How to attach your CodeBug to your Raspberry Pi

There's a guide to setting your CodeBug up in Raspberry Pi tether mode here:
http://www.codebug.org.uk/learn/activity/62/raspberry-pi-controlled-codebug-with-i2c/

Upload the tether software to your CodeBug from here:
https://github.com/codebugtools/codebug_i2c_tether/raw/master/firmware/codebug_i2c_tether.cbg

With the Pi turned off, attach the CodeBug to the GPIO pins on the Pi, either directly or using male to female link cables. (Make sure you don't have a USB cable or battery in your CodeBug)

Type in these commands on your Raspberry Pi:

```
sudo raspi-config
```

Choose advanced mode → enable I2C → yes to both options → reboot your Pi

```
sudo apt-get update
```

```
sudo apt-get install python3-codebug-i2c-tether
```

```
wget https://raw.githubusercontent.com/codebugtools/codebug_i2c_tether/master/example.py
```

```
python3 example.py
```

If it worked correctly you should see an arrow pointing up and left on your Codebug.

Using the CodeBug with our wireless doorbell kit for Raspberry Pi.

We modified CodeBug's example Python code to interact with our Wireless Doorbell Kit for the Raspberry Pi. The CodeBug displays the message “Ding Dong” when the doorbell is pushed. You can also hold down button A on the CodeBug to see the last time and date the bell was pushed.

Even if you're not using our wireless doorbell kit, you'll still find the examples useful if you want to understand how to make a CodeBug interact with a shell script on the Raspberry Pi.

Download the files from our server with:

```
wget www.securipi.co.uk/cbdb.zip
```

```
unzip cbdb.zip
```

```
chmod a+x test2.sh
```

If you open the two Python files (dingdong.py and doorbella.py) with the Nano editor you'll be able to see how the two Python programs and the shell script operate together. Run the shell script with

```
sudo ./test2.sh
```

Test2.sh shell script

```
#!/bin/sh
# This example assumes you have bought and installed our Wireless Doorbell kit
# http://www.amazon.co.uk/dp/B00UL9QBJ4 or http://www.amazon.co.uk/dp/B00UL3FRGS

# This sets the button detection script doorbella.py running in the background
python3 doorbella.py &

# Main loop
while true;  do
      valid="6454856"
      scan=`./RFSniffer`
            if [ "$scan" = "$valid" ]; then
                  d=`date +%d%m%y`
                  t=`date +%T`
                  echo "Doorbell pushed"
                  echo "Your code is " $scan
                  echo  "$t $d" >> doorbell.log
                  python3 dingdong.py

            else
                  echo "BAD READ:  your code and the valid don't match"
                  echo "Your correct valid code should be " $scan
                  fi
      sleep 5
done
exit
```

dingdong.py

```
import time
import codebug_i2c_tether
cb = codebug_i2c_tether.CodeBug()
cb.open()

for i in range(0,-45,-1):

    cb.write_text(i, 0, 'Ding Dong  ', direction="right")

    time.sleep(.1)
```

doorbella.py

```
import codebug_i2c_tether
import datetime
import time
import subprocess

cb = codebug_i2c_tether.CodeBug()
cb.open()


def scroll_message(message):
    length = len(message)
    for i in range(0,length*-5,-1):
        cb.write_text(i, 0, message, direction="right")
        time.sleep(.15)

while True:
    if cb.get_input('A'):
        last1_str = subprocess.getoutput("tail -n1 doorbell.log")
        scroll_message(last1_str)
    if cb.get_input('B'):
        last3_str = subprocess.getoutput("tail -n3 doorbell.log")
        scroll_message(last3_str)
```

If you run the test2.sh script and press button A on the CodeBug, you'll see the last time and date the doorbell was pressed scroll across the display. If you press button B you'll see the last three entries.