GROUP MEMBERS

KISA EMMANUEL B24259 S23B23/028

Otwiine Elizabeth B22719 S23B13/087

Nicole Johnson B24254 S23B23/020

Emmanuel Nsubuga B20811 M23B38/005

Switching on lights, opening and closing windows and doors automatically and remotely and detecting intrusion

Components for smart home security system

RF Transmitter and Receiver Module(433MHZ)

Ultrasonic sensors

ARDUINO UNO Board

Soldering iron

Wire Cutters

Jumper wires

9v Battery and connectors

ARDUINO sms alarm

Resistors(220 ohms)l

RTC module (to keep track of the time)

buzzers

Switches

Servo motor

LEDs

Buzzer

Bread board

Header connectors

Requirements

The Smart Home Security System we are working on should be able to do the following;

* Should be able to switch on lights automatically when someone enters the room.
* Should be able to carry out opening and closing doors in a home
* Should be able to detect intrusion on the home which can the through the use of alarms
* Transmitters should sense the human motion and transmit the data to receiver
* Receiver should be able to receive the data from transmitter and activate alarm system and camera or sending sms etc

Code for the different parts

**PIR with a LED**

int ledPin = 13; // choose the pin for the LED

int inputPin = 8; // choose the input pin (for PIR sensor)

int pirState = LOW; // we start, assuming no motion detected

int val = 0; // variable for reading the pin status

void setup() {

pinMode(ledPin, OUTPUT); // declare LED as output

pinMode(inputPin, INPUT); // declare sensor as input

Serial.begin(9600);

}

void loop(){

val = digitalRead(inputPin); // read input value

if (val == HIGH) // check if the input is HIGH

{

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

if (pirState == LOW)

{

Serial.println("Motion detected!"); // print on output change

pirState = HIGH;

}

}

else

{

digitalWrite(ledPin, LOW); // turn LED OFF

if (pirState == HIGH)

{

Serial.println("Motion ended!"); // print on output change

pirState = LOW;

}

}

}

**CODE FOR ULTRASONIC SENSOR WITH BUZZER**

// defines pins numbers

const int trigPin = 9;

const int echoPin = 10;

const int buzzer = 11;

const int ledPin = 13;

// defines variables

long duration;

int distance;

int safetyDistance;

void setup() {

pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

pinMode(buzzer, OUTPUT);

pinMode(ledPin, OUTPUT);

Serial.begin(9600); // Starts the serial communication

}

void loop() {

// Clears the trigPin

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance = duration \* 0.034 / 2;

safetyDistance = distance;

if (safetyDistance <= 5) {

tone(buzzer, 1000); // Produce a 1kHz tone on the passive buzzer

digitalWrite(ledPin, HIGH);

} else {

noTone(buzzer); // Stop the tone

digitalWrite(ledPin, LOW);

}

// Prints the distance on the Serial Monitor

Serial.print("Distance: ");

Serial.println(distance);

delay(100);

}

**Code for Micro servo motor with push button and arduino**

#include <Servo.h>;

// pushbutton pin

const int buttonPin = 8;

// servo pin

const int servoPin = 3;

Servo servo;

//create a variable to store a counter and set it to 0

int counter = 0;

void setup()

{

servo.attach (servoPin);

// Set up the pushbutton pins to be an input:

pinMode(buttonPin, INPUT);

}

void loop()

{

// local variable to hold the pushbutton states

int buttonState;

//read the digital state of buttonPin with digitalRead() function and store the //value in buttonState variable

buttonState = digitalRead(buttonPin);

//if the button is pressed increment counter and wait a tiny bit to give us some //time to release the button

if (buttonState == LOW) // light the LED

{

counter++;

delay(150);

}

if(counter == 0)

servo.write (20); // zero degrees

else if(counter == 1)

servo.write(90);

else if(counter == 2)

servo.write (150);

else if(counter == 3)

servo.write (180);

//else reset the counter to 0 which resets the servo to 0 degrees

else

counter = 0;

}

Code for Arduino with LDR and LED

const int ledPin = 13;

const int ldrPin = A0;

void setup() {

Serial.begin(9600);

pinMode(ledPin, OUTPUT);

pinMode(ldrPin, INPUT);

}

void loop() {

int ldrStatus = analogRead(ldrPin);

if (ldrStatus <=300) {

digitalWrite(ledPin, HIGH);

Serial.println("LDR is DARK, LED is ON");

}

else {

digitalWrite(ledPin, LOW);

Serial.println("---------------");

}

}

**CODE FOR LCD**

#include <LiquidCrystal\_I2C.h>

LiquidCrystal\_I2C lcd(0x27,16,2); // set the LCD address to 0x3F for a 16 chars and 2 line display

void setup() {

lcd.init();

lcd.clear();

lcd.backlight(); // Make sure backlight is on

// Print a message on both lines of the LCD.

lcd.setCursor(2,0); //Set cursor to character 2 on line 0

lcd.print("Hello world!");

lcd.setCursor(2,1); //Move cursor to character 2 on line 1

lcd.print("GROUP D");

}

void loop() {

}

**GSM CODE**

#include <SoftwareSerial.h>

// Create software serial object to communicate with SIM800L

SoftwareSerial mySerial(3, 2); // SIM800L Tx & Rx is connected to Arduino #3 & #2

void setup() {

// Begin serial communication with Arduino and Arduino IDE (Serial Monitor)

Serial.begin(9600);

// Begin serial communication with Arduino and SIM800L

mySerial.begin(9600);

Serial.println("Initializing...");

delay(1000);

// Handshake with SIM800L

mySerial.println("AT"); // Once the handshake test is successful, it will return OK

updateSerial();

// Configure SIM800L in TEXT mode

mySerial.println("AT+CMGF=1");

updateSerial();

// Configure how newly arrived SMS messages should be handled

mySerial.println("AT+CNMI=1,2,0,0,0");

updateSerial();

// Send an SMS

mySerial.println("AT+CMGS=\"+256741539324\"");

updateSerial();

mySerial.print("⚠️ Intruder Alert ⚠️ | ensubga019@gmail.com"); // SMS content

updateSerial();

mySerial.write(26); // ASCII code for Ctrl+Z to send the message

updateSerial();

// Make a call

mySerial.println("ATD+ +256741539324;");

updateSerial();

delay(50000); // Wait for 50 seconds

mySerial.println("ATH"); // Hang up the call

updateSerial();

}

void loop() {

updateSerial();

}

void updateSerial() {

delay(500);

while (Serial.available()) {

mySerial.write(Serial.read()); // Forward what Serial received to Software Serial Port

}

while (mySerial.available()) {

Serial.write(mySerial.read()); // Forward what Software Serial received to Serial Port

}

}

Merge 1

#include <LiquidCrystal\_I2C.h>

// defines pins numbers

const int trigPin = 9;

const int echoPin = 10;

const int buzzer = 11;

const int ledPin = 13;

// defines variables

long duration;

int distance;

int safetyDistance;

LiquidCrystal\_I2C lcd(0x27, 16, 2); // set the LCD address to 0x3F for a 16 chars and 2 line display

void setup() {

pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

pinMode(buzzer, OUTPUT);

pinMode(ledPin, OUTPUT);

Serial.begin(9600); // Starts the serial communication

lcd.init();

lcd.clear();

lcd.backlight(); // Make sure backlight is on

// Print a message on both lines of the LCD.

lcd.setCursor(2,0); //Set cursor to character 2 on line 0

lcd.print("Hello world!");

lcd.setCursor(2,1); //Move cursor to character 2 on line 1

lcd.print("GROUP D");

}

void loop() {

// Clears the trigPin

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance = duration \* 0.034 / 2;

safetyDistance = distance;

if (safetyDistance <= 5) {

tone(buzzer, 1000); // Produce a 1kHz tone on the passive buzzer

digitalWrite(ledPin, HIGH);

lcd.setCursor(0,0); //Set cursor to character 0 on line 0

lcd.print("Intrusion detected");

} else {

noTone(buzzer); // Stop the tone

digitalWrite(ledPin, LOW);

lcd.setCursor(0,0); //Set cursor to character 0 on line 0

lcd.print(" "); // Clear the line

}

// Prints the distance on the Serial Monitor

Serial.print("Distance: ");

Serial.println(distance);

delay(100);

}

**FULL CODE**