# Automatic Light - Version 1

25th April 2016

## Overview:

This was my first project involving home automation. It was relatively inexpensive and low-tech. However, despite this, the outcome was largely successful. By the end of this project, I was able to control my bedroom light using a basic web interface and the light would also turn on and off automatically by using a homemade door sensor and a pressure plate to detect whether someone walked in or out.

Here’s a demonstration of the system in action: <https://www.youtube.com/watch?v=cyKxYWvNoTQ>

In retrospect, I think this project could be improved in many ways:

1. Another pressure plate should be added to accurately determine if someone has walked in or out. One pressure plate in the room, one outside.
2. Wiring and circuitry could be neatened to decrease cost of wire and components, and also to improve aesthetics.

I may revisit this project in the future and make these improvements, the only problem that I have is that the raspberry pi that I am using doesn’t have much power output left. If I was to add another pressure plate or other sensor, the pi may not be able to power the solid state relay that controls the light. Furthermore, Mum may not be happy with me pulling up the carpet in the hallway to stick a pressure plate under the carpet.

## Breakdown:

## Inputs:

**Door Sensor:**

The door sensor consisted of a coin and a spring, when the door opened the circuit was broke. When the door was shut, current could flow.

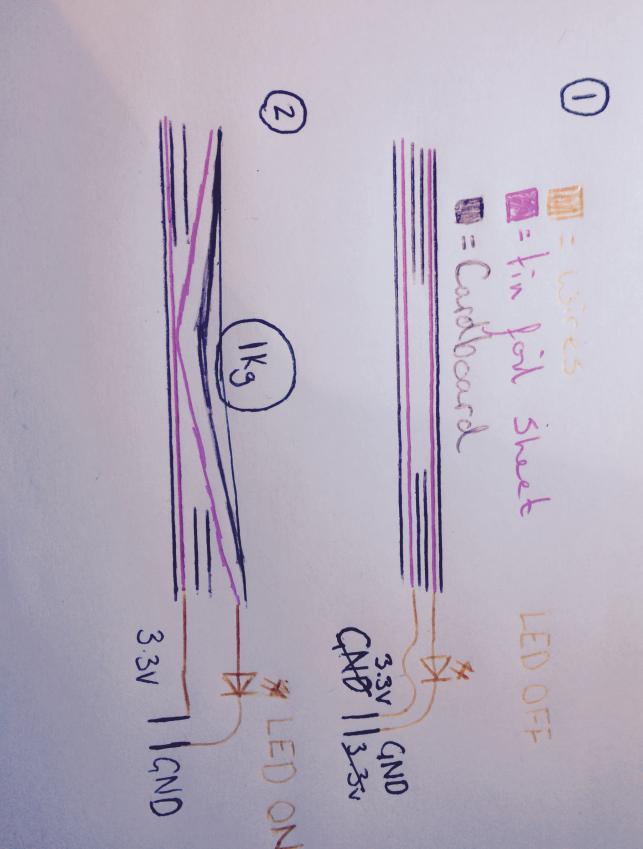
Here’s a clip of how it works:

<https://www.youtube.com/watch?v=tpZZki4fnNM>

**Pressure Plate:**

I made a pressure plate from some cardboard and tinfoil, I layered the cardboard and sandwiched two sheets of tinfoil between them, a potential difference is applied to the two sheets of tin foil. When the plate is stepped on, the two sheets touch and current flows. When the plate is stepped off, the circuit breaks.

Drawing of the concept:



Although the picture isn’t very clear, I assume you get the idea.

I used two pressure plates, one on the wall (hidden behind a poster), which acts as a normal light switch and one on the floor under a rug which is triggered when someone walks in.

**Light Switch:**

In a regular light switch, the mains wires are put in the back and when you switch it current is either blocked or allowed to flow. Instead of having mains current flowing through my light switch I had one node connected to 3.3v my pi and the other connected to an input pin on my pi. If I still had the power of the light controlled through the light switch I would not be able to control my light from the Raspberry Pi. This meant that the light could still be used like normal from the switch, on top of the automation.

## Outputs:

**Solid State Relay Connected to Mains Light:**

I used a Solid State Relay to control the high voltages of my bedroom light through the low voltages that the raspberry pi can output. Here is the Solid State Relay (SSR) that I used:



I managed to tuck the SSR into the wall behind the light switch and run the wire out the side of the switch.

Now, with these inputs and the output, I was able to start coding the Web Interface and the automation.

Here’s what the pi looks like with the circuit I soldered together, although this doesn’t really mean much:



## Automation Logic:

Once I began thinking about the logic of this problem, I quickly realised that a door sensor and a pressure plate were not really sufficient to fully determine whether a person has walked in or out of the room and this is one of the things that I would improve upon if I redone this project.

Despite the system’s limitations, I came up with certain sequences of inputs that would help decide on whether to turn the light on or off. I would store the order inputs were activated in an array of length four. The most recent input is the rightmost stored in the array, when new inputs are received the first item in the array is deleted and everything is shifted left one. For example, if someone was to open the door the input array would be:

[“”, “”, “”, “DO”]

If they then trod on the pressure plate, the input array would be:

[“”, “”, “DO”, “PA”]

If they then stepped off:

[“”, “DO”, “PA”, “PO”]

Where: PA = Plate Activated, i.e. Plate has been stepped on

PO = Plate Off, i.e. Plate has been stepped on

DO = Door Opened

DC = Door Closed

These are the sequences that I used:

### Walked In Sequences:

["DO", "DC", "PA", "PO"]

["DO", "PA", "DC", "PO"]

### Walked Out Sequences:

["PA", "PO", "DO", "DC"]

["PA", "DO", "PO", "DC"]

["DO", "DC", "DO", "DC"]

After implementing this, I would say that the system is about 70% accurate with detecting when a person walks in or out.

## Source Code:

Below the source code is stored as text, but I find that hard to read. Here’s source code in a more readable format:

<https://github.com/BenjaminThomas1999/Automatic-Light-Raspberry-Pi/blob/master/HardwareIO.py>

import time, os, thread

import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)

GPIO.setup(24, GPIO.OUT)

all\_inputs = []

class HwInput(object):

def \_\_init\_\_(self, PIN, name):

self.PIN = PIN

self.name = name

GPIO.setup(PIN, GPIO.IN, pull\_up\_down=GPIO.PUD\_DOWN)

all\_inputs.append(self)

self.history = [0, 0]

def state(self):

if not self.history[0] == self.history[1]:

if self.history[1] == 1:

return True

elif self.history[1] == 0:

return False

else:

return None

def update(self):

self.history[0] = self.history[1]

self.history[1] = GPIO.input(self.PIN)

sequence = ["" , "", "", ""]

door\_sensor = HwInput(18, "door\_sensor")

pressure\_plate = HwInput(23, "pressure\_plate")

wall\_pad = HwInput(25, "wall\_pad")

main\_switch = HwInput(4, "main\_switch")

light\_state = True

def cycle(arr, item):

arr.append(item)

return arr[1:]

def toggleLight():

global light\_state

light\_state = not light\_state

def ioUpdate():

global light\_state, sequence

for i in all\_inputs:#updates input states

i.update()

if wall\_pad.state() == True:

toggleLight()

if main\_switch.state() == True or main\_switch.state() == False:

light\_state = not light\_state

if door\_sensor.state() == True:

sequence = cycle(sequence, "DC")#DC = Door Closed

elif door\_sensor.state() == False:

sequence = cycle(sequence, "DO")#DO = Door Opened

elif pressure\_plate.state() == True:

sequence = cycle(sequence, "PA")#PA = Plate Activated

elif pressure\_plate.state() == False:

sequence = cycle(sequence, "PO")#PO = Plate Off

if sequence == ["DO", "PA", "PO", "DC"]:#could be walking in or out. Invert light state

light\_state = not light\_state

sequence = ["", "", "", ""]

elif sequence == ["DO", "DC", "PA", "PO"]:

light\_state = True

sequence = ["", "", "", ""]

elif sequence == ["DO", "PA", "DC", "PO"]:

light\_state = True

sequence = ["", "", "", ""]

elif sequence == ["PA", "PO", "DO", "DC"]:

light\_state = False

sequence = ["", "", "", ""]

elif sequence == ["PA", "DO", "PO", "DC"]:

light\_state = False

sequence = ["", "", "", ""]

elif sequence == ["DO", "DC", "DO", "DC"]:

light\_state = False

sequence = ["", "", "", ""]

elif sequence == ["PA", "PO", "PA", "PO"]:

light\_state = True

sequence = ["", "", "", ""]

web\_file = open("/var/www/html/output.txt")

web\_input = web\_file.read()

if web\_input == "1":

light\_state = True

elif web\_input == "0":

light\_state = False

web\_file.close()

web\_file = open("/var/www/html/output.txt", "w")#clear file

web\_file.close()

web\_output = open("/var/www/html/state.txt", "w")

if light\_state == True:

GPIO.output(24, False)

web\_output.write("on")

else:

GPIO.output(24, True)

web\_output.write("off")

web\_output.close()

def ioUpdateLoop():

while True:

ioUpdate()

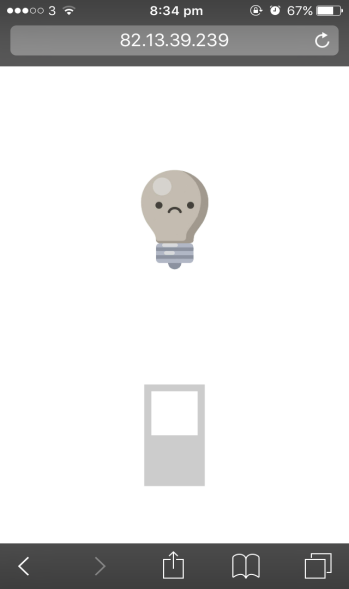
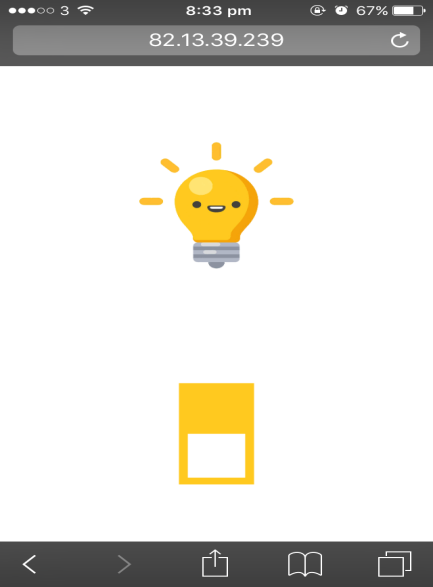
time.sleep(0.1)

if \_\_name\_\_ == '\_\_main\_\_':

ioUpdateLoop()

## The Web Interface:

Here’s two screen shots of the web interface when the light is turned on and off:



The light can be turned on or off by either pressing the light bulb or the switch on screen. The web page uses AJAX to automatically update the light state on screen if the light is turned on or off by physical methods and it also uses AJAX to send the on or off command to the Raspberry Pi Server without refreshing the page.

When an “On” command is sent to the server, a php script writes a 1 to output.txt in the server directory (/var/www/html)/ If an “Off” command is sent, a 0 is written to output.txt

This file (output.txt) is then read by the main script which controls the Inputs and Outputs, and this script turns the light on or off depending on the contents of output.txt and then it clears the file.

## Evaluation:

I would say that this project was largely successful for my first home automation attempt. As previously mentioned, the automation system works about 70% of the time which is acceptable. I think this project could be simplified significantly by adding a second pressure plate outside the door, this would make walk-in/walk-out detection much simpler as it would just be the order the plates were triggered. (The door sensor would no longer be necessary)

Furthermore, another issue with this project is that the automation system does not work if you don’t know to tread on the pressure plate and where the pressure plate is. This could be fixed by perhaps using a laser trigger system instead of pressure plates, however this would be hard to implement due to the power limitations of my Raspberry Pi