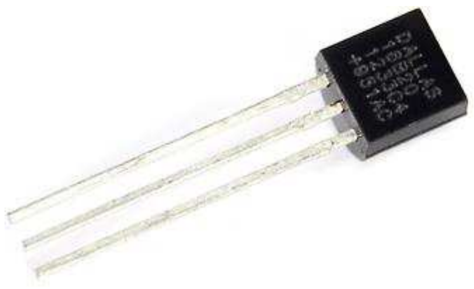
REUK.co.uk - The Renewable Energy Website

DS18B20 Temperature Sensor with Raspberry Pi

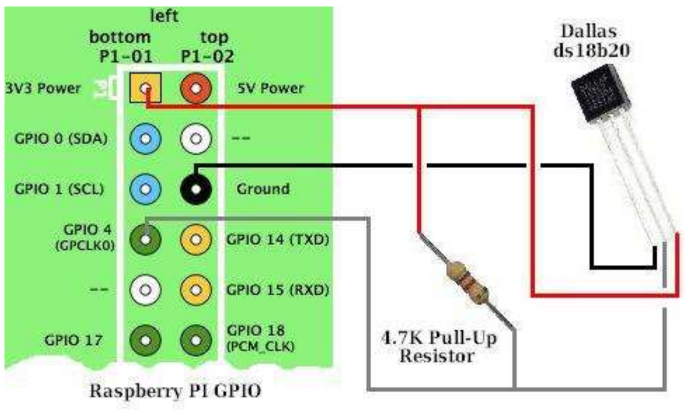
DS18B20 Temperature Sensor With Raspberry Pi

Measure temperatures with a 1-wire DS18B20 sensor and Raspberry Pi



The **Raspberry Pi** does not have any ADC (analogue to digital converter) pins in its GPIO, and so a **digital temperature sensor** should be used. The sensor of choice around the world is the **Dallas DS18B20 1-Wire temperature sensor** pictured above. They are relatively cheap, very easy to find, easy to use, and supply readings accurate to +/-0.5 degrees across the range -10 to +85 degrees Celcius.

In order to interface this with the **Raspberry Pi** we just need one resistor - a 4k7 which acts as a pull-up resistor - and connect everything up as shown below:



Raspberry Pi and DS18B20 Temperature Sensor - Python

Log on to your Raspberry Pi, and at the command prompt enter the following:

```
sudo modprobe w1-gpio && sudo modprobe w1_therm
```

to load in the 1-wire communication device kernal modules required to use the temperature sensor.

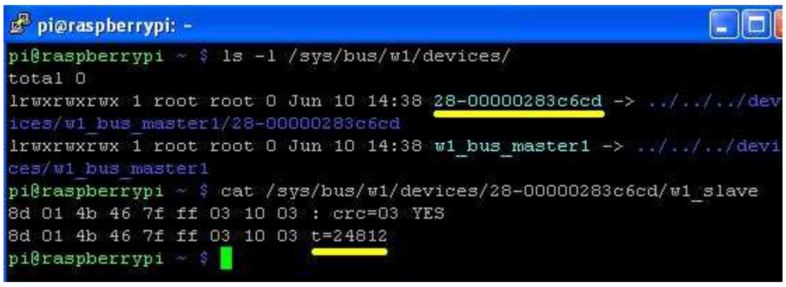
(to avoid having to manually enter this command every time you re-boot your Pi, edit using nano or similar the file "/etc/modules" adding the lines w1-gpio, and w1_therm to the end of the file.)

Now at the command prompt enter:

```
ls -l /sys/bus/w1/devices/
```

to see a list of the devices currently connected to your Raspberry Pi.

Your temperature sensor will appear with an address in the format **28-00000xxxxxx**. Below you can see that our temperature sensor address was 28-00000283c6cd. Each DS18B20 temperature sensor has a unique hard-coded address so that you can connect up multiple temperature sensors when required and still read them individually.



```
pi@raspberrypi ~ $ ls -l /sys/bus/w1/devices/
total 0
lrwxrwxrwx 1 root root 0 Jun 10 14:38 28-00000283c6cd -> ../../../dev
ices/w1_bus_master1/28-00000283c6cd
lrwxrwxrwx 1 root root 0 Jun 10 14:38 w1_bus_master1 -> ../../../devi
ces/w1_bus_master1
pi@raspberrypi ~ $ cat /sys/bus/w1/devices/28-00000283c6cd/w1_slave
8d 01 4b 46 7f ff 03 10 03 : crc=03 YES
8d 01 4b 46 7f ff 03 10 03 t=24812
pi@raspberrypi ~ $
```

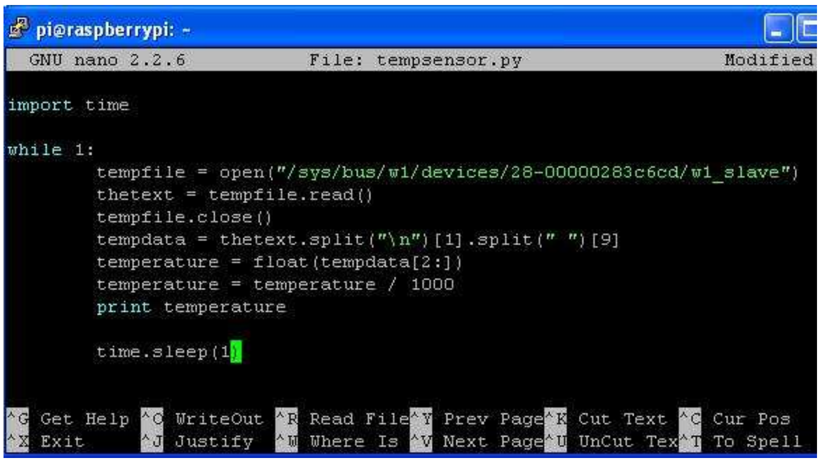
Having found your temperature sensor, type in the following at the command prompt to open the sensor file and view the temperature reading:

```
cat /sys/bus/w1/devices/28-00000283c6cd/w1_slave
```

(making sure to substitute in the address for your own temperature sensor of course.)

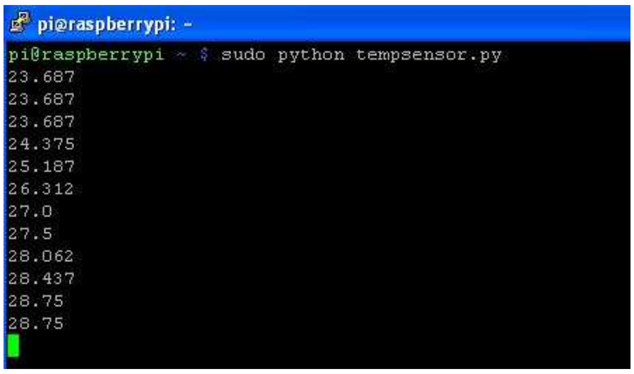
Above we can see that the output from this command is two lines of data. At the end of the second line after "t=" is a numerical value. This value is 1000 times the temperature of the sensor in degrees Celcius. Therefore in our example where we see t=24812, the temperature readings was 24812/1000=24.812 degrees Celcius.

Reading temperatures from the command line like this is not very useful. In general we will want the temperature to be measured at regular intervals and either displayed somewhere or logged in a file for later analysis. In order to achieve this we need to write a script. Here is our **Python** script to measure and display the temperature once every second:



```
  GNU nano 2.2.6            File: tempsensor.py                  Modified

import time

while 1:
        tempfile = open("/sys/bus/w1/devices/28-00000283c6cd/w1_slave")
        thetext = tempfile.read()
        tempfile.close()
        tempdata = thetext.split("\n")[1].split(" ")[9]
        temperature = float(tempdata[2:])
        temperature = temperature / 1000
        print temperature

        time.sleep(1)

^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 Tex^T To Spell
```

Without going into great detail (click here if you want to learn Python), once per second this script reads in the temperature sensor file, gets the second line, splits it into its space-delimited elements, gets the tenth of these elements (denominated '9' since '0' is the first element), dumps the first two characters ("t=") of this element, and turns what remains (24812 in our example) from a string into a numerical value which is then divided by 1000 to give us the temperature which is then printed to the screen.



```
pi@raspberrypi: ~
pi@raspberrypi ~ $ sudo python tempsensor.py
23.687
23.687
23.687
24.375
25.187
26.312
27.0
27.5
28.062
28.437
28.75
28.75
```

The above screenshot shows the script running while the **temperature sensor** is being held causing it to warm it up.

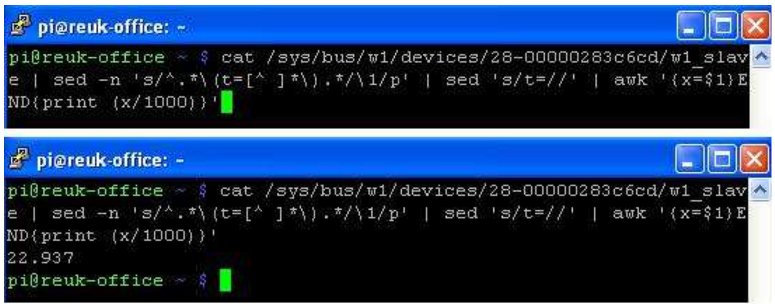
If you would like to connect more than one temperature sensor to your Raspberry Pi, then click here to read our article Connect Multiple Temperature Sensors with Raspberry Pi.

Quick Display Temperature on Command Line - Bash

Instead of writing a Python script to get the temperature reading from the sensor, it is possible to write a one line command which will read the sensor, extract the temperature from the rest of the returned data, and process the result using Bash on the Raspberry Pi command line.

There are many ways to achieve this, but here is one example using sed and awk:

```
cat /sys/bus/w1/devices/28-00000283c6cd/w1_slave | sed -n 's/^.*\(t=[^
     ]*\).*/\1/p' | sed 's/t=//' | awk '{x=$1}END{print(x/1000)}'
```



```
pi@reuk-office: ~
pi@reuk-office ~ $ cat /sys/bus/w1/devices/28-00000283c6cd/w1_slav
e | sed -n 's/^.*\(t=[^ ]*\).*/\1/p' | sed 's/t=//' | awk '{x=$1}E
ND{print (x/1000)}'
```

```
pi@reuk-office: ~
pi@reuk-office ~ $ cat /sys/bus/w1/devices/28-00000283c6cd/w1_slav
e | sed -n 's/^.*\(t=[^ ]*\).*/\1/p' | sed 's/t=//' | awk '{x=$1}E
ND{print (x/1000)}'
22.937
pi@reuk-office ~ $
```

The pipe character "|" is used to pass the output from one command into the next command. The script above first reads in the temperature from the sensor as before returning the two lines of data, the last part of the second line of which is t=XXXXX where XXXXX is the temperature in millidegrees Celcius. We then use sed to find this 'word' beginning with 't=' in the data and then pipe it to a second sed command which strips off the 't='. Finally the numerical result is piped through to an Awk command which divides it by 1000 and displays the temperature in degrees Celcius.

Obviously you would not want to be typing that long line in every time you wanted to check the temperature of your sensor, so you could create an **alias**:

```
alias checktemp="cat /sys/bus/w1/devi......x/1000)}"
```

so that whenever you type in checktemp at the prompt, the commands will be run.

Alternatively you can save the commands as a **shell script**. To do this open the text editor with the command:

```
sudo nano checktemp
```

which will create an empty file called checktemp. Type in:

```
#!/bin/bash
```

on the first line, and then enter the cat /sys/bus... etc commands on the second line. Press Ctrl-X to exit and save the file. To make this file executable (so that we can run it) enter:

```
chmod +x checktemp
```

and finally move it to the /usr/bin directory where most executable files are stored:

```
mv checktemp /usr/bin/
```

Now when you type in checktemp at the command prompt, your shell script will be run and the temperature reading from the sensor will be displayed.

Buy DS18B20 Temperature Sensors

DS18B20 temperature sensors are available as bare components shown at the top of this article, but now, as shown above, it is possible to purchase warterproof DS18B20 sensors - DS18B20 sensors fitted into stainless steel tubine filled with epoxy and with a decent length of cable attached. The set of five shown above for example were available for just **£6.53** including air mail delivery from Hong Kong. Click here for links to purchase Waterproof DS18B20 Temperature Sensors, or here for the bare component DS18B20 sensors.

Temperature Logger with DS18B20 and Raspberry Pi

Click here to find out how to turn your Raspberry Pi and DS18B20 into a Raspberry Pi Temperature Data Logger so you can view your temperature data from anywhere in the world over the internet.

Please take the time to **rate this article** (5 = excellent).
0, 1, 2, 3, 4, 5

Article Last Modified: 22:21, 24th Sep 2014

Comment on this Article

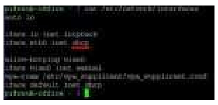
If you have any comments on this article, please email them to neil@reuk.co.uk.

Related Articles

More from the **REUK.co.uk website**:

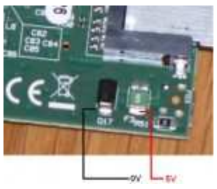
Solar Powered Raspberry Pi
Power your Raspberry Pi permanently from a PV solar panel

Setting up Static IP Address on Raspberry Pi
Find out how to give your Raspberry Pi a static IP address



Raspberry Pi Temperature Logger with Xively
Make a Raspberry Pi temperature data logger accessible online with Xively

Wire 5V Direct to Raspberry Pi
Find out how to connect 5V directly to Raspberry Pi instead of using micro USB



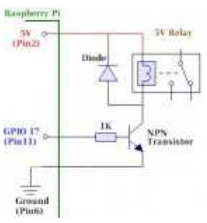
Connecting to Raspberry Pi from PC via SSH
Find out how to operate a Raspberry Pi directly from a Windows PC using SSH

Downloading Youtube Videos to your Raspberry Pi
Find out how to download Youtube videos to Raspberry Pi to Watch on TV

How to Rename Your Raspberry Pi
Find out how to set your choice of name for your Raspberry Pi

```
login as: pi
pi@192.168.1.75's password:
Linux home-tv 3.6.11+ #474 PREEMPT Thu
The programs included with the Debian
the exact distribution terms for each
individual files in /usr/share/doc/*/c
Debian GNU/Linux comes with ABSOLUTELY
permitted by applicable law.
Last login: Wed Oct  2 15:36:38 2013 f
pi@home-tv ~ $
```

Simple Raspberry Pi Relay Control over the Internet
Take your first steps in Raspberry Pi home automation - controlling a relay



Click one of the links below for further articles on the following **relevant topics**:
raspberry pi, electronics, education.

Site Search