UP877962

IoT Project report

Introduction

The project being proposed is a smart room sensor that incorporates a switched LED for the user to light the room, a humidity and temperature sensor that periodically takes readings and publishes them over the local area network to a HTML page and also will incorporate a motion sensor that has a passive and active mode when in passive mode the sensor will record the amount of detections it encounters and when set to active mode the sensor will also trigger a buzzer on the device whenever the sensor is activated.

This project would be useful for people who want to keep track of what is happening in a room including the amount of people accessing it and climate related data and also for being alerted to unauthorised entrances to the room, the remotely activated led could also be of use to light an area, this could be especially useful in rooms that are supposed to be climate controlled and where human intrusions could be a factor in affecting the said climate controls. .

The room sensor will work over the local area network using the TCP protocol and the recorded data will be available by way of a HTML page that contains all of the relevant switches and displays, the decision to use this method was due to the fact that it requires no additional effort by the user other than typing in the required IP address rather than having to register and log in with 3rd party software to access the relevant data, the HTML page is generated within the device using a buffer and regular expressions/pattern matching as well as GET requests to allow the integration of user controls within the page.

The LUA programming language will be used to write the underlying code for the room sensor, this is a lightweight language that is ideal for IoT devices that have limited memory and processing power. The code will be written in the Esplorer IDE using the previously supplied nodeMCU based firmware.

Design

The room sensor will incorporate the esp8266 microcontroller, an SR501 motion sensor with a piezo buzzer incorporated within the same logic within the program to activate upon the sensor giving a reading, a DHT11 temperature and humidity sensor that will take readings on a 3 second interval and publish these readings to a HTML page accessible over the local area network and also feature a switchable LED that can be controlled by the HTML page. The motion sensor features two different modes, it can work in a passive mode where it will record the number of detections and work in an active mode whereby upon movement detection a loud buzzer will sound. The room sensor was originally intended to incorporate a camera that would take an image when movement was detected instead of the buzzer sounding but difficulties in acquiring components meant that the decision was taken to incorporate a buzzer at this point the camera chosen was the OV2640 2mp camera that at least in a hardware respect is perfectly compatible with the esp8266 with the upper

input voltage at 3.3v. the image would have been displayed in a list element with the html page, however I do believe this may have caused issues with the page size if a lot of photos are uploaded due to limited memory on the esp8266. The room sensor has a working temperature range of -17c to 70c so can be used in most environments however a limitation with the DHT11 sensor is that it will not detect negative temperatures.

Wiring layout

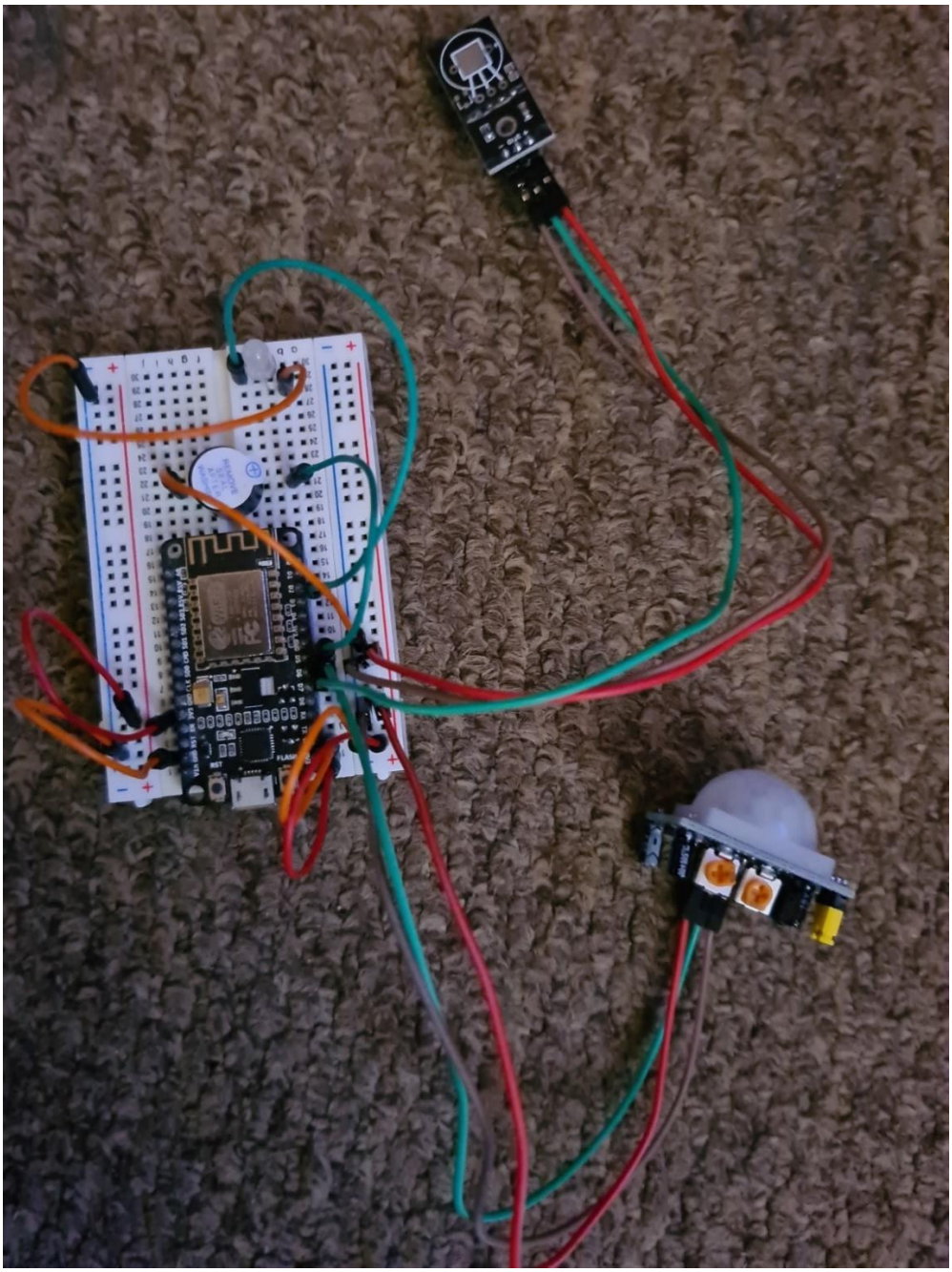
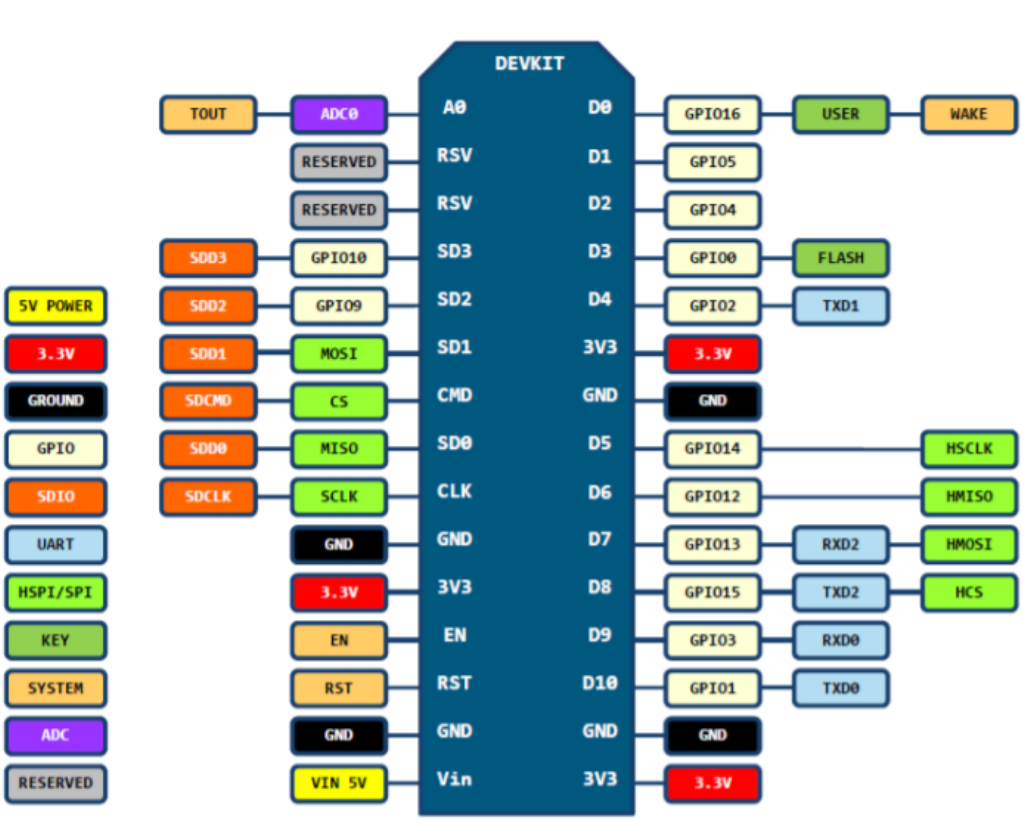


Figure 1 – an image of the room sensors wiring layout, for the purpose of keeping the wiring organised red cables have been used for 3.3v, orange/brown cables for GND and green cables for data connections.



Using the above pin definition, the room sensor is wired as follows:

- Both breadboard rails are powered and grounded using respective 3.3v and GND pins on each side.
- The user controlled LED bridges rows 29 to 28 of the breadboard with the anode end connected to row 29 and is powered and supplied data control by a cable from the D2 pin on the esp8266 to row 29 and then grounded form row 28 to the left hand ground rail.
- The DHT11 humidity sensor is grounded and powered via the right hand 3.3v and ground rail of the breadboard, the data connection is connected to the D7 pin on the esp8266.
- The SR501 motion sensor is grounded and powered through the right-hand ground rail of the breadboard and the data connection is connected to the D6 pin on the esp8266, the sensor is set to have a 3 second delay and the maximum range has been set at 7 metres.
- The active buzzer bridges the centre of row 21 on the breadboard and the positive side is connected to the D5 pin on the esp8266, this provides both power and data connections and then the negative side of the buzzer is connected to the right-hand sided ground rail.

There was no need to consider ohms law or add any resistance to any aspects of the room sensor as all components are compatible with the 3.3v output of the esp8266 and any additional resistance would just lower the loudness of the alarm when the motion sensor is set to its active mode which

would be counterproductive to its purpose. The left-hand power/ground rail was intentionally left mostly clear to allow for the inclusion of the camera.

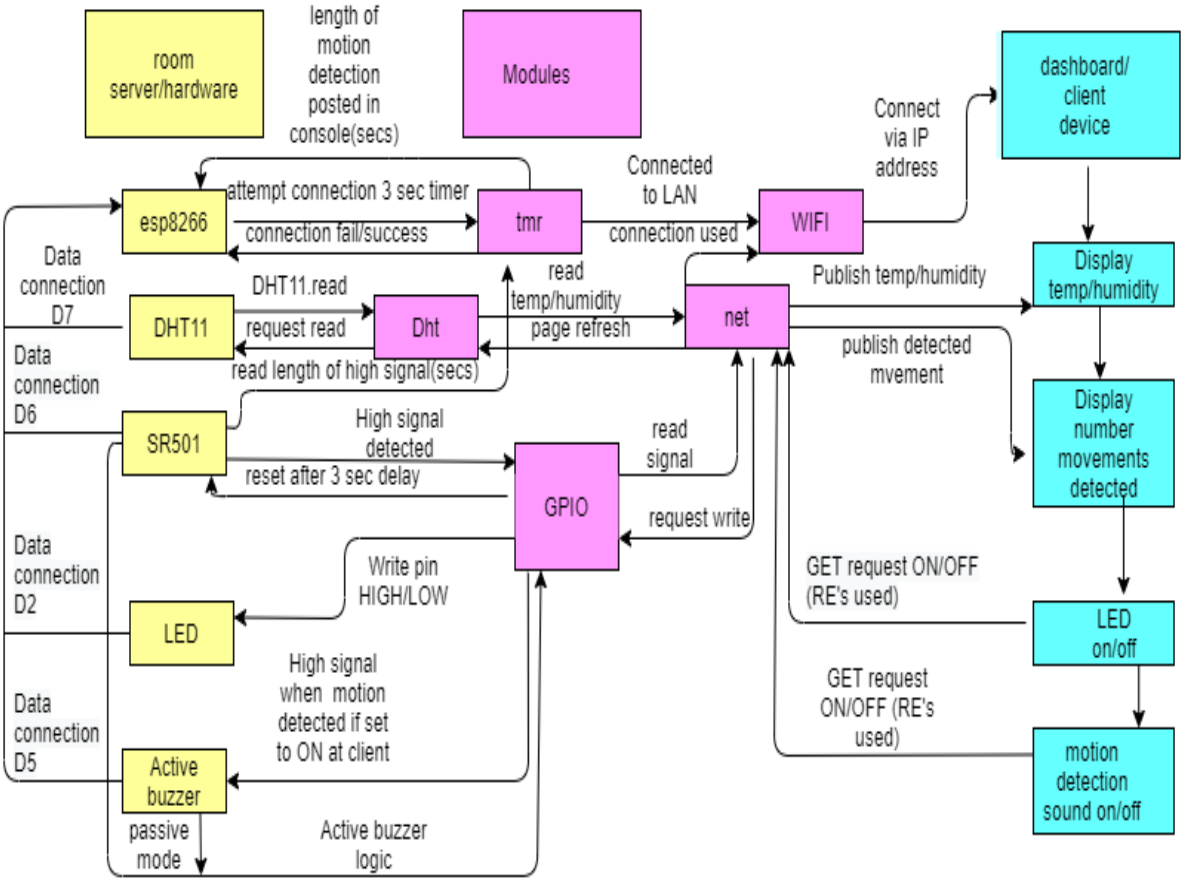
Modules

There are several modules that were used during building the room sensor, these modules and their usage are listed below:

- GPIO(general purpose input/output) module- this module is used to manage the data connections from the esp8266 to the peripheral sensors and components, generally most of these data connections use a combination of HIGH or LOW signals that roughly correspond to on or off commands to the sensor or the esp8266 with one notable exception being the onboard LED's in which case this is reversed.
- Tmr(timer) module – this module is used throughout the project for occasions when there is a need for a timed event such as for allowing multiple attempts to connect to the local area network and recording the duration of the motion detections.
- DHT module – this module manages the DHT11 temperature and humidity sensor that is incorporated into the room sensor.
- Wifi module – this module is used to manage connections to the local area network. The esp8266 also supports packet injection through the onboard wireless card.
- Net module – this module is used to send and receive data over the local area network and used in the creation of the webserver that serves the control page. The TCP protocol is used for communications.

Control flow

The Net and GPIO modules manage most of the data flow through the home sensors components, these were used as they were most applicable in the purpose of creating a plug and play device that required no additional sign up to 3rd parties or additional setup stages when initially setting up the system, GET requests were used to ask for changes from the TCP server implemented when the user wishes to activate/deactivate the onboard LED and sound for the motion sensor. Regular expressions were used to decode the GET requests to the server.



A diagram displaying the control flow for the room sensor system.

Cost

The devices build cost is roughly £10 if it is broken down into solely the individual components, However, with the need to buy an electronics kit to gain the needed components the actual build cost was ~£20 although many of the components in the kit were not used within the build at this point. The cost of the camera was £5 although as it was not possible to acquire it, it has been omitted from the design and build.

Source code

```
--Connect to the local areaa network on station mode
wifi.setmode(wifi.STATION)
station_cfg={}
--insert your AP credentials
station_cfg.ssid="VM071645-2G"
station_cfg.pwd="nogames4penfold"

station_cfg.save=false
wifi.sta.config(station_cfg)
mytimer=tmr.create()
mytimer:register(5000, 1, function()
   if wifi.sta.getip()==nil then
        print("Connecting to AP...\n")
   else
        ip, nm, gw=wifi.sta.getip()
        print("IP Info: \nIP Address: ",ip)
        print("Netmask: ",nm)
        print("Gateway Addr: ",gw,"\n")

   end
end)




--this pin controls the buzzer
buzzPin=5
gpio.mode(5, gpio.OUTPUT)


--pin controls the LED
lightPin=2
gpio.mode(2, gpio.OUTPUT)
gpio.write(lightPin,gpio.LOW)


--used as a switch for the buzzer
flag=0

--pin for the dht11 sensor
pinDHT=7

```

```
--controls the led on and off
function led(input)

if input == "ON1" then

lightPin=2
gpio.mode(2, gpio.OUTPUT)
gpio.write(lightPin,gpio.HIGH)


else if input == "OFF1" then
lightPin=2
gpio.mode(2, gpio.OUTPUT)
gpio.write(lightPin,gpio.LOW)


end
end

end




--starts the buzzer
function startBuzz()
if flag==1 then

gpio.write(buzzPin,gpio.HIGH)

end
end

--stops the buzzer
function stopBuzz()
flag=0
gpio.write(buzzPin,gpio.LOW)


end
```

```
motionPin=6
gpio.mode(motionPin, gpio.INT)

start=tmr.time()
lastEnd=0
count=0


--starts the motion sensor and keeps a count of times triggered and duration of trigger--limited by sensor
function startMotion()
  start=tmr.time()
  print('Motion detected!')
  count= count+1
  tmr.delay(1000000)
  gpio.trig(motionPin, "down", stopMotion)

  startBuzz()
end

--stops the motion sensor
function stopMotion()
  duration = tmr.time() - start
  print('motion ended after '..duration..' seconds.')
  tmr.delay(1000000)
  gpio.trig(motionPin, "up", startMotion)

  stopBuzz()
end

gpio.trig(motionPin, "up", startMotion)


mytimer:start()


--creates the TCP server
srv=net.createServer(net.TCP)
srv:listen(80,function(conn)
    conn:on("receive",function(conn,request)
        local status, temp, humi , temp_dec, humi_dec = dht.read11(pinDHT)

        --buffer fot the html code
```

```
126         --buffer fot the html code
127         local buf = ""
128         if status == dht.OK then
129             print("DHT Temperature:"..temp.." Humidity:"..humi.."\n")
130         end
131         buf=buf.."<html>"
132         buf=buf.."<head> <title>Room Sensor</title> <meta http-equiv=\"refresh\" content=\"3\"> </head>"
133         buf = buf.."<h1>ESP8266 Room sensor</h1>"
134         buf=buf.."<body><p>Temperature: "..temp.."."..temp_dec.."C</p>"
135         buf=buf.."<p>Humidity: "..humi.."."..humi_dec.."%RH</p>"
136
137         --RE to match the on/off commands and post GET req
138         local _, _, method, path, vars = string.find(request, "([A-Z]+) (.+)?(.+) HTTP")
139         if(method == nil)then
140             _, _, method, path = string.find(request, "([A-Z]+) (.+) HTTP")
141         end
142         local _GET = {}
143         if (vars ~= nil)then
144             for k, v in string.gmatch(vars, "(%w+)=(%w+)&*") do
145                 _GET[k] = v
146             end
147         end
148
149         buf = buf.."<p>LED <a href=\"?pin=ON1\"><button>ON</button></a> <a href=\"?pin=OFF1\"><button>OFF</button></a></p>"
150         buf = buf.."<p>Motion sensor Sound <a href=\"?pin=ON2\"><button>ON</button></a> <a href=\"?pin=OFF2\"><button>OFF</button></a></p>"
151         local _on,_off = "",""
152         if(_GET.pin == "ON1")then
153               gpio.write(lightPin, gpio.HIGH)
154         elseif(_GET.pin == "OFF1")then
155               gpio.write(lightPin, gpio.LOW)
156         elseif(_GET.pin == "ON2")then
157               flag=1
158               --gpio.write(switch, gpio.HIGH)
159         elseif(_GET.pin == "OFF2")then
160               --gpio.write(switch, gpio.LOW)
161               flag=0
162         end
163
164
165
166         buf=buf.."<p>"..count.."movements detected".."</p></body></html>"
167         conn:send(buf)
```

```
166         buf=buf.."<p>"..count.."movements detected".."</p></body></html>"
167         conn:send(buf)
168         conn:on("sent",function(conn) conn:close() end)
169     end)
170 end)
171 --Connect your mobile device to the AP and navigate to the IP address shown in the console
172
```

Discussion

One of the main insights drawn is the fact of how incredibly versatile such a low powered device can be, having never worked with electronic components before getting to see the code that underpins how these things work was very interesting and especially from a device the size of a matchbox with an 80hz clock speed and 50k of ram and built in Wi-Fi with injection the esp8266 is incredible. The low ram does appear to have its limitations when using a webpage to serve data as the webpage can only contain 50kb of data which is not enough for a full featured webpage, for the purposes of the project so far this is fine but any addition of style elements will use valuable memory and if the camera had been incorporated there would have been the need to limit the resolution and number of images stored as otherwise they could quickly overflow the device memory.

A major limitation with the DHT11 temperature and humidity sensor is that it cannot detect negative temperatures, this was not realised until late in the build and as such takes away from this element of the systems usefulness, it would still be fit for purpose in rooms where the temperature will NEVER go below zero but for any room where sub-zero temperatures are possible it would give false 0c readings.

The SR501 motion sensor also has several limitations the first being that the minimum reset time is 3 seconds meaning that after activation the sensor must wait 3 seconds before being able to be activated again, the code contains logic to detect the duration of movement detection but this is pretty useless with this sensor as the only way to record accurate times is to increase the reset time which then means the sensor is inactive for that period of time afterwards. Another limitation with this type of sensor is getting false positive readings from environmental elements such as sunlight or sources of heat within the room.

The advantages of the system at this point is that it is cheap at around £10 build cost and very simple to use and set up with the user only needing to plug it into a power source and get its IP address to connect to the dashboard and control the room sensor and get the readings, there is no need to register with any 3rd party or sign into anything however this also brings its own disadvantages such as the lack of a polished interface and potential security issues although as the data being recorded is not sensitive in nature so this should not present issues although if the camera had been incorporated there would have been a need to further consider the security of the device.

The design could be improved in many ways starting with improved sensors as those currently used have their limitations including possibly incorporating multiple sensors to one purpose to combine their readings to eliminate false positives , also incorporating the ability to send the user email alerts for motion detection would be useful as they could then act upon the detections in real time such as alerting someone of the intrusion and in much the same way sending some sort of alert if the temperature exceeds some given bound necessary for the room. Adding a login page would also be a future consideration once the device starts to incorporate any data that may be considered sensitive such as images.

The entire system shown in images and described in this report will be implemented with the obvious omission of the camera element.