1. Fingerprint sensor R307:
   1. Fingerprints enrollment:

#include <Adafruit\_Fingerprint.h>

// pin #2 is IN from sensor (GREEN wire)

// pin #3 is OUT from arduino (WHITE wire)

SoftwareSerial mySerial(2, 3);

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&mySerial);

uint8\_t id;

void setup()

{

Serial.begin(9600);

while (!Serial); // For Yun/Leo/Micro/Zero/...

delay(100);

Serial.println("\n\nAdafruit Fingerprint sensor enrollment");

// set the data rate for the sensor serial port

finger.begin(57600);

if (finger.verifyPassword()) {

Serial.println("Found fingerprint sensor!");

} else {

Serial.println("Did not find fingerprint sensor :(");

while (1) { delay(1); }

}

Serial.println(F("Reading sensor parameters"));

finger.getParameters();

Serial.print(F("Status: 0x")); Serial.println(finger.status\_reg, HEX);

Serial.print(F("Sys ID: 0x")); Serial.println(finger.system\_id, HEX);

Serial.print(F("Capacity: ")); Serial.println(finger.capacity);

Serial.print(F("Security level: ")); Serial.println(finger.security\_level);

Serial.print(F("Device address: ")); Serial.println(finger.device\_addr, HEX);

Serial.print(F("Packet len: ")); Serial.println(finger.packet\_len);

Serial.print(F("Baud rate: ")); Serial.println(finger.baud\_rate);

}

uint8\_t readnumber(void) {

uint8\_t num = 0;

while (num == 0) {

while (! Serial.available());

num = Serial.parseInt();

}

return num;

}

void loop(){

Serial.println("Ready to enroll a fingerprint!");

Serial.println("Please type in the ID # (from 1 to 127) you want to save this finger as...");

id = readnumber();

if (id == 0) {// ID #0 not allowed, try again!

return;

}

Serial.print("Enrolling ID #");

Serial.println(id);

while (! getFingerprintEnroll() );

}

uint8\_t getFingerprintEnroll() {

int p = -1;

Serial.print("Waiting for valid finger to enroll as #"); Serial.println(id);

while (p != FINGERPRINT\_OK) {

p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.println(".");

break;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

break;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

break;

default:

Serial.println("Unknown error");

break;

}

}

// OK success!

p = finger.image2Tz(1);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

return p;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

return p;

default:

Serial.println("Unknown error");

return p;

}

Serial.println("Remove finger");

delay(2000);

p = 0;

while (p != FINGERPRINT\_NOFINGER) {

p = finger.getImage();

}

Serial.print("ID "); Serial.println(id);

p = -1;

Serial.println("Place same finger again");

while (p != FINGERPRINT\_OK) {

p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.print(".");

break;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

break;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

break;

default:

Serial.println("Unknown error");

break;

}

}

// OK success!

p = finger.image2Tz(2);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

return p;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

return p;

default:

Serial.println("Unknown error");

return p;

}

// OK converted!

Serial.print("Creating model for #"); Serial.println(id);

p = finger.createModel();

if (p == FINGERPRINT\_OK) {

Serial.println("Prints matched!");

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

return p;

} else if (p == FINGERPRINT\_ENROLLMISMATCH) {

Serial.println("Fingerprints did not match");

return p;

} else {

Serial.println("Unknown error");

return p;

}

Serial.print("ID "); Serial.println(id);

p = finger.storeModel(id);

if (p == FINGERPRINT\_OK) {

Serial.println("Stored!");

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

return p;

} else if (p == FINGERPRINT\_BADLOCATION) {

Serial.println("Could not store in that location");

return p;

} else if (p == FINGERPRINT\_FLASHERR) {

Serial.println("Error writing to flash");

return p;

} else {

Serial.println("Unknown error");

return p;

}

return true;

}

* 1. Fingerprints check:

#include <Adafruit\_Fingerprint.h>

// pin #2 is IN from sensor (GREEN wire)

// pin #3 is OUT from arduino (WHITE wire)

SoftwareSerial mySerial(2, 3);

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&mySerial);

void setup(){

Serial.begin(9600);

while (!Serial); // For Yun/Leo/Micro/Zero/...

delay(100);

Serial.println("\n\nAdafruit finger detect test");

// set the data rate for the sensor serial port

finger.begin(57600);

delay(5);

if (finger.verifyPassword()) {

Serial.println("Found fingerprint sensor!");

} else {

Serial.println("Did not find fingerprint sensor :(");

while (1) { delay(1); }

}

Serial.println(F("Reading sensor parameters"));

finger.getParameters();

Serial.print(F("Status: 0x")); Serial.println(finger.status\_reg, HEX);

Serial.print(F("Sys ID: 0x")); Serial.println(finger.system\_id, HEX);

Serial.print(F("Capacity: ")); Serial.println(finger.capacity);

Serial.print(F("Security level: ")); Serial.println(finger.security\_level);

Serial.print(F("Device address: ")); Serial.println(finger.device\_addr, HEX);

Serial.print(F("Packet len: ")); Serial.println(finger.packet\_len);

Serial.print(F("Baud rate: ")); Serial.println(finger.baud\_rate);

finger.getTemplateCount();

if (finger.templateCount == 0) {

Serial.print("Sensor doesn't contain any fingerprint data. Please run the 'enroll' example.");

}

else {

Serial.println("Waiting for valid finger...");

Serial.print("Sensor contains "); Serial.print(finger.templateCount); Serial.println(" templates");

}

}

void loop(){

getFingerprintID();

delay(50);

}

uint8\_t getFingerprintID() {

uint8\_t p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.println("No finger detected");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

return p;

default:

Serial.println("Unknown error");

return p;

}

// OK success!

p = finger.image2Tz();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

return p;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

return p;

default:

Serial.println("Unknown error");

return p;

}

// OK converted!

p = finger.fingerSearch();

if (p == FINGERPRINT\_OK) {

Serial.println("Found a print match!");

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

return p;

} else if (p == FINGERPRINT\_NOTFOUND) {

Serial.println("Did not find a match");

return p;

} else {

Serial.println("Unknown error");

return p;

}

// found a match! “” PUT THE RESPONSE CODE HERE “”

Serial.print("Found ID #"); Serial.print(finger.fingerID);

Serial.print(" with confidence of "); Serial.println(finger.confidence);

return finger.fingerID;

}

1. Bluetooth module HC-05:

Uses serial communication protocol, so once connected to TX & RX, we’ll use “serial” function to communicate with mobile phone.

1. PIR movement sensor and turns on light when there is movement:

int ledPin = 13; // LED

int pirPin = 2; // PIR Out pin

int pirStat = 0; // PIR status

void setup() {

pinMode(ledPin, OUTPUT);

pinMode(pirPin, INPUT);

Serial.begin(9600);

}

void loop(){

pirStat = digitalRead(pirPin);

if (pirStat == HIGH) { // if motion detected

digitalWrite(ledPin, HIGH); // turn light ON

}

else {

digitalWrite(ledPin, LOW); // turn light OFF if we have no motion

}

}

1. Smoke Sensor MQ-2 and turns on alarm when smoke is detected:

int buzzer = 10;

int smokeA0 = A5;

// Threshold value

int sensorThres = 400;

void setup() {

pinMode(buzzer, OUTPUT);

pinMode(smokeA0, INPUT);

Serial.begin(9600);

}

void loop() {

int analogSensor = analogRead(smokeA0);

Serial.print("Pin A0: ");

Serial.println(analogSensor);

// Checks if it has reached the threshold value

if (analogSensor > sensorThres)

tone(buzzer, 1000, 200);

else

noTone(buzzer);

delay(100);

}

1. Servo motor and control garage door and 3 home lights with mobile phone using Bluetooth module:

#include <Servo.h>

Servo myservo; // create servo object to control a servo

int light1 = 5;

bool L1 = 0;

int light2 = 6;

bool L2 = 0;

int light3 = 7;

bool L3 = 0;

int pos = 0;

String control;

void setup() {

myservo.attach(9); // attaches the servo on pin 9 to the servo object

Serial.begin(9600);

}

void loop() {

control = Serial.read();

if( control == "GO"){ // Garage Open

for (pos = 180; pos >= 0; pos -= 1){ // goes from 180 degrees to 0 degrees

myservo.write(pos); // tells servo to go to position 0

delay(15); // waits 15ms for the servo to reach the position

}

}

else if( control == "GC" ){ // Garage Close

for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

}

}

if( control == "L1" ){

L1 = !L1; //Changes the state of L1 bool variable to switch light

digitalWrite(light1, L1);

}

if( control == "L2" ){

L1 = !L2; //Changes the state of L2 bool variable to switch light

digitalWrite(light2, L2);

}

if( control == "L3" ){

L1 = !L3; //Changes the state of L3 bool variable to switch light

digitalWrite(light3, L3);

}

}

1. Rain sensor with LCD to show weather condition and open rain cover:

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <Servo.h>

Servo myservo; // create servo object to control a servo

LiquidCrystal\_I2C lcd(0x27,20,4);

int pos = 0;

char control;

int sensorPin = 10;

void setup() {

Serial.begin(9600);

myservo.attach(9); // attaches the servo on pin 9 to the servo object

lcd.begin(16,2);

lcd.backlight();

}

void loop() {

bool level = digitalRead(sensorPin);

if (level == 1) {

Serial.println("There is rain!");

lcd.setCursor(0,0);

lcd.print(" There is rain! ");

for (pos = 180; pos >= 0; pos -= 1){ // goes from 180 degrees to 0 degrees

myservo.write(pos); // tells servo to go to position 0

delay(15);

}

else if (level == 0) {

Serial.println("There is no rain!");

lcd.setCursor(0,0);

lcd.print(" There is no rain! ");

for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15);

}

delay(1000);

}

1. LDR (photoresistor) to sense the outside light and control the LED:

int ldr=A0;

int value=0;

int LED = 3;

void setup() {

Serial.begin(9600);

pinMode(LED,OUTPUT);

}

void loop() {

value=analogRead(ldr);//Reads the Value of LDR(light).

Serial.println("LDR value is :");//Prints the value of LDR to Serial Monitor.

Serial.println(value);

if(value<300)

{

digitalWrite(LED,HIGH);//Turns on the light in the dark.

}

else

{

digitalWrite(LED,LOW);//Turns off the light in light.

}

}

1. Flame sensor and turn on alarm when flame detected:

int sensorPin = A0;

int sensorValue = 0;

int buzzer = 12;

void setup() {

pinMode(buzzer,OUTPUT);

Serial.begin(9600);

}

void loop(){

sensorValue = analogRead(sensorPin);

Serial.println(sensorValue);

if (sensorValue < 100){

Serial.println("Fire Detected");

tone(buzzer, 1000, 200);

delay(1000);

}

else

noTone(buzzer);

}

1. Temperature sensor (LM35) and turn on fan when temperature is high:

int val;

int tempPin = 1;

int fan = 3;

void setup(){

Serial.begin(9600);

pinMode(fan, OUTPUT);

}

void loop(){

val = analogRead(tempPin);

float mv = ( val/1024.0)\*5000;

float cel = mv/10;

Serial.print("TEMPRATURE = ");

Serial.print(cel);

Serial.print("\*C");

Serial.println();

if( cel > 25 )

digitalWrite(fan, HIGH); //Turns on the fan when temperature is high

else

digitalWrite(fan, LOW); //Turns off the fan when temperature is low

delay(1000);

}