Technical Brief 20100507 from Missing Link Electronics:

Using USB Hot-Plug For UMTS Short Message Service

This Technical Brief describes how the USB hot-plug capabilities of the MLE “Soft” Hardware Platform can be used to quickly connect USB-based consumer electronic devices. In an exemplary setup, we demonstrate how to connect an off-the-shelf Universal Mobile Telecommunications System (UMTS) modem device to extend the MLE “Soft” Hardware Platform with UMTS/GSM connectivity. We will present an Open Source based solution, which goes beyond a simple Short Message Service (SMS) and which can, for example, be used for remote monitoring and administration of sensor networks in the field.

— Technical Brief 20100507 —

The FPGA technology underneath the MLE “Soft” Hardware Platform allows very flexible connectivity to standard and - not so standard - application specific I/O. This facilitates quick integration of sensors and actuators to build proof-of-concepts for a wide range of embedded systems. The full integration of a programmable system-on-chip microcontroller with a complete GNU/Linux software stack provides scripting capabilities, plus a network stack with TCP/IP Internet connectivity, all ready to run.

In the following we will demonstrate how to make use of widely available mobile internet USB devices to extend the connectivity. Application examples are remote monitoring and administration of sensor networks etc.

The key feature of a platform is having a large set of readily available peripheral hardware that can easily be added. This is quite important, as building new hardware, even when using a platform based aproach, may take too much time. As important as having peripheral hardware components readily available is having a large set of ready-to-use software which supports those peripherals and allows to quickly build a proof-of-concept. We will show how to extend the MLE “Soft” Hardware Platform in general, and the MLE 1000 Series Rapid Prototyping System in particular, with Short Message Service (SMS).

Figure 1: Short Message Service for MLE 1000 RPS

We will utilize the USB connectivity of MLE. USB has become the world wide standard for extending personal computers with new interfaces and a large number of different devices is readily available, including UMTS modem devices. These UMTS modems can send and receive SMS. But how can USB-based extensions be integrated into the system quickly?

That is where open source software and the MLE Linux distribution come into play. USB is designed as a plug-and-play standard. When running the MLE Linux kernel with its

outstanding driver support plus a hotplug daemon like `udev`, getting the modem to work is as simple as plugging it into the USB port of any other Linux workstation.

When plugging a supported USB UMTS modem into any USB port of the MLE 1000 Series Rapid Prototyping System the Linux kernel will report something like the following:

```
root@mlerps1k:~# dmesg
.
.
.
[  314.921800] usb 1-2: new full speed USB device using c67x00 and address 3
[  315.124933] usb 1-2: New USB device found, idVendor=12d1, idProduct=1003
[  315.124993] usb 1-2: New USB device strings: Mfr=2, Product=1, SerialNumber=0
[  315.125026] usb 1-2: Product: HUAWEI Mobile
[  315.125053] usb 1-2: Manufacturer: HUAWEI Technology
[  315.128998] usb 1-2: configuration #1 chosen from 1 choice
[  316.285341] usbcore: registered new interface driver usbserial
[  316.285619] USB Serial support registered for generic
[  316.287857] usbcore: registered new interface driver usbserial_generic
[  316.287915] usbserial: USB Serial Driver core
[  316.524303] USB Serial support registered for GSM modem (1-port)
[  316.524772] option 1-2:1.0: GSM modem (1-port) converter detected
[  316.547732] usb 1-2: GSM modem (1-port) converter now attached to ttyUSB0
[  316.548088] option 1-2:1.1: GSM modem (1-port) converter detected
[  316.562381] usb 1-2: GSM modem (1-port) converter now attached to ttyUSB1
[  316.562882] usbcore: registered new interface driver option
[  316.562928] option: v0.7.2:USB Driver for GSM modems
root@mlerps1k:~#
```

The hotplug framework then creates new terminal device nodes for the discovered modem. In this case they are named `ttyUSB0` and `ttyUSB1`. Application software can access the modem via these device nodes. For transmitting SMS `ttyUSB0` is relevant, only. The MLE Linux distribution ships with a minimal client named `com` to connect to terminal devices. We will now complete the proof-of-concept part quickly using this client. We communicate directly and manually with the modem and instruct it to send an SMS.

```
root@mlerps1k:~# com /dev/ttyUSB0
C-a exit, C-x modem lines status
[STATUS]: RTS CTS DSR DCD DTR
AT
OK
ATE1
OK
AT+CPIN="<MyPin>"
OK
AT+COPS
OK
AT+COPS?
+COPS: 0,0,"<MyOperator>",2
```

```
OK
AT+CMGF=1
OK
AT+CMGS="<YourNumber>"
> Hello World from RPS1000 via AT-commands!
> <ctrl>+<z>

+CMGS: 7

OK
```

<YourNumber> now will receive a SMS stating `Hello World from RPS1000 via AT-commands!`. For a detailed look at what those AT-commands from above do, we recommend the Openmoko wiki [OMWIKI] and the SMS Tutorial by developershome.com [DEVHOME]. Receiving SMS can be tested in the same way using the `AT+CMGL="REC UNREAD"` command.

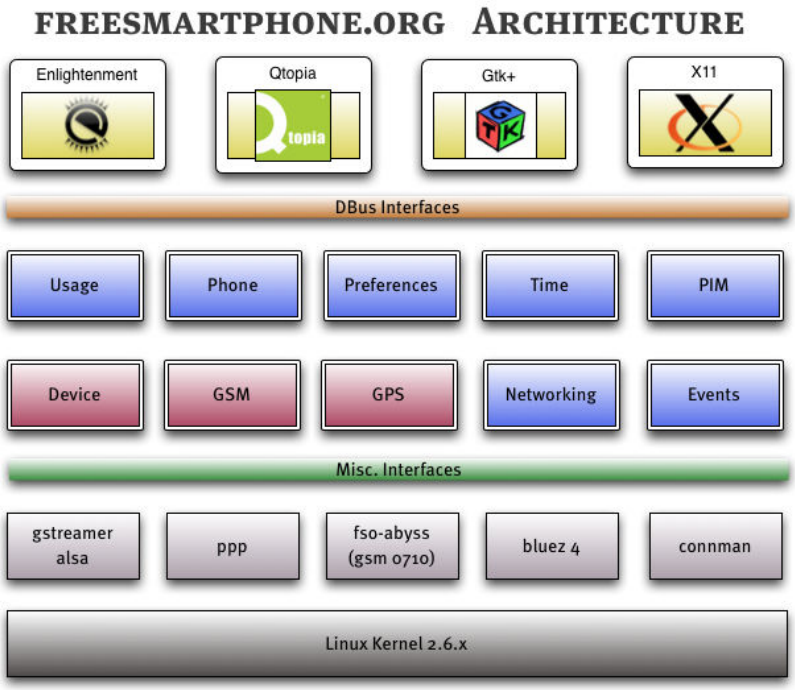

Figure 2: The Free Smartphone middleware. Courtesy of freesmartphone.org

Now that we have a proof-of-concept combining the MLE 1000 Series Rapid Prototyping System and an off the shelf USB UMTS stick, a more user friendly way of handling SMS transfers is to be found. For this purpose a variety of SMS frameworks are provided as open source software and are readily available for both, the MLE 1000 Series Rapid Prototyping System and for your desktop Linux system.

This will make development easier, as it can be done on any Linux workstation as long as you stick to the same framework you chose for the MLE “Soft” Hardware Platform. We

have tested the `libgsmd` from the Openmoko project, which provides a rather minimalistic, but easy to use command line tool `libgsmd-tool` for basic phone tasks. Depending on the application this might already be sufficient. For a more complex and complete solution the freesmartphone.org stack [FSO] as seen in Figure 2 is also available for the MLE 1000 Series Rapid Prototyping System.

The FSO framework includes an abstraction layer for the hardware, telephony services, networking, time, date and location services as well as data and user preferences storage. A lot of ready-to-run example software is available and maintained by the open source community. This enables you to quickly integrate technically any feature you know from modern smartphones into your prototype. Examples range from Internet based update and user application management, via skinable modern graphical user interfaces down to more basic tasks like integration and management of multiple network connections via multiple interfaces. Of course, this is a lot more than just transmitting SMS.

In this technical brief we have shown, how a proof-of-concept for SMS integration into a system based on the MLE 1000 Series Rapid Prototyping System can be developed quickly. We have used off-the-shelf consumer hardware such as a UMTS modem, connected to a standard USB interface and have taken huge benefits from the readily available open source frameworks and Linux drivers.

References

[OMWIKI] Openmoko, Inc.:
Openmoko wiki: Manually using SMS, retrieved April, 30th 2010.
http://wiki.openmoko.org/wiki/Manually_using_SMS#GSM_Modem_Setup

[DEVHOME] developershome.com:
developershome.com: SMS Tutorial, retrieved April, 30th 2010.
http://www.developershome.com/sms/

[FSO] freesmartphone.org:
freesmartphone.org: Free Smartphone, retrieved April, 30th 2010.
http://www.freesmartphone.org/