An Internet of Things  
Home Monitoring & Control System

Aaron Scally Joyce

G00314077

Supervisor:

Michelle Lynch, Galway-Mayo Institute of Technology



Course

Computer & Electronic Engineering

**Declaration**

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Computer & Electronic Engineering at Galway Mayo Institute of Technology. This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

Acknowledgments

Firstly I would like to start off by thanking anyone who has provided any kind of support throughout the progress of my 3rd year project.

Project Supervisor: Michelle Lynch

Table of Contents

1. Acknowledgements……………………………………………………………………..2
2. Table of Contents……………………………………………………………………….3
3. Summary………………………………………………………………………………..4
4. Introduction……………………………………………………………………………..5
5. Hardware Description…………………………………………………………………..6
   * .Intel Galileo Development board
   * Lights & Relays
   * GSM modem
   * Reed Switch
   * L.D.R
6. Software Description…….……………………………………………………….........7
   * C / C++
   * xHTML
   * AT Commands
7. Conclusion……………………………………………………………………………...8

**An Internet of Things  
Home Monitoring & Control System**

This report summarises the 3rd Year project of G.M.I.T Computer Electronic Engineering Student Aaron Scally Joyce. The project was inspired by the rapidly expanding interest in the internet of things technology. It is presented by a model house incorporating florescent house bulbs and communication technology such as Ethernet and GSM, interfacing with an Intel Galileo development platform.

The user of this project can control and monitor the state of lights in the house.

The project also utilizes P.I.R Sensors to monitor movement within the house whilst the owner is away and will notify user by text message if they’re triggered.

The Galileo is acting as a server sending a webpage to a client using the xHTML language where the user can then control light states and monitor the house temperature.

Another feature is the magnetic reed switches which are mounted to the door to notify user if the door has been opened. I’ve achieved my main goal which was to make a fully functional and operational home control and monitoring system utilising text message and internet.

**Introduction**

This project is based on the idea of the Internet of Things. Things that are usually separate house hold things can easily be controlled all at once from anywhere in the world. Simply put, this is the concept of basically connecting any device with an on and off switch to the Internet (and/or to each other). This includes everything from cell phones, washing machines and almost anything else you can think of. I demonstrate this with a small basic model of a house and household bulbs to show the projects ability to control things within the house by internet or G.S.M.

**Hardware Description**

My hardware is made up of the Intel Galileo Gen 2 development board because of its great ability to be implemented with Ethernet or Wi-Fi capabilities by its on board 100Mbps Ethernet port or its mini PCI-E port on the bottom of the board it also uses an Intel Quark SOC x1000 CPU which is clocked at 400Mhz and 256Mb of DDR3 Ram which makes this more than capable for the task at hand. The Intel Galileo runs a Linux image booted from the SD card slot which allows full Ethernet capabilities, and uses the Arduino header pins for its analogue and digital I/O.

For the light control, which could easily be replaced with any appliance of your choice, I used relays. For this project more specifically I intended to build my own relay board but with concerns for safety, working with mains electricity operating at 230V AC I chose to scrap my protoboard version and opt for a more robust factory made 8 channel relay board. The board comes from a reliable big manufacturer so I had no fears of safety standards and being compatible with the same frequency and voltage used here in Ireland. The relay board consists of 8 relays in a row, each has its own signal pin which closes the normally open relay when a low logic signal is received. The signal side of the board connected to the Galileo is protected by Photo couplers preventing if in worst case scenario the high voltage manages to come back through the relay board will be stopped from reaching the Intel Galileo by the photo couplers.

A GSM shield developed with the mobile company Telefonica for use with the Arduino was used to allow SMS communication. Since the Galileo uses the same Arduino header layout it was compatible with the Galileo but was not useable with the Arduino libraries so I had to use AT commands to communicate with the GSM shield. To utilise the AT commands I sent them through the Seral1 on the Arduino IDE and connected the Galileo’s serial tx and rx (pin 0 and 1) to the gsm shields tx and rx (pin 2 and 3).

I use a reed switch which is a switch activated or deactivated with the presence of a magnet. I use tis to allow the board to tell when a door or window has been opened and send a text accordingly. How it works is by two metal leaves inside a glass casing. The two leafs of metal are pulled together with the presence of a magnet and away when the magnet leaves breaking the connection to the I/O pin.

For the back door light I use a light dependant resistor (L.D.R) to allow the board to determine how bright or dark it is outside. When the light level falls below a certain threshold set by me, the light at the back door will then be activated.

For monitoring temperature I use the LM35 sensor which gives the house temperature and will then be displayed on the webpage for the user to see.

**Software Description**

The software I used was C / C++ through the Arduino IDE, xHTML for the website and originally some CSS for the webpage stylesheet which I later removed to neaten up the code and it was causing errors within the Arduino IDE.

The main part of the project would be based on C / C++ with the Intl Galileo doing all the monitoring of the states, sensors and light control. Form me this was enjoyable to write from my 2 / 3 years’ experience writing C. I didn’t have much errors or difficulties when writing, and when I did I wasn’t too long getting through them. From this project I would say it has helped improve my understanding of code and my writing skill has come up a level.

The webpage was probably the most confusing aspect to it as I’ve had no previous knowledge or experience with html of any kind so after some research I decided to write the webpage in xHTML because of its ability to work with a wide range of devices and operating systems as well as mobile. I also chose it because it’s modern and not outdated. This was probably the most time consuming aspect of the project trying to teach myself a new language but I feel I picked it up pretty quick with the help of forums and books.

For gsm like stated earlier I had to use AT commands. AT is short for “Attention”. Each command sent begins with the letters “AT” they’re used for controlling and communicating with modems of all kinds. Luckily there was a lot of resources online which helped me with the different types of AT errors and commands I needed. It also helped broaden my knowledge of how GSM works.

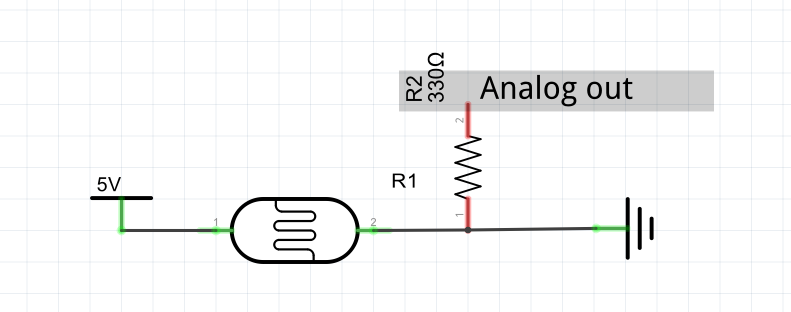
**Block diagram of the project:**



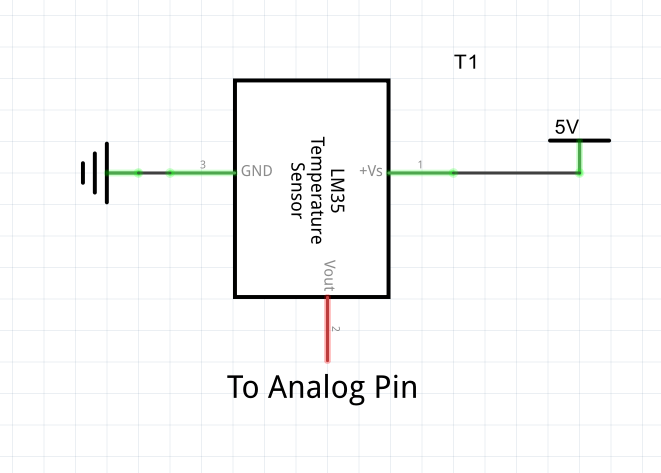


**Schematics**

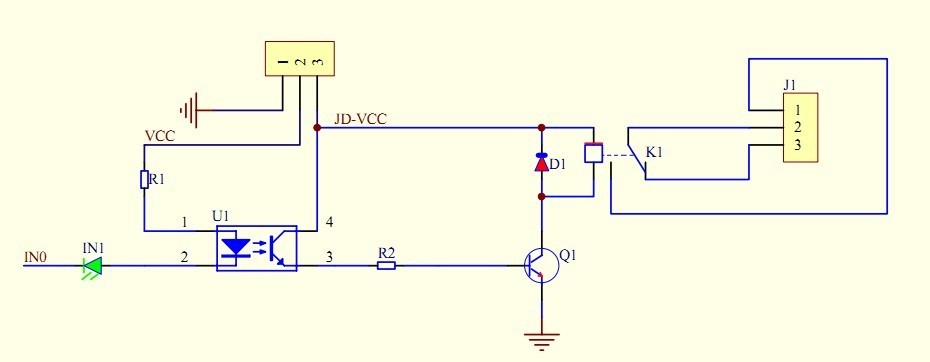
L.D.R



LM35



Single Relay Schematic



**System Integration**

I found integration pretty simple, mainly because I started out adding to my project piece by piece instead of getting all parts working separately. I started off with GSM and got the sending working with AT commands then moved on to receiving where I was stuck for a couple of weeks. I then moved onto reed switches and then to Ethernet. Here is where I had to break away from the main code and try get the web portion working separately. I found Ethernet confusing, understanding client and host and which I wanted the Galileo board to act as. Once that was operational I then moved onto writing the webpage which set me back quite some time. Having to learn the html was a challenge. An even bigger challenge was trying to get the webpage interacting with the hardware such as the relays. When it all seemed to work, incorporating it in with the already working gsm and reed switches took some time but nothing too difficult. Unfortunately I then found a bug. When a client is connected to the webpage and a GSM command comes in, the board carries out the command sent by gsm but is then reversed back to the state set by the webpage. I can get the webpage to read the change in status and display it to the webpage before its switched back, but cannot uncheck the tick box through GSM.

**Conclusion**

When I set out for project I was torn between two ideas. With some research I was intrigued with the advances with the idea of “the internet of things” and wanted to try it for myself. I was originally planning on using LED’s but went with fluorescent house bulbs because of its realism. I feel it actually shows people that house appliances running with mains 230v is easily controlled by your mobile or internet with the help of a development board. As well as controlling house appliances or lights I wanted to add some practicality to my project with GSM and a security element with reed switches on the door along with P.I.R movement sensors to let the user know if there is activity within the house while they are away. In the end I would consider my project successful. I have met my goals set out at the beginning of the year. I have a fully working home monitoring system with web control as well as having the ability to be controlled by text message. It is a compact design which I feel would be very easy to incorporate into any house, shed or whatever the need may be. It sends feedback of temperature and with the relay board more appliances can easily be added such as a house fan (if the temperature exceeds a certain point it turns on) or home heating. I am confident with my project and have a fully demonstrable model built with working light control by web or gsm and sensors with text and web feedback.

**Operating Instructions**

1. Supply power to the project by plugging it into a wall socket.
2. Connect the Ethernet cable to router or PC.
3. On the GSM modem press the power button to turn the modem on for communication.
4. Wait for a text message from the board confirming a full successful boot, and that it is ready to now receive commands.
5. In a web browser of your choice, on mobile or pc type the IP: “192.168.137.10” into the URL bar on top.
6. You will then be greeted with a nice UI clearly labelling the control options and states.
7. To use GSM you text the number of the sim which you’ve placed within the board.

The phone number that comes pre-installed: +353 85 220 7032

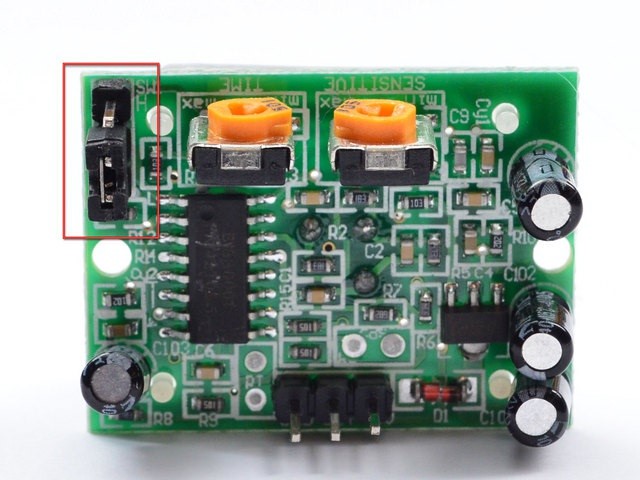
|  |  |
| --- | --- |
| Command | Result |
| On1 | Turns on light in room1 |
| Off1 | Turns off light in room1 |
| On2 | Turns on light in room2 |
| Off2 | Turns off light in room2 |
| On3 | Turns on light in room3 |
| Off3 | Turns off light in room3 |
| Ona | Turns on all lights |
| Offa | Turns off all lights |

To control room states over Text message, simply text one of the given commands to the number above. [\*\*Commands to be sent one at a time]

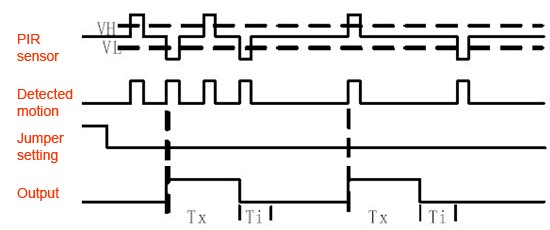
**Data Sheets**

P.I.R Movement Sensors:

To keep monitoring movement instead of going high for time period when triggered:

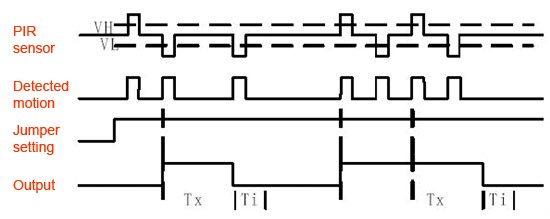


Now set up the testing board again. You may notice that when connecting up the PIR sensor as above, the LED does not stay on when moving in front of it but actually turns on and off every second or so. That is called "non-retriggering".



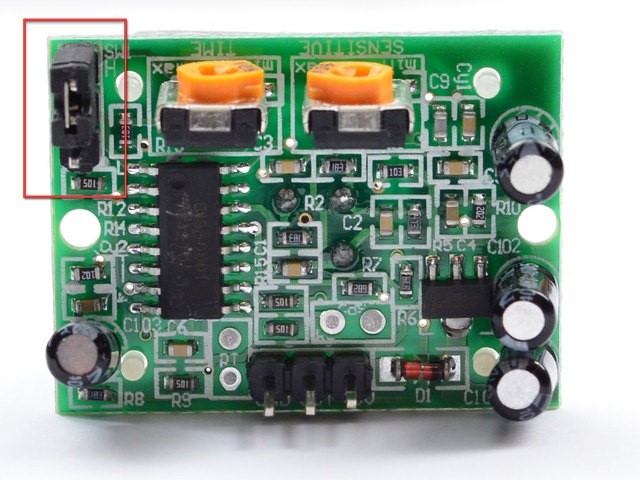
Now change the jumper so that it is in the **H** position. If you set up the test, you will notice that now the

LED *does* stay on the entire time that something is moving. That is called "retriggering".



(The graphs above are from the BISS0001 datasheet, they kind of suck)

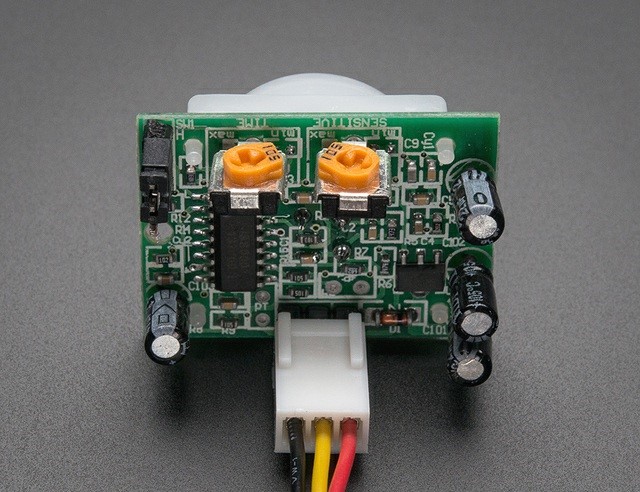
For most applications, "retriggering" (jumper in H position as shown below) mode is a little nicer.



If you need to connect the sensor to something edge-triggered, you'll want to set it to "no retriggering" (jumper in L position).

# Changing sensitivity

The Adafruit PIR has a trimpot on the back for adjusting sensitivity. You can adjust this if your PIR is too sensitive or not sensitive enough - clockwise makes it more sensitive.

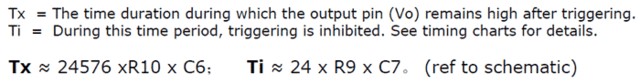


# Changing Pulse Time and Timeout Length

There are two 'timeouts' associated with the PIR sensor. One is the "**Tx**" timeout: how long the LED is lit after it detects movement - this is easy to adjust on Adafruit PIR's because there's a potentiometer.

The second is the "**Ti**" timeout which is how long the LED is guaranteed to be off when there is no movement. This one is not *easily* changed but if you're handy with a soldering iron it is within reason.

First, lets take a look at the BISS datasheet again



On Adafruit PIR sensors, there's a little trim potentiometer labeled **TIME.** This is a 1 Megaohm adjustable resistor which is added to a 10K series resistor. And **C6** is 0.01uF so

**Tx = 24576 x (10K + Rtime) x 0.01uF**

If the Rtime potentiometer is turned all the way down counter-clockwise (to 0 ohms) then

**Tx = 24576 x (10K) x 0.01uF = 2.5** seconds (approx)

If the Rtime potentiometer is turned all the way up clockwise to 1 Megaohm then

**Tx = 24576 x (1010K) x 0.01uF = 250** seconds (approx)

If RTime is in the middle, that'd be about 120 seconds (two minutes) so you can tweak it as necessary.

For example if you want motion from someone to turn on a fan for a minimum of 1 minute, set the Rtime potentiometer to about 1/4 the way around.

**The Code:**

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <OnewireKeypad.h>

#include <String.h>

#include <LiquidCrystal\_I2C.h>

#include <SPI.h>

#include <Ethernet.h>

int i**,** n**;**

int Relay1 **=** 4**;**

int Relay2 **=** 5**;**

int Relay3 **=** 6**;**

float tempC**;**

int reading**;**

int LDR\_Read**;**

int tempPin **=** A0**;** // pin for temperature sensor input

int LDR **=** A1**;** // pin for LDR input

String msg**;**

String MotionDetected **=** "Movement Has Been Detected!"**;** // defining different preset messsages

String Startup **=** "Setup and PIR Calibration Complete..Ready for commands!"**;**

String DoorMsg **=** "Door Has Been Opened"**;**

byte mac**[]** **=** **{**0x98**,** 0x4F**,** 0xEE**,** 0x01**,** 0x81**,** 0xEA**};**

IPAddress ip**(**192**,** 168**,** 137**,** 10**);** // sets the ip of webpage

char**\*** SensReadOut**[]** **=** **{**"OFF"**,** "ON"**};** // used for the webpage light status

EthernetServer server**(**80**);**

//sets the default status for lights

int Relay1Status **=** 0**;**

int Relay2Status **=** 0**;**

int Relay3Status **=** 0**;**

int Relay4Status **=** 0**;**

int BackdoorLight **=** 13**;**

// String to store the incoming HTTP request

String HTTPRequest**;**

char password**[]** **=** **{**1**,** 2**,** 3**,** 4**};**

char userInput**[**4**];**

int calibrationTime **=** 20**;** //the time we give the sensor to calibrate (10-60 secs according to the datasheet)

int pirRoom1 **=** 8**;** //the digital pin connected to the PIR sensor's output

int pirRoom2 **=** 9**;**

int pirRoom3 **=** 10**;**

int BackPir **=** 11**;**

int doorSwitch **=** 12**;**

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

int pir2State **=** LOW**;**

int pir3State **=** LOW**;**

int pirbackState **=** LOW**;**

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

char KEYS**[]** **=** **{**

'1'**,** '2'**,** '3'**,**

'4'**,** '5'**,** '6'**,**

'7'**,** '8'**,** '9'**,**

'\*'**,** '0'**,** '#'**,**

**};**

OnewireKeypad **<**Print**,** 12 **>** Keypad**(**Serial**,** KEYS**,** 4**,** 3**,** A0**,** 4700**,** 1000 **);**

void setup**()** **{**

//###### INPUTS ######

pinMode**(**pirRoom1**,** INPUT**);**

pinMode**(**pirRoom2**,** INPUT**);**

pinMode**(**pirRoom3**,** INPUT**);**

pinMode**(**BackPir**,** INPUT**);**

//###### OUTPUTS ######

pinMode**(**Relay1**,** OUTPUT**);**

pinMode**(**Relay2**,** OUTPUT**);**

pinMode**(**Relay3**,** OUTPUT**);**

pinMode**(**BackdoorLight**,**OUTPUT**);**

Serial**.**print**(**"calibrating sensor "**);**

**for(**int i **=** 0**;** i **<** calibrationTime**;** i**++){**

Serial**.**print**(**"."**);**

delay**(**1000**);**

**}**

Serial**.**println**(**" done"**);**

Serial**.**println**(**"SENSOR ACTIVE"**);**

delay**(**50**);**

Serial**.**begin**(**9600**);**

Serial1**.**begin**(**115200**);**// the GPRS baud rate

Serial1**.**println**(**"AT+CMGF=1"**);**

delay**(**2000**);**

Serial1**.**println**(**"AT+CMGD=1,4"**);**

delay**(**2000**);**

digitalWrite**(**Relay1**,** HIGH**);**

digitalWrite**(**Relay2**,** HIGH**);**

digitalWrite**(**Relay3**,** HIGH**);**

digitalWrite**(**BackdoorLight**,**LOW**);**

Ethernet**.**begin**(**mac**,** ip**);**

server**.**begin**();**

Serial**.**print**(**"server is at "**);**

Serial**.**println**(**Ethernet**.**localIP**());**

Serial**.**println**(**"Ready For Message"**);**

msg **=** Startup**;**

SendTextMessage**();**

**}**

void loop**()** **{**

LDR\_Read **=** analogRead**(**LDR**);**

Serial**.**print**(**"LDR Reading is: "**);**

Serial**.**println**(**LDR\_Read**);**

**if(**LDR\_Read **<**60**){**

digitalWrite**(**BackdoorLight**,**HIGH**);**

delay**(**5000**);}**

**else**

digitalWrite**(**BackdoorLight**,**LOW**);**

int doorState **=** digitalRead**(**doorSwitch**);**

Serial**.**print**(**"Your Door is "**);**

Serial**.**println**(**doorState**);**

**if(**doorState **==** LOW**)**

**{**

digitalWrite**(**BackdoorLight**,**HIGH**);**

Serial**.**println**(**"Door has been opened"**);**

msg **=** "Back door has been opened"**;**

SendTextMessage**();** // sends a text when door has been opened

**}**

**else** **if(**BackdoorLight**,**HIGH**){**

digitalWrite**(**BackdoorLight**,**LOW**);**

**}**

ReceiveTextMessage**();** // lookin for message for command to turn on or off lights

delay**(**100**);**

EthernetClient client **=** server**.**available**();**

// Serve web page & handle client requests

**if(**client**)** **{**

Serial**.**println**(**"New client connected"**);**

WebCheck**(**client**);**

**}**

// Close the connection

client**.**stop**();**

PIRSens**();**//calls for movement sensor scan

**}**

//%%%%%%%%%%%%%%%%%%%%%%{FUNCTIONS}%%%%%%%%%%%%%%%%%%%%

/\*

void Keypad() {

Keypad.SetHoldTime(100); // Key held time in ms

Keypad.SetDebounceTime(50);

Serial.println("Enter Password");

for(i=1;i<5;i++){

Serial.print("Enter Digit ");

Serial.println(i);

while(Keypad.Key\_State() == 0) {}

// data = serialRead();

if((Keypad.Key\_State() == 3))

userInput[i] = Keypad.Getkey();

Serial.print(userInput[i]);

while ((Keypad.Key\_State())) {}

}

Serial.print("Password Entered is: ");

for(i=0;i<4;i++){Serial.print(userInput[i]);}

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

}

}

\*/

void PIRSens**(){**

//############################ ROOM 1 ################################

val **=** digitalRead**(**pirRoom1**);** // read input value

**if** **(**val **==** LOW**)** **{** // check if the input is LOW

//digitalWrite(Relay1, LOW); // turn LED ON

**if** **(**pirState **==** LOW**)** **{**

// we have just turned on

Serial**.**println**(**"Motion ended in room 1"**);**

msg **=** "Movement Detected in Room1"**;**

SendTextMessage**();**

// We only want to print on the output change, not state

pirState **=** HIGH**;**

**}**

**}**

**else** **{**

//digitalWrite(Relay1, HIGH); // turn LED OFF

**if** **(**pirState **==** HIGH**){**

// we have just turned off

Serial**.**println**(**"Motion Detected in room 1"**);**

pirState **=** LOW**;**

**}**

**}**

//################################ ROOM 2 #############################

int pir2val **=** digitalRead**(**pirRoom2**);** // read input value

**if** **(**pir2val **==** LOW**)** **{** // check if the input is LOW

**if** **(**pir2State **==** HIGH**){**

// we have just turned off

Serial**.**println**(**"Motion ended in room2 !"**);**

// We only want to print on the output change, not state

pir2State **=** LOW**;**

**}**

**}**

**else** **{**

**if** **(**pir2State **==** LOW**)** **{**

// we have just turned on

Serial**.**println**(**"Motion detected in room2 "**);**

//msg = "Movement Detected in Room2";

//SendTextMessage();

pir2State **=** HIGH**;**

**}**

**}**

//\*\*\*\* These ports used are broken so the code is commented out \*\*\*\*\*\*\*\*

//########################## ROOM 3 ###############################

/\*

int pir3val = digitalRead(pirRoom3); // read input value

if (pir3val == LOW) { // check if the input is LOW

if (pir3State == HIGH){

// we have just turned off

Serial.println("Motion ended in room3 !");

// We only want to print on the output change, not state

pir3State = LOW;

}

}

else {

if (pir3State == LOW) {

// we have just turned on

Serial.println("Motion detected in room3 ");

// We only want to print on the output change, not state

pir3State = HIGH;

}

}

//################################# Back Door #############################

//if(LDR < 30){

do{

//Serial.println(LDR\_Read);

int BackPirVal = digitalRead(pirRoom2); // read input value

if (pir2val == LOW) { // check if the input is LOW

if (pir2State == HIGH){

// we have just turned off

Serial.println("Motion ended in room2 !");

// We only want to print on the output change, not state

pir2State = LOW;

}

}

else {

if (pir2State == LOW) {

// we have just turned on

Serial.println("Motion detected in room2 ");

//msg = "Movement Detected in Room2";

//SendTextMessage();

// We only want to print on the output change, not state

pir2State = HIGH;

}

}

}

while(LDR\_Read <50);

\*/

**}**

void SendTextMessage**()** **{**

Serial1**.**println**(**"AT+CMGS=\"+353876373466\""**);**

delay**(**2000**);**

Serial1**.**println**(**msg**);**

delay**(**1000**);**

Serial1**.**write**(**0x1A**);**//the ASCII code of the ctrl+z

delay**(**1000**);**

Serial1**.**println**();**

**}**

void ReceiveTextMessage**()** **{**

char t**;**

Serial1**.**println**(**"AT+CPMS=\"SM\""**);** // Lists SMS

delay**(**1000**);**

Serial1**.**println**(**"AT+CMGR=1"**);** // Reads SMS Address 1

delay**(**1000**);**

**while** **(**Serial1**.**available**()** **>** 0**)** **{**

t **=** Serial1**.**read**();**

Serial**.**print**(**t**);**

**if** **(**t **==** 'o' **||** t **==** 'O'**)** **{**

t **=** Serial1**.**read**();**

**if** **(**t **==** 'n'**)** **{**

n **=** Serial1**.**read**();**

**if** **(**n **==** '1'**)** **{**

digitalWrite**(**Relay1**,** LOW**);**

Relay1Status **=** 1**;**

**}**

**else** **if** **(**n **==** '2'**)** **{**

digitalWrite**(**Relay2**,** LOW**);**

Relay2Status **=** 1**;**

**}**

**else** **if** **(**n **==** '3'**)** **{**

digitalWrite**(**Relay3**,** LOW**);**

Relay3Status **=** 1**;**

**}**

**else** **if** **(**n **==** 'a' **||** n **==** 'A'**)** **{**

digitalWrite**(**Relay1**,** LOW**);**

digitalWrite**(**Relay2**,** LOW**);**

digitalWrite**(**Relay3**,** LOW**);**

//digitalWrite(Relay4, LOW);

//digitalWrite(Relay5, LOW);

Relay1Status **=** 1**;**

Relay2Status **=** 1**;**

Relay3Status **=** 1**;**

Serial1**.**println**(**"AT+CMGD=1,4"**);**

delay**(**500**);**

**}**

delay**(**1000**);**

Serial1**.**println**(**"AT+CMGD=1,4"**);**

delay**(**500**);**

**break;**

Serial**.**print**(**"End of Messgage"**);**

**}**

**else** **if** **(**t **!=** 'n' **&&** t **==** 'f'**)**

**{**

t **=** Serial1**.**read**();**

**if** **(**t **=** 'f'**)**

**{**

n **=** Serial1**.**read**();**

**if** **(**n **==** '1'**)** **{**

digitalWrite**(**Relay1**,** HIGH**);**

Relay1Status **=** 0**;**

**}**

**else** **if** **(**n **==** '2'**)** **{**

digitalWrite**(**Relay2**,** HIGH**);**

Relay2Status **=** 0**;**

**}**

**else** **if** **(**n **==** '3'**)** **{**

digitalWrite**(**Relay3**,** HIGH**);**

Relay3Status **=** 0**;**

**}**

**else** **if** **(**n **==** 'a' **||** n **==** 'A'**)** **{**

digitalWrite**(**Relay1**,** HIGH**);**

digitalWrite**(**Relay2**,** HIGH**);**

digitalWrite**(**Relay3**,** HIGH**);**

// digitalWrite(Relay4, HIGH);

//digitalWrite(Relay5, HIGH);

Relay1Status **=** 0**;**

Relay2Status **=** 0**;**

Relay3Status **=** 0**;**

Serial1**.**println**(**"AT+CMGD=1,4"**);**

delay**(**500**);**

**}**

//delay(1000);

Serial1**.**println**(**"AT+CMGD=1,4"**);**

delay**(**500**);**

Serial**.**println**(**"End of Messgage"**);**

**}**

**else** **{**

**break;**

**}**

**}**

**}**

**}**

**}**

void WebCheck**(**EthernetClient **&**client**)** **{**

reading **=** analogRead**(**tempPin**);**

tempC **=** reading **/** 9.31**;**

Serial**.**print**(**"Relay 1 Status: "**);**

Serial**.**println**(**Relay1Status**);**

Serial**.**println**(**tempC**);**

boolean currentLineIsBlank **=** **true;**

**while** **(**client**.**connected**())** **{**

**if** **(**client**.**available**())** **{**

char c **=** client**.**read**();**

HTTPRequest **+=** c**;**

Serial**.**write**(**c**);**

// if you've gotten to the end of the line (received a newline

// character) and the line is blank, the http request has ended,

// so you can send a reply

**if** **(**c **==** '\n' **&&** currentLineIsBlank**)** **{**

// send a standard http response header

client**.**println**(**"HTTP/1.1 200 OK"**);**

client**.**println**(**"Content-Type: text/html"**);**

client**.**println**(**"Connection: close"**);**

client**.**println**();**

client**.**println**(**"<!DOCTYPE HTML>"**);**

client**.**println**(**"<html>"**);**

// add a meta refresh tag, so the browser pulls again every 5 seconds:

client**.**println**(**"<meta http-equiv=\"refresh\" content=\"5\">"**);**

// output the value of each analog input pin

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HEAD \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

client**.**println**(**"<head> "**);**

client**.**println**(**"<style type=\"text/css\">table{font-family:arial;}</style>"**);**

client**.**print**(**"<title>Intel Galileo Project</title>"**);**

client**.**print**(**"</title>"**);**

client**.**println**(**"</header>"**);**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* BODY \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

client**.**println**(**"<body bgcolor=\"#6495ED\">"**);**//gives the color to the webpage background light blue

//client.println("<body>");

client**.**println**(**"<h1><center>Galileo Project</center></h1>"**);**

client**.**println**(**"<h2><center>Home Monitoring & Control System</center></h2><hr />"**);**

client**.**print**(**"<h3><center>House Temperature is "**);**

client**.**print**(**tempC**);**

client**.**println**(**" Celcius</center></h3><hr />"**);**

client**.**println**(**"<center><strong><table border=\".3\" width=\"50%\" cellpadding=\"10\">"**);**

client**.**println**(**"<tr><th colspan=\"2\"><h3>Room</h3></th><th><h3>State</strong></h3></tr>"**);**

// prints out the table on webpage uses port numbers to make the rows

**for** **(**int analogChannel **=** 4**;** analogChannel **<=** 6**;** analogChannel**++)** **{**

int sensorReading **=** digitalRead**(**analogChannel**);**

client**.**println**(**"<tr>"**);**

client**.**print**(**"<td colspan=\"2\">"**);**

client**.**print**(**"Room "**);**

client**.**print**(**analogChannel **-** 3**);**

client**.**print**(**" is "**);**

client**.**println**(**"</td>"**);**

**if** **(**sensorReading **>** 0**)**

**{**

client**.**print**(**"<td><center><input type=\"button\" id=\""**);**

client**.**print**(**analogChannel **-** 3**);**

client**.**print**(**"\"value=\""**);**

client**.**print**(**SensReadOut**[**0**]);**

client**.**print**(**"\""**);**

//client.read("");

**}**

**else**

**{**

client**.**print**(**"<td><center><input type=\"button\"value=\""**);**

client**.**print**(**SensReadOut**[**1**]);**

client**.**print**(**"\""**);**

**}**

client**.**print**(**"onclick=\"this.value=this.value=='ON'?'OFF':'ON';\"></input></center></td>"**);**

client**.**println**(**"</tr>"**);**

client**.**println**(**"<br />"**);**

**}**

client**.**print**(**"</table></strong></center>"**);**

// Use a HTML form to get the user input via a checkbox

client**.**print**(**"<center>"**);**

client**.**println**(**"<form method=\"get\">"**);**

// First check if the checkbox has been clicked

//defining all differant combinations of lights

**if(**HTTPRequest**.**indexOf**(**"GET /?Relay1=2"**)** **>** **-**1**)**

Relay1Status **=** 1**;**

**else** **if(**HTTPRequest**.**indexOf**(**"GET /?"**)** **>** **-**1**)**

Relay1Status **=** 0**;**

**if(**HTTPRequest**.**indexOf**(**"GET /?Relay2=3"**)** **>** **-**1**)**

Relay2Status **=** 1**;**

**else** **if(**HTTPRequest**.**indexOf**(**"GET /?"**)** **>** **-**1**)**

Relay2Status **=** 0**;**

**if(**HTTPRequest**.**indexOf**(**"GET /?Relay3=4"**)** **>** **-**1**)**

Relay3Status **=** 1**;**

**else** **if(**HTTPRequest**.**indexOf**(**"GET /?"**)** **>** **-**1**)**

Relay3Status **=** 0**;**

**if(**HTTPRequest**.**indexOf**(**"GET /?Relay2=3&Relay3=4"**)** **>** **-**1**){**

Relay2Status **=** 1**;**

Relay3Status **=** 1**;**

**}**

**if(**HTTPRequest**.**indexOf**(**"GET /?Relay1=2&Relay2=3"**)** **>** **-**1**){**

Relay1Status **=** 1**;**

Relay2Status **=** 1**;**

**}**

**if(**HTTPRequest**.**indexOf**(**"GET /?Relay1=2&Relay3=4"**)** **>** **-**1**){**

Relay1Status **=** 1**;**

Relay3Status **=** 1**;**

**}**

**if(**HTTPRequest**.**indexOf**(**"GET /?Relay1=2&Relay2=3&Relay3=4"**)** **>** **-**1**){**

Relay1Status **=** 1**;**

Relay2Status **=** 1**;**

Relay3Status **=** 1**;**

**}**

// Display the checked/unchecked box on the webpage as requested

// & turn on/off the Relay accrdingly

**if(**Relay1Status **==** 1**)** **{**

client**.**println**(**"<input type=\"checkbox\" name=\"Relay1\" value=\"2\" onclick=\"submit();\" checked>Relay1"**);**

digitalWrite**(**Relay1**,** LOW**);**

**}**

**else** **{**

client**.**println**(**"<input type=\"checkbox\" name=\"Relay1\" value=\"2\" onclick=\"submit();\">Relay1"**);**

digitalWrite**(**Relay1**,** HIGH**);**

**}**

**if(**Relay2Status **==** 1**)** **{**

client**.**println**(**"<input type=\"checkbox\" name=\"Relay2\" value=\"3\" onclick=\"submit();\" checked>Relay2"**);**

digitalWrite**(**Relay2**,** LOW**);**

**}**

**else** **{**

client**.**println**(**"<input type=\"checkbox\" name=\"Relay2\" value=\"3\" onclick=\"submit();\">Relay2"**);**

digitalWrite**(**Relay2**,** HIGH**);**

**}**

**if(**Relay3Status **==** 1**)** **{**

client**.**println**(**"<input type=\"checkbox\" name=\"Relay3\" value=\"4\" onclick=\"submit();\" checked>Relay3"**);**

digitalWrite**(**Relay3**,** LOW**);**

**}**

**else** **{**

client**.**println**(**"<input type=\"checkbox\" name=\"Relay3\" value=\"4\" onclick=\"submit();\">Relay3"**);**

digitalWrite**(**Relay3**,** HIGH**);**

**}**

// Write the closing HTML tags to the webpage

client**.**println**(**"</form></center>"**);**

client**.**println**(**"</body>"**);**

client**.**print**(**"<footer><hr /><em>Aaron Scally Joyce B.Eng Comp Electronic Eng G.M.I.T</em></footer>"**);**

client**.**println**(**"</html>"**);**

HTTPRequest **=** ""**;**

**break;**

**}**

**if** **(**c **==** '\n'**)** **{**

// you're starting a new line

currentLineIsBlank **=** **true;**

**}**

**else** **if** **(**c **!=** '\r'**)** **{**

// you've gotten a character on the current line

currentLineIsBlank **=** **false;**

**}**

**}**

**}**

// give the web browser time to receive the data

delay**(**1**);**

**}**