


```

COM12 - PuTTY
Type 'startx' to launch a graphical session

i2root@raspberrypi:~# i2cdetect -y 0
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20: 20  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
40:  --  --  --  --  --  --  -- 48 49  --  --  --  --  --  --
50: 50  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60: 60  --  --  --  --  --  -- 68  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --

root@raspberrypi:~# modprobe tmp102
root@raspberrypi:~# modprobe rtc-ds1307
root@raspberrypi:~# echo tmp102 0x48 > /sys/class/i2c-adapter/i2c-0/new_device
root@raspberrypi:~# echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-0/new_device
root@raspberrypi:~# hwclock -r
Sun 04 Nov 2012 02:09:41 UTC -0.282234 seconds
root@raspberrypi:~# sensors
tmp102-i2c-0-48
Adapter: bcm2708_i2c.0
temp1:    +35.8Â°C (high = +160.0Â°C, hyst = +150.0Â°C)

root@raspberrypi:~#

```

i2c bus device detect status and active TMP102 & RTC DS1307

48 -> tmp102 50 -> 24c32 60 -> MCP4725 68 -> RTC DS1307
 49 -> ads1015 20 -> 23008

11. U10 **FT232RL** & U12 **ZR3243** J4 USB input from PC or Mac or RS-Pi for USB to RS232 COM port, just connect a cable (5pin mini USB-B to A USB) then you have DB9 full 9pin RS232 function.

U12 **ZR3243** RS232 Transceiver Driver , You can have DB9 full 9pin RS232 output for PC or Mac or RS-Pi USB cable input .

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

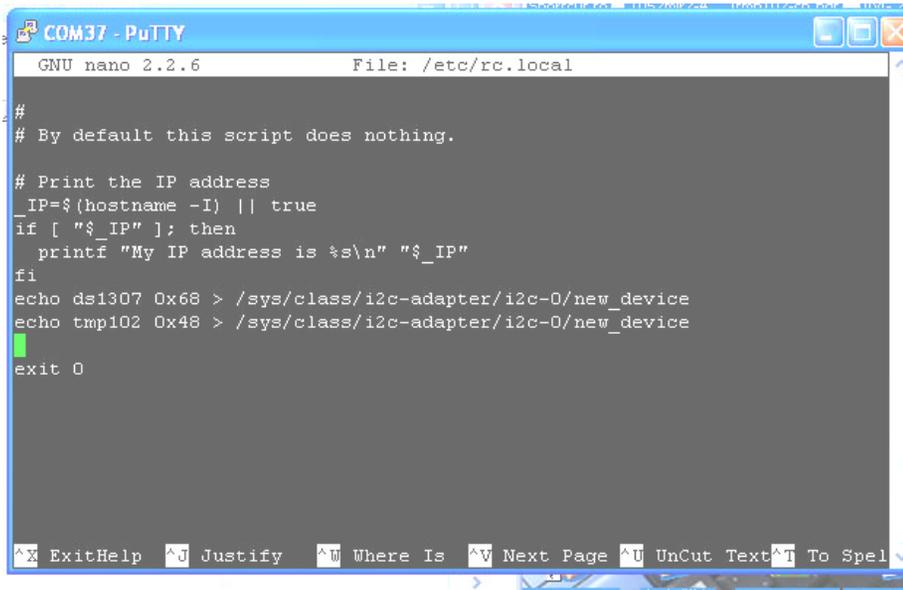
ExitHelp    Justify    Where Is    Next Page    UnCut Text    To Spel
1

```

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.

dec(0x15a0>>4) * 0.0625 = 21.625°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>

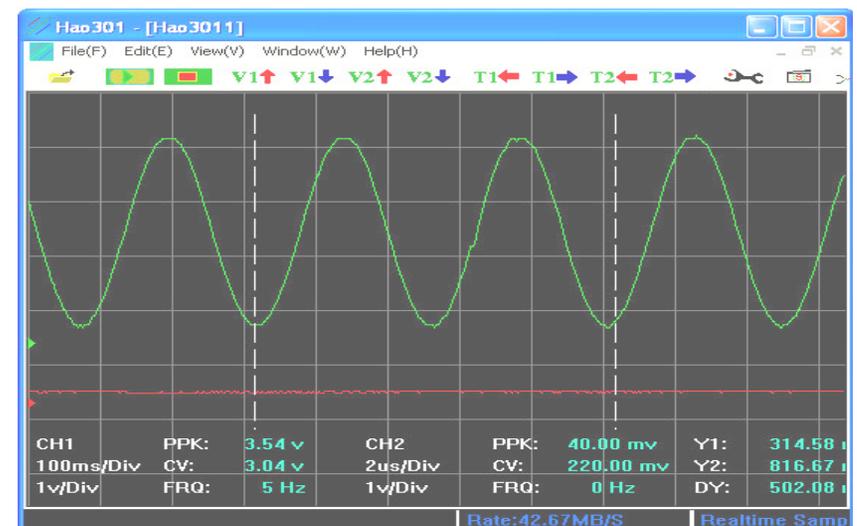
<http://www.aqilart.com/blog/tmp102-raspberry-pi>

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

* MCP4725 Digital to Analog Converter but our address are "60" all the sample can Download from our web site.

<http://learn.adafruit.com/mcp4725-12-bit-dac-with-raspberry-pi>

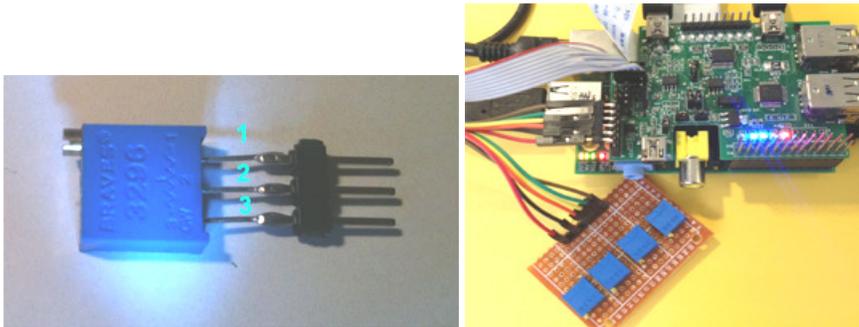
JP11 for analog output JP11 pin 1,2 (AOUT, GND)



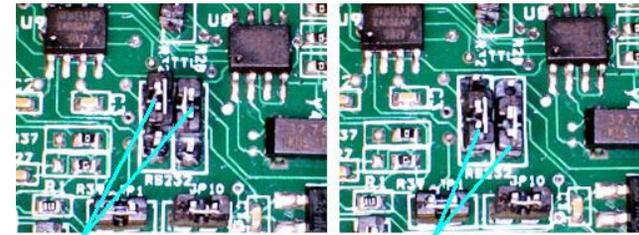
* ADS1015 12bit Analog-to-Digital Converter
demo 12 bit 4 channel input ads1015-49.py at address "49"

J7 INPUT (AIN0, AIN1,AIN2,AIN3,GND,3V3)

AIN0, AIN1, AIN2, AIN3 connect to **trimpot 10K OHM pin 2**
pin1 3v3 pin3 GND



* FT232RL the driver for USB to TTL can download from our web site, use this function , you need plug –in USB cable install driver first, then plug-in Rs-Pi hub board to Rs-pi board. * the USB to RS232 you can choose USB from PC or Mac or Rs-Pi, then your PC or Mac or Rs-Pi can have RS232 port full 9pin DB9



Setting for USB to TTL

Setting for USB

console Port

to RS232

Change R28 ,R32 Jumper you can change USB-TTL or USB-RS232 function

- * USB-TTL console function, access you Rs-Pi by a normal USB A to USB Mini B cable
- * USB-RS232 make your PC or Mac or Rs-Pi have full function 9 pin RS232 Port

For PC or Mac if you need driver you can download from our web site in number 19 FT232RL chipset

<http://www.pridopia.co.uk/support.html>

also the all the information for FT232RL

<http://www.ftdichip.com/Products/ICs/FT232R.htm>

Driver for PC or Mac from FTDI web site

<http://www.ftdichip.com/FTDrivers.htm>

```
COM21 - PuTTY
root@raspberrypi:/home/pi/ads1015# i2cdetect -y 0
  0 1 2 3 4 5 6 7 8 9 a b c d e f
00: -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- --
20: 20 -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- 48 49 -- -- --
50: 50 -- -- -- -- -- -- -- -- -- --
60: 60 -- -- -- -- -- 68 -- -- -- --
70: -- -- -- -- -- -- -- -- -- --

root@raspberrypi:/home/pi/ads1015# dir
Adafruit_ADS1x15.py  Adafruit_I2C.pyc  ads1015_example.py
Adafruit_ADS1x15.pyc ads1015-49.py      ads1015.py
root@raspberrypi:/home/pi/ads1015# ./ads1015-49.py
Reading ADS1015 Channel 0, 1, 2, 3 Total 4 Channel

Channel 0 = 1.485 V
Channel 1 = 1.869 V
Channel 2 = 0.981 V
Channel 3 = 1.371 V
root@raspberrypi:/home/pi/ads1015#
root@raspberrypi:/home/pi/ads1015#
```

ADS1015 use 4 trimpot 10K OHM for 4 channel Analog input

- **23008 23008 8 bit GPIO**

use 23008.py can set each bit output high or low

use 23008-20.py LED chaser program for Rs-Pi V1

use 23008-1-20.py for Rs-Pi V2

```
COM12 - PuTTY
root@raspberrypi:/home/pi# dir
24c32 4725 Desktop eeprog mcp230xx mcp4725 python_games rs-pi-pg
root@raspberrypi:/home/pi# cd mcp230xx
root@raspberrypi:/home/pi/mcp230xx# dir
23008.py 23017-22.py mcp23008.py mcp23017.wsgi
23017-21.py lcd1602.py mcp23017.py README
root@raspberrypi:/home/pi/mcp230xx# ./23008.py
Usage: mcp23008.py -b <bank> -o <output> -s <high|low>
root@raspberrypi:/home/pi/mcp230xx# ./23008.py -b a -o 0 -s high
Output GPA0 is already high.
root@raspberrypi:/home/pi/mcp230xx# ./23008.py -b a -o 1 -s high
Output GPA1 is already high.
root@raspberrypi:/home/pi/mcp230xx# ./23008.py -b a -o 2 -s high
Output GPA2 is already high.
root@raspberrypi:/home/pi/mcp230xx# ./23008.py -b a -o 1 -s low
Output GPA1 changed to low
root@raspberrypi:/home/pi/mcp230xx# ./23008.py -b a -o 3 -s high
Output GPA3 changed to high
root@raspberrypi:/home/pi/mcp230xx# ./23008.py -b a -o 4 -s high
Output GPA4 changed to high
```

- **24c32 EEPROM**

```
COM12 - PuTTY
root@raspberrypi:/home/pi/eeprog# dir
24c01.c 24cXX.o data2 eeprog.c eeprom_1.c i2c-dev.h WARNING
24cXX.c ChangeLog data4 eeprog.o eeprom_2 Makefile
24cXX.h data eeprog eeprom_1 eeprom_2.c README
root@raspberrypi:/home/pi/eeprog# ./eeprog /dev/i2c-0 0x50 -r 0:100 -f -x -16
eeprog 0.7.6, a 24Cxx EEPROM reader/writer
Copyright (c) 2003-2004 by Stefano Barbato - All rights reserved.
Bus: /dev/i2c-0, Address: 0x50, Mode: 16bit
Reading 100 bytes from 0x0
0000| 50 72 69 64 6f 70 69 61 20 ff ff ff ff ff ff ff
0010| ff ff
0020| ff ff
0030| ff ff
0040| ff ff
0050| ff ff
0060| ff ff ff ff

root@raspberrypi:/home/pi/eeprog# ./eeprog /dev/i2c-0 0x50 -r 0:100 -f -x -16
eeprog 0.7.6, a 24Cxx EEPROM reader/writer
Copyright (c) 2003-2004 by Stefano Barbato - All rights reserved.
Bus: /dev/i2c-0, Address: 0x50, Mode: 16bit
Reading 100 bytes from 0x0
Pridopia ~~~~~
~~~~~root@raspberrypi:/home/pi/eeprog#
root@raspberrypi:/home/pi/eeprog# ./eeprog /dev/i2c-0 0x50 -r 0 -f -x -16
eeprog 0.7.6, a 24Cxx EEPROM reader/writer
Copyright (c) 2003-2004 by Stefano Barbato - All rights reserved.
Bus: /dev/i2c-0, Address: 0x50, Mode: 16bit
Reading 1 bytes from 0x0
0000| 50
```

use eeprog 0.7.6 can read/write for 24c32 or 24cxx

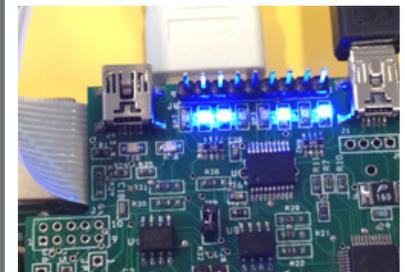
The Program can be download from

<http://www.codesink.org/eeprog.html>

New Pridopia **Pi_Scratch** interface software you can download from our web site, more easy to control by Raspberry pi

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

i2c address 20 23008 8 GPIO test



use command 8pin+"address"+a"bit (1 to 8)"

8pin+"address"+ a"10001000" --> 8bit mode

8pin20a5 --> output to address 20 bit 5

or use "8pin20a10010011" --> 8bit address 20 from 8 to 1 "10010011"

To keep Rs-Pi USB Hub board working properly, you need make sure the Vcc input for Rs-Pi above 4.75V, JP3 pin 1 Vcc, pin4 GND

Package Content

- 1x Rs-Pi USB Hub & I2C AD/DA & FT232RL-R1 board
- 1x USB to MINI USB hub input cable (for USB Hub input & Power input)
- 1x 2.0mm pitch 2x5 to DB9 com port cable with 30cm length
- 1x CR1220 3V Battery
- 1x Manual