medium that can be electrically erased and reprogrammed.