20140814 - Lesson 12 and 13 (Movement Sensing and Power Control)

The lesson this week will be using a pIR sensor, and a magnetic reed switch to display on the screen On/Off conditions. The 2nd part of the lesson will take what we learned from Lesson 12 and use the pIR sensor to turn on a outlet (or in our case a light bulb).

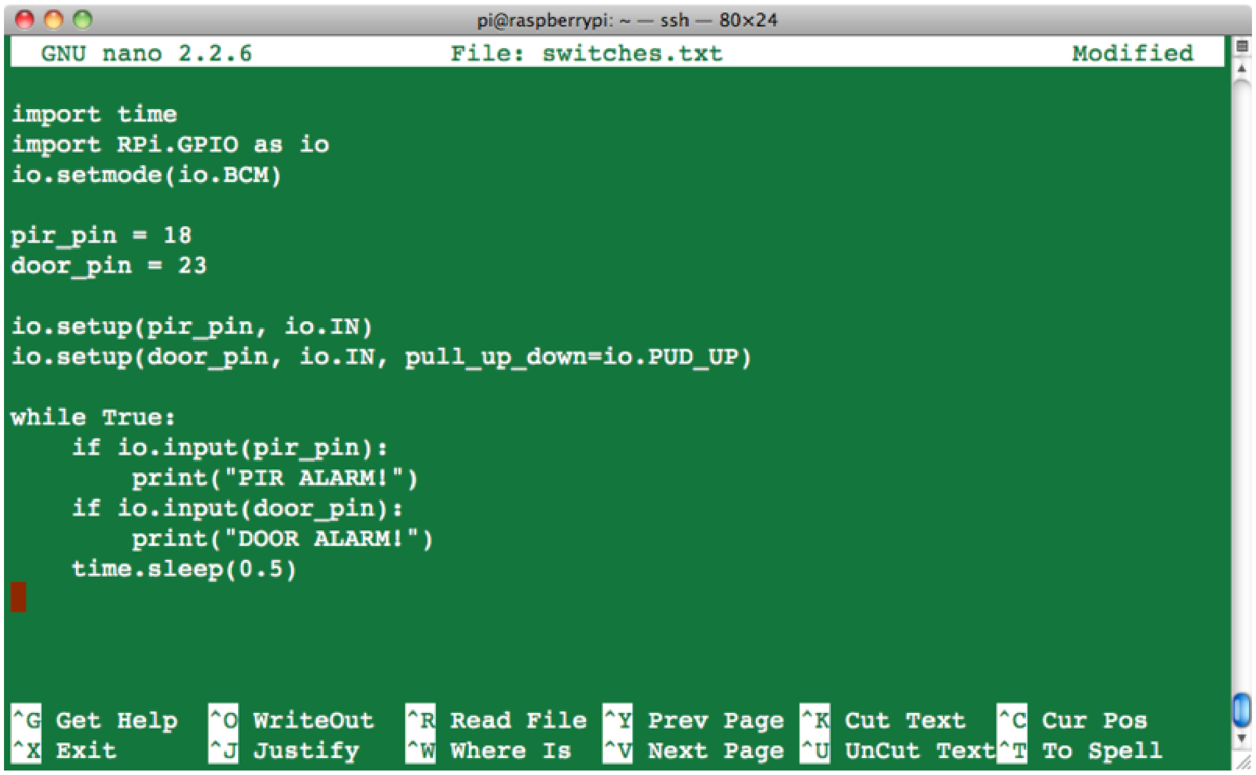
So to start off we need a pIR sensor, Magnetic reed switch, breakout board, breadboard, relay, batteries, a light bulb, and of course the Raspberry Pi.

https://learn.adafruit.com/adafruits-raspberry-pi-lesson-12-sensing-movement

The important thing to take away from Lesson 12 is how to activate the built in Pull-UP resistor for use with the reed-switch. Which is this code:

io.setup(door_pin, io.IN, pull_up_down=io.PUD_UP)

The pIR's output is 3.3v volt and doesn't need a pull-up because it is digital.



pi@raspberrypi: ~ — ssh — 80×24

```
  GNU nano 2.2.6              File: switches.txt                       Modified

import time
import RPi.GPIO as io
io.setmode(io.BCM)

pir_pin = 18
door_pin = 23

io.setup(pir_pin, io.IN)
io.setup(door_pin, io.IN, pull_up_down=io.PUD_UP)

while True:
    if io.input(pir_pin):
        print("PIR ALARM!")
    if io.input(door_pin):
        print("DOOR ALARM!")
    time.sleep(0.5)
```

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

So what else does the code say to do?
It will display the words: "PIR ALARM!" or "DOOR ALARM" if either one are triggered. pretty simple and straightforward.

remember to run the program you need to execute it with the sudo command:
sudo python switches.py

Once that works, you might asked what good is that going to do? Well, how about turning a light on for a period of time and turning it off again.

Lesson 13 is just that - using the pIR sensor and a relay we can turn on a light, the light will remain on for 20 seconds and turn off and wait for motion again.

https://learn.adafruit.com/adafruits-raspberry-pi-lesson-13-power-control

Really, You could use the reed switch to do the same thing, think about a closet with a bulb in it, you want it on while the door is open, if you forget to close the door the light will shut it's self off, of course the code would need to be changed to not trigger again because the door was still open. (anyways, that is just a thought)

In this lesson Adafruit uses something called a "Power Switch Tail 2" https://www.adafruit.com/products/268
This looks like a 110v extension of a outlet. It is really a controllable opto-ilsolated solid-state relay. On one side you have the high AC voltage (110v) on the other side you have a low 3.3v (5v) that when it goes high will turn on the AC voltage.

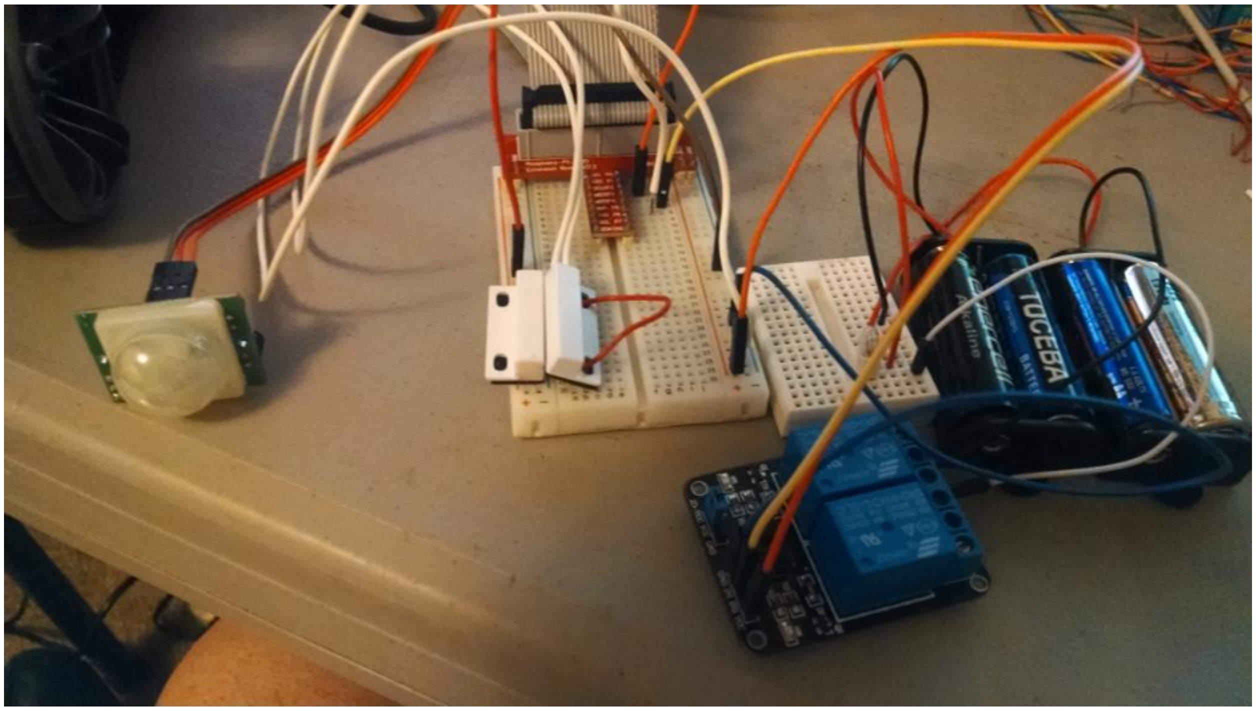
For this demo, I'm just going to use a common relay - my realy can handle 110v AC, but for now I'm just going to use 6 volts DC on one side, and 3.3v on the other. My relay is also a mechanical relay. Also it works just the opposite of the the "Tail" in that a low will activate my relay, so the code needs to be changed slightly for me.
This is the code from Adafruit. The code I used is the same making all the True to False, and all the False to True

```
import time
import RPi.GPIO as io
io.setmode(io.BCM)

pir_pin = 24
power_pin = 23

```

```
io.setup(pir_pin, io.IN)
io.setup(power_pin, io.OUT)
io.output(power_pin, False)

while True:
    if io.input(pir_pin):
        print("POWER ON")
        io.output(power_pin, True)
        time.sleep(20);
        print("POWER OFF")
        io.output(power_pin, False)
        time.sleep(5)
    time.sleep(1)
```

I also changed the pir_pin and power_pin because I had it all on the same breadboard, and didn’t make changes to the circuit.

So you can probably see some how this works already, but after the setup, and letting the Pi know if the pins will be input or output, the program waits for the pIR to trigger, it then activates the relay, the light will stay on for 20 seconds, it also displays on the screen that the power is on, after 20 seconds, it displays off, and turns off the relay - it waits for a total of 6 seconds before asking if the pIR has triggered again.
Why does it wait? It helps to prevent immediate retriggering - if your relay is clicking a lot, you may need to increase this wait time.



My final circuit was mounted to a board, which I didn't take a picture of, but other then how it was mounted, this is the same.

Next week we start some basics with the Beaglebone Black.

Aug 14, 2014 - Lesson 12 and 13 (Movement Sensing and Power Control)

The lesson this week will be using a pIR sensor, and a magnetic reed switch to display on the screen On/Off conditions. The 2nd part of the lesson will take what we learned from Lesson 12 and use the pIR sensor to turn on a outlet (or in our case a light bulb).

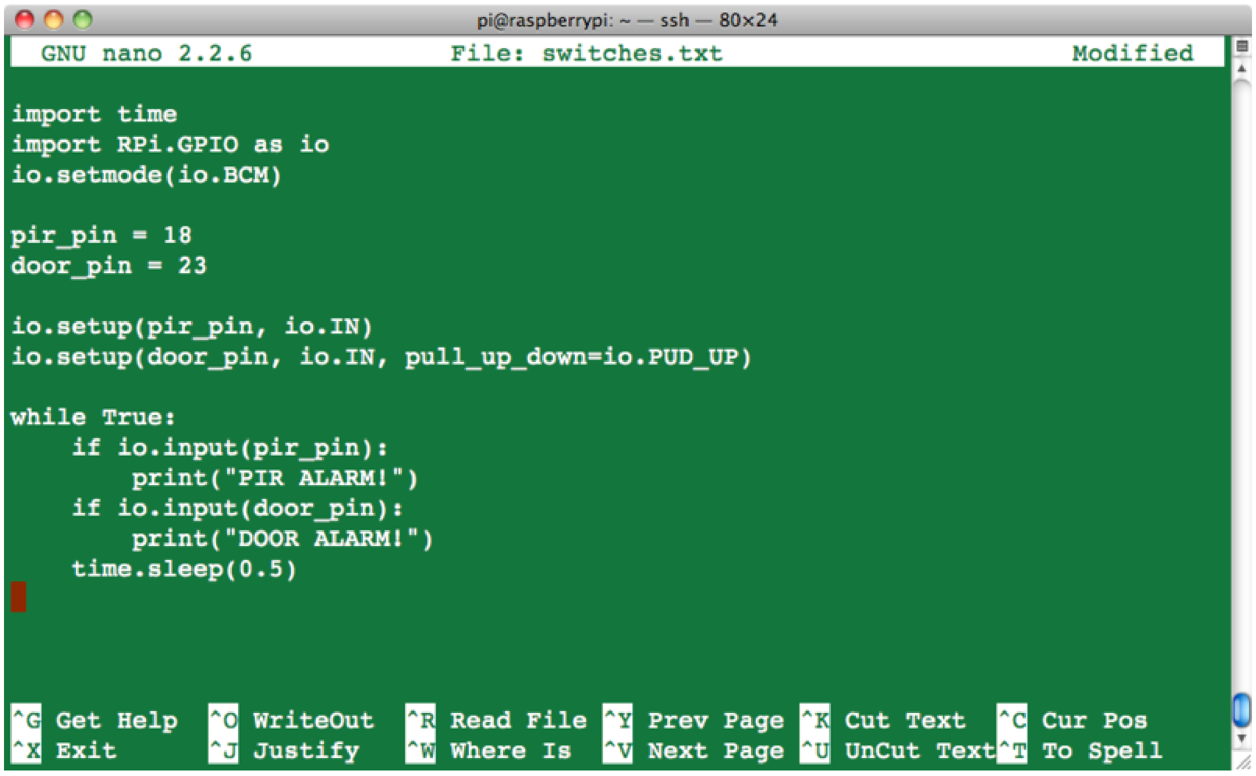
So to start off we need a pIR sensor, Magnetic reed switch, breakout board, breadboard, relay, batteries, a light bulb, and of course the Raspberry Pi.

https://learn.adafruit.com/adafruits-raspberry-pi-lesson-12-sensing-movement

The important thing to take away from Lesson 12 is how to activate the built in Pull-UP resistor for use with the reed-switch. Which is this code:

```
io.setup(door_pin, io.IN, pull_up_down=io.PUD_UP)
```

The pIR's output is 3.3v volt and doesn't need a pull-up because it is digital.



```
  GNU nano 2.2.6               File: switches.txt                     Modified

import time
import RPi.GPIO as io
io.setmode(io.BCM)

pir_pin = 18
door_pin = 23

io.setup(pir_pin, io.IN)
io.setup(door_pin, io.IN, pull_up_down=io.PUD_UP)

while True:
    if io.input(pir_pin):
        print("PIR ALARM!")
    if io.input(door_pin):
        print("DOOR ALARM!")
    time.sleep(0.5)

^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text^T To Spell
```

So what else does the code say to do?
It will display the words: “PIR ALARM!” or “DOOR ALARM” if either one are triggered. pretty simple and straightforward.

remember to run the program you need to execute it with the sudo command:
sudo python switches.py

Once that works, you might asked what good is that going to do? Well, how about turning a light on for a period of time and turning it off again.

Lesson 13 is just that - using the pIR sensor and a relay we can turn on a light, the light will remain on for 20 seconds and turn off and wait for motion again.

https://learn.adafruit.com/adafruits-raspberry-pi-lesson-13-power-control

Really, You could use the reed switch to do the same thing, think about a closet with a bulb in it, you want it on while the door is open, if you forget to close the door the light will shut it’s self off, of course the code would need to be changed to not trigger again because the door was still open. (anyways, that is just a thought)

In this lesson Adafruit uses something called a “Power Switch Tail 2” https://www.adafruit.com/products/268

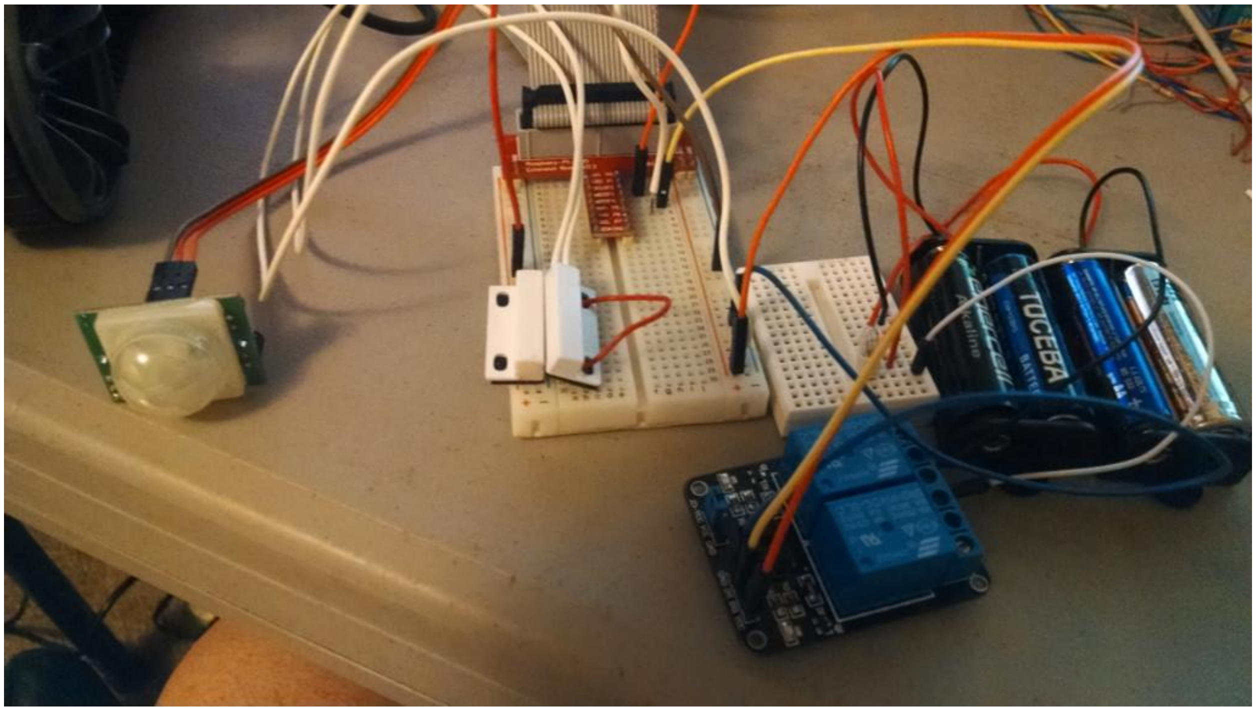
This looks like a 110v extension of a outlet. It is really a controllable opto-ilsolated solid-state relay. On one side you have the high AC voltage (110v) on the other side you have a low 3.3v (5v) that when it goes high will turn on the AC voltage.

For this demo, I’m just going to use a common relay - my realy can handle 110v AC, but for now I’m just going to use 6 volts DC on one side, and 3.3v on the other. My relay is also a mechanical relay. Also it works just the opposite of the the “Tail” in that a low will activate my relay, so the code needs to be changed slightly for me.
This is the code from Adafruit. The code I used is the same making all the True to False, and all the False to True

```
import time
import RPi.GPIO as io
io.setmode(io.BCM)

pir_pin = 24
power_pin = 23

io.setup(pir_pin, io.IN)
io.setup(power_pin, io.OUT)
io.output(power_pin, False)

while True:
   if io.input(pir_pin):
      print("POWER ON")
      io.output(power_pin, True)
      time.sleep(20);
      print("POWER OFF")
      io.output(power_pin, False)
      time.sleep(5)
   time.sleep(1)
```

I also changed the pir_pin and power_pin because I had it all on the same breadboard, and didn’t make changes to the circuit.

So you can probably see some how this works already, but after the setup, and letting the Pi know if the pins will be input or output, the program waits for the pIR to trigger, it then activates the relay, the light will stay on for 20 seconds, it also displays on the screen that the power is on, after 20 seconds, it displays off, and turns off the relay - it waits for a total of 6 seconds before asking if the pIR has triggered again.
Why does it wait? It helps to prevent immediate retriggering - if your relay is clicking a lot, you may need to increase this wait time.



My final circuit was mounted to a board, which I didn't take a picture of, but other then how it was mounted, this is the same.

Next week we start some basics with the Beaglebone Black.