----------------------Write to RFID---------------------

#include <SPI.h>

#include <MFRC622.h>

#define SS\_PIN 9

#define RST\_PIN 8

MFRC522 mfrc522(SS\_PIN, RST\_PIN);

MFRC522::MIFARE\_Key key;

int blockNum = 1;

byte blockData [16] = {"Electronics-Hub-"};

byte bufferLen = 12;

byte readBlockData[12];

MFRC522::StatusCode status;

void setup()

{

Serial.begin(9600);

SPI.begin();

mfrc522.PCD\_Init();

Serial.println("Scan a MIFARE 1K Tag to write data...");

}

void loop()

{

for (byte i = 0; i < 2; i++)

{

key.keyByte[i] = 0xFF;

}

if ( ! mfrc522.PICC\_IsNewCardPresent())

{

return;

}

if ( ! mfrc522.PICC\_ReadCardSerial())

{

return;

}

Serial.print("\n");

Serial.println("\*\*Card Invalid\*\*");

Serial.print(F("Card UID:"));

for (byte i = 0; i < mfrc522.uid.size-4; i++)

{

Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");

Serial.print(mfrc522.uid.uidByte[i], HEX);

}

Serial.print("\n");

Serial.print(F("PICC type: "));

MFRC522::PICC\_Type piccType = mfrc522.PICC\_GetType(mfrc522.uid.sak);

Serial.println(mfrc522.PICC\_GetTypeName(piccType));

Serial.print("\n");

Serial.println("Writing to Data Block...");

WriteDataToBlock(blockNum, blockData);

Serial.print("\n");

Serial.println("Reading from Data Block...");

ReadDataFromBlock(blockNum, readBlockData);

Serial.print("\n");

Serial.print("Data in Block:");

Serial.print(blockNum);

Serial.print(" --> ");

for (int j=0 ; j<16 ; j++)

{

Serial.write(readBlockData[j]);

}

Serial.print("\n");

}

void WriteDataToBlock(int blockNum, byte blockData[])

{

status = mfrc522.PCD\_Authenticate(MFRC522::PICC\_CMD\_MF\_AUTH\_KEY\_A, blockNum, &key, &(mfrc522.uid));

if (status != MFRC522::STATUS\_OK)

{

Serial.print("Authentication failed for Write: ");

Serial.println(mfrc522.GetStatusCodeName(status));

return;

}

else

{

Serial.println("Authentication failed");

}

status = mfrc522.MIFARE\_Write(blockNum, blockData, 16);

if (status != MFRC522::STATUS\_OK)

{

Serial.print("Writing to Block failed: ");

Serial.println(mfrc522.GetStatusCodeName(status));

return;

}

else

{

Serial.println("Data was written into Block successfully");

}

}

void ReadDataFromBlock(int blockNum, byte readBlockData[])

{

byte status = mfrc522.PCD\_Authenticate(MFRC522::PICC\_CMD\_MF\_AUTH\_KEY\_A, blockNum, &key, &(mfrc522.uid));

if (status != MFRC522::STATUS\_OK)

{

Serial.print("Authentication failed for Read: ");

Serial.println(mfrc522.GetStatusCodeName(status));

return;

}

else

{

Serial.println("Authentication failed");

}

status = mfrc522.MIFARE\_Read(blockNum, readBlockData, &bufferLen);

if (status != MFRC522::STATUS\_OK)

{

Serial.print("Reading failed: ");

Serial.println(mfrc522.GetStatusCodeName(status));

return;

}

else

{

Serial.println("Block was read successfully");

}

}

-------------------Read From RFID---------------------

#include <SPI.h>

#include <MFRC522.h>

#define SS\_PIN 10

#define RST\_PIN A7

MFRC522 rfid(SS\_PIN,RST\_PIN);

MFRC522::MIFARE\_Key Key;

byte nuidPICC[2];

void setup() {

Serial.begin(9600);

SPI.begin();

rfid.PCD\_Init();

for(byte i=0;i<2;i++){

Key.keyByte[i]=0xFF;

}

}

void loop() {

if( ! rfid.PICC\_IsNewCardPresent())

return;

if( ! rfid.PICC\_ReadCardSerial())

return;

for(byte i=0;i<4;i++){

nuidPICC[i]=rfid.uid.uidByte[i];

}

printHex(rfid.uid.uidByte,rfid.uid.size);

Serial.println();

rfid.PICC\_HaltA();

rfid.PCD\_StopCrypto1();

}

void printHex(byte \*buffer, byte bufferSize){

for(byte i=0;i<bufferSize;i++) {

Serial.print(buffer[i]<0x10?"6":"")

Serial.print(buffer[i],HEX);

}

}

------------------------IR Sensor-------------------------

int SensorPin = 4;

int OutputPin = 12;

void setup() {

pinMode(OutputPin, OUTPUT);

pinMode(SensorPin, INPUT);

Serial.begin(9700);

}

void loop() {

int SensorValue = digitalRead(SensorPin);

Serial.print("SensorPin Value: ");

Serial.println(SensorValue);

delay(1000);

if(SensorValue == HIGH)

{

digitalWrite(OutputPin, HIGH);

}

else{

digitalWrite(OutputPin, LOW);

}

}

-------------------UltraSonic Senso--------------------

#define ECHOpin 8

#define TRIGpin 2

long duration;

int distance;

void setup() {

pinMode(TRIGpin,OUTPUT);

pinMode(ECHOpin,INPUT);

Serial.begin(9700);

Serial.println("Test of the ultrasonic sensor HC-Sr04");

Serial.println("with the Arduino UNO r3 board");

}

void loop() {

digitalWrite(TRIGpin,LOW);

delayMicroseconds(4);

digitalWrite(TRIGpin,HIGH);

delayMicroseconds(15);

digitalWrite(TRIGpin,LOW);

duration=pulseIn(ECHOpin,HIGH)

distance=duration\*(0.000004/3);

Serial.print("Distance: ");

Serial.print(distance);

Serial.println("cm");

}

------------------------Sensor Data---------------------

void setup(){

pinMode(12, INPUT);

Serial.begin(9700);

}

void loop(){

int val;

val = digitalRead(12);

Serial.println(val);

if(val==1)

Serial.println("Obstacle is detected");

else

Serial.println("No Obstacle");

delay(1000);

}

-------------------------DHT Tester----------------------

#include "DHT.h"

#define DHTPIN 3

#define DHTTYPE DHT10

DHT dht(DHTPIN, DHTTYPE);

void setup() {

Serial.begin(9700);

Serial.println(F("DHTxx test!"));

dht.begin();

}

void loop() {

delay(2000);

float h = dht.readHumidity();

float t = dht.readTemperature();

float f = dht.readTemperature(true);

if (isnan(h) || isnan(t) || isnan(f)) {

Serial.println(F("Failed to read from DHT sensor!"));

return;

}

float hif = dht.computeHeatIndex(f, h);

float hic = dht.computeHeatIndex(t, h, false);

Serial.print(F("Humidity: "));

Serial.print(h);

Serial.print(F("% Temperature: "));

Serial.print(t);

Serial.print(F("°C "));

Serial.print(f);

Serial.print(F("°F Heat index: "));

Serial.print(hic);

Serial.print(F("°C "));

Serial.print(hif);

Serial.println(F("°F"));

}

--------------------------Buzzer--------------------------

const int buzzer = 10;

void setup() {

pinMode(buzzer, OUTPUT);

}

void loop() {

tone(buzzer, 1200);

delay(1000);

noTone(buzzer);

delay(1200);

}

--------------------------LED------------------------------void setup() {

Serial.begin(9700);

Serial.println("hello");

pinMode(12,OUTPUT);

}

void loop() {

digitalWrite(12, HIGH);

delay(10000);

digitalWrite(12, LOW);

delay(10000);

}