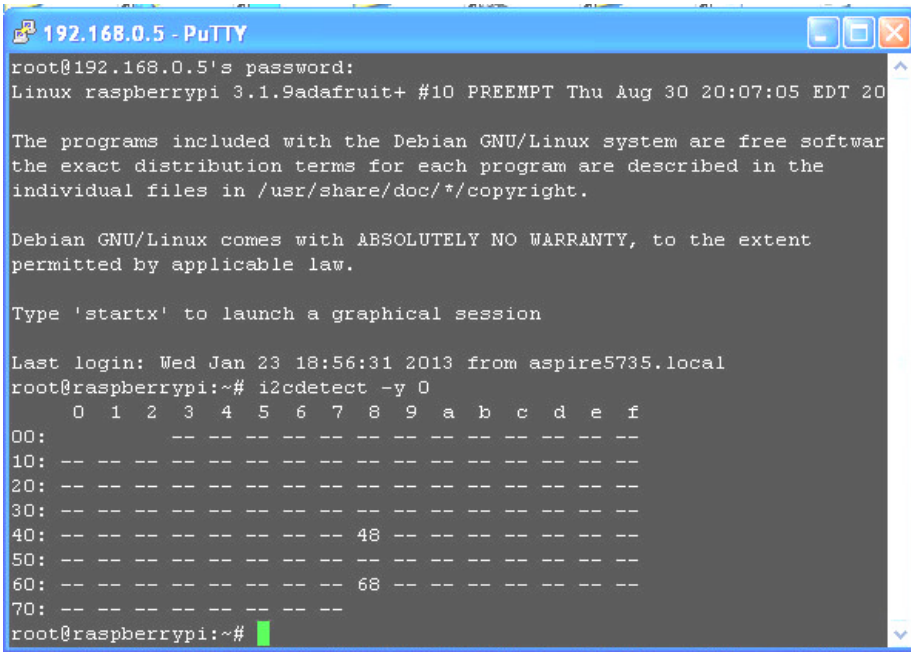


Rs-Pi RTC & Temp module Board User Manual

RTC DS1307 with CR1220 Battery

TMP102 Temperature Sensor



```
root@192.168.0.5's password:
Linux raspberrypi 3.1.9adafruit+ #10 PREEMPT Thu Aug 30 20:07:05 EDT 20
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Type 'startx' to launch a graphical session
Last login: Wed Jan 23 18:56:31 2013 from aspire5735.local
root@raspberrypi:~# i2cdetect -y 0
 0 1 2 3 4 5 6 7 8 9 a b c d e f
00: -- -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- 48 -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- 68 -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- -- -- -- -- -- -- --
root@raspberrypi:~#
```

i2c bus device detect status and active TMP102 & RTC DS1307

48 -> tmp102 68 -> RTC DS1307



First Install battery for RTC , "+" mark on top
RTC DS1307 - 68 in i2cdetect -y 0 or
i2cdetect -y 1 for Rs-Pi V2 you will see 68 in the
screen 68 -> RTC DS1307
48 -> tmp102

This requires a Raspberry Pi running a kernel with the RTC module and DS1307 module included. This is not true of the "Wheezy" distros or Occidentalis v0.1. This is for use with Occidentalis v0.2 or greater

then, load up the RTC module by running
sudo modprobe rtc-ds1307

Then, as root (type in **sudo bash**) run

echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-0/new_device (if you have a rev 1 Pi)

echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device (if you have a rev 2 Pi)

hwclock -r read time

hwclock -w write time in RTC

hwclock -s write time in System

hwclock --set --date="2013-08-21 08:00:12" --utc

write in custom Time in RTC

*TMP102 information

modprobe tmp102

echo tmp102 0x48 > /sys/class/i2c-adapter/i2c-0/new_device

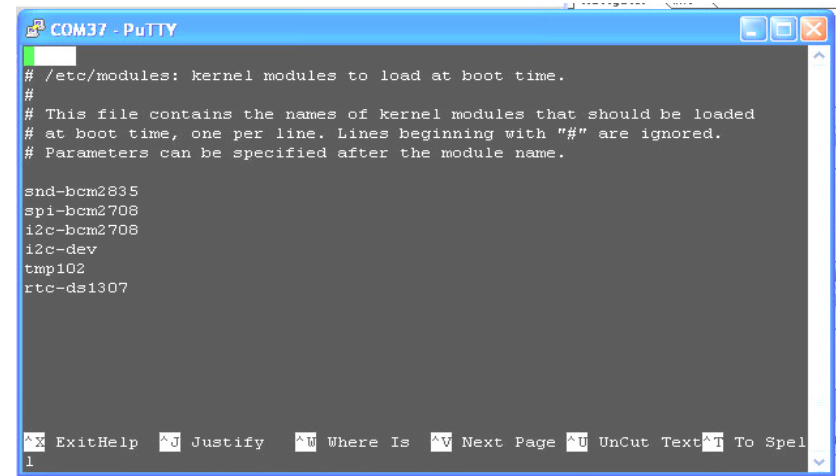
(if you have a rev 1 Pi)

echo tmp102 0x48 > /sys/class/i2c-adapter/i2c-1/new_device

(if you have a rev 2 Pi)

sensors show the temp

you'll want to add the RTC kernel module & temp tmp102 to the /etc/modules list, so its loaded when the machine boots. Run **sudo nano /etc/modules** and add **rtc-ds1307 & tmp102** at the end of the file



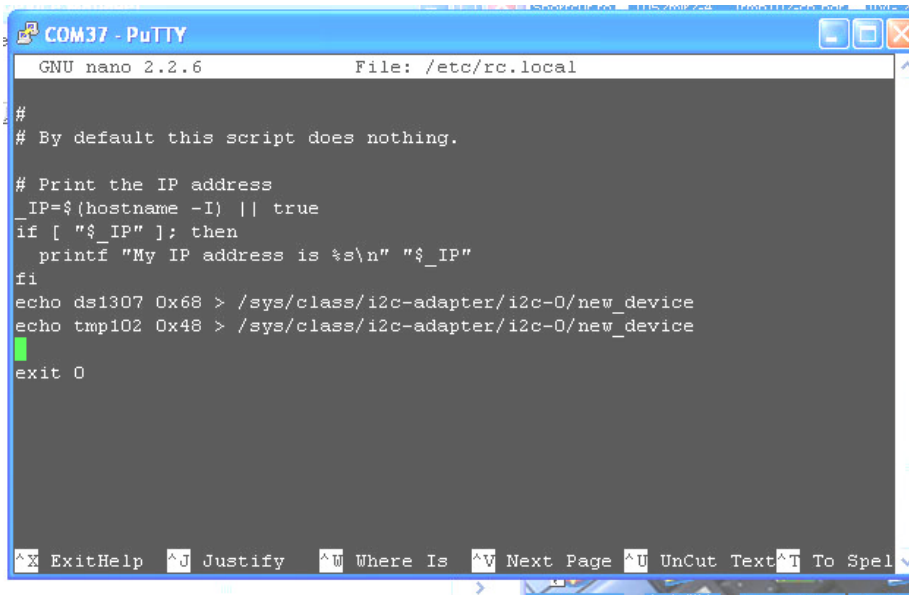
```
COM37 - PuTTY
# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.
# Parameters can be specified after the module name.

snd-bcm2835
spi-bcm2708
i2c-bcm2708
i2c-dev
tmp102
rtc-ds1307
```

Then you'll want to create the DS1307 device creation at boot, edit /etc/rc.local by running

sudo nano /etc/rc.local

and add **echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-0/new_device** before **exit 0**



```
COM37 - PuTTY
GNU nano 2.2.6      File: /etc/rc.local

#
# By default this script does nothing.

# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi
echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-0/new_device
echo tmp102 0x48 > /sys/class/i2c-adapter/i2c-0/new_device

exit 0
```

* **Adafruit Occidentalis v0.2** image support the TMP102 and RTC DS1307 if you need this driver, you can choose this.

The image can be download from

<http://learn.adafruit.com/adafruit-raspberry-pi-educational-linux-distro/occidentalis-v0-dot-2>

- **if your kernel without tmp102 module**

The '-y' option disables interactive mode for the command and the '0' is the I2C bus to scan. You can also run this command on the Pi's second I2C bus by specifying '1' instead.

We can see that it has found our TMP102 device at address 0x48.

To read the temperature from our temperature sensor, we use the i2cget command to read a single byte (Byte 1 - full degrees) from the temperature register (0x00) of the device.

```
pi@raspberrypi ~ $ i2cget -y 0 0x48 0x00 b
0x16
```

Converting this hexadecimal value to decimal, we get our temperature of 22°C.

If you want a more precision on the temperature, you can read both the full and fractional bytes from register 0 as follows:

```
pi@raspberrypi ~ $ i2cget -y 0 0x48 0x00 w
0xa015
```

This gives us byte 2 (0xa0) and byte 1 (0x15), but as a 16bit hexadecimal number and in the wrong order. To convert to °C, swap around the bytes, shift right by 4, convert to decimal and multiply by 0.0625.

E.g.

$\text{dec}(0x15a0 \gg 4) * 0.0625 = 21.625^\circ\text{C}$

tmp102 information

<http://www.element14.com/community/groups/raspberry-pi/blog/2012/07/26/is-it-done-yet-temperature-sensing-with-the-raspberry-pi#comment-16249>

<http://www.agilart.com/blog/tmp102-raspberry-pi>

<http://donalorrissey.blogspot.co.uk/2012/09/raspberry-pi-i2c-tutorial.html>

<http://www.rpi-blog.com/2012/07/raspberry-pi-temperature-sensor-using.html>

DS1307 RTC information

<http://learn.adafruit.com/adding-a-real-time-clock-to-raspberry-pi/set-rtc-time>

New Scratch Raspberry Pi GPIO control program

<http://www.pridopia.co.uk/rs-pi-set-scratch.html>

Package Content

- 1x Rs-Pi RTC & Temp module board
- 1x CR1220 3V Battery
- 1x Manual