GSM-DSA Application note

Summary

Preparing the raspberry pi environment

In order to make the most out of your GSM-DSA, it is best to make sure that the operation system is up to date. To do this we need to connect to the platform used in the GSM-DSA by using a software called PuTTY.

PuTTY is a SSH and telnet client, developed originally by Simon Tatham for the Windows platform. PuTTY is open source software that is available with source code and is developed and supported by a group of volunteers.

Wiring the GSM-DSA to your pc

Connect the GSM-DSA to a router where the network is configured to 192.168.170.x with a mask 255.255.255.0

Connect your pc to the same router. The router may use DHCP with the address of the GSM-DSA (192.168.170.151) excluded or it might be just a switch. In the latter case you must set your network address manually to a free number of the same group: 192.168.170.50 for example.

Installing PuTTY

PuTTY is free of charge and may be downloaded here:

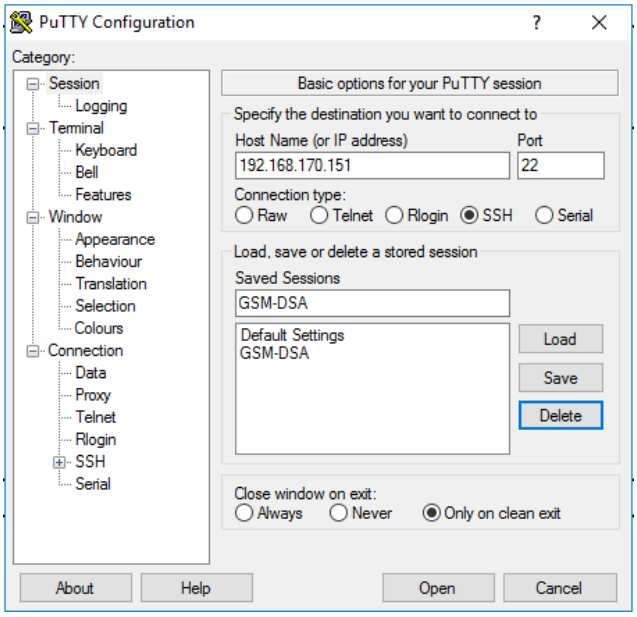
http://www.putty.org/

After the download, install and run it.

Connecting to the pi

After starting up PuTTY enter the host name and port used by your GSM-DSA. Per default this is 192.168.170.151 and port 22.

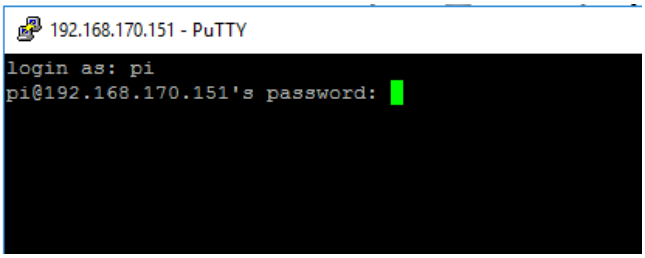
You can save this setup so you don't have to re-enter next time.



Click open.

Logging in

A black window appears with a login. Login as: pi. Password: raspberry



Updating Symbian

To do this your router or switch must be connected to the internet.

Enter: **sudo apt-get update**

If successful a list of get commands appear on the screen.
Now the update was downloaded.

Next enter: **sudo apt-get upgrade**

A message appears asking: pages will get upgraded continue? Y or N: Enter: **Y**
This might take a couple of minutes.

Setting a static Ethernet address

To change the network address of your GSM-DSA you need to edit the configuration file.

Enter: **nano /etc/dhcpcd.conf**

Use down arrow key to reach the end of the file and insert the following 4 lines of code at end of file:

```
interface eth0
static ip_address=192.168.170.151/22
static routers=192.168.170.1
static domain_name_servers=192.168.170.1
```

Press ctrl+X to leave and save with Y.

Setting a dynamic Ethernet address

Enter: **nano /etc/dhcpcd.conf**

Use down arrow key to reach the end of the file and delete the following 4 lines of code at end of file or add a # in front

```
# interface eth0
# static ip_address=192.168.170.151/22
# static routers=192.168.170.1
# static domain_name_servers=192.168.170.1
```

Press ctrl+X to leave and save with Y.

Now check if the interfaces config file is correct:

Then Enter: **nano /etc/network/interfaces**

```
# Please note that this file is written to be used with dhcpcd
# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'
# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

auto lo
iface lo inet loopback

iface eth0 inet dhcp

allow-hotplug wlan0
iface wlan0 inet manual
    wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf

allow-hotplug wlan1
iface wlan1 inet manual
    wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf

iface default inet dhcp
```

Make sure it looks similar to this. Specially the line iface eth0 inet dhcp.

Configuring a WiFi connection

Looking for available wifi networks:

Enter: **sudo iwlist wlan0 scan**

Write down the wifi network ESSD you need. In this example we call it MY-WIFI-NETWORK
Now your wifi network needs to be added to the configuration file.

Enter : **sudo nano /etc/wpa_supplicant/wpa_supplicant.conf**

Go to the bottom of the file and add the following:

```
network={
    ssid="MY-WIFI-NETWORK"
    psk="My_wifi_password"
}
```

Make sure to add the quotes for the ESSID and the password.

You can verify whether it has successfully connected using

ifconfig wlan0.

Check if the inet addr field has the IP address given by the WIFI router beside it. In this case the GSM-DSA has connected to the network. If not, check that your password and ESSID are correct.

Verification of connection

After changing the Ethernet or WiFi connection, verify that it works by rebooting.

Enter: **sudo reboot**
The GSM-DSA replies: The system is going down for reboot NOW!

To reconnect, address the new ip address with your PuTTY or browser and retry.

Changing the port for DGLUX5

Per default the port for DGLUX5 on the GSM-DSA is set to 80. To change this enter the following line in your PuTTY.

nano /opt/dsa/dglux-server/server.json

Find the line “port”: 80, and change 80 to a port of your choosing.
Find the line “httpsPort”: 443, and change 443 to a port of your choosing.

Press ctrl+X and Y for save.

Preparing the UPS and RTC

The Uninterruptable Power Supply prevents the GSM-DSA from crashing for a short term power loss. It will keep the device running for 2 minutes and will then perform a save shut down. This prevents a corruption of the data on the SD-card. Corruption may happen if the SD card is powered off while a write cycle to the card is happening.

The Real Time Clock provides a backup time source in case the GSM-DSA is not connected to an internet connection.

Verification of function of the real time clock

First we check if the rtc can be reached through its internal communication connection:

Enter: **i2cdetect -y 1**

A good answer looks like this:

```
#     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
#00:          -- -- -- -- -- -- -- -- -- -- -- -- --
#10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
#20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
#30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
#40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
#50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
#60: -- -- -- -- -- -- -- -- UU 69 6a 6b 6c 6d 6e 6f
#70: -- -- -- -- -- -- -- --
```

If there is no such reply: Follow instructions below to install RTC clock and make sure config file is correct as outlined below. Reboot. In rare cases it might be required to disconnect power and battery to make sure a full reboot by the UPS CPU is performed.

Then we check the firmware version of UPSpico

Enter: **sudo i2cget -y 1 0x69 0x26**

Reply: 0x30

This means V3.0

Verify power mode:

Enter: sudo i2cget -y 1 0x69 0x00 b

Reply 0x01 = Powered from power supply
Reply 0x02 = Battery powered.

Install UPS PICO V3.0 hardware RTC

Enter the following script in your PuTTY. Copy and right click to paste.

```
sudo apt-get install python-rpi.gpio
sudo apt-get install git python-dev python-serial python-smbus python-jinja2 python-xmltodict python-psutil python-pip
#Enter y (two times) to allow using additional diskspace
sudo pip install jinja2
sudo pip install xmltodict
sudo git clone https://github.com/modmypi/PiModules.git

cd PiModules/code/python/upspico/picofssd
sudo python setup.py install
sudo update-rc.d picofssd defaults
sudo update-rc.d picofssd enable
#Install RTC
sudo apt-get install i2c-tools
```

Enter: **sudo nano /etc/modules**
Insert the following 3 lines at end of file

```
i2c-bcm2708
i2c-dev
rtc-ds1307
```

Enter: **sudo nano /boot/config.txt**

add the following two lines at end of file:

```
enable_uart=1
dtoverlay=i2c-rtc,ds1307
```

NOTE: after accessing the raspi-config command and committing any changes, these lines need to be re-entered!

```
sudo reboot
sudo apt-get -y remove fake-hwclock
sudo update-rc.d -f fake-hwclock remove
```

Enter: **sudo nano /lib/udev/hwclock-set**

```
Delete or set commend char in front of these three lines:
if [ -e /run/systemd/system ] ; then
exit 0
fi
```

to

```
#if [ -e /run/systemd/system ] ; then
#exit 0
#fi
```

Setting the time

To set the time of the HW clock to the system time:

Enter: sudo hwclock –w

To set the source of the system time to the hwclock:

Enter: sudo hwclock --hctosys

To set he source of the system time back to the system:

Enter: sudo hwclock --systohc

Reading the time

Enter: **sudo hwclock –r**

UPS LED indicator

In total there are 4 LED on the board. From left to right they are

UPS – BAT – CHG - HOT

UPS LED	UPS LED is OFF	System is not running or is in Low Power Mode (only HW RTC is running)
	UPS LED is lighting continuously	System is booting or shutting down
	UPS LED is blinking every 600 ms	System is running on cable powering (after booting time)
	UPS LED is blinking every 1800 ms	System is running on battery powering
BAT LED	BAT LED is OFF	Battery is ok
	BAT LED is ON	Battery is missing or defect

CHG LED	CHG LED is OFF	Battery is not charging
	CHG LED is ON	Battery is being charged

Accessible registers of the UPS

0x69 UPS status registers specification

Enter: sudo i2cget -y 1 0x69 ADDRESS

ADDRESS	Name	Explanation
0x00 b	mode	Powering Mode – Read ONLY, Writing has no effect on the system and will be overwritten by UPS with the new value 0x01 - RPI_MODE (means cable powering mode) 0x02 - BAT_MODE
0x08 w	batlevel	Value of Battery Voltage in 0.01 VDC in BCD format 0x0490 means 4.9 VDC
0x0a w	rpilevel	Means value of Voltage supplying RPi on J8 5V Pin in 0.01 VDC in BCD format 0x0510 means 5.1 VDC
0x1b b	ntc	Temperature in Celsius degree of the embedded NTC1 sensor placed on the top of PCB. Values in BCD format. 0x35 means 35 °C.
0x22 w	ups_is_running	It is a 16 bit unsigned variable that value of it, is changing every 1 ms within the main loop of the firmware. Reading two times of this variable must return a different value (with interval longer than 1 ms), if not, means that system hangs-up, and need to be reset, if not restarted by other protection internal mechanism (watch-dog, and supervising watch dog). As these protection mechanisms are always restarting the system when something goes wrong, reason of existence of this variable is just to confirm to the remote user that everything is working well and give feedback to the remote user that system is running properly. As it is a mirror variable, writing to it nothing change, will be again re-written with the newer internal value.

0x6A RTC Registers specification

Enter: sudo i2cget -y 1 0x6a ADDRESS

ADDRESS	Name	Explanation
0x00 b	seconds	seconds in BCD
0x01 b	minutes	minutes in BCD
0x02 b	hours	hours in BCD
0x03 b	wday	week day in BCD
0x04 b	mday	month day in BCD
0x05 b	month	month in BCD
0x06 b	year	year in BCD

Setting the shut down time after power loss

Once there is a power failure, the UPS will execute a safe shut down procedure after a delay time is expired. After the safe shut down the system is in sleep mode, the RTC will keep running.

If power returns again the system will restart automatically.

The factory default for the delay time is set to 60 seconds. It can be verified through this command:

Enter: sudo i2cget -y 1 0x6b 0x01 b

Each number represents 1 minute. The default value is 0 and the highest value is 0xFE. If the value is set to 2, the shutdown delay time will be 1 Minute + 2 Minute = 3 Minutes.

A value of 0xff (255) disables this timer, and system will run on battery power until the battery discharges to 3.4V for LP battery and 2.8V for LF Battery type.

To change the value:

Enter: sudo i2cset –y 1 0x6b 0x01 0x##,

where ## represents the number you wish to change it to. For example

sudo i2cset –y 1 0x6b 0x01 0x04

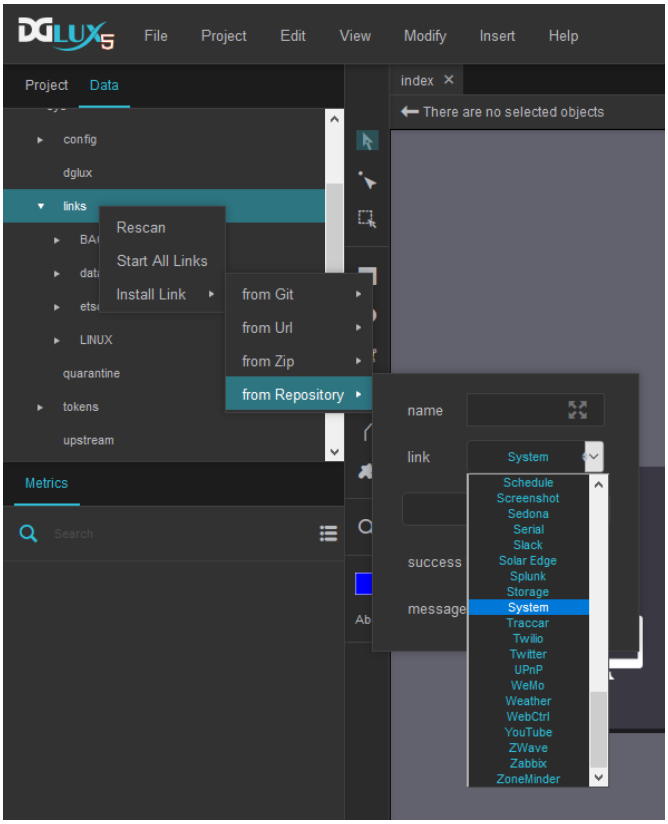
would change it to a 5 min delay.

Operation of UPS with DGLUX5

On DGLUX5 switch to data – sys – links.

Right click on links and select install link,

Then select from Repository, give it a name like SYSTEM for example and select System from the drop down list.



Under Downstream Linux-System is now available and commands can be written to it through right click.
Select Execute Command.
Enter the desired command string in the command form and press invoke. Result will be written under output.