Etherlux: A Novel Internet Connected Power Outlet

Rounok Joardar ©

Introduction

- A novel internet connected power outlet that can be controlled over the worldwide web has been developed

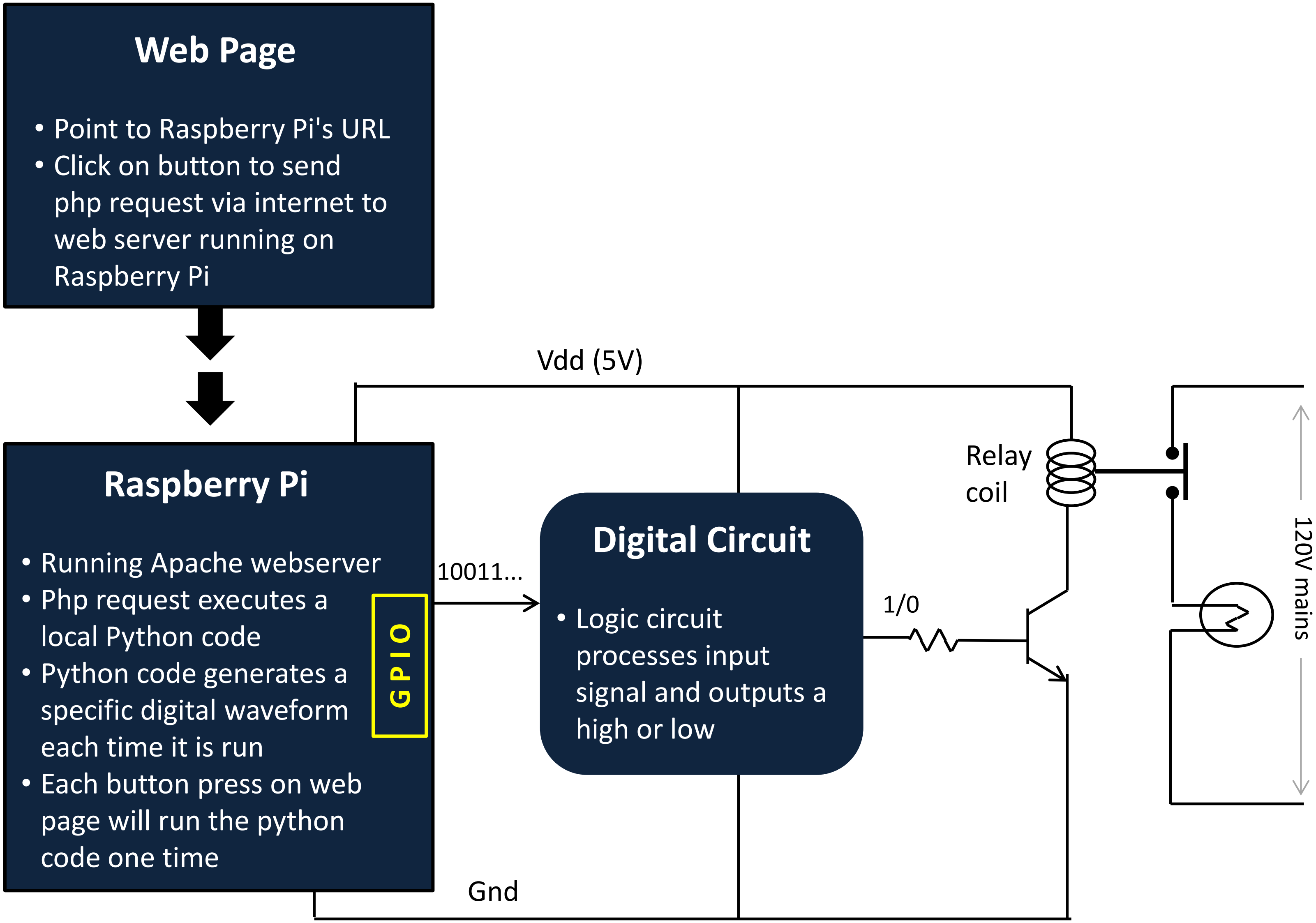
- The system uses a Raspberry Pi and simple digital circuitry

- The following slides describe in detail the internal workings of the system

- The main distinguishing features of this design are:
  - ability to control outlets well beyond simple on/off operations (e.g. timed on/off, duty cycle control, complementary behavior, etc. etc.)
  - low cost compared to similar units available today
  - easy upgrades to increase scale
  - robust peformance

- DIY enthusiasts can view detailed step-by-step construction process by visiting **http://www.rounok.com**

Concept



Web Page

- Point to Raspberry Pi's URL
- Click on button to send php request via internet to web server running on Raspberry Pi

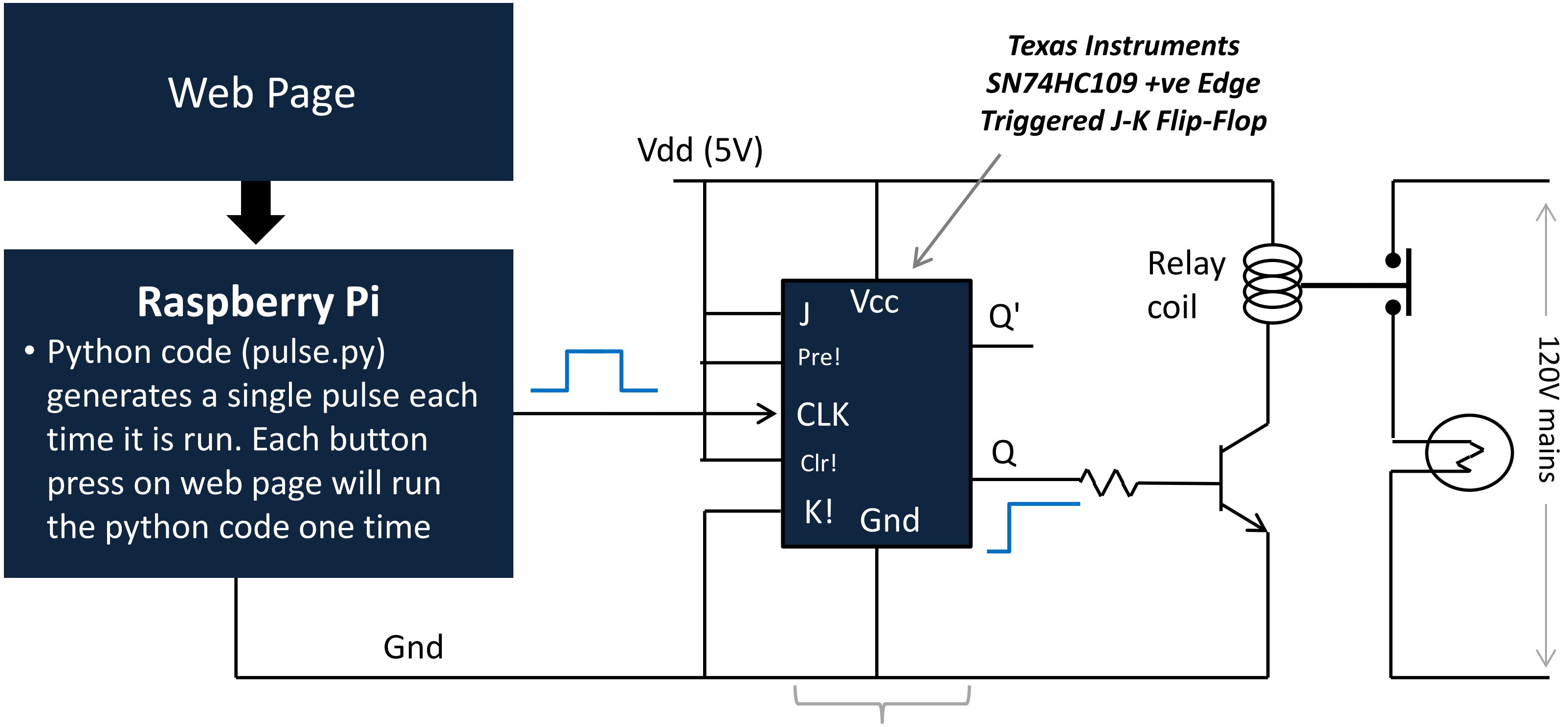
Raspberry Pi

- Running Apache webserver
- Php request executes a local Python code
- Python code generates a specific digital waveform each time it is run
- Each button press on web page will run the python code one time

Digital Circuit

- Logic circuit processes input signal and outputs a high or low

Simple Implementation of Concept



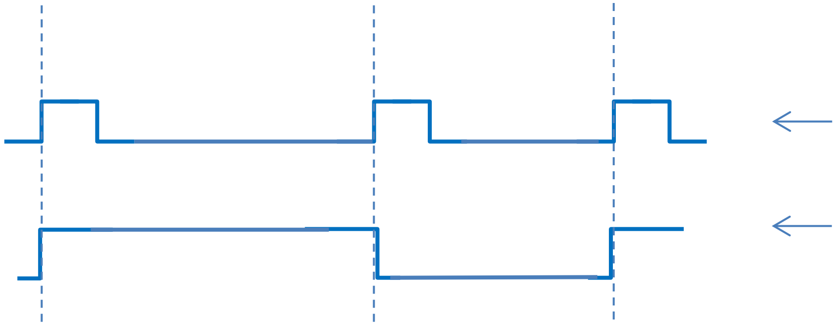
Web Page

Raspberry Pi

- Python code (pulse.py) generates a single pulse each time it is run. Each button press on web page will run the python code one time

Texas Instruments SN74HC109 +ve Edge Triggered J-K Flip-Flop

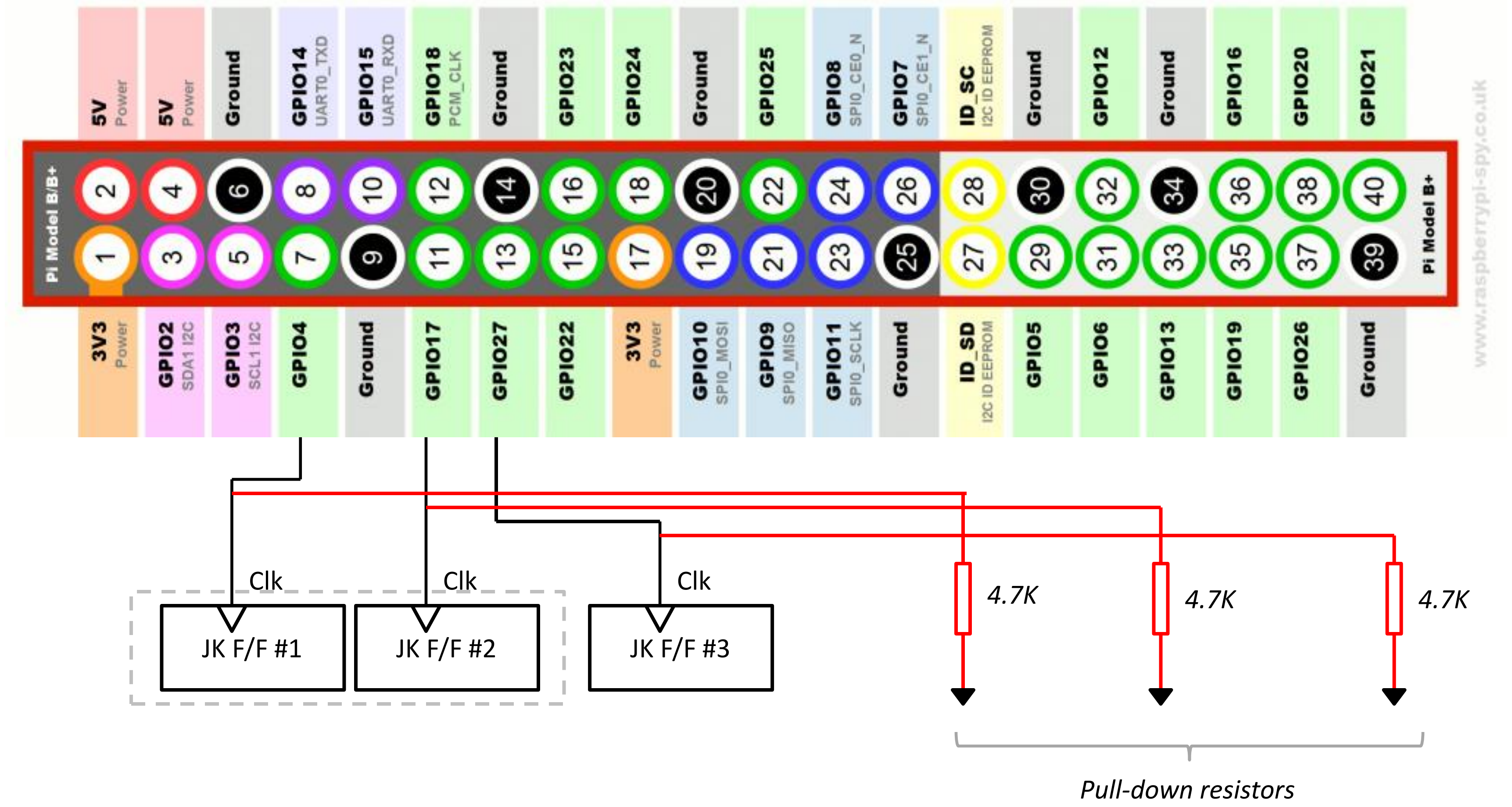
JK flip flop configured to toggle output every time a pulse appears at its clock terminal



Clock signal (from Pi)

Flip flop output at Q (positive edge trig'd)

Implementation Details



- Designed for 3 controlled outlets and one "always on" outlet
- GPIO ports 4, 17, and 27 are used
- Each port is connected to clock pin of a JK flip flop
- Pull-down resistors are used at GPIO ports to stabilize signals

Coding Details

```
<html>
<body>
<form action="runPython.php" method="post">
<input type="checkbox" name="lightNum[]" value="port04">Light01
<br>
<input type="checkbox" name="lightNum[]" value="port17">Light02
<br>
<input type="checkbox" name="lightNum[]" value="port27">Light03
<br>
<br>
<input type=submit value="Continue">
</form>
</body>
</html>
```

- Apache webserver installed and running on Raspberry Pi
- This is the simple html code that is present in the /var/www directory on the Raspberry Pi (file = lights.html)
- Sends request to the webserver to run php file "runPython.php" with the value "port04" or "port17" or "port27" depending on which outlet the user chooses to toggle

Coding Details (contd..)

```
<?php
header ('refresh:10; url=lights.html');

echo 'Light toggled at: ' . date('Y-m-d H:i:s');
echo '<br>';
foreach($_POST['lightNum'] as $key => $value){
    $item = $_POST['lightNum'][$key];
    echo 'Now toggling ' . $item;
    echo '<br>';
    exec('sudo python pulseGen.py ' . $item);
    echo 'Done. <br>';
}
echo '<br>Please wait. Redirecting to main page';
/* exec('sudo python pulseGen.py'); */

?>
```

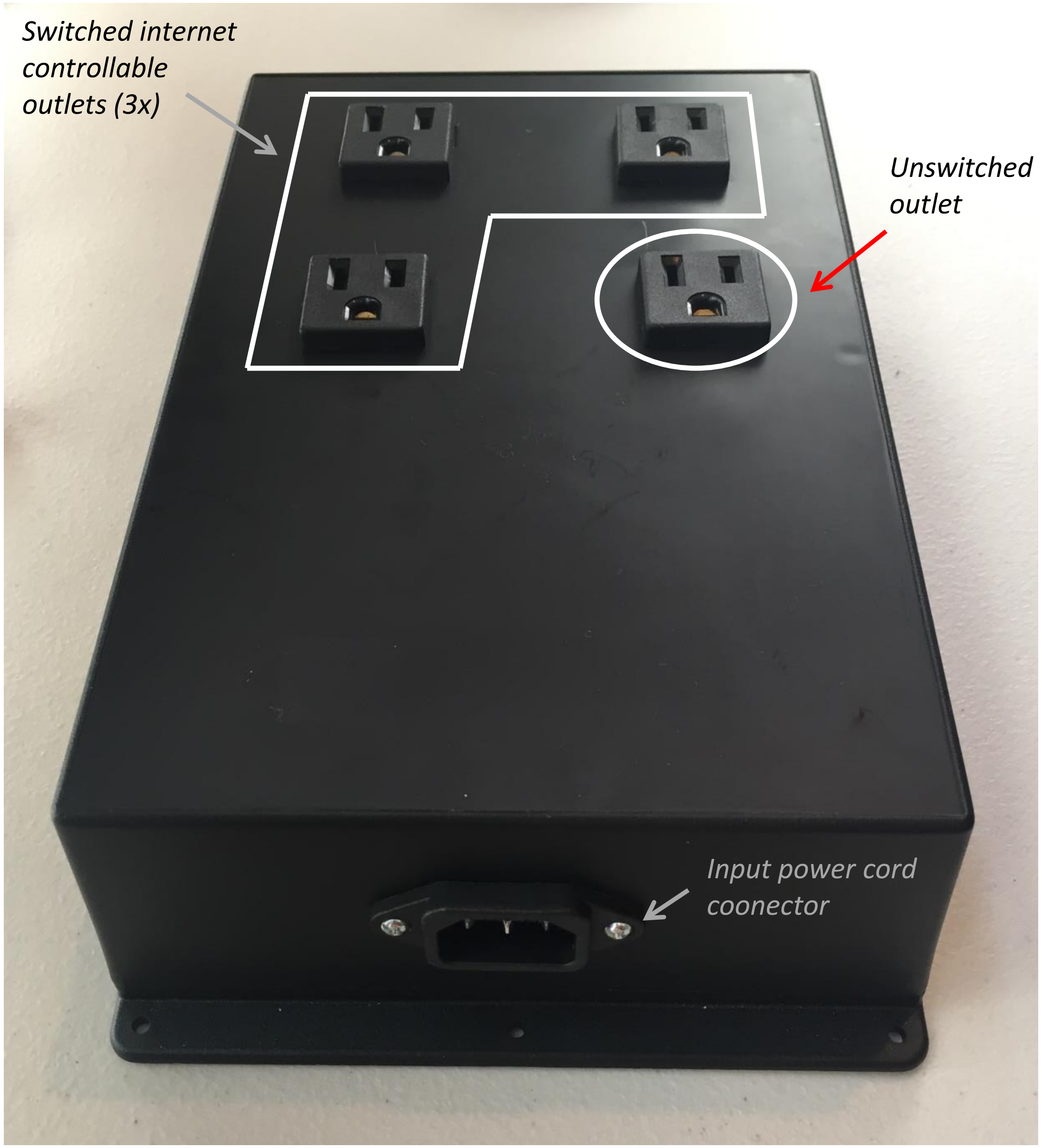
- **This is the php code that is present in the /var/www directory on the Raspberry Pi (file = runPython.php)**
- **Upon receiving request from the internet the webserver runs this file**
- **The php code in turn executes a Python script named "pulseGen.py" with the value "port04" or "port17" or "port27" depending on which outlet the user chose to toggle**

Coding Details (contd..)

```
import RPi.GPIO as GPIO
from time import sleep
import sys

GPIO.setmode( GPIO.BOARD )
GPIO.setup( 4 , GPIO.OUT )
GPIO.setup( 17 , GPIO.OUT )
GPIO.setup( 27 , GPIO.OUT )
GPIO.output( 4, False)
GPIO.output( 17, False)
GPIO.output( 27, False)

port = sys.argv[1]
pulseWidth = 5
portNum = 4
if port == "port04":
    portNum = 17
if port == "port17":
    portNum = 27
if port == "port27":
    portNum = 7
print( 'Active port = ' + str(portNum) )

if (portNum == 4 or portNum == 17 or portNum == 27):
    i = 1
else:
    i = 4
#print( str(i) )
while i < 2:
    print( 'Generating pulse on port ' + str(portNum) )
    GPIO.output( portNum , True )
    sleep ( pulseWidth )
    GPIO.output( portNum, False)
    sleep ( pulseWidth )
    i += 1
    print( 'Done' )
GPIO.cleanup()
```

- **This is the Python code that is present in the /var/www directory on the Raspberry Pi (file = pulseGen.py)**
- **When the php code executes this Python script, the GPIO output at "port04" or "port17" or "port27" is toggled, depending on which outlet the user chose**

Final Assembly



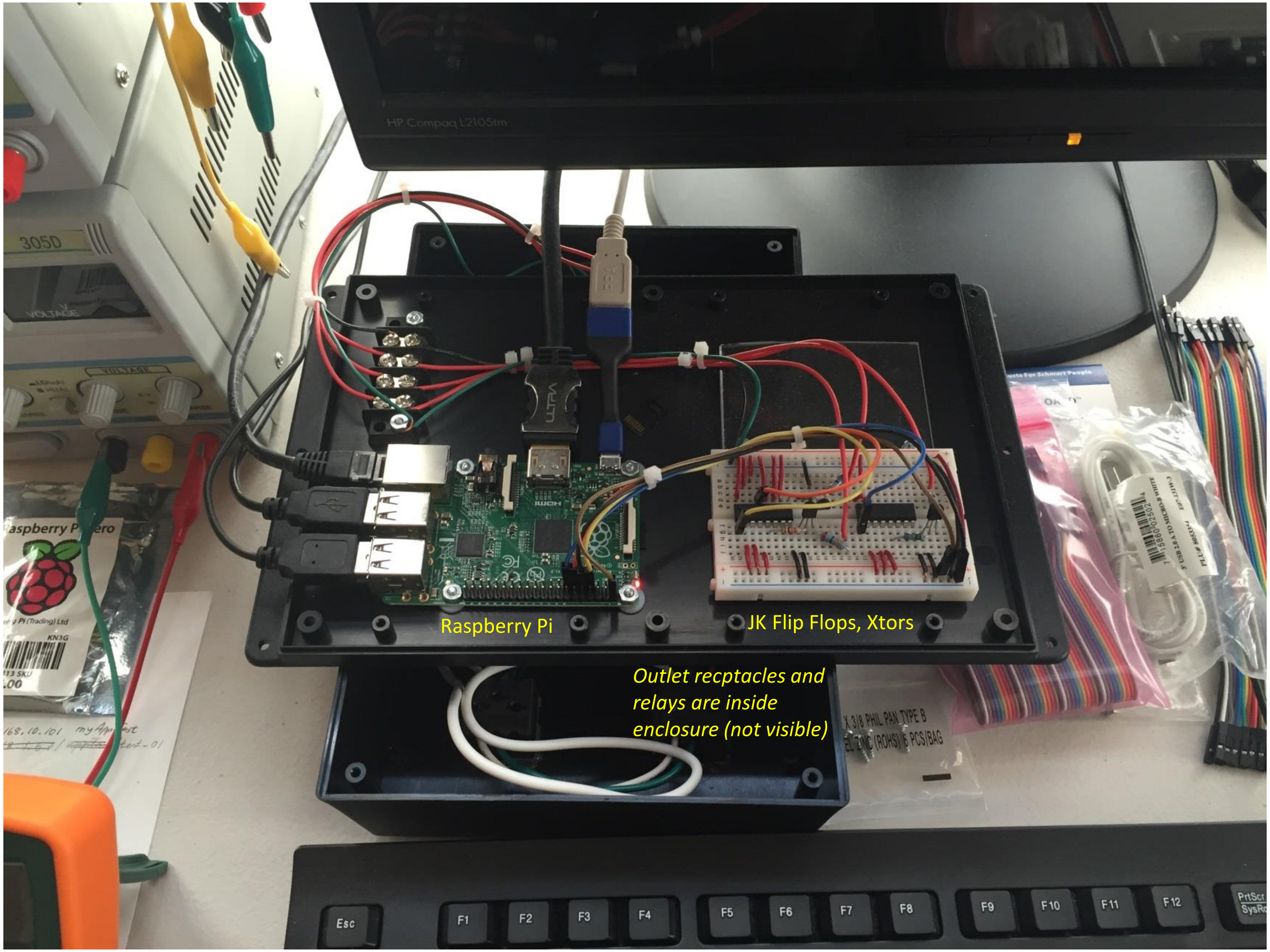
Switched internet controllable outlets (3x)
Unswitched outlet
Input power cord coonector

- This is the final assembled view of the "Etherlux" web controlled power outlet box

- Raspberry Pi with wi-fi adapter and other electroncis and relays are located inside the enclosure

- Internal view is shown in next slide

Final Assembly (contd..)


Raspberry Pi
JK Flip Flops, Xtors
Outlet recptacles and relays are inside enclosure (not visible)

- **Internal view of the "Etherlux" web controlled power outlet box**

Etherlux in Action

Short demo coming up next - don't go away :-)

- **The demo will first show a set of simple on/off functionality moves**
- **Then it will show the ability to strobe the outlets from a web page**
  - ***In practice this functionality can be used to control the speed of a fan or the output of a room heater***