Paweł Koszut
Institute of Telecommunications
Warsaw University of Technology

Intercepting GSM Communication Using Open-Source and Open-Hardware Technologies

Abstract: This paper presents current achievements of open-source community developers in the area of building GSM-enabled communication hardware and software. Open-source GPL licensed software makes constant and very significant contribution to IT industry. Together with so called open-hardware, it enables the cost of some expensive GSM devices, for example bugging devices used by intelligence to be lowered to the cost of raw electronic components and this can have potentially enormous influence on other branches of mobile telecommunication industry, similar to the impact open-source exerted on software development in recent years. Bugging devices can eavesdrop mobile communication based on A5/1 cryptanalysis achievements and a flaw found in GSM protocol stack.

1. Introduction

In 2003, when a team of researchers from Technion published their achievements in A5/1 cryptanalysis and GSM cryptographic protocol flaw [1], it became clear that mobile communication can no longer be considered secure. Although GSM bugging devices used by law-enforcement agencies are still very expensive and only governments and big companies can afford them, very interesting question arises : for how long ?
Because most of these devices employs proprietary software developed by manufacturers, their cost remains relatively high. By applying free open standards to software and hardware development, many contributors worldwide can share their common work and make it free for others who want to join a project, and also free for those who simply want to use it for their own purposes, both private and commercial. This is the same approach which resulted in emergence of Linux operating system and many more free professional software products. Apart from lowering the cost, there is also one more advantage of building intelligence devices using open standards – along with the fact that the production can be easily moved to Poland, this can significantly reduce security concern of trapdoors installed in these devices by foreign intelligence cooperating with big manufacturers abroad or secretly inserting malicious code – a phenomenon which occasionally happens. For example in 2006 a scandal broke out when Greek government admitted that mobile operator Vodafone found spying code installed in their software [2], and this enabled many government VIPs including Prime Minister to be bugged [3]. The availability of source code and the fact that random people worldwide can learn it and contribute, in obvious way reduce such risk.

Further from this paper you can learn the approach open-source community has chosen to develop

GSM-enabled devices, i.e. hardware and software solutions used to build it. This will help you better understand the process of transforming GSM radio waves into readable data with the use of free and open standards only. Finally this will demonstrate the importance of these achievements for our domestic industry and law-enforcement agencies.

2. Hardware platform

The project is based on a hardware platform which consists of usual PC (or laptop) and USRP board - Universal Software Radio Peripheral, which is a low-cost software defined radio equipment. It was developed by Ettus Research LLC as an open-hardware solution, and is currently well supported by open-source community developers, user mailing lists and a producer. USRP consists of a motherboard and pluggable daughterboards which cover wide range of frequencies from 0 Hz to almost 3 GHz and are constantly developed.

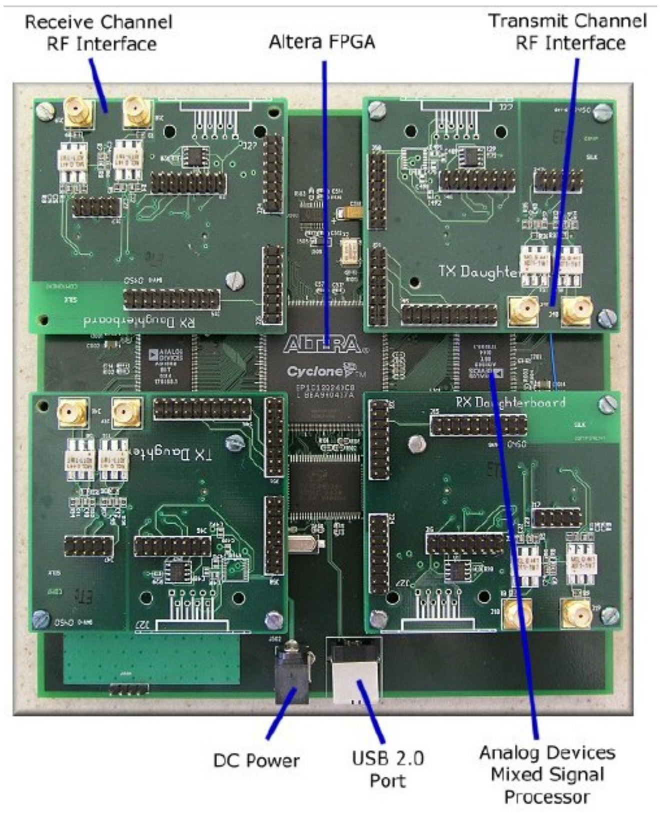

Receive Channel RF Interface
Altera FPGA
Transmit Channel RF Interface
DC Power
USB 2.0 Port
Analog Devices Mixed Signal Processor

Figure 1 : USRP motherboard and four daughterboards plugged in.

Wide range of frequencies are covered with the following daughterboards, which are pluggable into motherboard's four slots (two RX and two TX slots) :

- DC to 30 MHz receiver
- DC to 30 MHz transmitter
- 1 MHz to 250 MHz receiver
- 1 MHz to 250 MHz transmitter
- 50 to 860 MHz receiver

- 800 MHz to 2.4 GHz receiver
- 400-500 MHz transceiver
- 750-1050 MHz transceiver (including cell and ISM bands)
- 1150-1450 MHz transceiver
- 1.5-2.1 GHz transceiver (including PCS bands)
- 2.3-2.9 GHz transceiver (including ISM band)

Figure 2 : DBSRX daughterboard

For receiving GSM signals it is advisable to choose DBSRX daughterboard (see figure 2 above) and plug it into one of the RX slots. It is a receiver-only board but it covers 800 MHz – 2.4 GHz band which is enough to receive most GSM signals used worldwide. This daughterboard along with for example 900 MHz to 2.6 GHz PCB (Printed Circuit Board) antenna can be chosen by developers as a set of tools for further research and experiments. To learn more about USRP, see Dawei Shen's tutorial [4].

3. Software environment

Software environment consists of Linux operating system along with GnuRadio package installed. GnuRadio is a powerful signal processing software fully compatible with USRP boards. It consists of many high level signal processing blocks which perform certain operations - for example signal generation, filtering, demodulation etc.

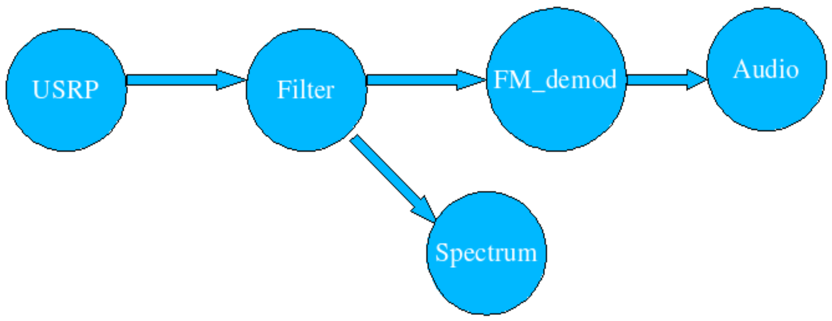


Figure 3 : GnuRadio blocks sample connections

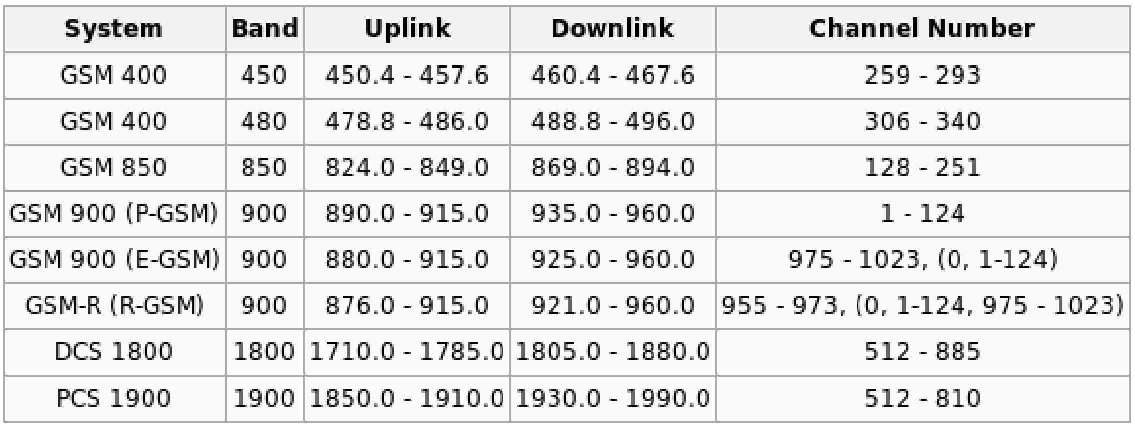
From the example above, you can learn how the blocks are tied together in order to develop applications. As you can see, USRP source block can be used to read samples of radio waves from USRP receiver, then the data stream can be processed by any configuration of gnuradio blocks, and finally it is thrown into spectrum viewer, audio output (sound card) or any other gnuradio sink block.

These signal processing blocks are implemented in C++ and are accessible from high level Python language as objects. In order to create an application, gnuradio blocks have to be tied together and properly configured if necessary. This simplifies application development process because programmers don't have to bother with implementation of some commonly used signal processing operations needed for their applications. However, they are still able to develop their own gnuradio signal processing blocks in C++. New blocks simply get input stream of data (usually type of complex, float, unsigned char, etc.) make any appropriate computations and return processed stream of output data. More on writing your own gnuradio blocks you can learn from Eric Blossom's documentation : How to Write a Signal Processing Block [5]

GnuRadio comes with many example applications which can be used along with USRP hardware, for example oscilloscope, spectrum browser, FM radio receiver. For our purposes you can find very comfortable to use gnuradio/gnuradio-examples/python/usrp/usrp_fft.py program (spectrum viewer) to explore the spectrum of GSM bands and to find GSM Base Transceiver Stations (BTS) transmitting in your area (more on that in the next chapter).

4. Preparation to receive GSM signals

After software and hardware is properly configured, developers can test their receiver using standard gnuradio spectrum browser application. From the following table you can learn the range of frequencies GSM operates.



System	Band	Uplink	Downlink	Channel Number
GSM 400	450	450.4 - 457.6	460.4 - 467.6	259 - 293
GSM 400	480	478.8 - 486.0	488.8 - 496.0	306 - 340
GSM 850	850	824.0 - 849.0	869.0 - 894.0	128 - 251
GSM 900 (P-GSM)	900	890.0 - 915.0	935.0 - 960.0	1 - 124
GSM 900 (E-GSM)	900	880.0 - 915.0	925.0 - 960.0	975 - 1023, (0, 1-124)
GSM-R (R-GSM)	900	876.0 - 915.0	921.0 - 960.0	955 - 973, (0, 1-124, 975 - 1023)
DCS 1800	1800	1710.0 - 1785.0	1805.0 - 1880.0	512 - 885
PCS 1900	1900	1850.0 - 1910.0	1930.0 - 1990.0	512 - 810

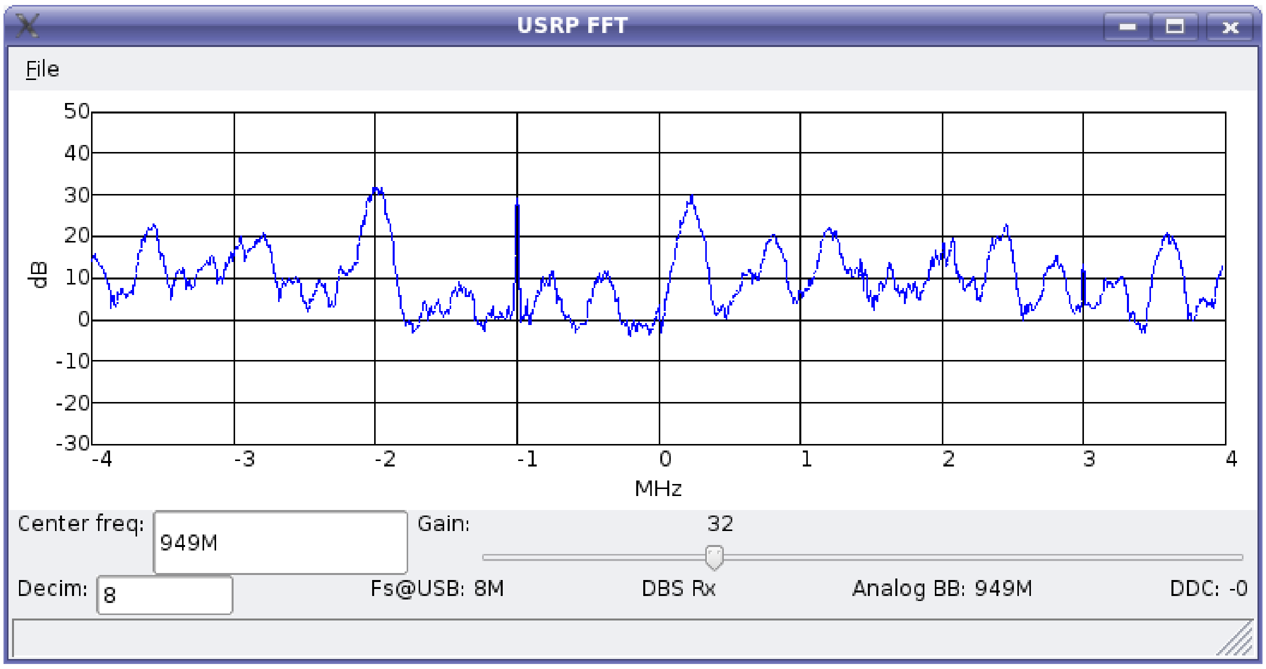
Figure 4 : GSM frequency bands

Since, we want to receive signals sent down from BTS to MS (Mobile Station), we focus on downlink range of frequencies. In most countries trying 935 MHz – 960 MHz and 1805 MHz – 1880 MHz is the most appropriate to start with.

Using the following example command :

gnuradio/gnuradio-examples/python/usrp/usrp_fft.py -R A -d 8 -g 32 -f 949M

we observe 8 MHz wide band of spectrum centered at 949 MHz, in which we find several BTSs, each one transmitting in a 200 kHz wide channel.



Center freq: 949M
Gain: 32
Decim: 8
Fs@USB: 8M
DBS Rx
Analog BB: 949M
DDC: -0

Figure 5 : GSM spectrum view

Not all of the channels are used in every area, some of them are reserved to be used in neighbouring cells (we see them as lower strength signals), or simply there is no need for more BTSs to operate in a specific population area.

On every active channel, a BTS transmits GMSK-modulated signal within eight consecutive timeslots numbered 0..7, each one lasts for 577 microseconds. In GMSK modulation, frequency changes +/- 67.708 kHz from a specified center frequency and because of differential modulation employed in GSM, consecutive stream of identical bits (for example 0000000 or 1111111) will result in constant frequency +67.708 kHz above center frequency, and similarly consecutive stream of differing bits (for example 01010101010101) will result in constant frequency -67.708 kHz below center frequency. This is used in Frequency Correction Channel (FCH) packets, which consist of all zeros (142 consecutive zeros), resulting in constant frequency signal being transmitted for 577 microseconds and enabling Mobile Stations to use it in order to correct their frequency offset during this period.

Using GnuRadio blocks we can build an application which tunes USRP to specified frequency, filter the signal and demodulate it into a stream of bits, but before we do so, lets look closer to the signal using fm demodulation block and oscilloscope which enables us to see frequency changes in time domain.

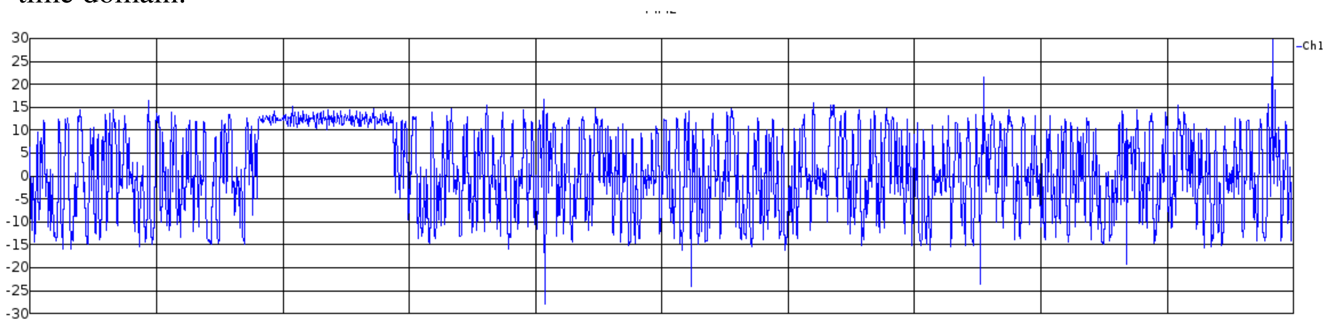
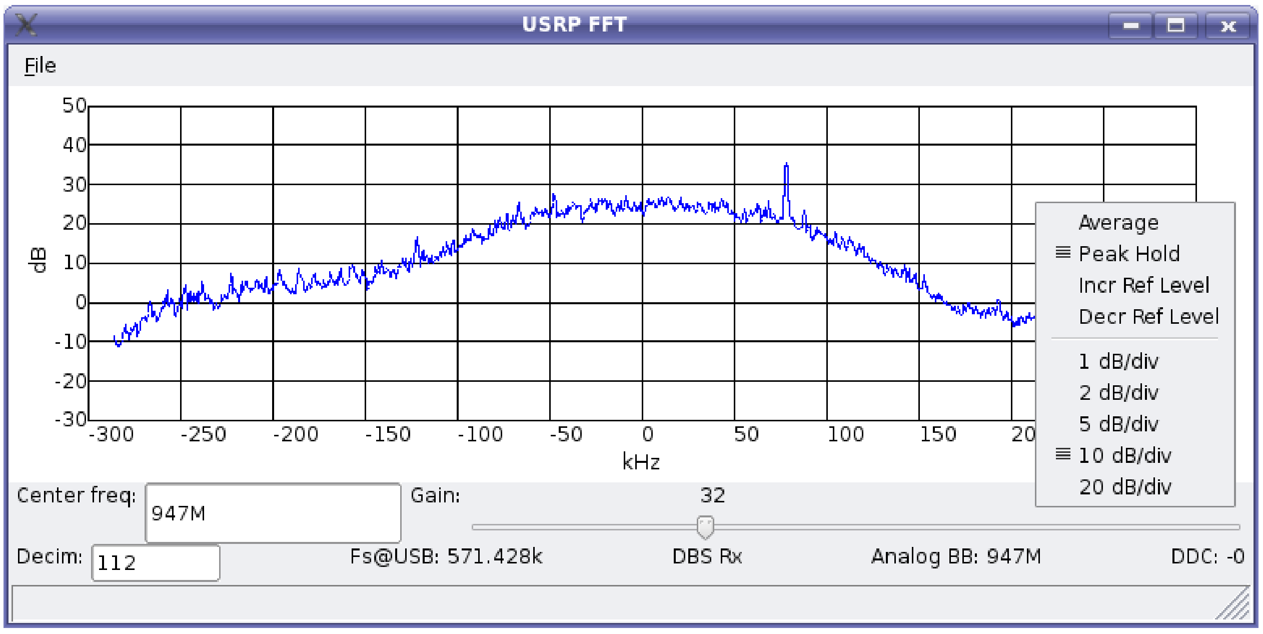


Figure 6 : Frequency changes in time domain

In the figure above, we clearly see one FCH packet with constant frequency +67.708 above center frequency, which lasts for 577 microseconds. This confirms that we receive and process accurate data from USRP.



Center freq: 947M
Gain: 32
Decim: 112
Fs@USB: 571.428k
DBS Rx
Analog BB: 947M
DDC: -0

Figure 7 : Radio spectrum of one single GSM channel

Lets focus on the strongest available signal we receive 2 MHz below the center frequency (949 MHz), so we tune our USRP to 947 MHz and change decimation rate to 112 in order to decrease the spectrum width, so it represents only one single GSM channel, see figure 7 above.

The result was obtained with the command :

gnuradio/gnuradio-examples/python/usrp/usrp_fft.py -R A -d 112 -g 32 -f 947M

and clearly shows a single pick located approximately +67 kHz above the center frequency (note:

"Peak Hold" option is helpful to catch the peak). This represents FCCH packet being sent periodically once every 10 packets in timeslot 0. For mobile phones, FCCH packets are used to precisely adjust to a carrier frequency, compensating for unavoidable frequency offset which depends on a specific piece of hardware. Also for USRP decoding is far better if the frequency offset, which ranges from several kHz to 32 kHz, is compensated. USRP offset can be estimated from a spectrum view but only with limited accuracy. Better estimation can be calculated in dedicated software which gives far more precise results.

5. Processing GSM signals

The easiest way to obtain GSM data is to get source signal from USRP tuned to a frequency on which a BTS operates, then filter the signal and demodulate it using existing gnuradio blocks.

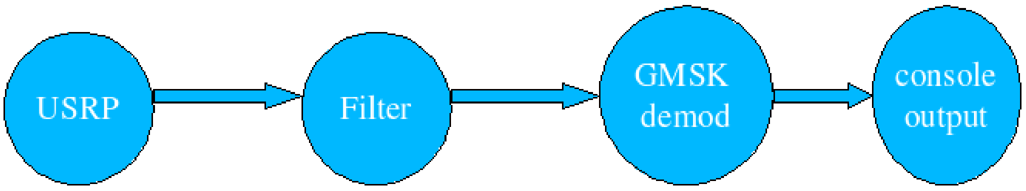
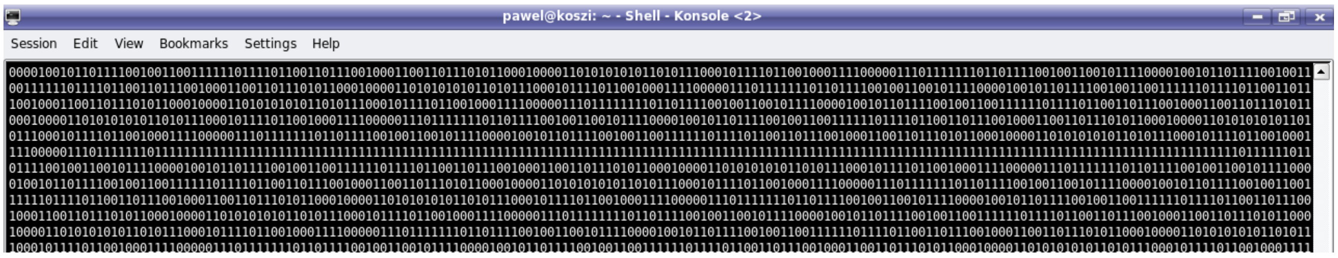


Figure 8 : Basic approach in GSM signal processing

This method is very simple and outputs a stream of bits from chosen GSM channel. Despite this approach doesn't use any correction methods and frequency offset isn't compensated during signal reception, the results are good enough to decode many interesting packets from the air.



```
00001001011011110010011001111110111101100110111001000110011011110101100010000110101010101101011100010111101100100011110000001110111111110110111100100110010111100001001011011110010011
001111110111101100110111001000110011011101011000100001101010101010110101110001011110110010001111000000111011111111011011110010011001011110000100101101111001001100111111011110110011011
10010001100110111010110001000011010101010101101011100010111101100100011110000001110111111110110111100100110010111100001001011011110010011001111110111101100110111001000110011011101011
0001000011010101010101101011100010111101100100011110000011101111111101101111001001100101111000010010110111100100110011111101111011001101110010001100110111010110001000011010101010101101
01110001011110110010001111000001110111111110110111100100110010111100001001011011110010011001111110111101100110111001000110011011101011000100001101010101011010111000101111011001000 1
11100000011101111111101111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111101111111011
0111100100110010111100001001011011110010011001111110111101100110111001000110011011101011000100001101010101010110101110001011110110010001111000001110111111110110111100100110010111100 0
0100101101111001001100111111011110110011011100100011001101110101100010000110101010101011010111000101111011001000111100000011101111111101101111001001100101111000010010110111100100110 01
1111101111011001101110010001100110111010110001000011010101010110101110001011110110010001111000000111011111111011011110010011001011110000100101101111001001100111111011110110011011100
10001100110111010110001000011010101010101101011100010111101100100011110000001110111111110110111100100110010111100001001011011110010011001111110111101100110111001000110011011101011000
1000011010101010101101011100010111101100100011110000011101111111101101111001001100101111000010010110111100100110011111101111011001101110010001100110111010110001000011010101010101101011
10001011110110010001111000001110111111110110111100100110010111100001001011011110010011001111110111101100110111001000110011011101011000100001101010101011010111000101111011001000111 1
```

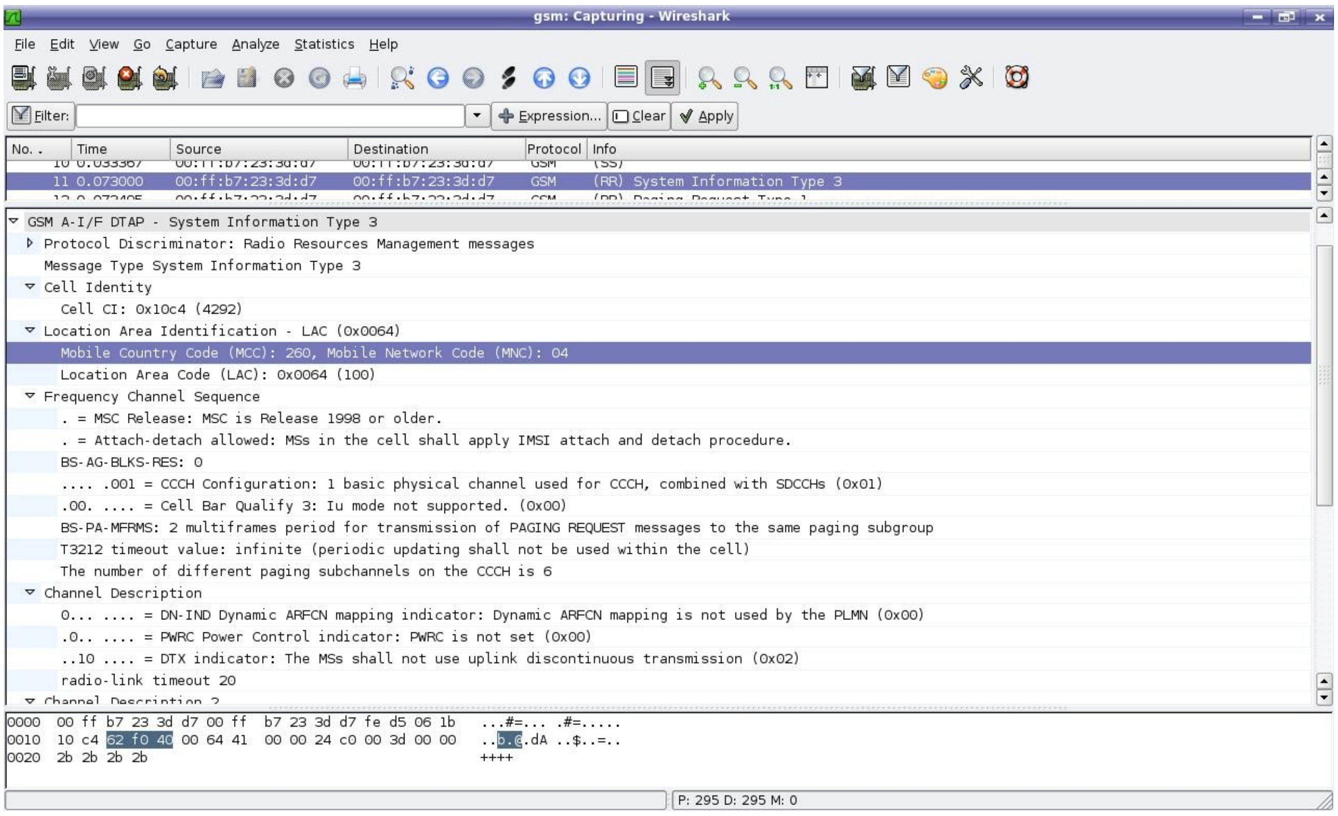
Figure 9 : GMSK-demodulated data

In the figure 9 you can see raw GMSK-demodulated data from single GSM channel, in which FCCH packet can be recognised because of its characteristic 142[1] consecutive 1's. Although FCCH packet consists of consecutive 0's (not 1's), data is seen as consecutive 1's because of differential encoding which should be dealt with (i.e. properly decoded) further in software.

This approach became a basis for first experiments and resulted in Joshua Lackey's software called GSSM[2] in which USRP board is seen in operating system as GSM network interface. The idea

1 There are even more consecutive 1's because of additional 3 trailing bits at the beginning and at the end of a packet.
2 Non-GPL licensed, but source code is available and can be used in educational purposes.

enabled network analysing tools with appropriate patches (definition of GSM packets) to present intercepted GSM packets sent from BTS.


gsm: Capturing - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression...
Clear
Apply

```
No. .  Time        Source              Destination         Protocol Info
   11 0.073000    00:ff:b7:23:3d:d7   00:ff:b7:23:3d:d7   GSM      (RR) System Information Type 3

GSM A-I/F DTAP - System Information Type 3
  Protocol Discriminator: Radio Resources Management messages
  Message Type System Information Type 3
  Cell Identity
    Cell CI: 0x10c4 (4292)
  Location Area Identification - LAC (0x0064)
    Mobile Country Code (MCC): 260, Mobile Network Code (MNC): 04
    Location Area Code (LAC): 0x0064 (100)
  Frequency Channel Sequence
    . = MSC Release: MSC is Release 1998 or older.
    . = Attach-detach allowed: MSs in the cell shall apply IMSI attach and detach procedure.
    BS-AG-BLKS-RES: 0
    .... .001 = CCCH Configuration: 1 basic physical channel used for CCCH, combined with SDCCHs (0x01)
    .00. .... = Cell Bar Qualify 3: Iu mode not supported. (0x00)
    BS-PA-MFRMS: 2 multiframes period for transmission of PAGING REQUEST messages to the same paging subgroup
    T3212 timeout value: infinite (periodic updating shall not be used within the cell)
    The number of different paging subchannels on the CCCH is 6
  Channel Description
    0... .... = DN-IND Dynamic ARFCN mapping indicator: Dynamic ARFCN mapping is not used by the PLMN (0x00)
    .0.. .... = PWRC Power Control indicator: PWRC is not set (0x00)
    ..10 .... = DTX indicator: The MSs shall not use uplink discontinuous transmission (0x02)
    radio-link timeout 20

0000  00 ff b7 23 3d d7 00 ff  b7 23 3d d7 fe d5 06 1b   ...#=... .#=.....
0010  10 c4 62 f0 40 00 64 41  00 00 24 c0 00 3d 00 00   ..b.@.dA ..$..=..
0020  2b 2b 2b 2b                                        ++++

P: 295 D: 295 M: 0
```

Figure 10 : Network analyser sniffing on virtual GSM interface

The figure above illustrates Wireshark (network analysing tool) with properly decoded System Information Type #3 packet. This includes Mobile Network Code (MNC) – unique value 04 was configured for experimental BTS in a laboratory at the Warsaw University of Technology. Among other fields we can see Mobile Country Code (MCC), Location Area Code (LAC) and Cell Identity (CI) values.

6. Summary

Current developments are focused on creating GPL-licensed version of software which will not only be able to apply error correction techniques thus limiting the number of invalid packets received, but will also interactively follow GSM traffic (frequencies, timeslots) based on information from received packets. Experiments, research and opinions suggest that the performance of USRP board is fast enough, so it can tune to different frequencies in short and adequate time, enabling interception of GSM packets also in communication where FH technique (frequency hopping) is used.

After this is confirmed, further work can include the areas of A5/1 cryptanalysis implementation [1] and experiments with active attacks on GSM infrastructure with the use of USRP transmit daughterboards.

REFERENCES

1: Elad Barkan, Eli Biham, Nathan Keller, Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication, 2003
2: BBC News, Greek boss at phone-tapping probe, 2006, http://news.bbc.co.uk/2/hi/europe/4788370.stm
3: BBC News, Greek government's phones tapped, 2006, http://news.bbc.co.uk/2/hi/europe/4674216.stm
4: Dawei Shen, The USRP Board, 2005, http://www.nd.edu/~jnl/sdr/docs/tutorials/4.pdf
5: Eric Blossom, How to Write a Signal Processing Block, 2006, http://www.gnu.org/software/gnuradio/doc/howto-write-a-block.html