Door Alarm with Bluetooth Control

İbrahim Mert GÜL, Selahattin BOZAN, Oğuzhan ELKAPMIŞ
ibrahimmertgul@gmail.com
seloo_bozan@hotmail.com
oguzhan.elkapmis@gmail.com

Abstract

Motion sensor triggers buzzer and led. Bluetooth devices activate or deactivate the motion sensor.

1. INTRODUCTION

Today, alarm systems are used in many different places and for different purposes. In this project, we worked on a motion-based alarm system that can be controlled by a bluetooth remote control. This project can be applied to houses and workplaces for security and such things.

2. MATERIAL AND METHODS

Material List

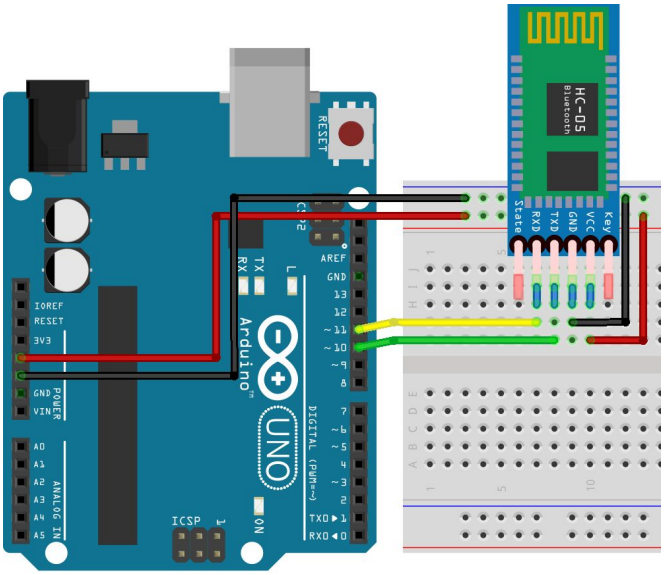
Name	Amount
Arduino Pro Mini	1
Motion Sensor(PIR HC-SR501)	1
Buzzer	1
HC-05 Bluetooth Modul	1
220Ω Resistor	1
Red (633nm) LED	1
PCB (100mm x 100mm)	1
USB Power Cable	1

Setup of HC-05 Bluetooth

Firstly be sure that about your HC-05 is master or slave mode. For this reason we have to connect modul with computer and run configuration software. We make use of a TERATERM software to configure as a slave with using USB-TTL converter.

We set the Boud Rate, Data, Parity and Stop values as shown in guides[3]. Also we set our modul's name, password and role(Slave).

Arduino and HC-05



We use 4 pins on HC-05 to make it work. 2 of them are Power and Ground, rest are RX and TX which are the pins that forward the data.

After the connections of the cable we have tested the arduino. To read data from HC-05 on arduino, we use *SoftwareSerial.h* library. Firstly learned how to read data[1], take it as a string[5](`readFromBluetooth()`) and printed data on Serial Port Screen.

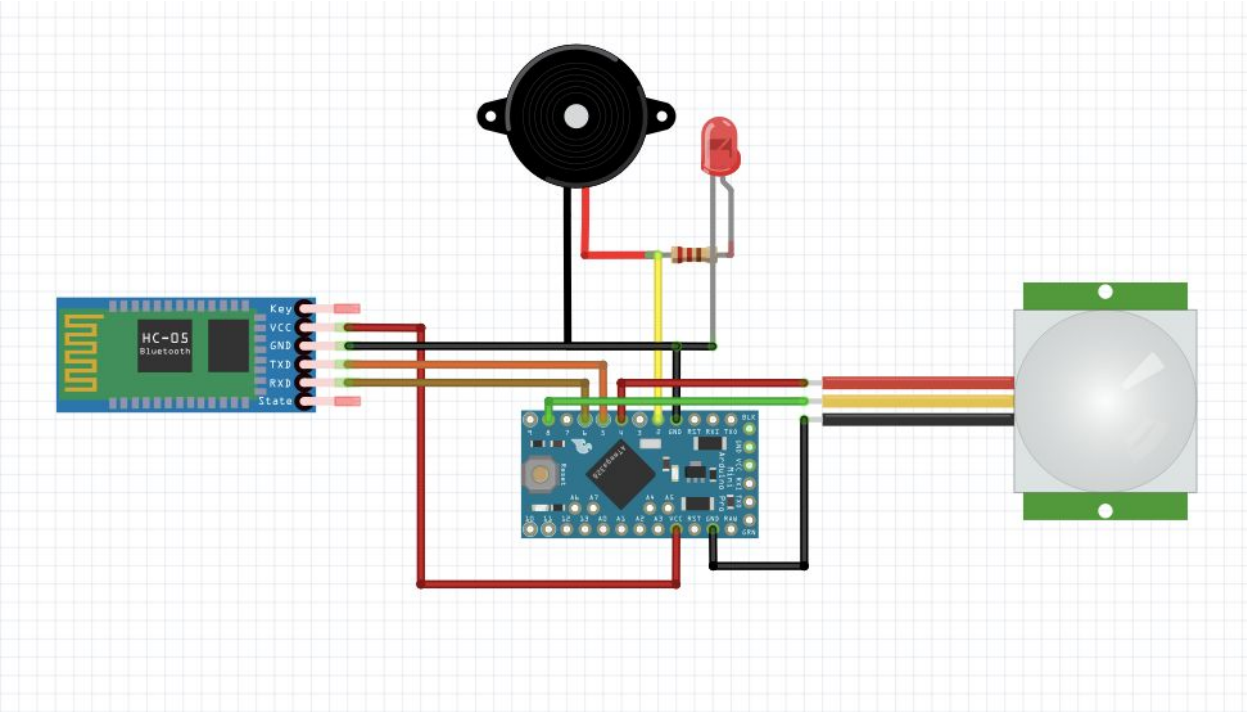
Sending test data is very easy with simple android app, called *Bluetooth Terminal*[4]

Motion Sensor

This sensor's range is 5-6 meters. If it is triggered, it sends signals for minimum 5 seconds. You can set the alarm time and sensitivity. There are two physical settings on motion sensor for it. We have used minimum alarm time and middle sensitivity settings.

Plans of the Project

Firstly we have connected the motion sensor and added alarm sound and alarm light. Arduino reads digital signal from sensor and sends digital signals to led[7] and buzzer[6]. We did not use any additional library to control sensor, buzzer and led. After that we determine two strings that accept from bluetooth to switch alarm. If alarm is closed, arduino does not read any data from motion sensor and shuts buzzer up and goes led off.



Bluetooth control interface

UI is made with Android Studio. Simply there are two buttons that send the data to HC-05.

```
btnOnOff.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if (state == 0) {
            tabbedActivity.bluetooth.send(String.valueOf(Consts.on),  CRLF: true);
            state = 1;
            btnOnOff.setText("ON");
            btnOnOff.setBackgroundColor(rootView.getResources().getColor(R.color.colorOn));
        } else {
            tabbedActivity.bluetooth.send(String.valueOf(Consts.off),  CRLF: true);
            state = 0;
            btnOnOff.setText("OFF");
            btnOnOff.setBackgroundColor(rootView.getResources().getColor(R.color.colorOff));
        }
    }
});
```

Test mechanism

We have tested the alarm on breadboard. To test more correctly we are going to put the sensors in a mechanism like smallbox with a door.

3. CONCLUSION

Project accomplished as accepted and open to renovative on UI side. If the Api document is written, it can be used on any device(with out IOS). .

ACKNOWLEDGEMENTS

Kaan Deniz Erciyes did not join. Android app with the help of Çağrı Aldemir. 3D design with the help of Ramazan Seyhan.

REFERENCES

[1] maker.robotistan.com/arduino-dersleri-17-hc-05-bluetooth-modulu-kullanimi/
[2] fritzing.org/home/
[3] www.instructables.com/id/HOW-TO-HC-05-Bluetooth-MODULE-AT-Commands-With-But/
[4] play.google.com/store/apps/details?id=Qwerty.BluetoothTerminal&hl=tr
[5] Autonomous Robotic Arm YA-40 Source Code (Video Link)
[6] maker.robotistan.com/arduino-dersleri-9-buzzer-ile-ses-cikisi-alma-2/
[7] https://maker.robotistan.com/arduino-ders-1-led-yakip-sondurme-blink/

APPENDIX

A. PROJECT SOURCE CODE

Ardunio:

```
#include <SoftwareSerial.h>
#define led 2
#define sensor 8
#define bluetoothTx 5
#define bluetoothRx 6
#define buzzerPin 2
#define Vcc 4
SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);
int val = 0;
int alarm = 0;


void setup() {
  pinMode(led, OUTPUT);
  pinMode(Vcc, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
  pinMode(sensor, INPUT);
  digitalWrite(buzzerPin, LOW);
  bluetooth.begin(38400);
  Serial.begin(9600);
  digitalWrite(Vcc, HIGH);
}

void loop(){
  String readedData = readFromBluetooth();
  digitalWrite(Vcc, HIGH);
  if(readedData.length() > 0) {
    Serial.print("DATA:");
    Serial.println(readedData);

    if(readedData == "101"){
      alarm = 1;
    }
    else if(readedData == "102"){
      alarm = 0;
    }
  }

  if (alarm){
    val = digitalRead(sensor);
    if (val == HIGH) {
      digitalWrite(led, HIGH);
      digitalWrite(buzzerPin, HIGH);
    }
    else {
      digitalWrite(led, LOW);
      digitalWrite(buzzerPin, LOW);
    }
  }
  else{
    digitalWrite(buzzerPin, LOW);
    digitalWrite(led, LOW);
  }
}
```

```
String readFromBluetooth() {
  String readString = "";
  while (bluetooth.available()) {
    char c = bluetooth.read();
    readString += c;
  }
  readString.trim();
  return readString;
}
```

Android(java):

```
package com.yazilimagi.robotkolbluetoothyonetim.Fragments;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;

import com.yazilimagi.robotkolbluetoothyonetim.Consts;
import com.yazilimagi.robotkolbluetoothyonetim.R;
import com.yazilimagi.robotkolbluetoothyonetim.TabbedActivity;


@SuppressLint("ValidFragment")
public class ControlFragment extends Fragment {

    Button btnOnOff;
    TabbedActivity tabbedActivity;
    int state;

    public ControlFragment(TabbedActivity tabbedActivity) {
        this.tabbedActivity = tabbedActivity;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
savedInstanceState) {

        final View rootView = inflater.inflate(R.layout.fragment_control, container,
false);
        state = 0;
        btnOnOff = rootView.findViewById(R.id.btnAltIleri);


        btnOnOff.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if (state == 0) {
                    tabbedActivity.bluetooth.send(String.valueOf(Consts.on), true);
                    state = 1;
```

```
                    btnOnOff.setText("ON");

btnOnOff.setBackgroundColor(rootView.getResources().getColor(R.color.colorOn));
                } else {
                    tabbedActivity.bluetooth.send(String.valueOf(Consts.off), true);
                    state = 0;
                    btnOnOff.setText("OFF");

btnOnOff.setBackgroundColor(rootView.getResources().getColor(R.color.colorOff));
                }
            }
        });



        return rootView;
    }


}
```