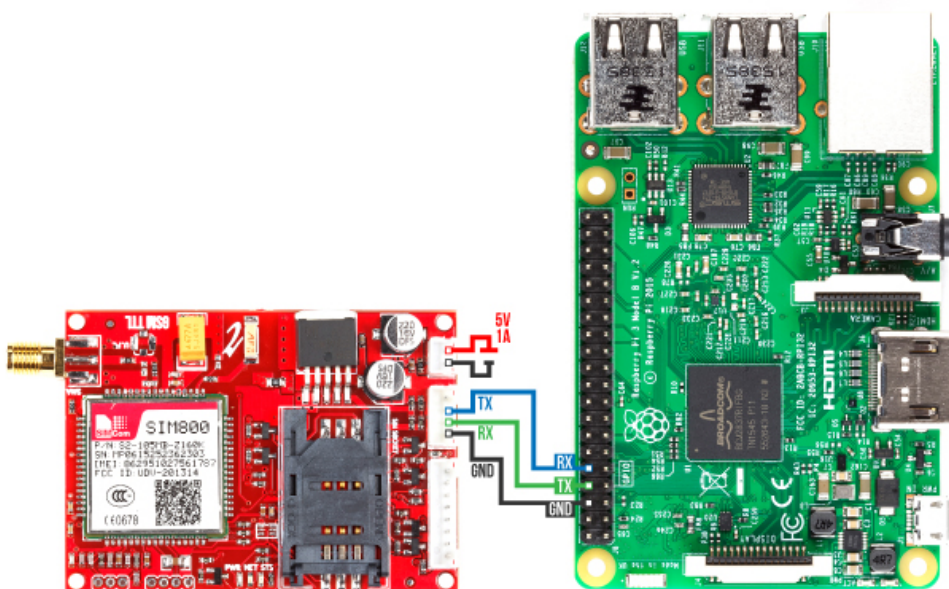


Raspberry Pi : How to access the Internet using GSM / GPRS Modem (SIM900/SIM800)

Here we are with yet another post on our favorite Pi. We had many different posts on Raspberry Pi as well as on GSM-GPRS connectivity. So what do we have here new? What we discussed so far focused either on GSM connectivity or TCP/IP connection over GPRS, here we intend to tether our Raspberry Pi to the internet through a GPRS cellular data connection. Lets see how its done..

Hardware Requirements

- [Raspberry Pi 3](#)
- [rhydoLABZ GSM/GPRS Modem](#)
- [Connecting Wires](#)
- 5v 1A power source (For GSM modem)

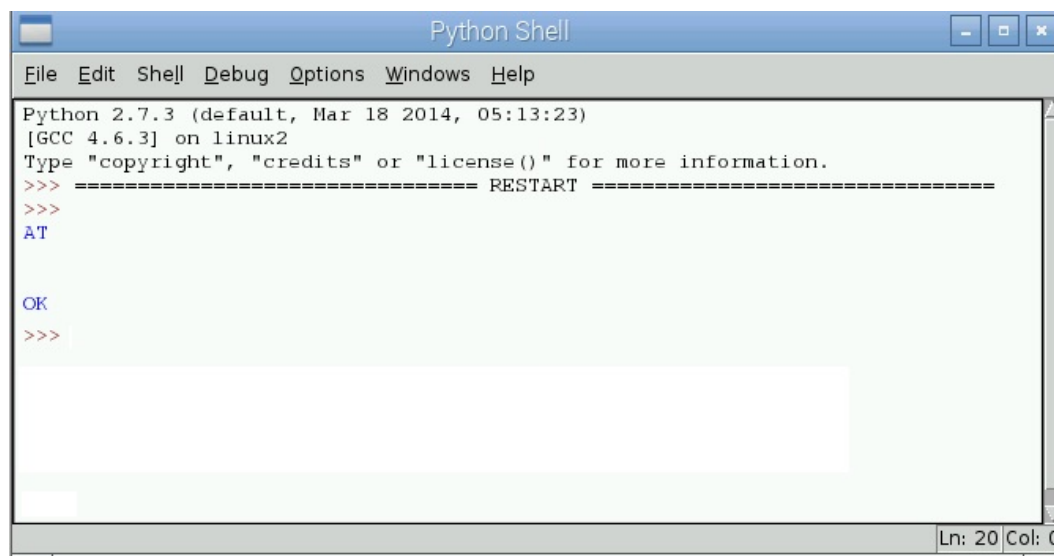


Make the connections as shown in the figure above. Provide separate power source (5V 1A) to the GSM modem for proper functioning as it drives much current. Insert a 2G activated SIM card, with working data connection, and power up the modules. It will take a few seconds for the SIM to get registered to the network. Now we need to check the communication between the RPi and the modem. For which, we make use of a python code that transmits an **AT** command and verifies whether an **OK** is received as acknowledgement. Open a python shell using **sudo idle** command and create a new file. Copy the code shown below, save and run the code.

If you are new to Raspberry Pi, we suggest reading [this post](#) before you proceed.

```
1 import serial
2 import os, time
3
4 # Enable Serial Communication
5 port = serial.Serial("/dev/ttyS0", baudrate=9600, timeout=1)
6
7 # Transmitting AT Commands to the Modem
8 # '\r\n' indicates the Enter key
9
10 port.write('AT'+'\r\n')
11 rcv = port.read(10)
12 print rcv
```

Run the code and if the connections are all working good, then we will receive an **OK** acknowledgement in the python shell.



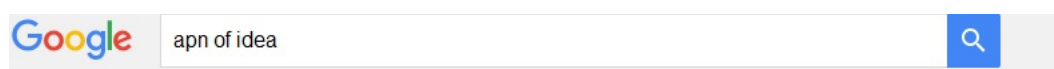
```
Python 2.7.3 (default, Mar 18 2014, 05:13:23)
[GCC 4.6.3] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
AT
OK
>>>
```

PPP Configuration

PPP or Point to Point Protocol establishes a Node to Node communication using serial interface. We make use of this, while accessing serial data connection on a PC. Here, using the serial connection, proper commands and PPP, we are about to access the internet on Pi.

APN

An Access Point Name (APN) is the name of a gateway between a GSM/GPRS network and the internet. The APN of the cellular network that we are using, must be know to us. You can either ask it out on google or contact the service provider. In this example, we have used [IDEA connection](#) and for the APN we asked it on Google.



All News Images Videos Maps More ▾ Search tools

About 6,06,000 results (0.43 seconds)

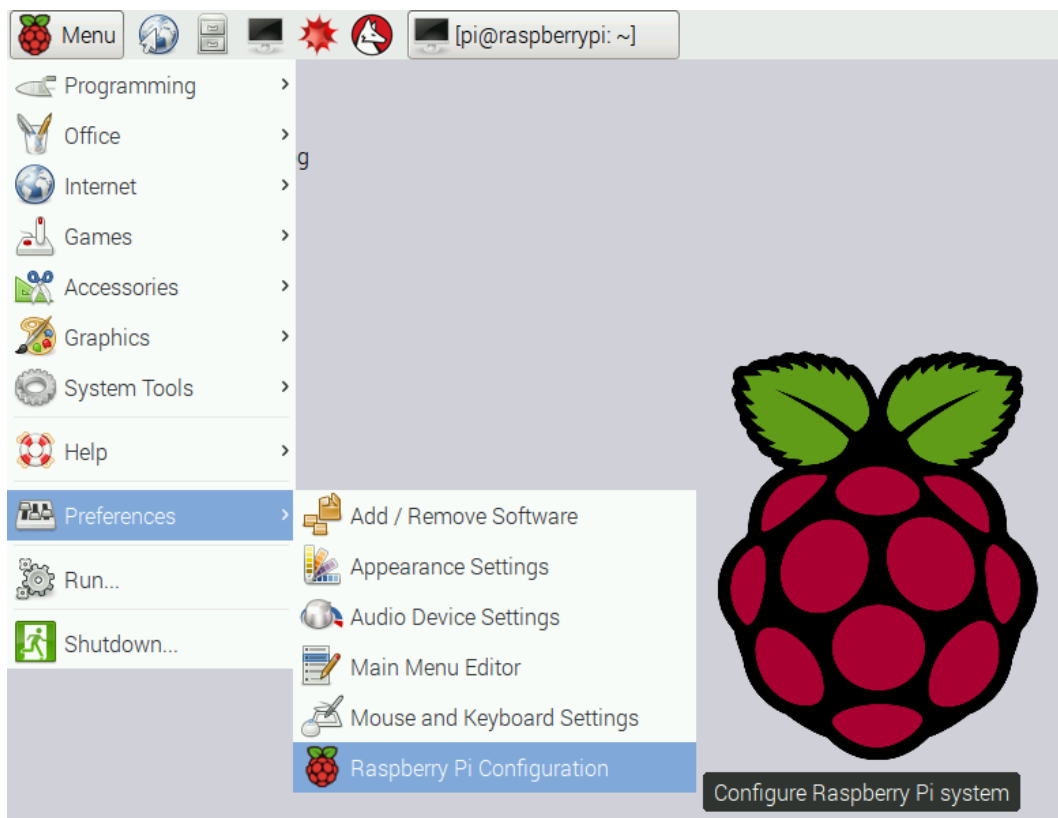
Idea APN for 2G/3G/GPRS

Setting Name	Value
Operator	IDEA
APN	imis/internet
Access Number	*99# or *99***#
Username	[blank]

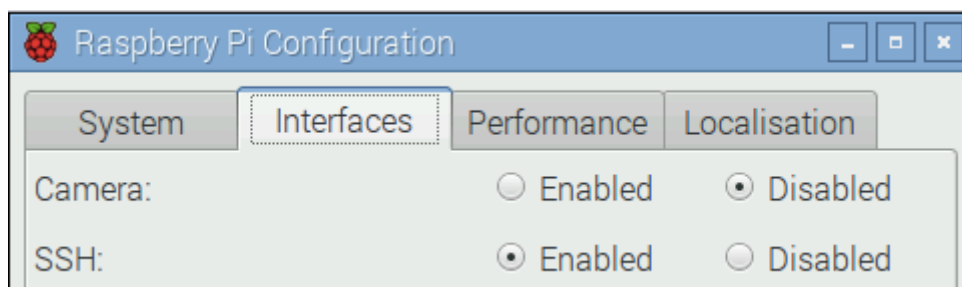
Installation

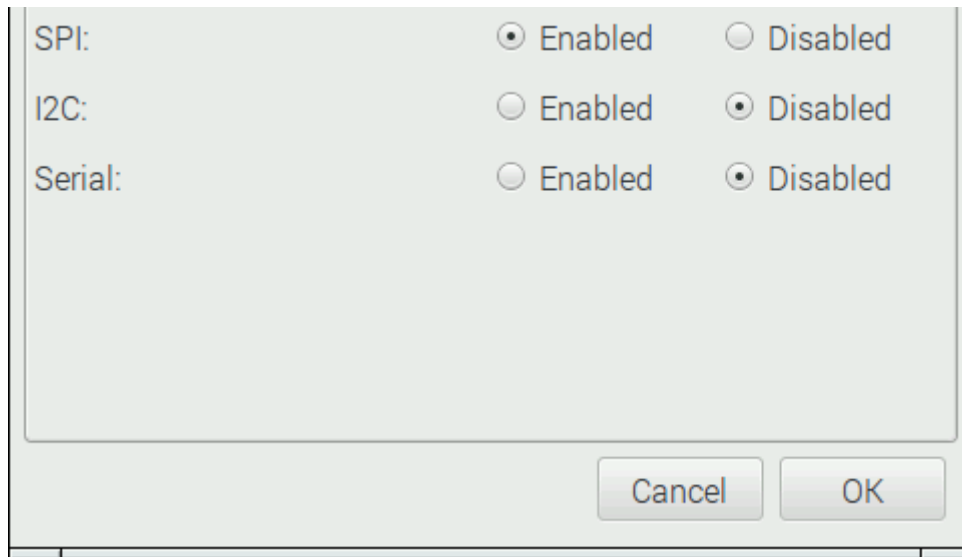
The first step is to disable the kernel's use of the hardware serial connection. By default, when the Raspberry Pi boots, it will use the serial connection to produce messages from the kernel and it will confuse the GSM modem. Follow the steps below

From Menu Select Preferences -> Raspberry Pi Configuration.



In this window, Select the interfaces tab and disable the serial option.





The next process is to install PPP Software. Make a suitable internet connection on the RPi by means of an Ethernet cable or WiFi. Open the LX Terminal and type:

```
1 sudo apt-get update
```

```
1 sudo apt-get install ppp screen elinks
```

After the installation we have to create a new PPP peer configuration. For this operation, we should login as root by entering **sudo -i** in the terminal and navigate to the peers directory,

```
1 cd /etc/ppp/peers/
```

Now open a new file **rnet** in a text editor by executing:

nano rnet

Copy the code shown below to the new file.

```
1 #imis/internet is the apn for idea connection
2 connect "/usr/sbin/chat -v -f /etc/chatscripts/gprs -T imis/internet"
3
4 # For Raspberry Pi3 use /dev/ttyS0 as the communication port:
5 /dev/ttyS0
6
7 # Baudrate
8 115200
9
10 # Assumes that your IP address is allocated dynamically by the ISP.
11 noipdefault
12
13 # Try to get the name server addresses from the ISP.
14 usepeerdns
15
16 # Use this connection as the default route to the internet.
17 defaultroute
18
19 # Makes PPPD "dial again" when the connection is lost.
```

```
20 persist
21
22 # Do not ask the remote to authenticate.
23 noauth
24
25 # No hardware flow control on the serial link with GSM Modem
26 nocrtscts
27
28 # No modem control lines with GSM Modem
29 local
```

From the above file, two parameters must be changed which depends on the modules used. In the beginning command, **connect**, after the section **-T**, the APN of the service provider should be given. Here we have used the APN of **Idea (imis/internet)**.

Then comes the communication port, as we have used Pi 3 in our example, the port name is **/dev/ttyS0**. Save the configuration file by pressing **ctrl+x**. This configuration file controls the options that will be set by PPPD, when the GSM Modem PPP connection is created. You can find a description of all the options in the PPPD main page.

In the above file, with in the line connect, a chat script is mentioned. To see this file, enter:

```
1 <strong>nano /etc/chatscripts/gprs</strong>
```

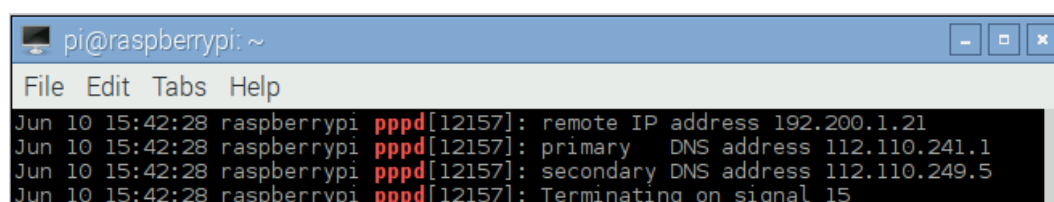
This will open a new file, which is not needed to be edited and this explains how a GPRS connection is created. If the SIM card needs a PIN to unlock, un comment the line **AT+CPIN=1234** and set the pin instead of 1234. Save the file and exit.

Next is to establish the connection. Before that make sure, all the steps described above are executed with no errors and proper wiring connections are also made.

Type, **sudo pon rnet** for creating the connection and it will exit with no response shown in the Lx terminal but the PPPD should be setting up the connection. To show the log for PPPD type,

```
1 <strong>cat /var/log/syslog | grep pppd</strong>
```

It will shows more messages and if the connection was established successfully, we can see the messages at the end as shown in the figure:



```
pi@raspberrypi: ~
File Edit Tabs Help
Jun 10 15:42:28 raspberrypi pppd[12157]: remote IP address 192.200.1.21
Jun 10 15:42:28 raspberrypi pppd[12157]: primary   DNS address 112.110.241.1
Jun 10 15:42:28 raspberrypi pppd[12157]: secondary DNS address 112.110.249.5
Jun 10 15:42:28 raspberrypi pppd[12157]: Terminating on signal 15
```

```

Jun 10 15:42:28 raspberrypi pppd[12157]: Connect time 0.0 minutes.
Jun 10 15:42:28 raspberrypi pppd[12157]: Sent 0 bytes, received 0 bytes.
Jun 10 15:42:28 raspberrypi pppd[12157]: Connection terminated.
Jun 10 15:42:29 raspberrypi pppd[12157]: Exit.
Jun 10 17:15:08 raspberrypi pppd[1181]: pppd 2.4.6 started by root, uid 0
Jun 10 17:15:12 raspberrypi pppd[1181]: Child process /usr/sbin/chat -v -f /etc/
chatscripts/gprs -T imis/internet (pid 1188) terminated with signal 15
Jun 10 17:15:12 raspberrypi pppd[1181]: Connect script failed
Jun 10 17:15:12 raspberrypi pppd[1181]: Exit.
Jun 10 17:53:48 raspberrypi pppd[804]: pppd 2.4.6 started by root, uid 0
Jun 10 17:53:49 raspberrypi pppd[804]: Serial connection established.
Jun 10 17:53:49 raspberrypi pppd[804]: Using interface ppp0
Jun 10 17:53:49 raspberrypi pppd[804]: Connect: ppp0 <-> /dev/ttyS0
Jun 10 17:53:50 raspberrypi pppd[804]: Remote message: Login OK
Jun 10 17:53:50 raspberrypi pppd[804]: PAP authentication succeeded
Jun 10 17:53:53 raspberrypi pppd[804]: local IP address 100.88.57.4
Jun 10 17:53:53 raspberrypi pppd[804]: remote IP address 192.200.1.21
Jun 10 17:53:53 raspberrypi pppd[804]: primary DNS address 112.110.241.1
Jun 10 17:53:53 raspberrypi pppd[804]: secondary DNS address 112.110.249.5
pi@raspberrypi:~ $

```

In the terminal type **ifconfig** and enter.

```

pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ sudo pon rnet
pi@raspberrypi:~ $ ifconfig
eth0      Link encap:Ethernet  HWaddr b8:27:eb:54:d4:8e
          inet6 addr: fe80::26db:526d:12c6:6fcc/64 Scope:Link
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:192 errors:0 dropped:0 overruns:0 frame:0
          TX packets:192 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:15552 (15.1 KiB)  TX bytes:15552 (15.1 KiB)

ppp0      Link encap:Point-to-Point Protocol
          inet addr:100.88.57.4  P-t-P:192.200.1.21  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:64 (64.0 B)  TX bytes:125 (125.0 B)

wlan0     Link encap:Ethernet  HWaddr b8:27:eb:01:81:db
          inet6 addr: fe80::ba27:ebff:fe01:81db/64 Scope:Link
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:47 errors:0 dropped:47 overruns:0 frame:0
          TX packets:23 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:12508 (12.2 KiB)  TX bytes:4487 (4.3 KiB)

pi@raspberrypi:~ $

```

A new section **PPP0** can be seen with the IP 192.200.1.21. In the modem the blue led will blink fast continuously as the connection is ON. To close the connection, type: **sudo poff rnet**. Well, rhydoLABZ modem is all you need now to start browsing on your Pi. The working is much similar to any of those commonly available internet dongles. We have made these to work with the PI, why don't we go for the same on a desktop PC?? We hope to have a post that will explain all of it. 😊 Keep reading our posts, and feel free to comment/share 😊