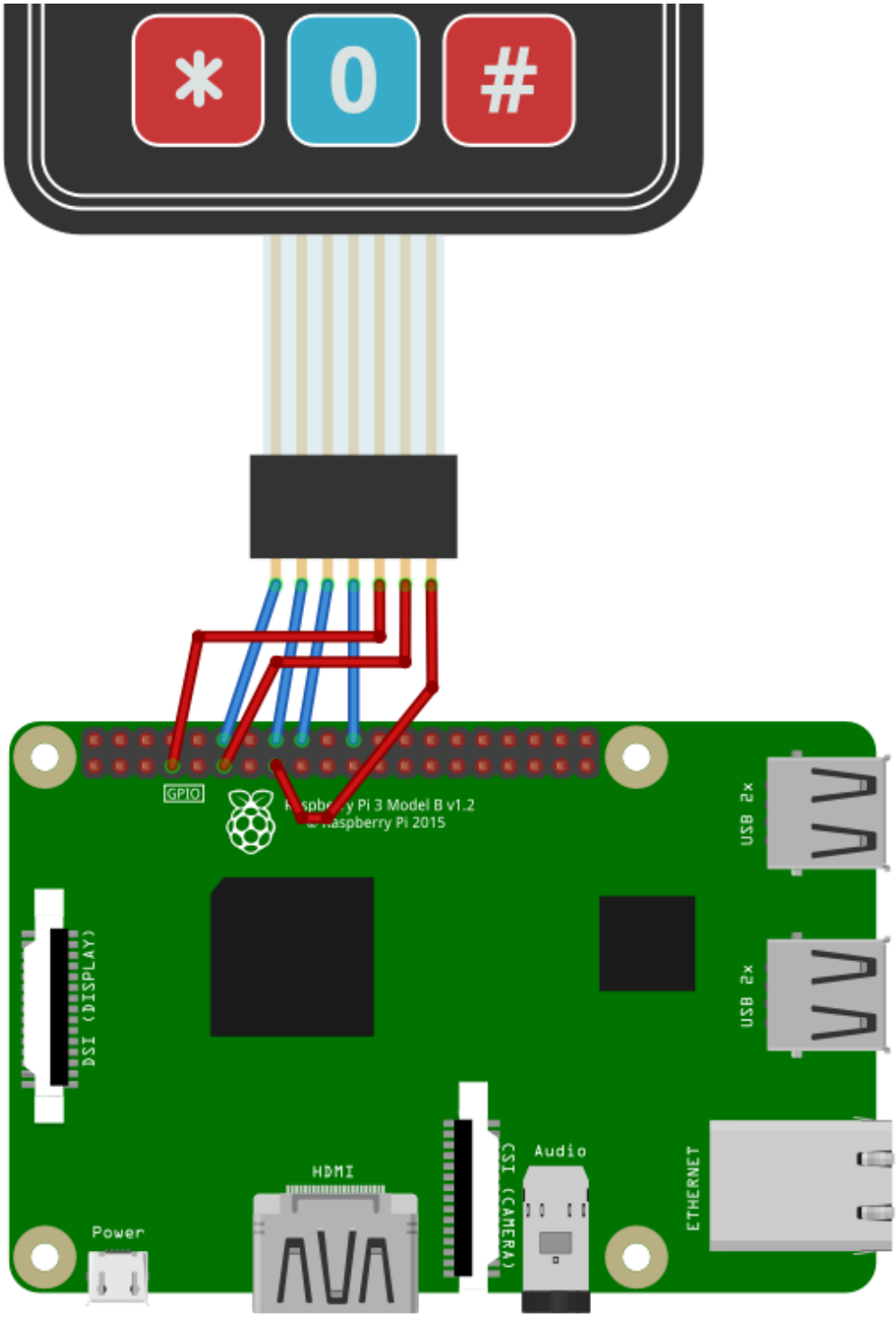
# Keypad and PIR alarm kit for Raspberry Pi

The keypad supplied is a four row, three column type with a adhesive backing that can easily be attached to the outside of a project box. The PIR module can detect movement in a room. The Python scripts can take an alarm photo using your Raspberry Pi camera and email it to your phone.

Here’s how you attach the keypad to your Raspberry Pi.




Left to right the outputs from the keypad are Row 1,2,3,4 (blue) and the Column 1,2,3 (red). There are seven male to female connector cables included with the kit.

The Python code uses the Raspberry Pi's internal pull-up resistors, otherwise you'd need a 10k resistor on each row line to stop the GPIO inputs floating around between 1 and 0.

The software that figures out which button was pressed first scans all the row lines for an input and then switches the GPIO inputs around and scans the column lines instead, to determine exactly which key was pressed. Fortunately there's some existing Python code to do this for us:

http://crumpspot.blogspot.com/2013/05/using-3x4-matrix-keypad-with-raspberry.html

matrixKeypad_RPi_GPIO.py
matrixKeypad_test.py

I cut and pasted both examples into a nano editor session and saved them, but they're also in our demo zip file. You can get that with:

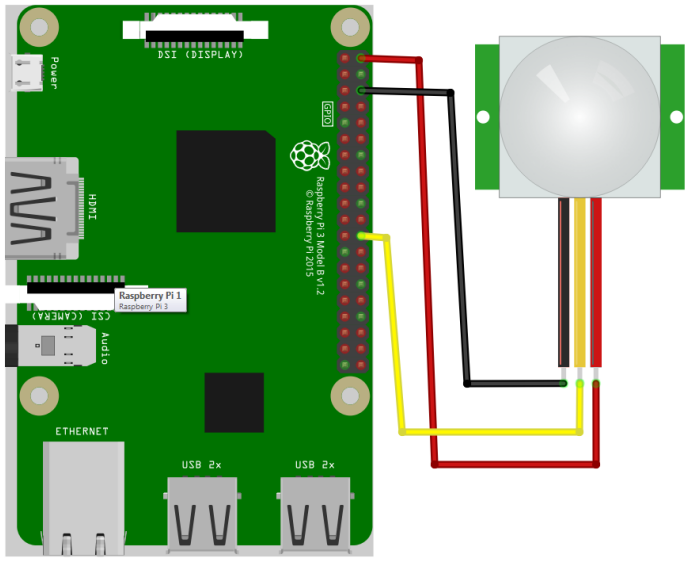
```
wget www.securipi.co.uk/keypad.zip
unzip keypad.zip
```

You can run the demo with

```
python matrixKeypad_test.py
```

and you'll be able to enter a 4 digit pin and see it printed.

We also built our own working alarm system demo, using the included PIR movement sensor connected to pins 5V PWR, GND and GPIO 8. There are three female to female connector cables included. The PIR outputs are printed on the PCB if you lift off the plastic dome – centre pin goes to GPIO 8. (There's a pinout guide at the end of this document).




You can test the PIR is working with:

```
python pirtest.py
```

and once you know the PIR is working correctly (1=movement, 0 = no movemet), run:

```
python alarm2.py
```

This script lets you input a 4 digit pin. If you enter the correct pin - which is 5678 - then the alarm will arm. Once the alarm is armed, any movement detected by the PIR will print a movement detected message to the console. There are other scripts that build on this to email you a photo of the intruder and write the time and date of PIR alerts to a log file.

```
from matrixKeypad_RPi_GPIO import keypad
from time import sleep
import RPi.GPIO as GPIO
import os

digitCount = 0
pin = [0,0,0,0]
finalPin = "0000"
validPin = "5678"
armState = 0
pirPin = 8

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(pirPin, GPIO.IN)

# Initialize the keypad class
kp = keypad()

def digitreturn():
    # Loop while waiting for a keypress
    r = None
    while r == None:
        r = kp.getKey()
        pirtest()
    return r

def message():
    os.system('clear')
    print("Please enter a 4 digit pin using the keypad")

def pirtest():
    if GPIO.input(pirPin) and armState == 1:
        print("Movement Detected. Intruders!")
        sleep(3)
message()

while True:
    digit = digitreturn()
    print digit
```

```
    pin[digitCount] = digit
    digitCount = digitCount + 1
    sleep(0.25)
    if digitCount == 4:
        finalPin = (str(pin[0]) + str(pin[1]) + str(pin[2]) +
str(pin[3]))
        print(finalPin)
        if validPin == finalPin:
            print("You entered the correct PIN")
            if armState == 0:
                print("alarm will arm in 5 seconds, get out now")
                sleep(5)
                print("armed")
                armState = 1
                sleep(1)
                message()
            elif armState == 1:
                print("alarm is now disarmed")
                armState = 0
        digitCount = 0
```

The next example (alarm3.py) takes a photo of the intruder using the Raspberry Pi camera and stores it to a time and date stamped file.

```
python alarm3.py
```

```
import picamera
from matrixKeypad_RPi_GPIO import keypad
from time import sleep
import RPi.GPIO as GPIO
import os
import datetime

digitCount = 0
pin = [0,0,0,0]
finalPin = "0000"
validPin = "5678"
armState = 0
pirPin = 8
camera = picamera.PiCamera()

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(pirPin, GPIO.IN)

# Initialize the keypad class
kp = keypad()

def digitreturn():
    # Loop while waiting for a keypress
```

```
    r = None
    while r == None:
        r = kp.getKey()
        pirtest()
    return r

def message():
    os.system('clear')
    print("Please enter a 4 digit pin using the keypad")

def pirtest():
    if GPIO.input(pirPin) and armState == 1:
        now = datetime.datetime.now().strftime("%y-%m-%d--%H-%M-
%S")
        print("Movement Detected " + now + ".  Taking a photo")
        camera.start_preview()
        sleep(3)
        camera.annotate_background = picamera.Color('black')
        camera.annotate_text =
datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')
        camera.capture('/home/pi/' + now + 'image.jpg')
        camera.stop_preview()


message()

while True:
    digit = digitreturn()
    print digit
    pin[digitCount] = digit
    digitCount = digitCount + 1
    sleep(0.25)
    if digitCount == 4:
        finalPin = (str(pin[0]) + str(pin[1]) + str(pin[2]) +
str(pin[3]))
        print(finalPin)
        if validPin == finalPin:
            print("You entered the correct PIN")
            if armState == 0:
                print("alarm will arm in 5 seconds, get out now")
                sleep(5)
                print("armed")
                armState = 1
                sleep(1)
                message()
            elif armState == 1:
                print("alarm is now disarmed")
                armState = 0
        digitCount = 0
```

The final example (alarm4.py) sends the photo of the intruder as an email attachment to your phone or PC. You will need to setup a new gmail address just for your Pi to use, and you'll also need to enable "less secure apps" access to it. Details below:

Setting up a spare Gmail account, just for your Pi to use.
The simplest way of sending emails & photos as attachments from your Pi, is to setup a new Gmail account for the Pi to use, even if you already have an existing Gmail account you use on your phone or PC. For one thing, it gets you 15GB of new cloud storage for your alert photos & secondly it removes the complication of generating application specific passwords for other apps on your existing Gmail account.

You need to create the new Gmail account in the web browser on your PC or Mac @
https://accounts.google.com/SignUp?service=mail
and note down the login & password for later. Don't use the # symbol in your password as it causes the Pi problems. Lower & upper-case letters & number combinations are always fine though.

Next, you need to set the new Gmail account to Enable "less secure apps". While logged in go to:
https://www.google.com/settings/u/0/security/lesssecureapps

Now we have a working email account, just for the Pi to use when sending emails. Any photos sent from the Pi will be backed-up in the Sent folder & you only need delete old photos if you get near to the 15GB limit. Emails from the Pi can be sent to any other email address on your phone or PC

Edit alarm4.py so that it includes your own new gmail email login and password for sending, and also the correct recipient destination email address.

```
nano alarm4.py
```

Once it's edited, run it with:

```
python alarm4.py
```

```
import picamera
from matrixKeypad_RPi_GPIO import keypad
from time import sleep
import RPi.GPIO as GPIO
import os
import datetime

#email setup below
from email.MIMEMultipart import MIMEMultipart
from email.MIMEText import MIMEText
from email.MIMEImage import MIMEImage
import smtplib


digitCount = 0
pin = [0,0,0,0]
finalPin = "0000"
validPin = "5678"
armState = 0
```

```
pirPin = 8
camera = picamera.PiCamera()

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(pirPin, GPIO.IN)

# Initialize the keypad class
kp = keypad()

def digitreturn():
    # Loop while waiting for a keypress
    r = None
    while r == None:
        r = kp.getKey()
        pirtest()
    return r

def message():
    os.system('clear')
    print("Please enter a 4 digit pin using the keypad")

def pirtest():
    if GPIO.input(pirPin) and armState == 1:
        now = datetime.datetime.now().strftime("%y-%m-%d--%H-%M-
%S")
        print("Movement Detected " + now + ".  Taking a photo")
        camera.start_preview()
        sleep(3)
        camera.annotate_background = picamera.Color('black')
        camera.annotate_text =
datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')
        camera.capture('/home/pi/' + now + 'image.jpg')
        camera.stop_preview()
        # Send an email section. Make a new gmail account just for
your Pi to use.  Then enable "less secure apps"
        #
https://www.google.com/settings/u/0/security/lesssecureapps
        msg = MIMEMultipart()
        msg.attach(MIMEImage(file('/home/pi/' + now +
'image.jpg').read()))
        mailer = smtplib.SMTP('smtp.gmail.com:587')
        mailer.starttls()
        # Login and password of your new gmail address go here
        mailer.login("myraspberrypi","password")
        # Recipient is the regular email account on your phone or
PC.
        mailer.sendmail("myraspberrypi@gmail.com",
["recipient@gmail.com"], msg.as_string())
        mailer.close()
```

```
message()

while True:
    digit = digitreturn()
    print digit
    pin[digitCount] = digit
    digitCount = digitCount + 1
    sleep(0.25)
    if digitCount == 4:
        finalPin = (str(pin[0]) + str(pin[1]) + str(pin[2]) +
str(pin[3]))
        print(finalPin)
        if validPin == finalPin:
            print("You entered the correct PIN")
            if armState == 0:
                print("alarm will arm in 5 seconds, get out now")
                sleep(5)
                print("armed")
                armState = 1
                sleep(1)
                message()
            elif armState == 1:
                print("alarm is now disarmed")
                armState = 0
        digitCount = 0
```

Here are the Raspberry Pi pinouts, for you reference: