VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ANALÝZA GSM SÍTĚ POMOCÍ OPEN SOURCE SW RÁDIA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

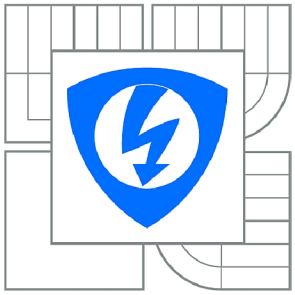
AUTOR PRÁCE
AUTHOR

Bc. PETR KILIAN

BRNO 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

ANALÝZA GSM SÍTĚ POMOCÍ OPEN SOURCE SW RÁDIA

ANALYSIS OF GSM NETWORK, USING OPEN SOURCE SW RADIO

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

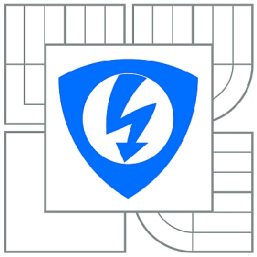
AUTOR PRÁCE — Bc. PETR KILIAN

AUTHOR

VEDOUCÍ PRÁCE — Ing. PAVEL ŠILHAVÝ, Ph.D.

SUPERVISOR

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student: Bc. Petr Kilian ***ID:*** 119476
Ročník: 2 ***Akademický rok:*** 2012/2013

NÁZEV TÉMATU:

Analýza GSM sítě pomocí open source SW rádia

POKYNY PRO VYPRACOVÁNÍ:

S pomocí open source projektů, vývojového rádiového KITu USRP1 a open source PBX realizujte testovací GSM síť. Nastudujte možnosti analýzy GSM sítě pomocí open source projektů. S pomocí vybraných open source projektů vytvořte aplikaci pro vyhodnocení parametrů testované GSM sítě. Věnujte se i na bezpečnost komunikace v GSM síti. V rámci diplomové práce rovněž vytvořte laboratorní úlohu.

DOPORUČENÁ LITERATURA:

[1] Clark, C. Software Defined Radio: with GNU Radio and USRP. McGraw-Hill Professional, New York. 2008, ISBN: 978-0071498838.
[2] Harte, L. Introduction to GSM: Radio Channels, Logical Channels, and Network Operation. Althos Publishing. 2005. ISBN: 978-1932813043.
[3] Bosse, J.G.. Signaling in telecommunication networks. John Wiley & Sons, Ltd. En-gland 2002 , ISBN 0-471-66288-7.

Termín zadání: 11.2.2013 ***Termín odevzdání:*** 29.5.2013

Vedoucí práce: Ing. Pavel Šilhavý, Ph.D.
Konzultanti diplomové práce:

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Tato práce se zabývá budováním a analýzami GSM sítí pomocí open-source projektů. Pro budování se používá softwarově definované rádio – zde kit USRP1, na kterém je pomocí GNU Radia zprovozněn software OpenBTS. Jako ústředna je použita PBX Asterisk. Pro analýzy testované sítě byly použity projekty OsmocomBB (na Motorole C123) a Airprobe (na Nokii 3310). Součástí jsou postupy instalace a realizace. V práci byly vytvořeny dva skripty na zjednodušení práce s projekty pro analýzy a laboratorní úloha.

KLÍČOVÁ SLOVA

GSM, USRP, open source, OpenBTS, GNURadio, analýza, Osmocom, Airprobe, gammu, Nokia, Motorola

ABSTRACT

This master thesis is focused on analysis and creation of GSM network, using open-source projects. For such creation is used software defined radio – in this thesis kit USRP1 which uses OpenBTS with GNURadio. As a switchboard is used PBX Asterisk and for testing GSM network was used OsmocomBB (on Motorola C123) and Airprobe (on Nokia 3310) projects. Installation and implementation are part of this thesis. In thesis have been also created two scripts for simplify of the work with projects and one laboratory task.

KEYWORDS

GSM, USRP, open source, OpenBTS, GNURadio, analysis, Osmocom, Airprobe, gammu, Nokia, Motorola

KILIAN, P. *Analýza GSM sítě pomocí open source SW rádia.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2013. 75 s. Vedoucí diplomové práce Ing. Pavel Šilhavý, Ph.D.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma „Analýza GSM sítě pomocí open-source SW rádia“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následku porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonu (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne ....

podpis autora

PODĚKOVÁNÍ

Děkuji vedoucímu své diplomové práce Ing. Pavlu Šilhavému, Ph.D. za velmi užitečnou pomoc a metodické vedení při zpracovávání této práce.

V Brně dne ....

(podpis autora)

SIX research centre
sensor, information and communication systems

Faculty of Electrical Engineering
and Communication

Brno University of Technology
Purkynova 118, CZ-61200 Brno, Czechia

http://www.six.feec.vutbr.cz

Výzkum popsaný v této habilitační práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

2007-13
OP Výzkum a vývoj pro inovace

OBSAH

ÚVOD

V posledních letech se objevuje čím dál více aktivit spojených s upozorňováním na problémy systému GSM, který byl do jisté doby považovaného za dokonalý. Když se pak k těmto aktivitám přidávaly další projekty, které měli za úkol například snížit náklady současného hardwarového vybavení – potřebného pro provoz GSM, vznikly nové možnosti. Možnosti testovat, realizovat a analyzovat vlastní sítě GSM za zlomek původní ceny. Později pak bylo možné takovéto sítě provozovat tam, kde není dostatek financí. Nyní takovéto projekty odhalují nejen problémy GSM systému, ale začínají se orientovat na mnohem novější a modernější systémy jako UMTS, či LTE. Cílem takovýchto realizací je zamezit pozdnímu odhalení bezpečnostních problémů.

Tato práce využívá kit USRP1 k realizaci sítě. Pro propojování hovorů je zvolena PBX Asterisk. Jako ovladač kitu slouží software UHD. Ten využívá GNURadio k softwarovému vytváření specifických částí rádiového zařízení. Stěžejním projektem je pak software OpenBTS, který ve schématu prezentuje základnovou stanici s řídící jednotkou.

Pro open-source analýzy byly zvoleny projekty OsmocomBB a Airprobe. Limitace na hardwarové vybavení u Airprobe však nedovolila tento projekt plně využít. U aplikací projektu OsmocomBB se pracuje se složitými výrazy – proto je v práci vytvořen skript pro zjednodušení práce. U Airporbe je vytvořen skript, který postup automatizuje a z dat získaných během hovoru vypíše hodnoty úrovně signálu a kvality přenosu.

V rámci navržené laboratorní úlohy si studenti vyzkouší vybudovat svou GSM síť, kterou budou pomocí OsmocomBB analyzovat.

Celá práce je doplněna ukázkami.

1 SYSTÉM GSM

Global System for Mobile Communications (GSM) je jedním z nejpoužívanějších systémů pro přenos hlasu a částečně i dat. Použitím buněk dokáže tato digitální radiotelefonní síť pokrýt rozsáhlé území za použití relativně malého množství kanálů. Počátky systému GSM se datují k roku 1982, kdy tento koncept začala vyvíjet a standardizovat Conference of European Posts and Telecommunications (CEPT). Prvního komerčního nasazení se tato síť dočkala v roce 1991 a dočkala se takového rozmachu, že za méně než třináct let od svého nasazení bylo na světě evidována miliarda GSM zařízení ve 205 zemích a oblastech po celém světě. Po nasazení systému GSM probíhal neustále jeho vývoj – byly například rozšířeny a zavedeny funkce přenosu zpráv, ale postupem času vznikal požadavek na zvýšení přenosové rychlosti přenosu dat. Proto byly zavedeny nové služby (jako např. GPRS), které začali k GSM neodmyslitelně patřit [1, 2].

1.1 GSM technologie

Pro svou funkci využívá GSM několik frekvenčních pásem. Původní GSM systém (označovaný jako GSM 900) je rozdělen do dvou pásem o šířce 25 MHz. První pásmo 890–915 MHz je určeno pro vysílání ze strany mobilního zařízení a druhé 935–960 MHz pro příjem signálu od základnové stanice. Takto je k dispozici 125 kanálů o šířce 200 kHz. Druhým v Evropě hojně používaným pásmem je GSM 1800. S šířkami pásem 1710–1785 MHz pro vysílání mobilní stanice a 1805–1880 MHz pro příjem. Kanálů o šířce 200 kHz pak v tomto pásmu existuje 375. Mezi další kmitočtová pásma patří GSM 1900, GSM 850 a GSM 400.

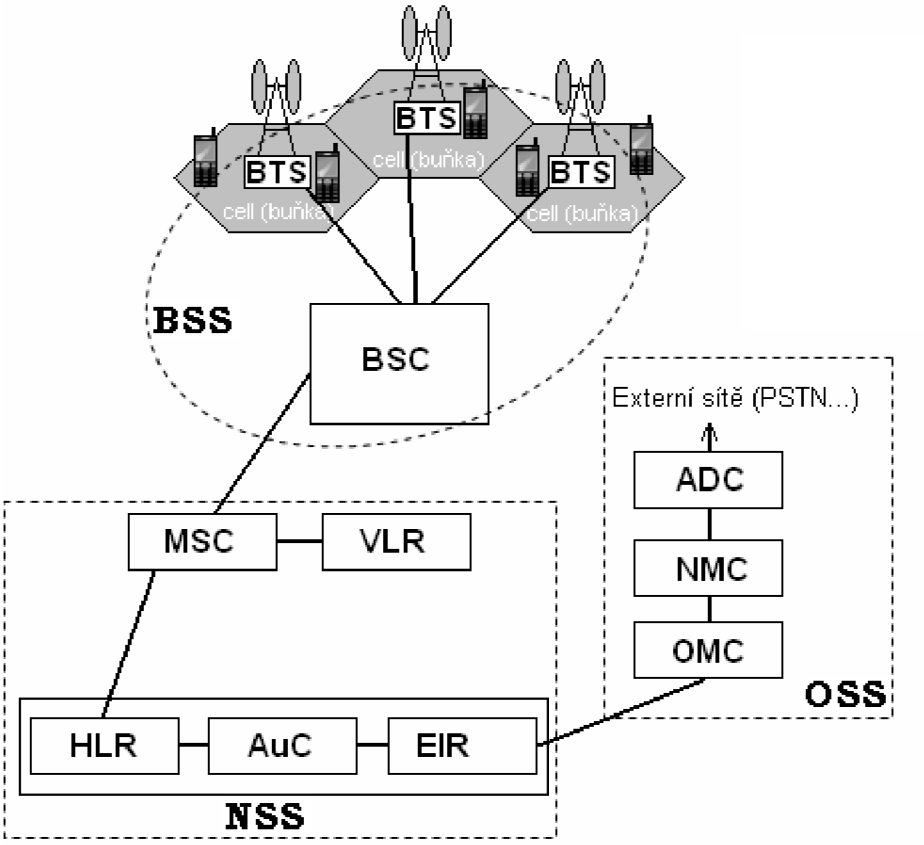
Mimo frekvenční dělení využívá systém GSM metodu vícenásobného přístupu s časovým dělením (Time Division Multiple Access – TDMA). Ta dovoluje sdílet radiový kanál více uživatelům. V radiovém kanálu je každému uživateli přidělen tzv. time-slot, který v přidělenou dobu může využívat pouze jeden účastník. Z takovýchto time-slotů se vytvoří pravidelně se opakující rámec. Systém GSM má v jednom rámci osm time-slotů. To znamená, že na jednom rádiovém kanálu může teoreticky současně probíhat 8 nebo 16 (při poloviční rychlosti) hovorů.

Při pohybu stanice je nutné přepínat mezi buňkami resp. jejich kanály. Tato činnost se nazývá handover. Jednotlivé buňky však mají různý průměr podle předpokládané hustoty provozu. Proto je pro handover důležitá kvalita všech dostupných kanálů, přičemž je nedílnou součástí jejich vyhodnocení. V GSM probíhá měření signálů mobilní stanicí. Výsledky jsou předány síti, která po vyhodnocení rozhoduje o přepnutí spojení.[1, 2].

1.2 Architektura sítě

GSM síť je rozdělena do jednotlivých bloků podle své činnosti. Tyto bloky je možné předem rozdělit do tří subsystémů:

- Subsystém základnových stanic (BSS)
- Síťový a přepojovací subsystém (NSS)
- Operační subsystém (OSS)



Obr. 1.1: Architektura GSM sítě [2]

Subsystém základnových stanic se skládá z jedné nebo více základnových stanic BTS na jedinou řídící jednotku BSC. Takto může být schéma obsaženo v celé síti vícekrát. BTS má za úkol zprostředkovat rádiové spojení s mobilními stanicemi, přičemž se využívá rádiové rozhraní nazývající se U_m. BSC řídí provoz svých BTS a má na starosti např. handover, přidělování rádiových kanálů atd. Pro BTS a jejich správnou funkci jsou též důležité obvody kmitočtové a časové reference pro vytváření synchronizačních kanálů.

Síťový a přepojovací (spínací) subsystém NSS se zabývá přepojováním hovorů a řízením komunikace v rámci mobilní sítě a dále i účastníky dalších telekomunikačních sítí. Mobilních rádiových ústředen MSC se v celé síti může nalézat více (např. jedna v rámci města). Právě tato část architektury sítě GSM zajišťuje všechny spínací funkce. Veškerá potřebná data jsou uložena v databázích návštěvnického (VLR) a domovského lokačního registru (HLR). V HLR se nacházejí všechny údaje u účastnicích patřících do sítě, včetně služeb. Pro autentizaci účastníků je pak využíván blok AuC – centrum autentičnosti. V návštěvnickém lokačním registru se nachází aktuální data o účastnicích kteří se nachází v příslušné oblasti jednotky MSC. Posledním blokem v NSS je registr mobilních stanic EIR, v němž jsou uloženy záznamy o odcizených či neoprávněně užívaných mobilních stanicích.

Operační systém OSS se zabývá údržbou hardwaru, řízením provozu a monitorováním sítě. Jednotlivými bloky jsou administrativní centrum ADC, centrum managementu sítě NMC a provozní a servisní centrum. Odtud je možné vést napojení k dalším sítím [2].

1.3 Kanály GSM systému

V systému GSM probíhá přenos informací po tzv. kanálech. V první řadě jsou rozlišovány dva druhy kanálů – logické a fyzické. Fyzickými kanály se rozumí přenosové kanály s číslem timeslotu a rádiového kanálu, přičemž jsou do nich vkládány kanály logické. Toto vkládání se nazývá mapování logických kanálů do fyzických. Jednotlivé logické kanály jsou časově odděleny .

Logické kanály jsou dvojího typu. Prvním jsou provozní kanály (TCH – Traffic Channel), které slouží k přenosu hovorových nebo uživatelských dat. Tyto kanály je možno dále rozdělit na TCH/F (Full Rate) nebo TCH/H (Half Rate), kdy F značí plnou a H poloviční rychlost.

Druhým typem logických kanálů jsou signalizační (řídící) kanály SC (Signal Channel). Jak název napovídá jsou určeny k přenosu signalizace a řízení sítě. Tyto kanály se dále dělí na 3 podskupiny:

- Vysílané řídící kanály BCCH (Broadcast Control Channel) – pro předávání údajů ze sítě (od BTS) k mobilním stanicím
- Společné řídící kanály CCCH (Common Control Channel) – obousměrné kanály pro vzájemnou signalizaci při vstupu MS do sítě a při sestavení hovoru
- Vyhrazené řídící kanály DCCH (Dedicated Control Channel) – obousměrné kanály k předávání zpráv a informací mezi MS a BTS

Pomocí vysílaných řídících kanálů (BCCH) zásobuje BTS mobilní stanici údaji pro registraci do systému jako je číslo kanálu, číslo sekvence, kód identifikující síť, korekce výkonu MS atd. BCCH dále obsahuje synchronizační kanál SCH (Synchronization Channel) pro časovou synchronizaci a frekvenčně korekční kanál FCCH (Frequency Correction Channel) pro kmitočtovou korekci MS.

Mezi společné řídící kanály (CCCH) patří paging kanál PCH (Paging Channel), který oznamuje mobilní stanici příchozí hovor. Druhým kanálem této podskupiny je kanál náhodného přístupu RACH (Random Access Channel) sloužící MS k žádosti o přidělení kanálu pro účely hovoru. Třetím je kanál pro řízení přístupu AGCH (Access Grant Control Channel), kterým je mobilní stanice informována o čísle kanálu pro spojení, tedy přidělení TCH.

Jedním vyhrazených řídících kanálů (DCCH) je obousměrný samostatný jednoúčelový kanál SDCCH (Stand-alone Dedicated Channel), který přenáší signalizaci a vzájemnou komunikaci mezi MS a sítí před přidělením TCH. Dalším kanálem je pomalý sdružený řídící kanál SACCH (Slow Associated Control Channel) sloužící pro údržbu a kontrolu, zejména pro přenos informací o parametrech rádiového signálu změřeného MS, jež jsou potřeba k případnému handoveru. Posledním kanálem této podskupiny je tzv. přepadový kanál FACCH, který používá se je-li nutné zaslat důležitou zprávu – např. o potřebě handoveru v průběhu hovoru [2, 23].

1.4 Mobilní stanice a její identifikace

Součástí architektury sítě GSM je též mobilní stanice, která umožňuje využití a připojení do této sítě. Skládá se z vysokofrekvenční části, obvodů pro zpracování signálu, mikroprocesorových obvodů a pamětí, napájecího zdroje atd. Jednou s nejdůležitějších částí

každé mobilní stanice je Subscriber Identity Module známý jak SIM karta, bez které není možné komunikovat se sítí s výjimkou tísňového volání. SIM karta obsahuje mimo jiné údaje, které jednoznačně identifikují účastníka.

Tímto údajem je IMSI (International Mobile Subscriber Identity), které účastníka identifikuje na mezinárodní úrovni. Celé IMSI může obsahovat až 15 číslic, kdy první 3 číslice určují mezinárodní příslušnost SIM karty a další 3 určují operátora, avšak takto by bylo nutné odesílat celé číslo při každé žádosti o službu. Proto se využívá TMSI (Temporary Mobile Station Identity). Toto číslo je přidělováno účastníkům na určitém území a zároveň je uloženo ve VLR databázi. Při přesunu na jiné území je mobilní stanici přiděleno TMSI nové, které si mobilní stanice na SIM kartě přepíše. Úvodní přidělení TMSI, např. při zapnutí stanice, probíhá na základě IMSI.

Identifikovat stanici je však také možné přes číslo International Mobile Equipment Identifier (IMEI), které je uloženo v paměti mobilního zařízení. Toto číslo zůstává stejné i při změně SIM karty. Tím je možno zajistit při odcizení mobilního zařízení jeho nefunkčnost nebo pouze jeho identifikaci.

BTS rozlišují pro identifikaci BSIC (Base Station Identity Code) – pro jedinečnou identifikaci základnové stanice. Dále cell ID pro identifikaci buňky. To vše je ještě obaleno identifikací lokace pomocí LAC (Location Area Code) [1, 2]

1.5 Bezpečnost GSM

Nutnost zabezpečení je u sytému GSM jasná. Jedná se o bezdrátovou síť, tudíž je k ní možno přistupovat odkudkoliv. V GSM jsou tak pro zabezpečení informací použity čtyři základní způsoby zabezpečení a to skrz: použití SIM karty, anonymitu díky TMSI, ověření totožnosti a šifrování signalizačních a hovorových dat.

Pro ověření totožnosti se používá algoritmus nazvaný A3. Důležitou úlohu zde hraje AuC. V ní probíhá generace náhodného čísla a poté i přepočet a porovnání výsledků odezvy. Generované náhodné číslo (RAND) je odesláno mobilní stanici, která na základě znalostí tajného individuálního klíče *Ki* a algoritmu provede výpočet a výsledek zašle zpět do AuC.

Pro šifrování a dešifrování dat, hovoru a signalizace se používá algoritmus A5, který je normalizovaný pro všechny sítě GSM. Pro tento algoritmus se využívá šifrovací klíč *Kc* vypočítávaný algoritmem A8 opět z tajného individuálního klíče *Ki*. [2, 3]

Ki se v síti nepřenáší, protože je uložen jak v AuC tak na SIM. Na SIM kartě je zabezpečeno omezení manipulace s tímto klíčem tak, aby tuto hodnotu nebylo možné jednoduše zjistit. Postupem času však vyšly na povrch metody jak se k hodnotě *Ki* dostat. Jednou z nich je získání její hodnoty přímo ze SIM karty, kdy pomocí dané SIM karty a speciálního zařízení je možné analyzovat odezvu SIM na různé RAND. Při správném postupu pak stačí cca 180 000 dotazů ke zjištění tajného klíče *Ki*.

Druhou metodou je odposlech hovoru, kdy se útočí na algoritmus A5. Jeho provedení je ve dvou verzích a to slabší A5/2 a silnější (použitá i u nás) A5/1. Verzi A5/2 se podařilo prolomit dříve a tento útok je popsán např. Ianem Goldbergem z University of Californa at Berkley. U A5/1 se útok jevil jako výpočetně náročný. Nakonec však vznikly dvě varianty popsané v dokumentu „Real Time Cryptoanalysis of A5/1 on a PC.“ U první varianty útoku stačí zachytit úvodní dvě minuty hovoru, přičemž následný výpočet *Ki* trvá v řádu sekund. Druhou variantou je odposlech úvodních dvou sekund. Poté je výpočet náročnější a však jeho délka je počítána v minutách [3].

2 KONCEPCE SOFTWAROVÉHO RÁDIA PRO GSM

Softwarově definované rádio je čím dál více se rozvíjející moderní technologií. Jeho výhoda, resp. základní myšlenka tkví v tom, že hlavní funkce rádiové přijímače a vysílače jsou provedeny hardwarem, který je řízen a doplňován softwarovým vybavením. Softwarem lze tak např. nastavovat frekvence, aktivovat (de)modulátory, filtry, komunikační protokoly atd. Tím vzniká velmi široce nastavitelné rádiové zařízení, které je možné téměř okamžitě rekonfigurovat dle aktuálních požadavků. Významný podíl použitelnosti softwarového rádia zaujímají mobilní buňkové systémy jako GSM, EDGE, UMTS [4, 5]. Mezi projekty či výrobce zabývající se výrobou kitů softwarového rádia patří např. Perseus SDR, WINRADiO, umtrx, Sora a další. V této práci je využito jednoho z vývojových kitů od Ettus Research – USRP1 (Universal Software Radio Peripheral).

2.1 Kit USRP1

Standardním vybavením USRP1 jsou 4 vysokorychlostní 12-ti bitové analogově-digitální konvertory s rychlostí 64MS/s (milionu vzorků za sekundu); dále 4 vysokorychlostní převodníky digitálně-analogové se 14 bity na vzorek dosahující rychlost 128MS/s. Veškeré vstupy a výstupy jsou připojeny k programovatelnému hradlovému poli (FPGA) Cyclone od firmy Altera. Aby mohl celý kit komunikovat s počítačem, je zde provedeno připojení přes USB port. Nutností však u tohoto kitu je připojení k USB verze 2.0. Na nižších verzích připojení nefunguje.

Vývojový kit USRP1 je možno doplnit o tzv. Daughterboards. Ty se volí podle požadavků na rádiový rozsah vytvářené aplikace. Vezme-li se v potaz, že základní deska kitu má čtyři vstupní a čtyři výstupní kanály (dle převodníků), je možné z každého páru převodníků vytvořit dva komplexní vstupy (přijímače) a 2 komplexní výstupy pro vysílání [6, 7].

Pro použití vývojového kitu USRP1 pro GSM technologii, jej bylo nutné doplnit o dvě daughterboards transceiveru RFX900, laditelné od 800MHz do 1GHz a s výstupním výkonem 200mW. Ty jsou také dále doplněny o antény VERT900. Pro dobrou funkci sítě musel být USRP1 doplněn o 52 MHz generátor hodinového impulsu. Původní 64MHz generátor, který je v kitu USRP1 obsažen, není vzhledem k náročnosti na přesnost hodinového signálu v GSM síti příliš vhodný. Mobilní stanice měli při původní konfiguraci s připojením, či vůbec rozpoznáním sítě, velký problém.

Mimo dvě daughterboards RFX900 byla k dispozici ještě deska WBX. Ta dokáže fungovat na frekvencích od 50 MHz do 2,2 GHz s výstupním výkonem okolo 100 mW. To zajišťuje možnost posunout experiment s GSM do jiného frekvenčního pásma. O důvodech použití je pojednáno později (v kapitole 3: Realizace).

2.2 Software GNU Radio

Z filozofie softwarového rádia je jasné použití ovladače hardwaru USRP kitu (zkr. UHD) a také softwaru, který zjednodušuje práci se s rádiem. Pro rodinu USRP zařízení firmy Ettus lze s UHD ovladačem, který je pro všechny zařízení rodiny stejný, využít LabVIEW,

Matlab se Simulinkem, případně používat UHD samostatně nebo (jako u této práce) využít toolkit GNU Radio. Použití GNU Radio a UHD je nejčastější volbou.

GNU Radio je open-source softwarové řešení poskytující signálové procesní bloky pro softwarová rádia. Dokáže fungovat jak na operačních systémech Windows tak Mac, avšak největší výkonnost s nejmenšími problémy zajišťují distribuce Linuxu. I když je GNURadio nejčastěji používáno na zařízení USRP od Ettus, dá se dále použít pro zařízení z projektů Perseus, Comedi nebo k testům na zvukové kartě.

GNU Radio obsahuje softwarové řešení filtrů, kanálového kódování, synchronizační prvky, ekvalizéry, de/modulátory, vocodéry, de/kodéry a taky spoustu dalších prvků, které se v rádiových systémech vyskytují. Výhodou softwaru je práce s bloky – jednotlivými prvky, které je možné mezi sebou propojovat. Tak je možné vytvořit konkrétní blok a přidat jej. Mezi bloky se pak mohou předávat data ať už v bitech, bajtech, vektorech, či složitějších datových typech.

Kvůli výkonu je software GNU Radio, či spíše jeho výpočetně náročné části, realizován v C++. Je tak možné využívat i systémy v reálném čase. Větší část aplikace však využívá programovací jazyk Python.

U tohoto softwaru není nutná znalost programování. V první řadě je možné použití grafického prostředí GNU Radio Companion, které se v zásadě podobá prostředí Simulink. Dále jsou také předpřipraveny různé nástroje a programy, které provádějí různé akce se signály. Takovýmto nástrojem je třeba uhd_fft, fungující jako jednoduchý spektrální analyzátor. [8]

2.3 OpenBTS

OpenBTS (Open Base Transceiver Station) je Unixovou aplikací užívající softwarové rádio pro vytváření GSM bezdrátové sítě, respektive jejího přístupového bodu. Síť je provedena jako klasická síť druhé generace s možností použití klasických telefonů. Pro spojování jednotlivých hovorů je nutné použití přídavného SIP softswitche nebo pobočkové ústředny PBX (Private branch exchange), které v GSM architektuře zastupují blok MSC. Přes tyto ústředny je pak také možné spojovat volání do rozsáhlejších externích sítí. Díky zavedeným standardům GSM technologie tak může být vytvořena VoIP (Voice over Internet Protocol) síť, která má mnohem nižší provozní i pořizovací náklady a má široké možnosti použití. Celá aplikace je napsána v C++ a platí na ní otevřená licence Affero General Public License (AGPL) verze 3 [9, 10]. V této práci bude využíváno OpenBTS poslední verze, tedy 2.8 s aktuálním kódem.

Největším přínosem projektu OpenBTS bylo odhalení bezpečnostní chyby systému GSM, kdy se ukázalo, že mobilní stanice před přístupem do sítě neověřuje BTS. Možnost využití této slabiny však byla před vyvinutím OpenBTS považována za nerealizovatelnou, vzhledem k vysokým pořizovacím cenám BTS [10, 11].

Softwarové řešení OpenBTS zastupuje funkce rozhraní U_m v GSM architektuře. Ta zhruba odpovídá vrstvám OSI modelu, takže rozhraní U_m (a tím pádem i SW OpenBTS) tvoří tři jeho nejnižší vrstvy: nejnižší je fyzická, pak linková (spojová) a síťová.

Třetí - síťová vrstva obstarává tři podvrstvy, s nimiž musí účastník navázat spojení postupně od nejnižší k nejvyšší podvrstvě. Pokud však spojení opouští, postupuje od nejvyšší k nejnižší vrstvě. První podvrstvou je Radio Ressource řídící přiřazování a uvolňování logických kanálů na rádiovém spojení a bývá ukončen v BSC. Druhou podvrstvou je Mobility

managment ověřující uživatele a sledující jejich pohyb, přičemž je spojen s VLR nebo HLR registry. Třetí je Call Control, která má propojovat telefonní hovory. V OpenBTS však tato podvrstva není. K takovému úkolu využívá Softswitche či PBX. A tak jako standardní GSM si vystačí pouze se dvěmi podvrstvami.

Na linkové vrstvě zajišťuje veškerou činnost protokol LAPDm. Ten vychází ze staršího protokolu HDLC avšak je zjednodušen a poupraven.

2.3.1 Fyzická vrstva Um GSM / OpenBTS

Na fyzické vrstvě U_m rozhraní GSM standardu se řeší problematika ve třech podvrstvách: radiomodemu, mutiplexování a časování a také kódování.

OpenBTS P2.8 podporuje na úrovni radiomodemu standardně GMSK modulaci se šířkou kanálu 200 MHz. Součástí je také podpora pásem GSM850, PGSM900 (EGSM900), DCS1800, PCS1900. Používají se frekvenční pár - zvlášť pro uplink (do sítě) i pro downlink (do stanice) – pro přenos ve stejném čase. Tyto frekvence mají od sebe odstup v závislosti na zvoleném kanále, respektive indexu značeném ARFCN (Absolute radio-frequency channel number). Hodnoty tohoto indexu a přidělené páry frekvencí radiového kanálu jsou definovány ve standardu GSM. Nutno ještě poznamenat, že každý radiový kanál je časově multiplexován do osmi pravidelně se opakujících timeslotů.

V rámci multiplexování a časování je těchto osm timeslotů zařazeno do takzvaného TDMA rámce. Tyto rámce, tedy je-li jich 26, se dále spojují do multirámce. Multirámec vzniká i spojením 51 TDMA rámců, avšak pouze v případě, že se jedná o signalizaci. O časování se v GSM architektuře stará řídící BTS pomocí tzv. logických kanálů. Všechny hodiny v mobilní stanici se pak „naladí“ podle přijatých signálů od BTS.

Poslední problematikou fyzické vrstvy u rozhraní U_m GSM standardu je kódování – zabezpečení proti chybám, jež poskytuje dopřednou korekci chyb. Obecně využívá každý GSM kanál paritní bity, konvoluční ½ kódování a čtyř nebo osmi burstové prokládání. Díky použití těchto technik a následného Viterbiho dekódování při příjmu je možné obnovit rámce poškozené s 25% chybou [12].

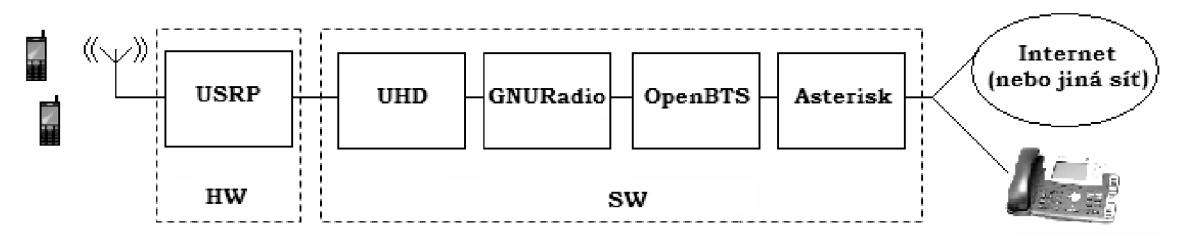
2.4 Asterisk

Asterisk je výkonné, flexibilní a rozšiřitelné řešení v oblasti telekomunikačního softwaru. Poskytuje kompletní řešení softwarové PBX ve formě open source projektu, který běží na platformách Linux a Unix. Má široké možnosti použití a to například: VoIP gateway, server interaktivního hlasového průvodce, softwarová ústředna, pobočková ústředna, překlad čísel a další. Podporuje široké spektrum kodeků (μ-law, A-law, Speex, GSM atd.), zahrnuje VoIP služby (SIP, H.323, IAX a MGCP) i tradiční TDM technologie (T1, ISDN, analogové POTS a PSTN a další). Jeho architektura je rozdělena na jádro a zaváděné moduly API (Application Programming Interface), čímž je docíleno zefektivnění výkonnosti [14].

Pro OpenBTS je Asterisk nejpoužívanějším a nejvíce podporovaným řešením. Při takovéto aplikaci GSM se využívá služby SIP. Každá mobilní stanice je tak registrována jako SIP uživatel, který se identifikuje pomocí IMSI SIM karty. Každé IMSI pak musí být přidáno do konfiguračních souborů Asterisku [13].

3 REALIZACE TESTOVACÍ GSM SÍTĚ

Pro realizaci testovací sítě na vývojovém kitu USRP je, kromě něj, potřeba počítače a mobilních stanic se SIM kartami. Jak jednotlivé projekty realizace navazují na sebe popisuje Obr. 3.1.



Obr. 3.1: Návaznost projektů při realizaci GSM

Minimální hardwarová výbava počítače není nijak blíže specifikovaná, pouze jediný požadavek je na USB port verze 2.0, pro rozhraní přístupu k USB. Softwarově je v tomto projektu počítač vybaven operačním systémem Linux v distribuci Ubuntu, ovladačem UHD a softwary GNU Radio, OpenBTS a Asterisk. K jednotlivým částem – jejich instalaci a nastavení – jsou věnovány samostatné podkapitoly.

Díky přidání generátoru hodinového impulsů v kitu z 64 MHz na 52 MHz neexistuje žádný speciální požadavek na mobilní stanice. Je možno použít jakoukoliv s podporou GSM, teoreticky i právě připojenou do jiné sítě. Pro práci je však nutné mít v této stanici SIM kartu, která má přidělené IMSI.

Veškerou instalací se prolínal problém s již starším kitem USRP1. Ten přestává být postupem času podporován a je snaha jej nahradit novějším USRP2. V případě volby verze operačního systému byal dána přednost starší (ještě podporované) verzi - Ubuntu 10.04 Lucid. Tato verze je totiž dobře otestována pro celou konfiguraci.

Důležitým aspektem při realizaci práce je fakt, že na vysílání v daných pásmech GSM není uděleno oprávnění od Českého telekomunikačního úřadu (ČTÚ). Původně bylo pro vysílání použito původních desek RFX900 na pásmo GSM 900 MHz. Při realizaci bylo samozřejmě důkladně proměřováno a zjišťováno, zda na daném kanálu nemůže dojít k rušení cizích BTS. Problém v pásmu GSM 900 je ten, že zde není žádný „neobsazený“ kanál. Proto byla zvolena daughterboard WBX. V pásmu GSM 1900 (PCS) by však pro Evropu určené telefony měly problém s nalezením sítě.

Pásmo GSM 1800 (DCS) má dle gsmweb.cz několik nepřiřazených kanálů. Měřením bylo zjištěno, že na nepřiřazených kanálech 512 – 517 (eventuelně 571 – 580) je v oblasti laboratoře skutečně rádiový klid. Proto bylo pro další měření a analýzu použito kanálů z tohoto rozsahu.

3.1 Instalace GNU Radio (ver. 3.4.2)

Instalace softwaru GNU Radio má několik možných způsobů. Problémem však je to, že se i se sebelepším návodem může vyskytnout spousta nečekaných chyb. GNU Radio využívá ke své funkci velké množství knihoven. U tohoto projektu byla instalace (nejen GNU Radia) prováděna hned několikrát. Poznatkem z tohoto je fakt, že i návod vystavený přímo pro danou

verzi i distribuci operačního systému na stránkách GNU Radia [8] neobsahoval všechny potřebné knihovny. Nejedná se však jen o knihovny, ale i podpůrné programy i nastavení kitu nebo instalace UHD. Ovšem tento způsob je možný i když dle mého názoru nejvíce zrádný. Pro instalaci tímto způsobem je možné využít i kvalitně zpracované návody indonéského internetového poskytovatele TelekomSpeedy, viz [15]. Není ani velkou překážkou, že tyto návody jsou v Indonéštině.

Tím asi nejjednodušším způsobem, který je zmíněn na stránkách projektu GNU Radio je použít pre-kompilovaný binární balíček pro Ubuntu resp. Fedoru. Ten je pouze zaveden do instalace přes *„apt-get install gnuradio“* resp. *„yum install gnuradio.“* Problémem však je nainstalovaná verze GNU Radia - 3.2.2. Tato verze je z roku 2009 a za tu dobu vývoj došel až k současné verzi 3.6.2.

Nejschůdnějším řešením, který je doporučován i na stránkách projektu, je použití kompilačního skriptu od Marcuse Leecha. Odkaz na stažení skriptu je na webu GNU Radia. Tento skript je použitelný pro linux distribuce Fedora, Ubuntu, Redhat, Debian různých vývojích verzí. Provede veškerá nastavení a doplnění balíčků. Nejnovější verze softwaru (GNU Radio) i ovladače (UHD) si přes Git (systém správy verzí) stahuje skript sám. V současnosti (prosinec 2012) je aktuální verzí GNU Radia 3.6.2. Bylo již zmíněno, že kit USRP1 je starší a doprovází jej problémy. V tomto případě je problém z ovladačem *libusrp*, který GNU Radio pro USRP1 využívá. Podpora *libusrp* byla zrušena od verze 3.5. To znamená, že poslední verzí GNU Radia, kterou lze použít, je verze 3.4.2.

V kompilačním skriptu však není nijak možné přenastavení na požadovanou, či již staženou verzi. Tím ovšem nemusí být skript „odepsán.“ Stačí po dokončení instalace skriptem stáhnout a nainstalovat onu požadovanou verzi – v tomto případě 3.4.2. I tak by mělo použití skriptu instalaci ulehčit. Takovýto postup byl volen i v této práci.

Je tedy třeba začít kompilačním skriptem. Ve spuštěném terminálu Ubuntu je vhodné se přemístit do své (oblíbené či nové) složky. Následuje stažení jako spustitelný skript a spuštění. Při spouštění je dobré nastavit „verbose,“ aby se instalace stala čitelnou:

```
~$ wget http://www.sbrac.org/files/build-gnuradio && chmod a+x
   ./build-gnuradio
~$ ./build-gnuradio -v
```

Ihned po spuštění se kompilační skript dotáže na pokračování instalace a dále na vlastnictví SUDO privilegií. Obě je nutno potvrdit s např.„y“ Po doplnění hesla již bude kompilační skript pracovat zcela sám. Celý tento proces zabírá zhruba půl až tři čtvrtě hodiny času a je zakončen hláškou: „All Done“

Nejnovější verze GNU Radio je i s ovladači a knihovnami nainstalována. Nyní je nutné tuto verzi přeinstalovat na starší. Veškeré knihovny a nastavení jsou již hotová a reinstalací se nic nezničí. Tudíž se teď stáhne a rozbalí požadovaná verze:

```
~$ wget http://gnuradio.org/redmine/attachments/download/279
   /gnuradio-3.4.2.tar.gz
~$ tar -xf gnuradio-3.4.2.tar.gz
```

Jak bylo již zmíněno, tento USRP kit používá externí hodinový oscilátor a tím pádem je změněna frekvence (z 64 na 52 MHz). Z tohoto důvodu je dle [15] doporučeno upravení kódu ve dvou souborech, které jsou s hodinovým taktem spojeny. Pro takovouto změnu je možno

použít oblíbený textový editor – zde je to gedit (je třeba jej doinstalovat). V prvním souboru (předpokladem je, že se nacházím ve stejné složce, kde byl soubor rozbalen):

```
~$  gedit gnuradio-3.4.2/usrp/host/lib/usrp_basic.cc
```

se na řádku 110 nachází:

```
d_verbose (false), d_fpga_master_clock_freq(64000000), d_db(2)
```

které je třeba opravit na:

```
d_verbose (false), d_fpga_master_clock_freq(52000000), d_db(2)
```

Podobným způsobem je také nutné ještě přepsat soubor:

```
~$  gnuradio-3.4.2/usrp/host/lib/db_bitshark_rx.cc
```

kde je ještě přeopraven řádek 151 na hodnotu:

```
set_clock_scheme(0,52000000);
```

Takto už nic nebrání nainstalovat GNU Radio požadované verze. Je možné, že bude při instalaci vyžadována nějaký balíček (knihovna), ale systém vše nahlásí. Při této instalaci chyběl jen balíček sdcc (doinstalován pomocí *„apt-get install sdcc“*). Následující postupně zadané příkazy provádí přesun do složky, konfiguraci, kompilaci a instalaci celého projektu:

```
~$  cd gnuradio-3.4.2
~$  ./configure --disable-usrp2 --enable-usrp
~$  make all
~$  make
~$  make check
~$  make install
~$  sudo ldconfig
```

Software je nainstalován. Zbývá upravit cestu k Pythonu v souboru:

```
~$  gedit ~/.bashrc
```

a do něj přidat:

```
PYTHONPATH = / usr/local/lib/python2.7/dist-packages
PKG_CONFIG_PATH = / usr / local / lib / pkgconfig /
```

Je vhodné tento krok ještě potvrdit v systému zadáním:

```
~$  sudo ldconfig
~$  export PYTHONPATH=/usr/local/lib/python2.7/dist-packages
~$  export PKG_CONFIG_PATH=/usr/local/lib/pkgconfig/
```

Správnou instalaci verze je možno ověřit zadáním příkazu:

```
~$  gnuradio-companion --version
```

Ten by měl potvrdit verzi 3.4.2. Po té je možné zapnout a připojit USRP a zadat příkaz:

```
~$  uhd_find_devices
```

Jeho správný výsledek vypadá podobně jako tento:

```
linux; GNU C++ version 4.4.3; Boost_104000; UHD_003.005.000-26-
    gb65a3924
```

```
--------------------------------------------------
-- UHD Device 0
--------------------------------------------------
Device Address:
    type: usrp1
    name:
    serial: E4R22Y3U
```

3.2 Instalace OpenBTS (ver. P2.8)

Před instalací OpenBTS je obdobně jako u GNU Radio nutné doinstalovat podpůrné balíčky knihoven. Tyto knihovny jsou dobře vypsány již na stránkách projektu OpenBTS viz [9]. Je tedy v terminálu Ubuntu zadáno:

```
~$ sudo apt-get install autoconf libtool libosip2-dev libortp-
   dev libusb-1.0-0-dev g++ sqlite3 libsqlite3-dev erlang
   libreadline6-dev libboost-all-dev subversion
```

Tím by mělo být vše potřebné připraveno. Pro pořádek by se měl uživatel přesunout do své složky (kde má již GNU Radio) a pak může dojít ke stažení aktuálního kódu OpenBTS pomocí nástroje subversion přímo ze stránek OpenBTS takto:

```
~$ svn co http://wush.net/svn/range/software/public openbts
```

Nyní již je možné přejít ke konfiguraci, kompilaci a instalaci. Tu provede následující sekvence příkazů hned po přechodu do příslušné složky. Pro doboru funkci je nutné provádět příkazy s právy uživatele root (přičemž po instalaci je tento mód zrušen):

```
~$ sudo su
~# cd openbts/openbts/trunk
~# autoreconf -i
~# ./configure --with-usrp1 --with-singledb
~# make clean
~# make
~# make install
~# exit
```

Takto provedená konfigurace (příkazem ./configure) bude využívat pouze jedné daughtboard, na níž bude provádět jak vysílání (Tx), tak příjem (Rx), druhá deska je takto nevyužitá. Pokud by byla provedena konfigurace bez volby „*--with-singledb*“, kit by standardně využíval dvou daughterboard tak, že první (strana A) je využita pro vysílání (Tx) a druhá (strana B) pro příjem (Rx). Konfigurace se provádí dle požadavků. V tomto případě však při využití obou desek (tedy jedné pro Tx a druhé pro Rx) docházelo k nestabilitám celého projektu. Proto byla raději zvolena volba jedné desky. Pro další zařízení od firmy Ettus se konfigurace provádí také jiným způsobem.

Po této instalaci je nutné propojení souborů transceiveru užívaného hardwaru. Provádí se na dvou místech – v prvním se vytvoří odkaz, v druhém kopie souboru. Předpokladem je, že se uživatel nachází tam, kde skončil, tedy stále ve složce *openbts/trunk/*:

```
~$ cd apps/
~$ ln -s ../Transceiver52M/transceiver .
~$ cd ..
```

```
~$  sudo cp Transceiver52M/std_inband.rbf
    /usr/local/share/usrp/rev4/
```

OpenBTS verze 2.8 využívá již oproti předchozím verzím pro své nastavení databáze. Proto je posledním krokem její vytvoření. Vytvoření probíhá s použitím ukázkového souboru, avšak jeho umístění musí být na specifickém místě systému, kde je pro OpenBTS nutné vyvořit složku. Následující příkazy počítají opět s umístění z předchozího bodu (tedy ve složce *openbts/trunk*):

```
~$  sudo mkdir /etc/OpenBTS
~$  sudo sqlite3 -init ./apps/OpenBTS.example.sql
    /etc/OpenBTS/OpenBTS.db ".quit"
```

Takto vytvořenou síť by již mobilní stanice měla být schopna rozpoznat, ovšem pro správnou funkci je ještě nutné podniknout pár kroků.

3.3 Asterisk, nastavení a databáze

Jednou z nutností je nainstalování PBX Asterisk. Jednou z možností, jak tento software do systému dostat, je přes připravený balíček linuxu. Ten ve verzi Ubuntu 10.04 nainstaluje verzi 1.6.2.5. V této práci však byla zvoleno stažení stabilní a certifikované verze Asterisku 1.8.11 LTS.

Stažení této verze proběhlo přes internetový prohlížeč přímo ze stránek *asterisk.org*. Jeho stažení bylo provedeno do souboru *Downloads*. Ten je dobré extrahovat (např. jako zde do své složky) a vzhledem k jeho dlouhém názvu jej trochu upravit. To provedou následující příkazy, jež vycházejí z předpokladu, že se nacházíme ve své složce – která se v tomto případě nachází v domovském adresáři.

```
~$  tar -xf ../Downloads/certified-asterisk-1.8.11-current.tar.gz
~$  mv certified-asterisk-1.8.11-current/ -T asterisk-1.8.11
```

Rozbalený Asterisk stačí už jen dle návodu s readme souboru nainstalovat:

```
~$  sudo su
~#  cd asterisk-1.8.11/
~#  ./configure
~#  make
~#  make install
~#  exit
```

S projektem OpenBTS bylo staženo mnoho dalších podpůrných prostředků. Nacházejí se ve stažené složce pojmenované openbts. V tuto chvíli zaměříme zájem na soubory *subscriberRegistry, smqueue*.

První jmenovaný je novinkou ve verzi OpentBTS 2.8. Dříve totiž OpenBTS používal SIP registr Asterisku jako náhradu HLR. Nyní se pro to užívá nová komponenta založená na sqlite3 databázi. [9]

V tomto případě pak subscriberRegistry slouží jako podpora Sipauthserve, o němž je pojednáno vzápětí. Předtím je nutné nastavení databáze registru (SubscriberRegistry):

```
((nacházíme se ve své složce s projektem))
~$  cd opentbts/subscriberRegistry/trunk/configFiles/
~$  sudo mkdir -p /var/lib/asterisk/sqlite3dir
```

```
~$  sudo sqlite3 -init subscriberRegistryInit.sql
    /var/lib/asterisk/sqlite3dir/sqlite3.db ".quit"
```

Nyní k Sipauthserve. Sipauthserve je daemon poskytující autentizační služby protokolu SIP. Jeho konfigurační proměnnou je SIP.Proxy.Registration. Ta by měla směřovat k jeho hostname a portu [9]. Nyní je třeba provést zkompilování subscriberRegistry, aby byl Sipauthserve spustitelný:

```
((nacházíme se ve své složce s projektem))
~$  cd opentbts/subscriberRegistry/trunk/
~$  make
```

A opět je nutno provést konfiguraci pomocí databáze. Předpokladem je umístění z posledním kroku:

```
~$  sudo sqlite3 -init sipauthserve.example.sql
    /etc/OpenBTS/sipauthserve.db ".quit"
```

Na konec je ještě možné přidat do celého projektu podporu SMS zpráv. Ten je realizován servisem **Smqueue**, který rovněž obsahuje stažený projekt OpenBTS. Nacházíme-li se ve složce s celým naším projektem zadáme:

```
~$  cd openbts/smqueue/trunk/
~$  autoreconf -i
~$  ./configure
~$  make
```

Smqueue závisí také na konfiguračním databázovém souboru. Ten je potřeba opět vytvořit a přenést dle předlohy [9, 15]. Opět uvažujeme, že se nacházíme ve stejné složce, kde byly ukončeny předchozí příkazy:

```
~$  cd smqueue/
~$  sudo sqlite3 -init smqueue/smqueue.example.sql
    /etc/OpenBTS/smqueue.db ".quit"
```

Nyní, ještě před závěrečný nastavením PBX Asterisk, je dobré si pro spolupráci s PBX a pro budoucí ulehčení práce s přenosem čísel IMSI poupravit nastavení databáze. Pro nastavení databáze by nainstalován uživatelsky jednoduchý a přívětivý program *sqliterbrowser*, pomocí:

```
~$  sudo apt-get install sqlitebrowser
```

Pomocí sqlitebrowseru je pak dobré editovat soubor ObenBTS.db. Otevřeme jej pod právy roota:

```
~$  sudo su
~#  sqlitebrowser /etc/OpenBTS/OpenBTS.db
```

Důležitým krokem je také změna položky „Control.LUR.OpenRegistration“ sloužící k otevřené registraci k OpenBTS. Pokud je hodnota registru nulová, neumožní žádnému IMSI přístup do sítě. Při pokusu o připojení by nám totiž odesílala SMS o nenalezení IMSI v databázi. Řešením je nastavení otevřené registrace pomocí registru „**Control.LUR.OpenRegistration**“ na hodnotu „ **.*** “ nebo „**^[0-9]{16}$**“. První možnost je o něco vhodnější. Dále je možné editovat další položky jako např. označení sítě pomocí LAC, MNC, MCC, cell id. Hlavním parametrem změny však bude položka „**GSM.Radio.C0**“ pro

nastavení kanálu (ARFCN) a analogicky také „**GSM.Radio.Band**“ pro přiřazení správného pásma GSM daného kanálu.

Nyní je možné spuštění celého OpenBTS. V rámci tohoto nastavení bude potřeba spouštět všechny aplikace zvlášť v samostatném terminálu. Navíc je defaultně nutné spustit zvlášť OpenBTS a její konzoly. Následující příkazy vycházejí z předpokladu, že se nacházíme ve své složce s projektem. Je více než vhodné je zadávat z pozice root (*sudo su*) a každý pak spustit v samostatném terminálu:

```
~#  asterisk -vvvvvc
~#  openbts/smqueue/trunk/smqueue/smqueue
~#  openbts/subscriberRegistry/trunk/sipauthserve
~#  openbts/openbts/trunk/apps/OpenBTS
~#  openbts/openbts/trunk/apps/OpenBTSCLI
```

3.3.1 Testovací konfigurace PBX Asterisk

Pro otestování projektu byly v Asterisku vytvořeni 2 uživatelé (mobilní stanice), kteří si mohou navzájem spojit. Také byla pro tyto účely vytvořena hovorová smyčka. Vychází se ze standardní konfigurace Asterisk souborů *extensioin.conf* a *sip.conf*.

Při této konfiguraci však bylo potřeba znát IMSI čísla SIM karet, které se budou do sítě připojovat. Metod jak tato čísla získat je několik – u každé z nich je nutné mít spuštěn OpenBTS projekt. Při použití otevřené registrace nastavením „ **.*** „ přijde účastníkovi po registraci uvítací SMS obsahující jeho číslo IMSI. Pokud by se tak ale nestalo, musí uživatel získat IMSI zadáním příkazu do konzole OpenBTSCLI:

```
OpenBTS> tmsis
```

Po jeho zadání se vypíše seznam registrovaných účastníků s čísly IMSI. Poslední možnost je více použitelná při nastavení „Open.Registration“ hodnotou „**^[0-9]{16}$**“. Tyto pokusy o přihlášení do testovací sítě se také logují do okna se spuštěným Asteriskem, z něhož je možné IMSI jednoduše zkopírovat (stejně jako u OpenBTSCLI) a přidat do konfiguračních souborů.

Zadávané číslo do projektu musí být ve formátu např.: IMSI012345678901234

Do souboru *sip.conf* (v /etc/asterisk/) se přidávají údaje registrovaných účastníků. Celý soubor *sip.conf* tak může vypadat například takto:

```
[general]
context=context
bindport=5060
dislallow=all
allow=gsm

[IMSI_prvni_sim]
type=friend
host=dynamic
context=context
canreinvite=no
callerid=777
```

```
[IMSI_druhe_sim]
type=friend
host=dynamic
context=context
canreinvite=no
callerid=888
```

Soubor *extensions.conf* slouží k propojování hovorů. Za jednotlivé IMSI bylo z důvodu zajištění správné funkce přidána adresa a port spojovací smyčky. (pozn.: Pokud by jej daný uživatel nezadal musel by změnit v databázi *OpenBTS.db* položku „Sip.Proxy.Registration" na port „5060"). V rámci tohoto souboru je také vytvořena jednoduchá smyčka hlasového automatu na lince 333. Pro tento projekt může soubor *extension.conf* vypadat například takto:

```
[context]
exten => 777,1,Dial(sip/IMSI_prvni_sim@127.0.0.1:5062)
exten => 888,1,Dial(sip/IMSI_druhe_sim@127.0.0.1:5062)
exten => 333,1,Answer
exten => 333,2,Playback(pbx-invalid)
exten => 333,3,Playback(vm-nobodyavail)
exten => 333,4,GoTo(2)
```

Po restartu Asterisku (zadáním do jeho konzole: **`core restart now`**) nebo opětovném spuštění všech náležitostí projektu by měla být realizovaná síť plně funkční.

3.4 Poznatky při realizaci

Prvním poznatek při realizaci se týkal samotného kitu USRP1. O něm již bylo zmíněno, že je již starší a jeho podpora zůstala na verzi GNU Radio 3.4.2, avšak stále je v nabídce firmy Ettus. Jeho primárním problémem pro realizaci GSM je již také zmíněny nevhodný generátor hodinového impulsu. Tento problém však byl vyřešen před vznikem této práce.

U GNU Radia se vychází z již stálé verze. Pro správnou instalaci je však třeba najít návod (a ne jen jeden), který pracuje s přesnou požadovanou konfigurací. Odlišnosti těchto návodů jsou třeba jen v detailech, i tak mnohdy nedokáží obsáhnout přesný problém, který se při instalaci podnikne. Je třeba dávat si pak o to větší pozor při realizaci se změněným generátorem časového impulsu.

Úplně jinou kapitolou je software OpenBTS. Ve valné většině se OpenBTS stahuje a instaluje přímo poslední verze umístěná na stránkách projektu (pomocí *svn*). Tento kód je však neustále ve vývoji a probíhá u něj dolaďování, vylepšování a oprava kódu. Je to dvojsečné, protože každý kdo instaluje, má jinou konfiguraci a nastavení. Co se tak jednou opravou jednomu zlepší, může dalšímu způsobit problémy při realizaci jeho projektu. V mnoha návodech jsou vidět návody na různé „hacky," vztahující se právě k dané verzi – lépe řečeno revizi instalace OpenBTS vztažené k danému datu. Tyto revize již ale sehnat nelze. Na základě toho je třeba počítat s tím, že pokud vyvstane v realizaci projektu problém, nemusí být způsoben špatným postupem.

U programů (či jejich částí) se při práci objevily problémy s neukončením tzv. „visením" programů. Tím nečekaným se stal *sipauthserve*, jehož visení by bylo spíše pozitivním přínosem v rámci úspory jednoho terminálového okna. Více čekaný či v návodech zmíněný je problém s neukončeným *transceiver*. Ten je součástí OpenBTS a vzniká

nestandardním nebo neočekávaným ukončením aplikace OpenBTS, např. při pádu programu špatné konfigurace. Tato skutečnost se při opětovném pokusu o spuštění oznámí hláškou:

```
bind() failed: Address already in use
terminate called after throwing an instance of 'SocketError'
Aborted
```

a dojde k ukončení aplikace OpenBTS. Potom je nutné daný proces ručně vyhledat pomocí:

```
~#  ps | grep transceiver
```

a poté jej podle zjištěného PID (identifikátoru procesu) ukončit příkazem ***kill*** s číslem procesu *transceiver* a nebo jednodušeji příkazem:

```
~#  killall transceiver
```

3.4.1 Problém virtuálních strojů

Jednou z věcí, které si kladla tato práce za úkol, bylo vyzkoušet, zda lze celý projekt realizovat na virtuálním stroji. Důvodem je možnost jednoduché přenositelnosti celého virtuálního počítače. Je tedy nutné, aby všechny aplikace fungovaly v pořádku. Z počátku nebyly nalezeny žádné dokumenty, které by přímo říkali, že to nelze. Pouze zde byl požadavek na dostatečný výkon a pár zpráv o lehce zpomalených fungujících projektech viz [17]. Vzhledem k přítomnosti Oracle VM VirtualBox na školních počítačích, byl pro virtualizaci zvolen právě tento program.

I když byl celý projekt nainstalován „krkolomným“ způsobem – jednalo se o vůbec první dokončenou instalaci podle pouze jednoho zdroje – vykazovalo největší problémy připojení kitu USRP1. Při pokusech o spojení se objevovaly chyby typů:

- odpojení USRP při jeho nalezení a stažení firmware
- neustále zůstával viset proces *transciever*
- pokud se povedlo spojit s jednotkou, tak po spuštění OpenBTS a následném pokusu připojení k OpenBTS došlo k pádu celé aplikace
- absolutně nemožná jakékoliv obnova spojení pomocí odpojení a připojení přes VirtualBox – celý stroj se na několik minut zasekl
- v několika případech se virtuální stroj zasekl úplně

Navíc na školním virtuálním PC byl celý virtuální stroj značně zpomalený, což bylo způsobeno nízkým výkonem hardwaru. Na privátním PC se virtuální stroj celkem držel. V (nevirtuálním) operačním systému privátního PC byl proveden pokus o vylepšení spojení externí instalací ovladače pro celý kit. Zprvu se to zdálo jako dobré řešení, které ke konci vedlo opět k pádu celého VirtualBoxu.

Z obav o nízký výpočetní výkon byl virtuální stroj přenesen na server, který zajišťoval výkon více než dostatečný. Původní nainstalovaná konfigurace se však chovala téměř stejně. Přeinstalování GNU Radia s celým problémem nijak nepokročilo. Navíc dále proběhlo na serveru ve virtuálním stroji pár pokusů o reinstalaci ovladačů, avšak celé snažení na VirtualBoxu ukončilo zamrznutí VM (Virtual Machine) i na serveru.

Po těchto nezdarech byl objeven článek (viz [16]), který shrnoval veškeré pokusy instalace GNU Radia s USRP na VirtualBox. Samotný VirtualBox má totiž špatně naprogramované připojení USB, které se snaží komunikovat i se systém na kterém VM běží.

A navíc je nutné podporu pro USB 2.0 do VirtualBoxu doinstalovat (v tomto projektu byla podpora od začátku). Připojení zařízení, stažení firmwaru a následné zablokování provádí VirtualBox, jenž si blokuje vlastní port, kvůli změně názvu [16]. Jednoduše řečeno pro aplikaci USRP není VirtualBox naprosto vhodný.

Opačným případem je VMware (pod licencí pro nekomerční použití). Ten dokáže „protunelovat" USB port přímo do virtuálního stroje bez závislosti, či nějaké odezvy reálného systému s plno podporou USB 2.0. Celý projekt začal bez problému fungovat hned po první instalaci. Bylo však nutné dodržet určitý postup spouštění projektu, kdy je nutné po připojení USRP nejprve stáhnout firmware pomocí *uhd_find_device*.

3.5 Výstupy první realizace sítě

V této podkapitole jsou výstupy získané přímo z tohoto projektu. Jsou zde odezvy na jednotlivé příkazy nainstalovaných softwarů, ukázky správných hlášení pro různé aplikace, či také výpis asterisku u spojeného hovoru a další. Vše se vztahuje k první realizaci sítě na deskách RFX900.

3.5.1 Odezvy správně nainstalovaného GNU Radio a UHD

Aplikací, kterou je třeba si zvyknout spouštět na začátku, je již několikrát zmiňovaný *uhd_find_device*. Je to aplikace stažená s ovladačem UHD a stahuje firmwarekit. Je nainstalovaná, takže ji lze spouštět kdekoliv z terminálu. Výsledkem je:

```
~$ uhd_find_devices
linux; GNU C++ version 4.4.3; Boost_104000; UHD_003.005.000-26-
    gb65a3924

--------------------------------------------------
-- UHD Device 0
--------------------------------------------------
Device Address:
    type: usrp1
    name:
    serial: E4R22Y3U
```

Další aplikací dodávanou s UHD je *uhd_usrp_probe*. Ta vypisuje možnosti daného zařízení. Samotný výpis je dlouhý a tak zde bude zobrazeno pouze pár řádků ze začátku. U tohoto výpisu je možné si všimnout, že aplikace stále počítá s původním generátorem časového impulsu:

```
 linux; GNU C++ version 4.4.3; Boost_104000; UHD_003.005.000-26-
    gb65a3924

-- Opening a USRP1 device...
-- Loading FPGA image: /usr/local/share/uhd/images/usrp1_fpga.rbf...
    done
-- Using FPGA clock rate of 64.000000MHz...
  ____________________________________________________
 /
|       Device: USRP1 Device
|     ______________________________________________
|    /
```

```
|   |       Mboard: USRP1
|   |   serial: E4R22Y3U
|   |
|   |   Time sources: none
|   |   Clock sources: internal
|   |   Sensors:
|   |     ____________________________________________________
|   |    /
|   |   |       RX DSP: 0
|   |   |   Freq range: -32.000 to 32.000 Mhz
|   |     ____________________________________________________
|   |    /
|   |   |       RX DSP: 1
|   |   |   Freq range: -32.000 to 32.000 Mhz
|   |     ____________________________________________________
|   |    /
|   |   |       RX Dboard: A
|   |   |   ID: RFX900 (0x0025)
|   |   |   Serial: E1R11X2R9

                              <zbytek vynechán>
```

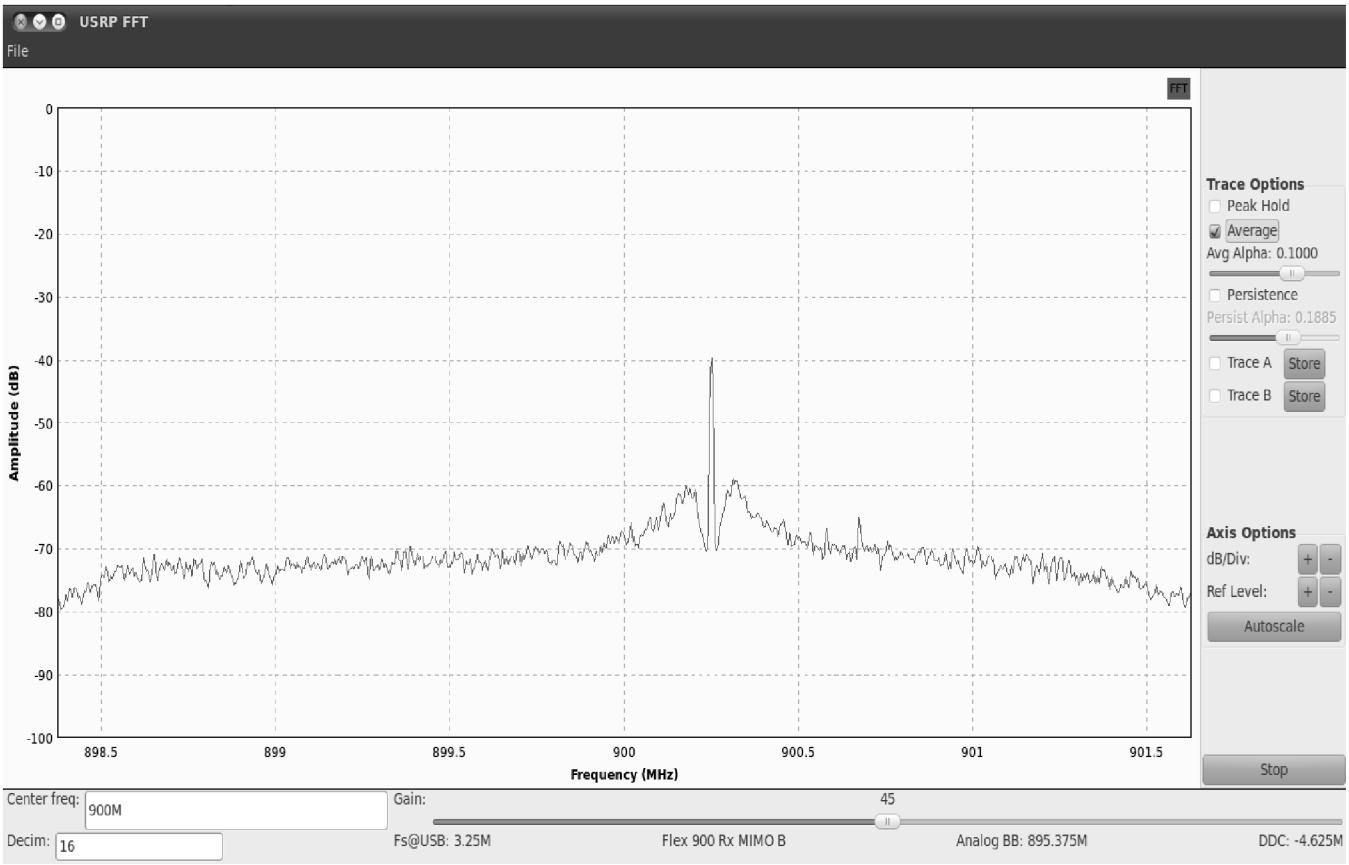
Pro ověření správnosti nainstalovaného GNU Radia je možné použít například dříve zmíněný „*gnuradio-companion --version*“ či jeho obdobu „*gnuradio-config-info -v*“. Tyto nástroje však ověří jenom verzi. Lepším nástrojem je „benchmark“ testující propustnost USB portů k zařízení. Nachází se v ukázkách dodávaných GNU Radio a výstup byl následující:

```
~$ /test_radio_bts/gnuradio-3.4.2/gnuradio-
    examples/python/usrp/usrp_benchmark_usb.py

Testing 2MB/sec... usb_throughput = 2M
ntotal    = 1000000
nright    = 999128
runlength = 999128
delta     = 872
OK
Testing 4MB/sec... usb_throughput = 4M
ntotal    = 2000000
nright    = 1999450
runlength = 1999450
delta     = 550
OK
Testing 8MB/sec... usb_throughput = 8M
ntotal    = 4000000
nright    = 3998606
runlength = 3998606
delta     = 1394
OK
Testing 16MB/sec... usb_throughput = 16M
ntotal    = 8000000
nright    = 7995275
runlength = 7995275
delta     = 4725
OK
Testing 32MB/sec... usb_throughput = 32M
ntotal    = 16000000
nright    = 15994398
runlength = 15994398
```

Celý kit s GNU Radiem lze využít jako spektrální analyzátor, díky nainstalované aplikaci *„usrp_fft.py“*. Pokud bez problému tato aplikace funguje, je výkon systému dostatečný. Obr. 3.2 ukazuje příklad naměřeného spektra pomocí tohoto nástroje. V porovnání se změřeným spektrem pomocí spektrálního analyzátoru GW Instek – GSP-830 však byla např. špička zobrazeného signálu posunuta o 0,2 MHz výše.

Při skenování spektra pomocí nástroje *usrp_fft.py* je také třeba dát si pozor na drobné vysílání kitu na aktuální skenované frekvenci, která je nastavena v kolonce „Center freq.“ Na nastavené frekvenci se na obrazovce spektra objevuje drobná špička (jen o pár jednotek dB více) identifikující signál, který ve spektru ve skutečnosti není, protože jej vytváří samotný kit.



Obr. 3.2: Nástroj usrp_fft.py

3.5.2 Ověření volného kanálu

Většina nastavení je použita z ukázkových souborů. Vzhledem k tomu, že se vlastně v pásmu GSM 900 jedná při budování sítě o nelegální aktivitu, bylo nutné alespoň ověřit zda na daném kanále již někdo nevysílá a zjistit na kterých frekvencích je přenos nastaven - tedy který kanál je zvolen - případně jej změnit.

Výchozí nastavený kanál (v databázi OpenBTS) byl kanál 51. Odpovídající frekvence jsou tak 900,2 MHz ve směru od mobilní k základnové stanici a 945,2 MHz od základnové stanice k telefonu. Dle stránek gsmweb.cz, které obsahují informace o všech obsazeních kanálů, poloze BTS, respektive všech náležitostí, které se české GSM sítě týkají, bylo zjištěno, že kanál 51 není žádným operátorem v místě vysílání obsazen.

Tento fakt byl potvrzen nejen měřením na spektrálním analyzátoru GW Instek (výsledek ukazuje Obr. 3.3), ale také programem pro kalibraci s názvem „kalibrate-uhd“. Jedná se o aplikaci, která je primárně určená pro kalibraci vnitřních kitu USRP. Během svého běhu testuje s využitím USRP rádia frekvence resp. kanály zvoleného GSM pásma. Pokud na dané frekvenci již nějaká BTS vysílá, aplikaci otestuje odstup interních hodin kitu USRP od hodinového signálu přicházejícího od vysílací BTS. Tento rozdíl přepočítá, vypíše spolu s kanálem, frekvencí a naměřeným výkonem do konzole a pokračuje na další kanál. Tímto způsobem pak lze jednoduše zjistit obsazení dostupných kanálu zvoleného GSM pásma. Program lze stáhnout z internetu, což společně s kompilací provedou následující příkazy (je dobré zvolit vhodný adresář):

```
~$ wget http://ttsou.github.com/kalibrate-uhd/kal-v0.4.1.tar.bz2
~$ tar -xf kal-v0.4.1.tar.bz2
~$ cd kal-v0.4.1/
~$ ./bootstrap
~$ ./configure
~$ make
```

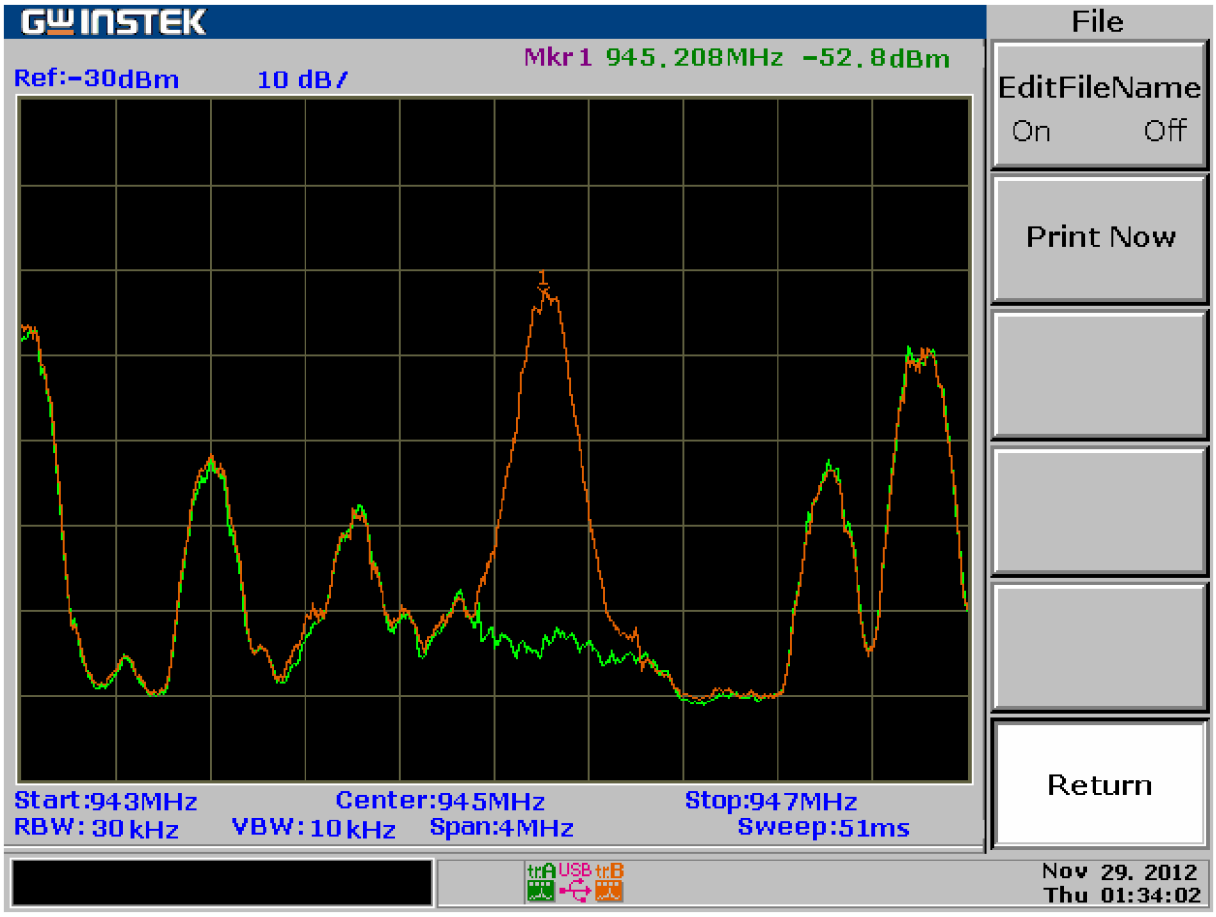
Nyní následuje výpis získaný aplikací kalibrate-uhd v laboratoři PA-339. Podobná situace panuje i v laboratoři SC-5.34. Je z něj patrné, že kanál 51 není na tomto místě využit:

```
~$ ./kal -s GSM900

kal: Scanning for GSM-900 base stations.
GSM-900:
    chan: 1 (935.2MHz +  26Hz)      power: 3579.98
    chan: 13 (937.6MHz +  16Hz)     power: 2473.56
    chan: 15 (938.0MHz +  24Hz)     power: 2543.84
    chan: 18 (938.6MHz +  16Hz)     power: 6401.51
    chan: 37 (942.4MHz + 32.275kHz) power: 21007.91
    chan: 38 (942.6MHz + 32.360kHz) power: 17844.53
    chan: 40 (943.0MHz +  37Hz)     power: 24892.68
    chan: 44 (943.8MHz +   2Hz)     power: 18379.63
    chan: 57 (946.4MHz +  21Hz)     power: 8732.86
    chan: 59 (946.8MHz +  24Hz)     power: 22704.89
    chan: 75 (950.0MHz +  26Hz)     power: 5867.46
    chan: 84 (951.8MHz +  20Hz)     power: 23693.40
    chan: 89 (952.8MHz - 338Hz)     power: 15788.80
    chan: 93 (953.6MHz +  15Hz)     power: 64714.02
    chan: 105 (956.0MHz +   6Hz)    power: 131690.64
    chan: 106 (956.2MHz +  13Hz)    power: 31198.37
```

3.5.3 GSM síť - realizace, připojení a volání

Již bylo zmíněno, že při realizaci projektu byl využit spektrální analyzátor GW Instek - GSP-830. Ten posloužil také jako ukazatel na vytvořenou GSM ve frekvenčním spektru. Při spuštění aplikace OpenBTS byl zaznamenán rozdíl na zvolené frekvenci. Tento rozdíl byl pouze ve směru od základnové stanice. V opačném směru zaznamenán nebyl, protože k výraznějšímu vysílání dochází při realizaci hovoru. Na Obr. 3.3 ukazuje zelená křivka stav, který je v oblasti před spuštěním. Oranžová křivka reprezentuje stav po spuštění OpenBTS. Ukazatel 1 (označen i Mkr 1) v obrázku zobrazuje signál na kanále 51, který je právě touto aplikací vyvolán.



Obr. 3.3: Zaznamenané frekvenční spektrum před (zeleně) a po (oranžově) spuštění OpenBTS

Pro zjišťování parametrů na spuštěném OpenBTS slouží jeho konzola OpenBTSCLI. Přes ni je možné zjišťovat mnoho údajů. Jednak to mohou být údaje z databáze - jako MCC, MNC atd., nebo výpis aktuálního zatížení. Zde je příklad výpisu nastavené identifikace sítě:

```
OpenBTS> cellid
MCC=001 MNC=01 LAC=1000 CI=10
```

Užitečným příkazem je příkaz „load“ – zatížení. Vypisuje aktuální zatížení z hlediska aktivních logických kanálů a délek front. Mezi vypisované hlídané kanály patří např. řídící SDCCH (Subscriber Dedicated Control Channel), provozní kanál TCH/F, velikost tabulky TMSI a další [12]. Výpis před registrací MS do sítě by následující:

```
OpenBTS> load
SDCCH load: 0/4
TCH/F load: 0/7
AGCH/PCH load: 0,0
Paging table size: 0
Transactions: 2
T3122: 2000 ms
```

Úlohy vypisování spojovacích stavů se při plně spuštěném projektu stará asterisk. Vypisuje jak pokusy o připojení do sítě tak spojování hovorů. První výpis zobrazuje pokus o přihlášení neregistrované SIM karty (reprezentovanou IMSI) do sítě:

```
[Nov 28 07:14:03] NOTICE[512]: chan_sip.c:25568
    handle_request_register: Registration from
```

```
'IMSI230015200333318 <sip:IMSI230015200333318@127.0.0.1>'
failed for '127.0.0.1:5062' - No matching peer found
```

Na mobilním zařízení je nutno síť vybrat ručně a to v nastavení telefonu. Síť se oznamuje v rozličných telefonech různě: např.: 01 001, Test, Range. Na obrazovce mobilního zařízení se pak většinou zobrazuje název sítě jako „Range.“ Výpis u úspěšného přihlášení registrované SIM karty vypadá takto:

```
-- Registered SIP 'IMSI230012500584348' at 127.0.0.1:5062
    > Saved useragent "OpenBTS P2.8TRUNK Build Date Dec 9
2012" for peer IMSI230012500584348
```

Volání uživatele na definovanou smyčku Asterisku se vypíše:

```
== Using SIP RTP CoS mark 5
    -- Executing [333@context:1] Answer("SIP/IMSI230012500584348-
    0000000b", "") in new stack
    -- Executing [333@context:2]
    Playback("SIP/IMSI230012500584348-0000000b", "pbx-invalid")
    in new stack
    -- <SIP/IMSI230012500584348-0000000b> Playing 'pbx-
    invalid.gsm' (language 'en')
    -- Executing [333@context:3]
    Playback("SIP/IMSI230012500584348-0000000b", "vm-
    nobodyavail") in new stack
    -- <SIP/IMSI230012500584348-0000000b> Playing 'vm-
    nobodyavail.gsm' (language 'en')
    -- Executing [333@context:4] Goto("SIP/IMSI230012500584348-
    0000000b", "2") in new stack
    -- Goto (context,333,2)
    -- Executing [333@context:2]
    Playback("SIP/IMSI230012500584348-0000000b", "pbx-invalid")
    in new stack
    -- <SIP/IMSI230012500584348-0000000b> Playing 'pbx-
    invalid.gsm' (language 'en')
== Spawn extension (context, 333, 2) exited non-zero on
    'SIP/IMSI230012500584348-0000000b'
```

A nakonec zde bude uveden výpis, při volání uživatele 2 s číslem 888, uživateli 1 s číslem 777:

```
== Using SIP RTP CoS mark 5
    -- Executing [777@context:1] Dial("SIP/IMSI230012500584348-
    0000000e", "sip/IMSI230012000258589") in new stack
== Using SIP RTP CoS mark 5
    -- Called sip/IMSI230012000258589
    -- SIP/IMSI230012000258589-0000000f is ringing
    -- SIP/IMSI230012000258589-0000000f is ringing
    -- SIP/IMSI230012000258589-0000000f is ringing
    -- SIP/IMSI230012000258589-0000000f answered
    SIP/IMSI230012500584348-0000000e
    -- Locally bridging SIP/IMSI230012500584348-0000000e and
    SIP/IMSI230012000258589-0000000f
== Spawn extension (context, 777, 1) exited non-zero on
    'SIP/IMSI230012500584348-0000000e'
```

Uvedený výpis ukazuje spojování uživatelů (mezi řádky s oznamováním použití SIP a RTP protokolu). Dále vypíše vyzvánění (v tomto případě třikrát), přijetí (odpověď „answered“) od volaného. Samotné spojení a trvání hovoru znázorňuje řádek s „Locally bridging.“ Konec hovoru oznamuje poslední řádek.

Kvalita hovoru nebyla nijak zvláště zkoumána, avšak za dobu spojení se neukázal problém, který by měl na kvalitu hovoru vliv.

3.6 Úprava spouštěcího skriptu projektu OpenBTS

Neustálé spouštění pětice programů pro realizaci GSM sítě pomocí OpenBTS bývalo zdlouhavé a nekomfortní. Na stránkách projektu OpenBTS jsou skripty pro automatické spuštění všech částí, avšak nelogicky pouze pro ústředny Freescale a Yate.

Proto byl jeden ze skriptů (pro Freeswitch) upraven tak, aby spouštěl ústřednu Asterisk a celý projekt by tak šel spustit pouze jedním příkazem.Výhoda tohoto skriptu spočívá také v tom, že se postará i o vypnutí procesů asterisk i transceiver, které zůstávají často „viset.“

Úpravy skriptu nejsou nijak zásadní. Nejprve je zvolena cesta k souborům projektu OpenBTS. Spouštěcí skript počítá s umístěním v souboru před složkou openbts.V původním skriptu také byla definována cesta pro ústřednu. U Asterisku to ale není potřeba, protože se dá spouštět z terminálu jako program. Proto byly položky týkající se umístění zakomentovány. Dál byl upraven název programu ústředny na asterisk za *killall.* Poslední úpravou byl spouštěcí příkaz ústředny, který byl vložen se stejnými parametry při spouštění jako by se spouštěl ručně. Vše ostatní zůstalo ponecháno.

Aby šel ještě skript bez problému spustit bez externích programů, je zapotřebí změnit práva přístupu k souboru na spouštěcí např:

```
~$ chmod 777 obts_startup.sh
```

Poté by měl být skript funkční a stačí jej spustit, buďto z pozice root nebo také normálního uživatele. Přitom ale dojde k vyzvání k zadání administrátorského hesla.

```
~$ ./obts_startup.sh
```

4 Open-source projekty pro analýzy GSM sítě

Pro fungování každé sítě je také důležité její testování. Nejsou-li k dispozici komerční projekty pro analýzu (jako třeba Tems) pomohou různé vytvořené open-source projekty analyzující GSM síť. Pro vytvořenou GSM síť je ale také možné mnohdy použít interních nástrojů, nalézajících se přímo v realizujícím softwaru. V tomto případě, kdy bude používán software OpenBTS, tato možnost taktéž existuje.

4.1 Projekt OsmocomBB

Základní projekt Osmocom (Open Source Mobile Communication) je soubor open-source softwaru pro oblast mobilních komunikací. Realizuje celou řadu projektů mezi nimiž je jak GSM/GPRS, tak bezšňůrová (DECT) či satelitní telefonie. Tyto projekty mají za úkol podpořit inovace, výzkumy a experimenty na rozsáhlých komunikačních systémech. Společně pak tvoří jednu z největších sbírek nástrojů pro realizaci různých částí GSM architektury. Mezi projekty realizované v rámci projektu Osmocom patří například OsmoBTS (vytvářející 2 vrstvy A-bis rozhraní BTS), OpenBSC (softwarová realizace BSC), Softsim (softwarový emulátor sim karty), OsmocomGMR pro satelitní telekomunikaci. Dílčí projekty spolu mohou spolupracovat.

Pro práci s GSM sítěmi je určen dílčí projekt s názvem OsmocomBB (Osmocom Base Band). Jeho cílem je vytvořit plnohodnotné open-source GSM zařízení (stack) 1 až 3 vrstvy. Za celou dobu své existence se stal nejspíše nejvíce užívaným open-source projektem pro implementace, vytváření a testování GSM sítí. Projekt vznikl k účelům zvýšení bezpečnosti u zařízení na veřejné síti, dále pro vzdělávání a také výzkum v oblasti GSM.

OsmocomBB používá pro své aplikace mobilní telefony, které jsou postaveny na čipech Calypso. U těchto telefonů se totiž podařilo zjistit, jakým způsobem pracují a jak v něm spustit vlastní kód. Mezi takové patří v hojném zástupu telefony Motorola (např. C115, C123, C140 a spoustu dalších) či jeden Sony Ericsson, Pirelli (od firmy Foxconn) či open-source mobil s názvem OpenMoko. Do všech jmenovaných telefonů je možné nahrávat svůj firmware, takže tento telefon využívá svůj hardware tak, jak zrovna potřebujeme. Pro tyto potřeby poskytuje Osmocom propracovaný systém.

4.1.1 Aplikace

Aplikace, které OsmocomBB poskytuje, je možné rozdělit do skupin:

- aplikace běžící na čipu telefonu
- aplikace běžící v počítači; komunikující přes sériové rozhraní

Druhé jmenované je možné dále rozdělit na aplikace pro správu firmwaru (nahrávání..) a aplikace GSM vrstvy L2/3.

K aplikacím běžícím na čipu telefonu patří:

- *layer1* – jednoduchý představitel první GSM vrstvy; sama o sobě tato aplikace nic nedělá – čeká na příkazy z vyšších vrstev
- *loader.bin* – zavaděč pro flash paměti
- *l1test* – testovací verze vylepšené aplikace layer1

- *bootloader* – univerzální zavaděč pro telefony s Calypso čipy.
- *hello_world* – výpis klasického „Hello world“
- *rssi* – aplikace určená ke sledování přijímaného signálu; dokáže měřit jednotlivé kanály i celé spektrum.

Aplikacemi pro správu jsou:
- *osmocon* – konzolový nástroj propojující firmware v telefonu s aplikacemi na počítači.
- *osmoload* – slouží k zápisu, mazání a prohlížení flash paměti
- *calypso_pll* – zobrazení možných kombinací násobičky/děličky pro výstupní frekvence
- *rita_pll* – zobrazení možných kombinací násobičky/děličky pro výstupní frekvence

Mezi aplikace pro GSM vrstvy L2/3 patří:
- *mobile* – aplikace vytvářející chování standardního mobilního telefonu rozšířené o zajímavé funkce
- *cell_log* – skenuje dostupné frekvence a sbírá informace z logického kanálu BCCH (Broadcast Control CHannel). Díky tomu je možné vytvořit seznam použitých ARFCN s údaji o úrovni, MNC, MCC a informacemi o systému
- *ccch_scan* – aplikace se synchronizuje s ARFCN, zaznamenává měření výkonu a údaje z logického kanálu CCCH (Common Control Channel)
- *bcch_scan* – předchůdce aplikace cell_log
- *cbch_sniff* – zaznamenává informační zprávy vysílané o dané buňce

Osmocon, tedy aplikace sloužící k propojení PC - telefon, slouží jednoduše řečeno k nahrání a spuštění firmware v telefonu. Navíc při běhu všech nahraných programů (firmware) vypisuje status do terminálu, respektive výstupy běžících programů. Nahrání i komunikace telefonu s PC se děje přes sériový port (resp. USB kabel), který je napojen k telefonu – zpravidla do sluchátkového vstupu.

Aplikace **rssi.bin** dělá z telefonu jednoduchý a přenosný analyzátor pro GSM sítě. Běží na čipu telefonu a veškeré důležité údaje vypisuje na obrazovku zařízení a po nahrání nemusí být připojen kabelem k PC. Pokud je připojen, vypisují se statusy do terminálu, ale to nejdůležitější této aplikace je stále na obrazovce telefonu. Aplikace rssi.bin má několik možných zobrazení a možností voleb:
- Zobrazení ARFCN
    - možnost naladění na libovolné ARFCN ve všech frekvenčních pásmech GSM potvrzujícím se zobrazením dané frekvence v Mhz
    - volba směru (downlink / uplink)
    - zobrazení úrovně signálu v dBm
    - nastavení zobrazení sloupcového grafu pro vizualizaci
    - akustické oznamování síly signálu

- Zobrazení spektra
    - přecházení mezi jednotlivými kanály při zobrazení spektra
    - volba měřítka

- Zobrazení Synchronizace
    - zobrazení systémových informací o dané buňce (i síti)
    - např. MCC, MNC, LAC, cell id, výkonová úroveň, BSIC atd.
    - zobrazení úrovně každého přijatého timeslotu zvlášť
    - měření vzdálenosti a zpoždění signálu dané BTS

Aplikace vrstvy L2/3 jako jsou ccch_scan, cell_log i cbch_snif potřebují pro svou funkci předehrát do telefonu firmware layer1. Poté jsou jednotlivé programy schopné výměny dat prostřednictvím minimalistického rozhraní mezi vrstvami L1 a L23. Toto rozhraní používá určité typy zpráv, kterými aplikace vyšších vrstev vysílají příkazy/dotazy a nižší vrstva na ně po provedení odpovídá. Aplikace je jednak vypisuje do spuštěné konzole a navíc tyto informace odesílá k dalšímu zpracování do protokolového analyzátoru Wireshark. K této činnosti používá protokol GSMTAP.

GSMTAP je pseudo-hlavičkový (přidává další hlavičky před zprávu) protokol, který balí GSM data z rádiového rozhraní U_m do UDP/IP paketů. Má přidělený port 4729. Podpora tohoto protokolu do analyzátoru je přidána od verze Wireshark 1.4.0. Tento protokol umí využít pro balení dat z GSM sítě nejen OsmocomBB, ale také Airprobe a dokonce OpenBTS [19].

4.1.2 Zprovoznění a kompilace projektu

Pro zprovoznění projektu je třeba mít projektem podporovaný telefon (informace o nich jsou na stránkách projektu). Pokud by byla potřeba odposlouchávat další telefony na větší vzdálenost než je pár metrů, je potřeba vyměnit nebo přemostit hardwarový filtr k odstranění nepotřebného signálu. Dále je třeba mít propojovací kabel. Telefon má sériový port přiveden na sluchátkový 2,5mm jack pod 3,3V. Jako druhý konec kabelu pro připojení do PC je možné využít sériový konektor standardu RS-232 (kde je nutné dát si pozor na velikost provozního napětí), ale v současné době je možné využít již USB varianty. Pro účely přímo tohoto projektu totiž různé společnosti prodávají speciální kabely, které obsahují jeden ze tří možných integrovaných čipů - PL2303, FT232 nebo CP2102. Za nejspolehlivější a nejvhodnější je považován typ s čipem FT232.

Projekt OsmocomBB je z hlediska své otevřenosti dělán na operační systém Linux a jeho distribuce. Je taky nutné upozornit, že zavaděč firmware je náročný na časování, tudíž při použití virtualizovaných strojů (např. pomocí VMware, VirtualBox atd.) může docházet k problémům a nespolehlivosti připojení.

Před samotnou kompilací projektu OsmocomBB je nutné operační systém doplnit o *libosmocore* – tedy knihovnu funkcí pro Osmocom projekty, které jsou součástí jak OsomocomBB tak OpenBSC, a také o Toolchain. Toolchain, respektive pro BB GnuArmToolchain, slouží k překladu – crosskompilaci – kódu pro ARM procesory nacházející se v telefonu. Pro jeho instalaci je možno použít několik možností a zde je použitá ta nejkomplexnější. Předtím je však třeba začít doplněním distribuce linuxu (zde Ubuntu ver. 10.04) o balíčky knihoven, které bude celá kompilace vyžadovat. Většina knihoven je určena pro toolchain:

```
~$ sudo apt-get install gcc libtool git-core pkg-config gcc
   shtool build-essential libgmp3-dev libmpfr-dev libx11-6
   libx11-dev texinfo flex bison libncurses5 libncurses5-dbg
   libncurses5-dev libncursesw5 libncursesw5-dbg libncursesw5-
   dev zlibc zlib1g-dev libmpfr-dev libmpc-dev src
```

Po tomto kroku je možné přistoupit k instalaci libosmocore – stažením přes git a kompilací. Po zvolení vhodného umístění v terminálu stačí zadat:

```
~$ git clone git://git.osmocom.org/libosmocore.gitcd libosmocore/
~$ autoreconf -i
~$ ./configure
~$ make
~$ sudo make install
~$ cd ..
```

U toolchainu je předpokladem opět vhodné umístění (složka) jako v předešlém kroku. Celý crosskompilátor se sestává se tří částí: GCC, Binutils a Newlib. Dohromady je sestaví skript dostupný na stránkách Osmocomu:

```
~$ mkdir gnu-arm/
~$ wget http://bb.osmocom.org/trac/raw-attachment/\
   /wiki/GnuArmToolchain/gnu-arm-build.2.sh
~$ chmod +x gnu-arm-build.2.sh
~$ mkdir build install src
~$ cd src/
~$ wget http://ftp.gnu.org/gnu/gcc/gcc-4.5.2/gcc-4.5.2.tar.bz2
~$ wget http://ftp.gnu.org/gnu/binutils/binutils-2.21.1a.tar.bz2
~$ wget ftp://sources.redhat.com/pub/newlib/newlib-1.19.0.tar.gz
~$ cd ..
~$ ./gnu-arm-build.sh
~$ cd ..
```

Kompilace je docela časově náročná a měla by končit výpisem: *Build Complete!*

Nyní je již možné přistoupit ke kompilaci OsmocomuBB. Postup není v ničem složitý. Pro povolení vysílání telefonu je však ale potřeba editovat Makefile projektu. Toto vysílání využívá například aplikace rssi.bin k měření zpoždění signálu – těžko říci, jak moc jsou výsledky relevantní, když není upraven filtr v telefonu. Na začátek se tedy provede stažení projektu a úprava souboru Makefile.

```
~$ git clone git://git.osmocom.org/osmocom-bb.git
~$ cd osmocom-bb
~$ git pull --rebase
~$ gedit src/target/firmware/Makefile
```

V tomto Makefile je pak třeba odkomentovat (smazáním #) řádek pro povolení vysílání:

```
CFLAGS += -DCONFIG_TX_ENABLE
```

Dále už je jen třeba před samotnou kompilací určit cestu k Toolchain exportem cesty, který je také uveden v následující sekvenci příkazů:

```
~$ cd src
~$ export PATH=$PATH:/<cesta ke složce gnu-arm>/install/bin
~$ make
```

Celý projekt je tímto připraven k použití [19].

4.1.3 Použití jednotlivých aplikací

Jako první při testu projektu je možno vyzkoušet aplikaci „**Hello World**!“ Program je totiž jednoduchý a nemá žádné požadavky navíc, jako třeba rssi.bin, či vlastně téměř všechny další aplikace. Problém je v tom, že telefon umí k nahrávání využít maximálně 64 kB paměti, i když hardwarová dispozice je větší. V takovém případě je nutné provést „chainload“, který přepíše možné využití paměti a pak je již možno nahrávat cokoliv. Zpět, ale k „Hello world!“ Nejjednodušší příkaz na nahrání firmware má strukturu:

```
~$  ./osmocon -p /dev/ttyUSB0 -m c123xor
    ../../target/firmware/board/Typ_telefonu/FIRMWARE.compalram.bin
```

Přičemž je nutné nalézat se ve složce, kde je aplikace osmocon umístěná – tudíž nalézáme-li se ve složce osmocom-bb je cesta:

```
~$  cd src/host/osmocon
```

Program *osmocon* má v možnostech více parametrů než jen dva uvedené. Jejich výčet je možné vyvolat parametrem *–h*.

Po připojení vypnutého telefonu k PC se „Hello world“ na Motorole C123 rozjíždí zadáním:

```
~$  ./osmocon -p /dev/ttyUSB0 -m c123xor
    ../../target/firmware/board/compal_e88/hello_world.compalram.bin
```

Po stisknutí enteru se samo od sebe nic dít nebude. Program totiž čeká na interakci uživatele, která dovolí telefonu nahrát firmware. Tím je krátké stisknutí červeného tlačítka, které slouží k zapnutí telefonu. Stisk však musí být krátký, aby k zapnutí telefonu nedošlo. Poté by mělo dojít k postupnému nahrávání firmware, jež je avizováno výpisem v terminálu.

Mnohdy se stává že se nahrávání nezdaří a zasekne se - obecně, když se 15 sekund nic neděje. V tomto případě je nutné příkaz v terminálu zrušit (ctrl + c), vytáhnout baterku v telefonu a odpojit a znovu připojit kabel.

Po úspěšném nahrání, jak aplikace „Hello world“ tak i ostatních, dojde k zobrazování zachycených informací na displeji telefonu a zároveň k výpisu informací do terminálu. V případě „Hello World“ jsou to informace o stavu baterie a napájeni.

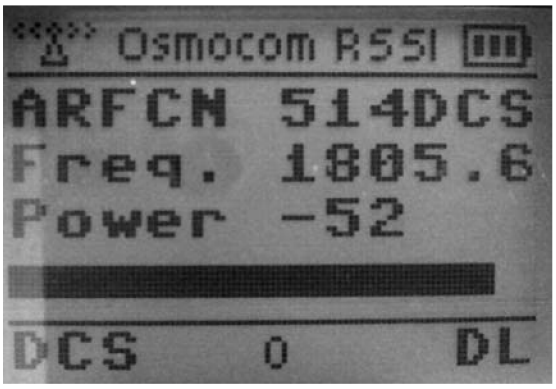
Aplikace v telefonu běží i po přerušení programu v terminálu i po odpojení kabelu. Telefon lze normálně vypnout červeným tlačítkem. Pro nahrání další aplikace je možné pokračovat dalším zadání příkazů. Je možné, že se program zasekne. V takovémto případě je vhodné vytáhnout baterii, či rozpojit a připojit kabel.

Aplikace rssi.bin je již onou aplikací náročnější na paměť. Pro její nastartování je nutné použít i „chainload.“ Postup pro nahrání je stejný jako v předchozím případě.

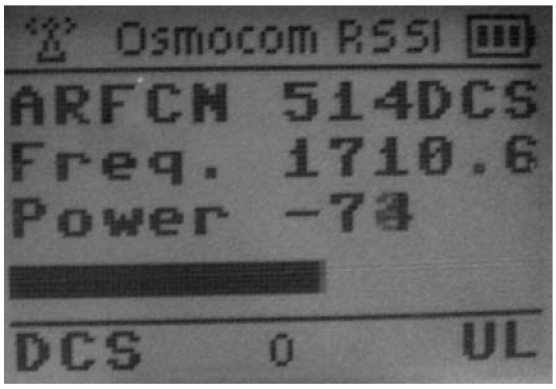
```
~$  ./osmocon -p /dev/ttyUSB0 -m c123xor -c
    ../../target/firmware/board/compal_e88/rssi.highram.bin
    ../../target/firmware/board/compal_e88/chainload.compalram.bin
```

Tato aplikace se plně ovládá na klávesnici telefonu. Aplikace se zdá být docela intuitivní. U základní obrazovky zobrazující číslo kanálu, frekvenci kanálu a sílu signálu, lze měnit jak normy (DCS, PCS), tak i směr analýzy (downlink – obr. 4.1; uplink – obr. 4.2). To se provádí přidruženými tlačítky pod displejem. Volba frekvence je možná buď to po každém

kanále – tlačítky vlevo, vpravo – nebo přímým zadáním ARFCN na číselníku. Tlačítka „nahoru, dolů“ ovládají hlasitost (resp. i zapínání) akustického oznamování síly signálu. Pro zadržení maximální hodnoty naměřené síly signálu slouží prostřední tlačítko.

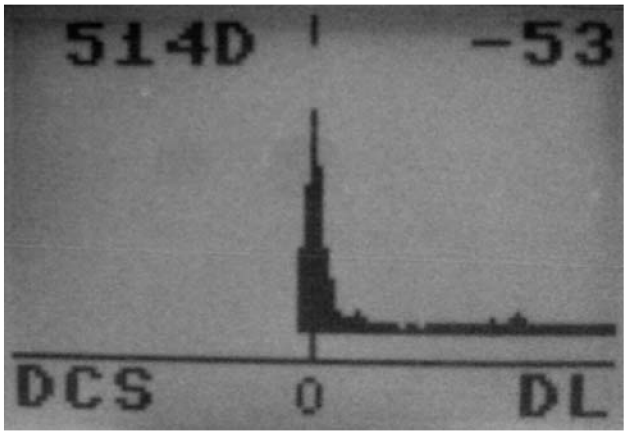


Obr. 4.1: Úroveň signálu směru downlink



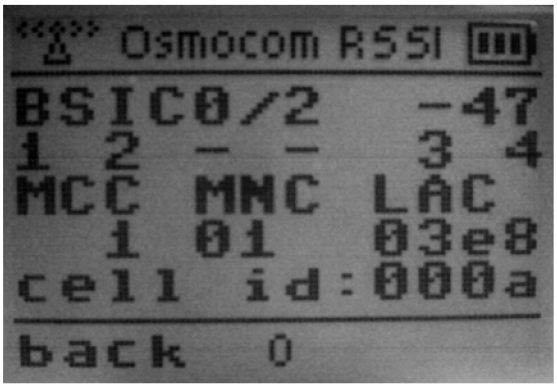
Obr. 4.2: Úroveň signálu směru uplink

Další obrazovkou (obr. 4.3) je „zobrazení spektra“ do níž se přechází stisknutím hvězdičky. Obrazovka v tomto módu zobrazuje spektrum, v tomto případě - sílu signálu na jednotlivých GSM kanálech (ARFCN). Celým spektrem se dá pohybovat (vlevo, vpravo) po jednotlivých kanálech, šipkami nahoru a dolů se celé spektrum zvětšuje a zmenšuje (volba měřítka).

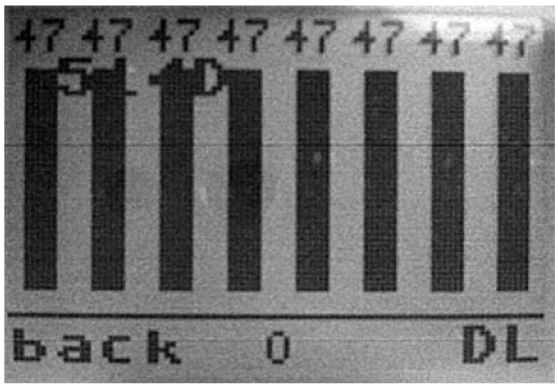


Obr. 4.3: Zobrazení spektra pomocí rssi.bin

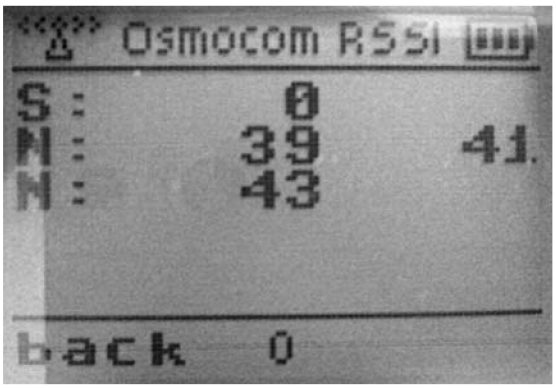
Zelené tlačítko (přijetí hovoru) se používá k přechodu na obrazovku synchronizace. Původní obrazovka (obr. 4.4) zobrazuje informace o základnové stanici (BSIC, cell id), síti (MCC, MNC, LAC) s doplňkem o aktuální sílu signálu. Při stisku tlačítka „nahoru“ se přechází do zobrazení, které zobrazuje úroveň signálu každého timeslotu (obr. 4.5). Pokud na původní obrazovce dojde ke stisku tlačítka „dolů“ zobrazí se kanály současné i sousedních buněk (obr. 4.7). Opětovným stiskem zeleného tlačítka se telefon pokusí o měření vzdálenosti a zpoždění (obr. 4.6) od dané BTS [19]. Tato volba však nefunguje zcela korektně u neupraveného filtru v telefonu.



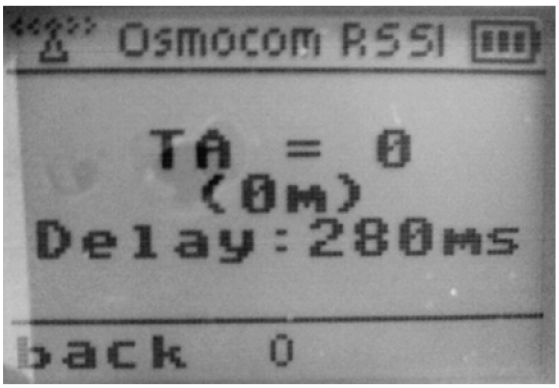
Obr. 4.4: Informace o BTS



Obr. 4.5: Úroveň signálu v timeslotech



Obr. 4.7: Výpis sousedních buněk



Obr. 4.6: Měření vzdálenosti a zpoždění

Pro analýzy na vyšší úrovni se již využívá aktivního spojení mobilního zařízení s počítačem. Na obrazovce telefonu se již žádné informace nevyskytují – pouze informace o využití první vrstvy tak jak je vidět na obr. 8. Využívají se tedy již vzpomenuté vrstvy aplikace vrstvy L2/3. Aplikace **cell_log** (v hodně zjednodušené verzi i **bcch_scan**) vypisuje informace o všech naměřených údajích. Aplikaci jde nastavit i tak, aby zachytávala pouze jeden kanál. Pro takovéto testování je však lepší využít aplikaci **ccch_scan** nebo **cbch_sniff**. Při testování těchto dvou aplikací nebyly pozorovány zásadní změny funkce. Lišil se například výpis do terminálu. Ve skutečnosti až na pár detailů vypadaly výpisy pro Wireshark jednotlivých funkcí (cell_log, bcch_scan, ccch_scan, cbch_sniff) téměř stejně. Více bude popsáno v další podkapitole na provedeném měření.

Nahrání projektu vyšší vrstvy se provádí podobně jako u předchozích příkladů. Je ale nutné si připravit 2 terminály. V prvním terminálu se nachystá aplikace pro ovládání první vrstvy. Pokud je vypnutý telefon (motorola C123) připojen k PC, je možné nahrát ovladač první vrsty osmoconem – zadáním do terminálu (umístění je stejné jako v předchozích případech osmocom-bb/src/host/osmocon:

```
~$  ./osmocon -p /dev/ttyUSB0 -m c123xor
    ../../target/firmware/board/compal_e88/layer1.compalram.bin
```

Opět pokračujeme krátkým stiskem červeného tlačítka pro nahrání softwaru do telefonu a sledujeme terminál. Správné nahrání ovladače první vrstvy je signalizováno na telefonu tak jak zobrazuje obr. 4.8. V okně prvního terminálu by se měli časem vypisovat „Battery Status.“

Obr. 4.8: Motorola C123 se aplikací Layer 1

Druhý terminál nahrává samotnou aplikaci L2/3 vrstvy podle druhu analýzy (ve skutečnosti spíše podle druhu aplikace). V terminálu je však nutné zvolit jiné umístění než v předchozích případech, protože zde již není nutné používat nástroj *osmocon*. Pokud jsme ve složce celého projektu osmocom-bb, k aplikacím se dostaneme:

```
~$  cd src/host/layer23/src/misc/
```

Jak již bylo řečeno GSMTAP balí zprávy aplikací a odesílá je do Wiresharku. Protože však PC zamítá tyto zprávy (ICMP zprávou) může se stát, že vysílač vysílající GSMTAP zprávy bude reagovat nepřiměřeně a zprávy přestane vysílat. Je proto předtím vhodné nastavit paketový filtr Linuxu k zahazování GSMTAP zpráv, protože i tak je Wireshark zachytí. Víme li tedy že GSMTAP používá port 4729 pak stačí zadat:

```
~$  sudo iptables –A INPUT –p UDP --dport 4729 –j DROP          [24]
```

Při volání jednotlivých aplikací je možné zadávat parametry, jejichž výpis je možné vyvolat parametrem *–h* u každé aplikace. Kromě aplikace *cell_log* jsou u zbylých tří aplikací možné nastavitelné parametry shodné. Aplikace *bcch_scan, ccch_scan, cbch_sniff* je tedy možné spustit s parametry, kde *–s* definuje socket, *-i* ip adresu smyčky pro odesílání GSMTAP a *–a* pro definici ARFCN kanál (v tomto případě je zvolen kanál, na kterém běží naše OpenBTS) např.:

```
~$  ./ccch_scan –s /tmp/osmocom_l2 –a 514 –i 127.0.0.1
```

Výstup terminálu je možno přesměrovat do souboru použitím *&>* s názvem souboru.

Dále můžeme použít aplikaci *cell_log*. Možných parametrů má více než než 3 předchozí aplikace. V tomto případě je to třeba parametr pro zápis analýzy do logfile:

```
~$  ./cell_log –s /tmp/osmocom_l2 –l cell_log_analyza.log –i
    127.0.0.1
```

Při běhu aplikací vyšší vrstvy vypisuje terminál první vrstvy obrovské množství surových údajů, které přeposílá ke zpracování.

Během celé analýzy mohou nastávat různé problémy – jako například komunikace s mobilním zařízením, chyby projektu apod. V tom případě nezbývá nic jiného než zkusit aplikaci vyšší vrstvy zastavit a spustit znovu, případně i aplikaci pro ovládání první vrstvy, či restart mobilního zařízení vytažením baterie.

4.2 Projekt Airprobe a Nokia 3310

AirProbe (dříve GSM-Sniffer) si klade za úkol vytvořit nástroj pro analýzu GSM mobilních standardů za účelem zjištění podrobností a nedokonalostí současného systému. Má sloužit pro vývojáře pracující s projekty OpenBTS, OpenBSC apod. Navíc spolupracuje s kity USRP. AirProbe je rozdělen na tři hlavní části (moduly).

Prvním z nich je modul Akvizice. Ten je hardwarově závislý a poskytuje všechny prostředky potřebné pro příjem a digitalizaci rádiového signálu. Protože je hardwarově závislý, má omezené funkce, avšak většinu funkcí tento modu zdědil od GNU Radia.

Modul demodulace se, jak vyplývá z jeho názvu, zabývá demodulací signálu. V tomto případě se spíše jedná o bitový tok zachycený modulem akvizice.

Posledním modulem je modul analýzy. Poskytuje všechny protokoly potřebné k dekódování a rozebrání přijatých dat.

Mezi základní funkce v rámci projektu patří sledování spektra. Za pomocí dodatečných softwarových nástrojů jako spektrální analyzér (Baudline) nebo převodní kalkulačky kanálu (arfcncalc) lze pak zachytit vysílání přímo na požadovaných kanálech. Při sledování spektra je také možno GMSK modulaci a její fázový posun.

Přímo samotná aplikace AirProbe poskytuje kolekci nástrojů a kód, umožňující základní protokolovou analýzu GSM rádiového prostředí. Problémem je, že tyto nástroje nejsou zcela dokončené a tak tedy není možné považovat aplikaci za plnohodnotný nástroj, avšak určité funkce použít lze. AirProbe může třeba pracovat jako GSM přijímač a tak může zachytávat údaje z přenosu. Tyto údaje se pak pokouší dekódovat pomocí zabudovaného A5 dekodéru. Pokud je dekódování úspěšné, je pro vizualizaci a nakládání se zachycenými daty využit Wireshark [20].

Problémem při použití Airporbe společně s kity USRP je fakt, že je projekt určen pro zachytávání na daughtboards DBSRX. Pro tuto diplomovou práci však byly k dispozici desky RFX900, WBX, LFTX a LFRX. Jako náhrada byla vyzkoušena jak WBX, tak RFX900. Ty však Airprobe rozeznat nedokázal.

Na oficiálních stránkách Airprobe je popsána i možnost dekódovat síť pomocí mobilního telefonu Nokia 3310. Zachytávání provozu zprostředkovává program *gammu* a pro dekódování dat je možné použít Wireshark, či program gsmdecode zprostředkovaný právě Airprobem. Použitý telefon pak vypisuje dění – zprávy, které sám používá. Tzn. že nezachytává dění v celé síti, ale pouze dění mezi sítí a samotným telefonem.

4.2.1 Projekt Airprobe s telefonem Nokia 3310 a Gammu

K zachytávání dat bude tedy v tomto případě sloužit Nokia 3310 propojená s PC konektorem D-SUB (standardu RS-232) sériovým kabelem s názvem „F-BUS a M2BUS s automatickým přepínáním." Jako software je použit Gammu.

Gammu je utilita pro příkazový řádek k ovládání různých druhů telefonů od českého autora. K jejímu používání slouží knihovna libGammu, která je jádrem celého projektu. LibGammu totiž obsahuje abstrakční vrstvu, která dokáže pracovat s různými telefony různých výrobců. Celá aplikace je napsána v jazyce C. Mimo získání informací o telefonu a síti dokáže gammu zálohovat či stahovat SMS i MMS, přistupovat ke kontaktům a úkolům a jejich export a import [22]. Jako pomoc pro zachytávání/dekódování je nutno ještě použít soubor *nhm5_587.txt*, který napomáhá dekódování trasovacích typů a ukládat je do GSM podadresáře.

Jednou částí Airprobe, která jde bez problému zkompilovat je program **gsmdecode**. Ten slouží k interpretaci resp. dekódování zachycených dat a to jak dalších aplikací projektu Airprobe (gsm-receiver a gsm-tvoid) tak i programu gammu. Prezentace je podobná tomu, jak to umí Wireshark. **Gsmdecode** má několik možností, jak je možné data dekódovat. Ve výchozím nastavení (-x) je po spuštění dekódován OpenGPA .xml formát. Tento formát je poskytován mobilním telefonem Nokia 3310 (obecně typy DCT3). Dále umí gsmdecode dekódovat formáty B a Bbis, tedy raw formáty kanálů SDCCH a BCCH. Ty jsou však zachytávány kitem USRP a aplikacemi Airprobu.

4.2.2 Příprava, kompilace, zachytávání a dekódování

Při instalaci gsmdecode je možné využít jak projektu Airprobe, tak i samotné oddělené verze programu. Zde je použit celý projekt Airprobe, avšak oficiální verze již není dlouhou dobu spravována. O aktualizaci a správu projektu se stará skupina Gnumonks, od kterých je použita verze v této práci. Předpokladem pro stažení a kompilaci jsou balíky knihoven (mnohé jsou již instalovány z předchozích kapitol):

```
~$  sudo apt-get install gcc g++ cpp gpp make automake git
```

Nyní bude pomocí gitu stažen Airprobe ze stránek gnumonks. Na vhodném místě v adresáři je třeba zadat:

```
~$  git clone git://git.gnumonks.org/airprobe.git
```

Z celého Airporobe stačí tedy jenom Gsmdecode, jehož kompilace (s vrácením na původní umístění posledním příkazem) se provádí zadáním:

```
~$  cd airprobe/gsmdecode
~$  ./bootstrap
~$  ./configure
~$  make
~$  cd ../../
```

Výhoda Gammu spočívá v tom, že je oficiálním instalačním balíčkem pro Ubuntu. Současně s ním se ještě instaluje balík s názvem dialog:

```
~#  sudo apt-get install gammu dialog
~#  sudo ldconfig
```

Nakonec se ještě musí vytvořit adresář (např. gsm_log), do kterého se budou ukládat zachycené logy. Do toho adresáře je také vhodné si umístit pomocný soubor (*nhm,* je uvedený i odkaz kde je možné stáhnout) pro zachytávání, aby nemusel být později hledán:

```
~$ mkdir gsm_log
~$ cd gsm_log
~$ wget svn.berlin.ccc.de/projects/airprobe/attachment/wiki/tracelog/\
   nhm5_587.txt --no-check-certificate
```

Nyní je možné kabelem propojit telefon s počítačem. Na koncovce telefonu je přepínačem nastaveno MBUS. Po propojení se nastaví ještě Gammu v konfiguračním souboru:

```
~$ sudo gammu-config
```

Konfigurace je pak takováto:

```
port = /dev/ttyS0
model = 6110
connection = mbus
synchronizetime = yes
logfile =
logformat = nothing
use_locking = yes
gammuloc =
```

V tomto kroku je již vše nastaveno a je možné začít se samotným zachytáváním. Zachytávání běží po dobu, dokud není zrušeno „ctrl+c“ a výstup je zaznamenávám do souboru *out.xml*. Předpokladem pro spuštění je umístění v adresáři zde nazvaném *gsm_log*. Zachytávání začíná po zadání:

```
~$ gammu --nokiadebug nhm5_587.txt v20-25,v18-19
```

Pokud by se delší dobu nic nedělo nebo by zachytávání skončilo chybou (o nepřipojeném zařízení), je účinným pomocníkem spustit zachytávání znovu a zkusit změnit polohu přepínače na FBUS a zpět na MBUS.

Jak již bylo řečeno, zachycená data je tedy možné dekódovat pomocí Wiresharku. Zde je dekódování provedeno gsmdecode z Airprobe. Následující příkaz provede výpis dekódovaných dat do terminálu (předpokladem je umístění z předchozího kroku):

```
~$ ../airprobe/gsmdecode/src/gsmdecode -x <out.xml
```

Volitelně je však možné zapsat dekódovaná data do souboru.

```
~$ ../airprobe/gsmdecode/src/gsmdecode -x <out.xml > decode.out
```

Také je možné případné vypisování hledaných veličin [20].

4.3 Netmonitor

Jednou se zajímavých „aplikací“ pro analýzu GSM sítě je aplikace Netmonitor. Nejedná se ale o open-source software a data získaná z této aplikace se zobrazují pouze na displeji telefonu. Tato aplikace – či spíše servisní menu telefonu je dostupné na téměř všech telefonech Nokia i u jiných výrobců. V současné době chytrých telefonů existuje spousta

aplikací, které dokáží různé informace získané ze sítě uživateli interpretovat. Ve starších telefonech však těchto informací, které telefon využívá, bývá více. U generace Nokia DCT-3 lze toto servisní menu aktivovat pomocí datového kabelu. Mezi tyto telefony patří i Nokia 3310 použitá v této práci.

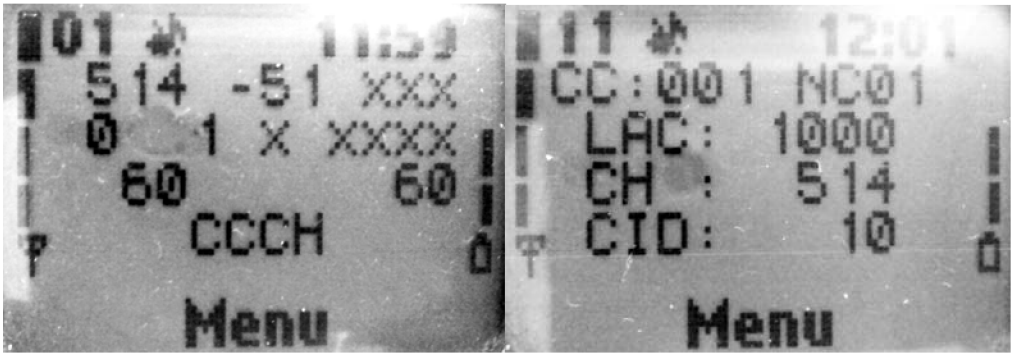
Pro aktivaci Nermonitoru je možné použít gammu. Stačí připojit telefon k PC, nastavit připojení pomocí gammu-config tak, jak je to popsáno v předchozí podkapitole. Následně už stačí jen v terminálu zadat:

```
~#  gammu -nokianetmonitor 243
```

Po jeho zadání se zpravidla na posledním místě v menu telefonu zobrazí položka Netmonitor. Po jejím vyvolání čeká telefon na zadání čísla, které určuje číslo obrazovky s danými údaji. Tyto údaje se zobrazují na „pozadí“ displeje telefonu. Nokia rozlišuje až 243 možných obrazovek, avšak zda je možné je použít záleží na typu použitého telefonu. Některá čísla jsou také nevyužita z důvodu logického oddělení daných funkcí. Pro orientaci v jednotlivých obrazovkách je vhodné využít jeden z odkazovaných návodů na Netmonitor viz [25, 26].

Po aktivaci Netmonitoru (zadáním libovolného čísla v jeho menu) se lze mezi jednotlivými obrazovkami pohybovat šipkami na telefonu. Obrazovky pak přeskakují na čísla dostupných testů. Na obrazovce je mnohdy změť čísel a symbolů, které bez návodu vůbec nic neříkají. Pokud se ale člověk orientuje ve zkratkách GSM systému, je možné využít dlouhého stisknutí hvězdičky (*). Namísto čísel se vypíšou zkratky určující co daný parametr znamená. Některé obrazovky vyžadují složitější nastavení (jako např. BTS Test – obrazovka 17). V tomto případě už je použití návodu nezbytné.

Počet zobrazovaných parametrů je úctyhodný. Hned první obrazovka (01) obsahuje parametry jako je číslo kanálu, síla přijímaného i vysílaného signálu, číslo Timeslotu, hodnota zpoždění (Timing Advance), kvalita přijímaného signálu a údaje pro reselekci C1 a C2. V dalších obrazovkách je možné postupně najít například: údaje o síti a BTS, chybovost přenosu, parametry sousedních buněk, TMSI, šifrování, údaje o baterii či softwaru telefonu, informace o handoverch a spoustu dalších [26]. Pro deaktivaci obrazovek na pozadí pohotovostní obrazovky stačí v menu Netmonitoru zadat „test“ 0.



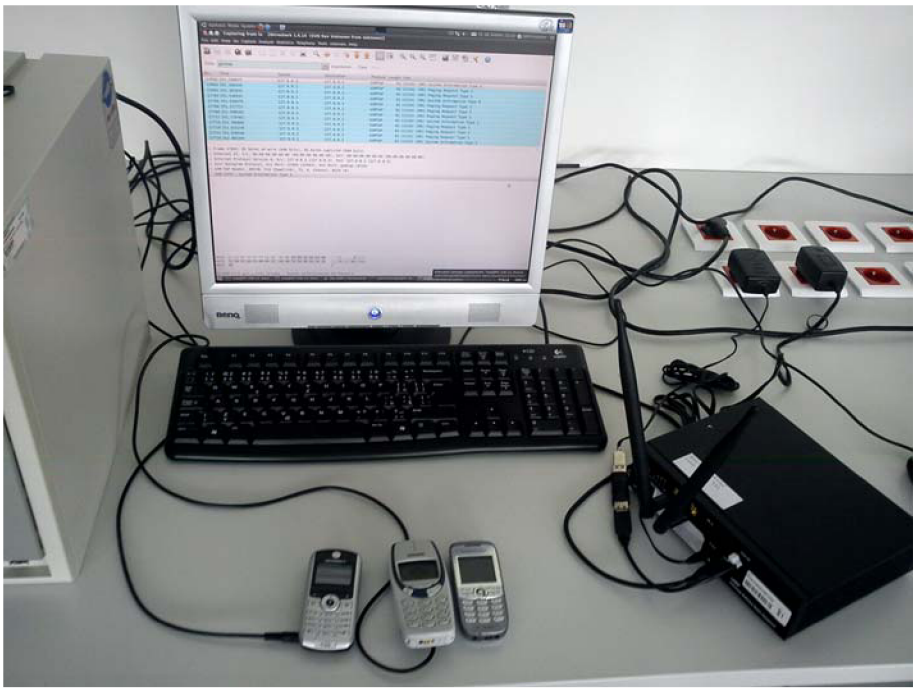
Obr. 4.9: Obrazovky 01 a 11 u aplikace Netmonitor se zachycenými údaji testovací sítě

5 Analýza vytvořené GSM sítě

Na experimentálním pracovišti (Obr. 5.1; seznam zařízení a softwaru viz. tab.5.1) byla pomocí SDR kitu USRP1 a počítače se softwarem OpenBTS vybudována GSM síť. Jak již bylo naznačeno, byla použita daughterboard WBX, s níž byl použit pro vybudování neobsazený kanál (ARFCN) 514 v pásmu GSM 1800 (DCS). Kanál 514 používá pro downlink (od BTS k mobilní stanici) frekvenci 1805,6 MHz a pro opačný směr uplink 1710,6 MHz. Pro ověření neobsazení frekvencí byl použit nástroj GNU Radia pro skenování spektra usrp_fft a také RSSI měřiče od OsmocomBB. Hodnoty signálu na daných frekvencích byly okolo -110 dB.

Tabulka 5.1: Seznam použitých zařízení a softwaru na experimentálním pracovišti

Počítač	Intel Pentium 4 (3,0 GHz), 1 GB Ram, Ubuntu 10.04 (lucid)
SDR Kit	USRP1, Daughterboard WBX, 52 MHz zdroj hodinového signálu, Antény VERT900
GNU Radio	Verze 3.4.2
OpenBTS	Verze P2.8
Asterisk	Verze 1.8.11
Wireshark	Verze 1.6.14
Airprobe; OsmocomBB	Z git://git.gnumonks.org/airprobe.git; git://git.osmocom.org/osmocom-bb/ (obojí staženo v březnu 2013)
Mobilní telefony	Nokia 3310 (+ F/Bus kabel), Sony Ericsson J210i, Motorola C123 (+ FTDI kabel). Telefony obsahují karty SIM



Obr. 5.1: Experimentální pracoviště

Analýz na vytvořené síti bylo provedeno několik. V první řadě byla vybudovaná síť proměřena opět nástrojem ***rssi.bin***. Tato analýza byla provedena pro ověření funkčnosti sítě a demonstraci funkce daného nástroje. Zaznamenané výsledky popisují obrázky Obr. 4.1, Obr. 4.2, Obr. 4.3, Obr. 4.4, Obr. 4.5, Obr. 4.6, Obr. 4.7, Obr. 4.8 v předchozí kapitole. Hodnota signálu uplink byla testována při sestaveném spojení, kdy MS vysílají. Tuto hodnotu, stejně jako hodnotu zpoždění a vzdálenosti je nutné považovat za velmi orientační z důvodu ponechání původního filtru v telefonu Motorola C123. Na Obr. 4.7 jsou vypsány sousední buňky – čísla těchto buněk jsou vyplněny v konfigurační databázi OpenBTS. Stejně je tak tomu i u hodnot při definici sítě. Např. u definice síťového LAC je v databázi zadána hodnota 1000 a synchronizační obrazovka *rssi.bin* zobrazuje tuto hodnotu hexadecimálně – tedy 03E8. Na Obr. 4.3 zobrazeného spektra je vidět parazitní jev, který se u WBX daughterboards objevoval stále a u RFX900 méně často. A to takový, že signál buzený deskou, ruší vedlejší přilehlé kanály. Tento jev se nepodařilo za dobu realizace eliminovat.

Při provádění analýzy s pomocí jiných aplikací – ať už od Osmocomu nebo Airporbe, byl použit stejný scénář:

1. Zahájení zachytávání pomocí zvolené aplikace začíná na již do sítě registrovaných telefonech Nokia 3310 (NO) a Sony Ericsson J210i (SE).
2. Po 10 až 20 sekundách od spuštění zachytávání je na SE zahájeno volání na číslo NO.
3. NO vyzvání (obr. 5.2) a po několika sekundách je hovor touto stanicí přijat.
4. Po uplynutí 10 sekund je tento hovor stanicí SE ukončen.
5. Zachytávání je vypnuto.

Pro srovnání byly aplikace *ccch_scan* (Osmocomu) a zachytávání pomocí *gammu* (pro Airprobe) spuštěny zároveň.

Obr. 5.2: Připojování hovoru - vyzvánění

Obrázek 5.3: Výpis IMSI a aktivních hovorů v konzole OpenBTS

5.1 Analýza pomocí Osmocom aplikací

U Osmocom aplikací byly porovnány na testovacím scénáři aplikace *ccch_scan, cbch_sniff, cell_log i bcch_scan*. Aplikace *bcch_scan* však nebyla zcela vhodná, protože i když byl zvolen kanál, na kterém má provádět měření, zachytávala i ostatní dostupné kanály. Proto zde nebude popisována. Navíc je jejím nástupcem *cell_log*, který funguje lépe a také s ním byly učiněny dvě měření.

5.1.1 Aplikace cell_log proměřující celé spektrum GSM

První měření aplikací *cell_log* bylo provedeno dle jejího specifického zaměření. Tím je zachycení celého spektra GSM – to znamená všech dostupných kanálů ARFCN v dané oblasti. V nastavených parametrech aplikace (viz kapitola 4.1. Použití jednotlivých aplikací Osmocomu BB) je nastaveno posílání zpráv GSMTAP do smyčky (pro Wireshark), což pro ilustraci provozu není špatné, ale u *cell_log* stačí sledovat její výpis do terminálu. V něm *cell_log* zaznamenává naměřené hodnoty úrovně signálu na ARFCN, na kterých se nějaká úroveň vyskytuje a na kanálech se silným signálem, který dokáže rozpoznat, vypíše provozovatele sítě.

Po celém měření (díky nastavení parametru *–l*) vytvoří aplikace logfile s výsledkem měření. Po každém spuštění provede aplikace rychlé proměření spektra. Výsledky jsou v logu zaznamenány, ale jsou vybrány pouze ty ARFCN, na kterých nebyl rozpoznán signál, např (první hodnota určuje číslo kanálu; dalšími hodnotami je úroveň signálu):

```
arfcn 60 -96 -107 -104 -102 -106 -96 -92 -105 -103 -103 -95 -101
```

Log dále obsahuje skupiny hodnot (uvozené sysinfo) vztahující se k jednotlivým ARFCN. Výpis obsahuje časovou značku pořízení hodnoty, BSIC, úroveň signálu, vzdálenost (TA) a nedekódované hodnoty System Information (1 až 4), např.:

```
[sysinfo]
arfcn 59
time 1369042824
bsic 3,0
rxlev -78
si1 55 06 19 00 00 00 1f 00 00 01 c0 04 1e 00 00 07 e0 00 00 5d 00 00
si2 59 06 1a 00 00 01 c0 00 00 f0 00 04 00 00 00 40 00 00 00 ff 5d 00
si2ter 49 06 03 8f 31 20 00 00 00 00 01 00 00 00 00 00 00 00 00 2b 2b 2b
si3 49 06 1b 0c e3 32 f0 10 60 50 c8 00 14 43 65 45 5d 00 00 46 ad 2b
si4 31 06 1c 32 f0 10 60 50 65 45 5d 00 00 45 2b 2b 2b 2b 2b 2b 2b 2b
ta 1
```

Pro přehled o daném stavu spektra GSM tento výpis dostačuje. V oblasti experimentálního pracoviště jsou tak aktivní ARFCN s úrovní (v dB): 9, 34, 40, 44, 45, 47, 55, 59, 75, 80, 82, 93, 94, 107, 612, 788 a taky testovací síť na ARCN 514.

5.1.2 Cbch_sniff, ccch_scan, cell_log a Wireshark

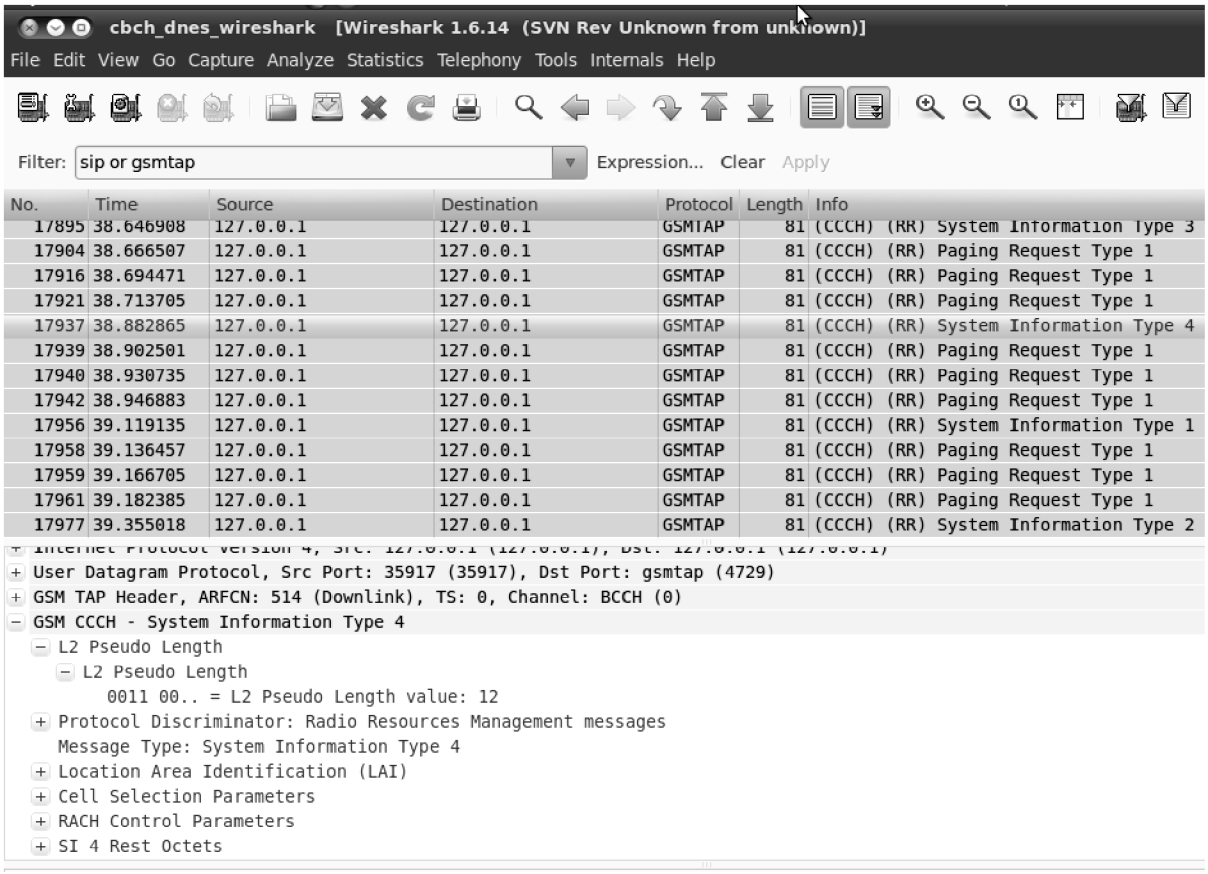
Jak již bylo zmíněno, všechny aplikace využívají pro přípravu dat pro Wireshark protokol GSMTAP, který tato data posílá do místní smyčky PC na adresu 127.0.0.1 (loopback). Na vybudované síti pomocí USRP používá tuto smyčku i OpenBTS pro komunikaci s PC (na protokolu UDP) a také Asterisk pro výpis spojování (protokoly SIP a SDP). Proto je dobré tato nežádoucí data ve Wiresharku odfiltrovat a zobrazovat si data pouze pro GSMTAP. Komunikace Asterisku ale není tak hojná a pro identifikaci místa hovoru je dobré si ji ve výpisu Wiresharku ponechat.

Pro aplikaci *cell_log* je v tomto případě (kdy plánujeme zachytávat pouze kanál 514) nutné přidat k původnímu příkazu v předchozí kapitole parametr *–A 514*. Při porovnání aplikací – respektive jejich výpisu zachyceného při provádění testovacího schématu bylo zjištěno, že GSMTAP vysílač odesílá do Wiresharku naprosto stejná data. Těmito daty je možno rozumět zprávy GSM sítě v logických kanálech CCCH a BCCH, které Wireshark dále identifikuje jako:

- Paging Request Type 1
- System Information Type 1, 2, 3 a 4
- Immediate Assigment

Dochází tak k zaznamenávání všech těchto zpráv, které však jsou pouze ve směru od BTS (downlink). Mobilní stanice uskutečňující hovor byly od sebe i naslouchací Motoroly vzdáleny jen několik desítek centimetrů. I když má naslouchací Motorola původní filtr provozu, na takto nízkou vzdálenost by měla vysílací signál mobilních stanic dobře slyšet. Z toho vyplývá, že všechny aplikace nedokážou pro GSMTAP resp. Wireshark zachytit a prezentovat komunikaci ve směru uplink.

Obr. 5.4 znázorňuje zachycená data v analyzátoru Wirereshark. V tomto případě je zde výpis po aplikaci *cbch_sniff*, velmi podobná situace panuje v prezentovaných datech po aplikacích *ccch_scan* a *cell_log*.



Obr. 5.4: Zachycená data od aplikace cbch_sniff ve Wiresharku

Pole prezentující ve výpisu komunikace u každé zachycené zprávy zdroj i cíl dat, protokol, délku dat a také informace o dekódovaném log. kanálu a typu zprávy se nemění. Kromě čísla a času zprávy se tak ve výpisu mění pouze druh zprávy. Wireshark data dekóduje jako GSMTAP protokol, z čehož vychází i zdroj a cíl smyčky a shodná délka. Typ zprávy určuje jako Radio Resouces Management Massages (RR), ale infotag log. kanálu je vždy CCCH, což ve skutečnosti není pravda. Tuto skutečnost popírá i samotný Wireshark, když v hlavičce GSMTAP určuje použitý kanál správně – je také znázorněno na obr. 5.4.

Hlavička GSMTAP dále obsahuje číslo kanálu, verzi, délku, typ přenášených dat i číslo rámce. Také jsou zde hodnoty odstupu signálu od šumu (Signal/Noise Ratio – určená v dB) a síly signálu (Signal Level – v dBm). Prezentované hodnoty těchto dvou parametrů jsou více než udivující. Například v některých zprávách dosahují hodnoty úrovně signálu hodnot 95, 206 nebo dokonce 250 dBm. Je patrné, že tyto hodnoty nejsou absolutně reálné. Chyba bude nejspíše někde u prezentace těchto dat.

Dále jsou dekódovany hodnoty dle druhu zprávy v další skupině hodnot. Zprávy druhu *Paging Request Type 1* se často opakují (jsou stejné) a ve skutečnosti neobsahují žádné konkrétní hodnoty. Pouze pár zpráv tohoto druhu obsahuje IMSI jedné z MS a určení typu TCH.

Zprávy *System Information* Type 1 až 4 obsahují podle typu různá množství zachycených hodnot. Type 1 má dvě skupiny nic neříkajících hodnot. Type 2 má navíc seznam sousedních ARFCN. Nejpočetnější na hodnoty je zpráva Type 3, která mj. nese identifikátor oblasti, popis kontrolního kanálu, či hodnoty pro handover. Podobné hodnoty obsahuje i Type 4.

Poslední zachycená *Immediate Assigment* nese informace o nastavení a volbě parametrů pro sestavení hovoru. Mezi těmito parametry je například určení, zda je TCH full rate či half rate, číslo timeslotu, zvlášť potvrzení čísla ARFCN, použití frekvenčního hoppingu atd.

5.1.3 Výstup cbch_sniff do terminálu

Aplikace L2/3 vrstvy OsmocomBB se liší v tom, co vypisují do terminálu. C***bch_sniff*** vypisuje do terminálu, že došlo k zachycení zprávy *System Information Type 1 nebo 4*. Na základě zpráv *SI Type 4* se pak snaží zachytit logický kanál CBCH (Cell Broadcast Control Channell). Na testovací síti se mu však tento kanál zachytit nedaří, jak znázorňuje výňatek výpis aplikace:

```
<0001> app_cbch_sniff.c:88 New SYSTEM INFORMATION 4
<0001> app_cbch_sniff.c:47 no CBCH chan_nr found
<0001> app_cbch_sniff.c:88 New SYSTEM INFORMATION 4
<0001> app_cbch_sniff.c:47 no CBCH chan_nr found
<0001> app_cbch_sniff.c:82 New SYSTEM INFORMATION 1
<0001> sysinfo.c:616 Now updating previously received SYSTEM
    INFORMATION 4
<0001> app_cbch_sniff.c:47 no CBCH chan_nr found
<0001> app_cbch_sniff.c:88 New SYSTEM INFORMATION 4
<0001> app_cbch_sniff.c:47 no CBCH chan_nr found
<000c> l1ctl.c:249 Dropping frame with 79 bit errors
```

Výňatek zachyceného výpisu obsahuje záměrně poslední řádek s nastalou chybou. Některá měření byla na tyto chyby bohatá.

5.1.4 Výstup ccch_scan do terminálu

Terminálový výpis apikace *ccch_scan* na testovacím scénaři byl následující:

```
<0001> app_ccch_scan.c:75 SI1 received.
<0001> app_ccch_scan.c:198 GSM48 IMM ASS (ra=0x4f, chan_nr=0x09,
    ARFCN=514, TS=1, SS=0, TSC=2)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to imsi
    M(230026700839559)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to imsi
    M(230026700839559)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to imsi
    M(230026700839559)
<0001> app_ccch_scan.c:198 GSM48 IMM ASS (ra=0x24, chan_nr=0x0a,
    ARFCN=514, TS=2, SS=0, TSC=2)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to imsi
    M(230026700839559)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to imsi
    M(230026700839559)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to imsi
    M(230026700839559)
```

Testovaní bylo pro ověření provedeno několikrát. Vždy byla zachycená situace podobná. V úvodu je oznámeno zachycení zprávy *System Information Type 1*. Všechny další zprávy se vážou k sestavování hovoru. Žádné další zprávy aplikace i při opakovaných pokusech nevypsala.

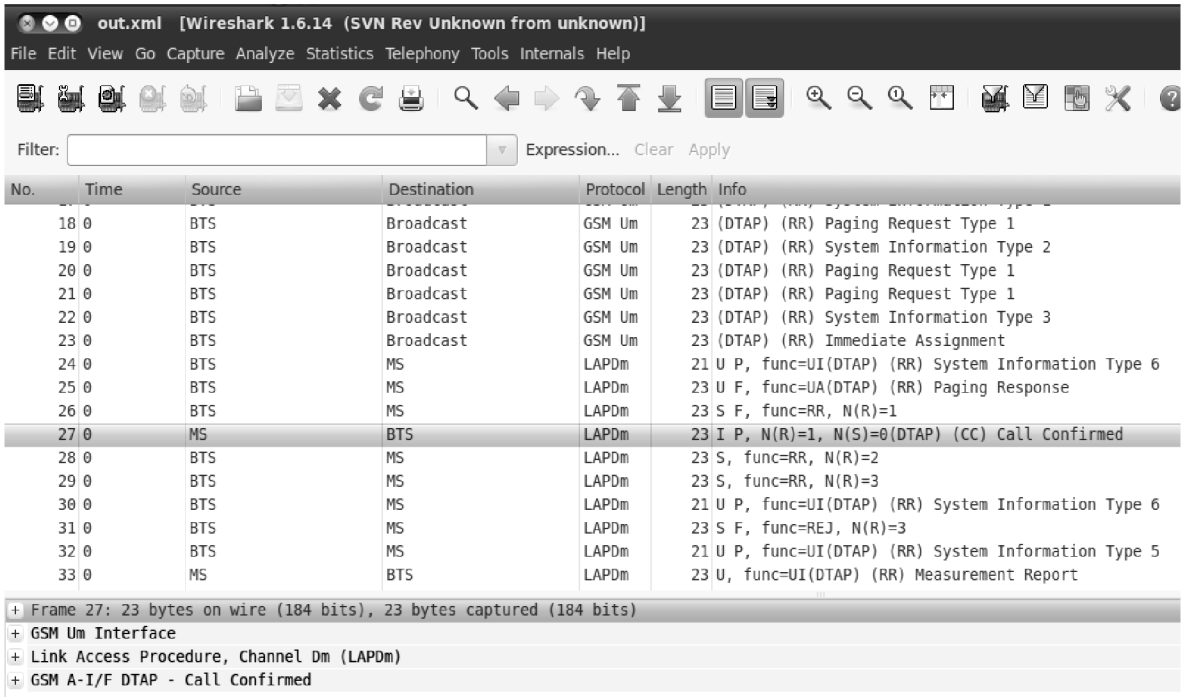
Při sestavování hovoru aplikace oznamuje jednak přidělený Timeslot (TS), ARFCN a kód Tranining Sequence (TSC). V dalších zprávách (pomocí pagingu) BTS vyhledává cílovou MS oznámením jeho IMSI a také druhem použitého TCH.

5.1.5 Cell_log zachytávající jeden kanál

Úprava aplikace *cell_log* pro zachytávání jednoho ARFCN kanálu má za následek to, že terminál bude zaplavován zprávami o měření úrovně signálu na zvoleném kanálu. Je-li zvoleno zachýtávání do logu jsou všechna měření zaznamenávána a výpis pak vypadá stejně jako u nespecifikovaného kanálu (viz předchozí kapitola). Společně se zprávami uvozené *sysinfo* obsahuje výpis též informace o měření síly signálu uvozené *power*.

5.2 Analýza telefonem Nokia

Při používání Nokie 3310 pro analýzu GSM sítě je patrná následující nevýhoda. Data je nejprve nutno zachytit (pomocí *gammu*) a až následně po skončení zachytávání dekódovat Wiresharkem či programem *gsmdecode* z projektu Airprobe. Oba dekodéry je však možné porovnat. Zachytávají se data přenášená v obou směrech (downlink i uplink) mezi MS Nokia a BTS. Žádná další komunikace na testovací síti tak zaznamenávána není.



Obr. 5.5: Komunikace mezi BTS a MS Nokia dekódována pomocí Wireshark

5.2.1 Dékódování dat Wiresharkem

Výhodou Wiresharku je jeho přehlednost – viz Obr. 5.5 zobrazující začátek budování spojení. Na obrázku je patrné rozlišení zpráv zdroje (source) a cíle (destination) zpráv a dochází k rozlišování i všesměrových zpráv, ale pouze těch, které využila Nokia. Zachycená škála zpráv ve Wiresharku je větší než u aplikací OsmocomBB a jsou to zprávy typu:

- Paging Request Type 1
- System Information Type 1, 2, 3, 4
- System Information Type 5, 6 (během hovoru)
- Immediate Assigment
- Paging Response
- Call Confirmed
- Measurement Report (během hovoru)
- Disconect, Release, Release Complete (při ukončování hovoru)
- A přenášené informační zprávy a funkce

Každá zpráva dekódovaná Wiresharkem obsahuje pole „**GSM Um Interface**.“ V něm se nachází minimálně informace o směru vysílání (Direction) a využívaném kanál (CCCH, BCCH, FACCH nebo SACCH). U zpráv System Information, Paging a Immediate Assigment obsahuje toto pole ARFCN, pásmo, frekvenci, BSIC, číslo rámce TDMA, počet chyb a časový posun.

Další pole u dekódovaných dat obsahuje hodnoty dle protokolu – GSM Um na Broadcast nebo LAPDm na hovor. U zpráv zachycených během hovoru (včetně jeho budování a ukončení) se vyskytují data obou zmíněných protokolů, avšak parametry u LAPDm neříkají o síti nic. Toto pole navíc definuje zachycenou zprávu.

Hodnoty ve zprávě *Paging Request Type 1* nejsou nijak zajímavé. Většina z nich je – dá se říct prázdná a opakující se. Neprázdné zprávy (v tomto případě 2) obsahují IMSI dané MS a určení TCH.

Zprávy *System Information* Type 1 až 4 jsou stejné jako v předchozí podkapitole u OsmocomBB aplikací. Zde jsou navíc zachyceny zprávy Type 5 a 6, které jsou jejich obdobou. Rozdíl je v tom, že jsou tyto zprávy zasílány stanici během uskutečněného hovoru. K tomu využívají taky jiný logický kanál.

Immediate Assigment nese stejné informace o nastavení a volbě parametrů pro sestavení hovoru jako u Osmocomu. Pro připomenutí je mezi těmito parametry například určení, zda je TCH full rate či half rate, číslo timeslotu, zvlášť potvrzení čísla ARFCN, použití frekvenčního hoppingu atd.

Bezpochyby zajímavou zprávou je *Paging Response*. Ta nese množství informací týkající se šifrování či použitých algoritmů. Celá zpráva je takovým informátorem o nastavení sítě a podporovaných funkcích. Navíc je tato zpráva směrovaná pro Nokii a tak nese její IMSI. MS na tuto zprávu odpovídá *Call Confirmed* nesoucí informace o možnostech MS.

Během hovoru odesílá MS základnové stanici *Measurement Report*. Jak název napovídá MS proměřuje spojení s BTS. Mezi měřenými parametry je síla přijímaného signálu a také kvalita přijímaného signálu. Kvalita (RXQUAL) je ve dvou hodnotách: full – kdy dochází k měření všech burstů a sub – kdy jsou měřeny jen skutečně vyslané při zapnuté funkci DTX (Downlink Discountinuos Transmittion). Jak ale Wireshark zobrazuje, tak v testované síti funkce DTX použita není. Kvalita se vyjadřuje v procentech jako Bit Error Rate (BER).

5.2.2 Dekódování dat pomocí Airprobe

Dekódovací program Airprobu Gsmdecode je určen na výpis informací do terminálu. Ten však nedokáže vypsat takové množství řádků a proto je dobré přesměrovat výstup do souboru. Zachycená a dekódovaná data tvoří v tomto případě 9340 řádků a tak jsou takto

dekódovaná data značně nepřehledná. Ve Wiresharku může být zhruba obdobné množství dat, ale rozdělení po jednotlivých zprávách (v počtu 370) situaci značně zpřehlední.

Ve výpisu dekódovaných dat jsou taktéž hexa hodnoty (vždy zhruba 2 řádky na zprávu) z těch však nic vyčíst nelze. Oproti Wiresharku Gsmdecode nedekóduje (nezobrazuje) hodnoty „Um Interface.“ U všech dat tak chybí například ARFCN. Zprávy a jejich hodnoty jsou však naprosto shodné s Wiresharkem. Data také nejsou tak roztříděna jako například druhá zachycená (a dekódovaná) zpráva, která vypadá takto:

```
HEX l2_data_out_Bbis:462 Format Bbis DATA
000: 31 06 1c 00 f1 10 03 e8  - 60 40 79 04 00 2b 2b 2b
001: 2b 2b 2b 2b 2b 2b 2b
    0: 31 001100-- Pseudo Length: 12
    1: 06 0------- Direction: From originating site
    1: 06 -000---- 0 TransactionID
    1: 06 ----0110 Radio Resouce Management
    2: 1c 00011100 RRsystemInfo4-C
    3: 00 001      Mobile Country Code (UNKNOWN)
    4: f1 01f      Mobile Network Code ((null))
    6: 03 1000     [0x03e8] Local Area Code
    8: 60 011----- Cell Reselect Hyst. :  6 db RXLEV
    8: 60 ---xxxxx Max Tx power level: 0
    9: 40 0------- No additional cells in SysInfo 7-8
    9: 40 -1------ New establishm cause: supported
    9: 40 --xxxxxx RXLEV Access Min permitted = -110 + 0dB
   10: 79 01------ Max. of retransmiss : 2
   10: 79 --1110-- slots to spread TX : 32
   10: 79 ------0- The cell is barred  : no
   10: 79 -------1 Cell reestabl.i.cell: not allowed
   11: 04 -----1-- Emergency call EC 10: not allowed
   11: 04 00000--- Acc ctrl cl 11-15: 0 = permitted, 1 = forbidden
   11: 04 ------00 Acc ctrl cl  8- 9: 0 = permitted, 1 = forbidden
   11: 04 -------0 Ordinary subscribers (8)
   11: 04 ------0- Ordinary subscribers (9)
   11: 04 -----1-- Emergency call (10): Everyone
   11: 04 ----0--- Operator Specific (11)
   11: 04 ---0---- Security service (12)
   11: 04 --0----- Public service (13)
   11: 04 -0------ Emergency service (14)
   11: 04 0------- Network Operator (15)
   12: 00 00000000 Acc ctrl cl  0- 7: 0 = permitted, 1 = forbidden
   12: 00 00000000 Ordinary subscribers (0-7)
l2_RRsystemInfo4C:1638 TRUNKATED (0x0xbfff261d - 0x0xbfff261d)
```

I když se zdá použití nástroje Gsmdecode neperspektivní, existuje zde důvod proč jej použít. Jak popisuje návod viz [20], je možné pomocí gsmdecode „lovit“ z dat určité informace. Použitím linuxu a jeho funkcí lze vypisovat a zacházet s daty, které jsou pro uživatele zrovna důležité a nemusí je tak složitě hledat. Například je možné vyhledat IMSI, ARFCN, LAC, či různě zacházet s daty jako je měření kvality nebo úrovně signálu, kdy je možné si je například vypsat a seřadit.

Na ukázku bude provedeno vypsání IMSI. Pro jednoduchost budou vypsány celé řádky obsahující řetězec IMSI. Nacházíme-li se ve složce zachycených dat (zde byla označena jako složka gsm_log, která se nachází v adresáři z celým naším projektem), provede se výpis IMSI zadáním:

```
~$  ../airprobe/gsmdecode/src/gsmdecode -x <out.xml | grep "7/odd"
```

Výpis vypadá na tomto analyzovaném souboru takto:

```
 6: 03 -------- ID(7/odd): 230026700839559
 6: 03 -------- ID(7/odd): 230026700839559
12: 03 -------- ID(7/odd): 230026700839559
```

Pokud bychom chtěli dostat čistě hodnoty kvality přenosu v BER můžeme použít awk, které nám vypíše žádanou hodnotu daného řádku (zde devátou). Dále je použita funkce *sort* k seřazení hodnot a *uniq -c* k sečtění výskytů dané hodnoty:

```
~$  ../airprobe/gsmdecode/src/gsmdecode -x <out.xml | grep "Quality
    Full" | awk '{print $9}' | sort | uniq -c
```

Naměřené hodnoty BER jsou takovéto:

```
94 ~0.14%%
22 ~0.57%%
 6 ~0.s28%%
 4 ~1.13%%
 2 ~18.10%%
20 ~2.25%%
 3 ~4.53%%
 5 ~9.05%%
```

To znamená, že nejvíce činila hodnota RXQUAL 0,14% BER a to u 94 zpráv.

Podobným způsobem tak lze vypsat jakoukoliv hodnotu. Předtím je ale nutné si zachycený a dekódovaný soubor projít k určení přesné fráze pro vyhledávání.

5.3 Vytvořené skripty pro automatizaci analýz

V rámci této diplomové práce byly vytvořeny dva skripty pro BASH linuxu. Ty mají za úkol jednak práci s projekty zjednodušit a dále ji tak automatizovat.

5.3.1 Skript pro práci s OsmocomBB

Skript pojmenovaný *run_osmo.sh* si klade za úkol odstranit zacházení se složitými příkazy pro nahrávání aplikací do zařízení. Příkazy jsou stejné jako v příslušné kapitole. Skript je vytvořen na Ubuntu 10.04 a je určen pro zařízení Motorola C123 přpojené přes USB port. Nastavení správné cesty k projektu OsmocomBB je možné provést hned na čtvrtém řádku skriptu.

Po spuštění aplikace může být uživatel vyzván k zadání hesla uživatele root. Poté se vybírá daná aplikace zadáním jednoho z písmen (a enteru) dle možností takto:

- **h** – spouští *hello_world*
- **r** – spouští *rssi.bin*
- **a** – spouští *cell_log* se skenováním všech ARFCN
- **l** – spouští *cell_log* na jednom ARFCN
- **c** – spouští *ccch_scan*
- **s** – spouští *cbch_sniff*

U všech aplikací je nutná spolupráce uživatele, který stiskne krátce červené tlačítko k nahrání programu do zařízení.

Volby pro nahrávání programů *hello_world* a *rssi.bin* neprovází žádná další aktivita. Mohou pouze nastat standardní problémy – jako zaseknutí nahrávání firmware.

U aplikací určených pro jeden kanál (tedy *l, c, s*) vyžaduje skript navíc zadání ARFCN. Pro čtyři volby (*a, l, c, s*) spouští skript program Wireshark pro analýzu zachyceného provozu přes GSMTAP. Wireshark je nastaven tak, aby došlo ihned po jeho spuštění k zachytávání na smyčce (loopback). Na jeho spuštění čeká skript 15 sekund. Uživatel může během této doby již nastavit filtr na protokol gsmtap. Protože jsou tyto aplikace L2/3, spouští skript další terminál. Ten pak používá jako ovladač Layer1 a původní terminál používá pro samotnou aplikaci. V tomto případě se nahrávání povoluje (stiskem červeného tlačítka) hned při otevření druhého terminálu.

Všechny analýzy je možné kdykoliv ukončit zadáním **ctrl+c** v terminálu (terminálech), případně dále zastavit zachytávání Wiresharkem nebo jej zavřít.

U aplikací s *cell_log* se generuje také výsledný log se získanými hodnotami. Ty se ukládají do stejné složky, ve které se nachází skript *run_osmo.sh*. Pro aplikaci na jeden kanál má log název „cell_log_kanal.log“. Pro všechny kanály je to soubor „cell_log_analyza.log.“

5.3.2 Skript pro automatizaci analýzy na Nokii

Skript pojmenovaný *n_gammu.sh* automatizuje kroky pro analýzu, které jsou popsány v kapitole 4.2. Využívá se tedy Nokia 3310, gammu a gsmdecode od Airprobe. Před prvním spuštěním je nutné zkontrolovat a opravit cestu k programu gsmdecode.

Skript je určen do separátního adresáře, ve kterém se bude nalézat spolu s klíčovacím souborem *nhm5_587.txt*. Jeho přítomnost testuje první podmínka skriptu. Pokud tato podmínka projde, spustí se konfigurační soubor *gammu-config*.

Po konfiguraci se spustí samotné zachytávání. Ukončení zachytávání provede uživatel stiskem „enter.“ Veškerá zachycená data se uloží do adresáře *out/*, který skript také vytvoří. Mezi vytvořenými soubory pak jsou:

- all.out – obsahující všechny zachycené a dekódované data
- imsi.out – vypíše zachycené IMSI Nokie
- quality.out – vypíše kvalitu (Bit Error Rate) během hovoru
- rxlev.out – vypíše hodnoty síly signálu během hovoru

Aby došlo k naplnění souborů imsi, quality a rxlev, musí zachytávající Nokie provést hovor. Daná data jsou totiž přenášena během hovoru. Nakonec skript tyto tři soubory otevře.

V souboru *quality.out* a *rxlev.out* jsou data seřazena dle výskytu. Tzn., že v prvním sloupci je počet výskytů hodnoty, která je vypsána hned vedle v druhém sloupci.

Díky tomuto skriptu tak nemusíme zadávat množství příkazů.

ZÁVĚR

Cílem této diplomové práce bylo ověření si a případně získání znalostí o standardu GSM. Bylo zde pojednáno o architektuře GSM sítě – jejich jednotlivých částech. Protože byla práce zaměřena také na její analýzy, byly popsány kanály, které GSM využívá. Neméně důležité jsou identifikátory uživatelů v síti a také popis zajištění bezpečnosti.

Jednou z hlavních částí řešeného problému v této práci je realizace vlastní GSM sítě za pomocí kitu USRP1 a open-source softwarů. Hlavním softwarem byl OpenBTS, který vytváří rozhraní klasické BTS v síti. Pro propojení a zacházení mezi kitem a OpenBTS slouží software GNURadio, určený pro softwarové rádia. Posledním použitým softwarem je pobočková ústředna Asterisk, sloužící k propojování hovorů.

Realizaci a také následná analýza takovéto „amatérské“ GSM sítě provázela řada neduhů a problémů. Tyto praktické zkušenosti provázejí celou práci. Spouštení projektu bývá zdlouhavé a tak byl upraven skript, který celou síť spustil.

Druhá z hlavních částí práce se zabývá analýzami GSM sítí pomocí open-source projektů. Těmito projekty byl OsmocomBB a Airprobe spojený s utilitou gammu. Jako doplněk analýzy byl zmíněn Netmonitor pro mobilní telefony Nokia, avšak ten není open-source softwarem. U všech projektů je popsána jejich instalace a zprovoznění.

U projektu OsmocomBB byly pro analýzu GSM sítí vybrány všechny možné aplikace pro analýzu – tedy rssi.bin, ccch_scan, cell_log (eventuelně bcch_scan) a cbch_sniff. Jejich výstupy byly popsány a porovnávány (na síti vytvořené pomocí OpenBTS) pomocí testovacího scénáře, definovaném v úvodu kapitoly 5.

Na obdobném scénáři byla provedena analýza pomocí utility gammu a části projektu Airprobe – s názvem gsmdecode – pro dekódování zachycených dat Nokií 3310. Výstupy obou projektů jsou v textu porovnány, avšak lepší projekt pro GSM analýzu na testovací síti určen nebyl. Oba dva z projektů mají své určité výhody a nevýhody.

Například u projektu OsmocomBB bylo očekáváno více výsledků (zejména i ve směru od mobilních účastníků). Zachytávaná data pak byla ovlivňována velkou nestabilitou celého projektu a taky faktem, že testovací GSM síť, není až tak úplně kvalitní plnohodnotnou sítí.

U projektu Airprobe je limitující fakt, že části projektu určené k zachytávání dat, jsou určeny pouze na jeden (resp. dva) druhy daughterboards pro kit USRP. Takto pak byla zachytávaná data spojena vždy jen s Nokií, které toto zachytávání prováděla.

V rámci práce byly vytvořeny skripty pro zjednodušení práce s oběma projekty. Navíc byla také vytvořena laboratorní úloha, která se nachází v přílohách (označena jako příloha B).

SEZNAM POUŽITÉ LITERATURY

[1] HARTE, Lawrence. *Introduction to global system for mobile communication (GSM): Physical channels, network, and operation*. Fuquay-Varina, NC: Althos, 2005. ISBN 978-193-2813-043. Dostupné z: http://www.scribd.com/doc/47864497/28612852-Introduction-to-GSM

[2] HANUS, Stanislav. *Skripta do předmětu BRMK: Rádiové a mobilní komunikace*. Brno: Ústav radioelektroniky FEKT VUT, 2005.

[3] FLÉGL, Petr. *Základní prehled zabezpecení GSM*. Praha, 2007. Dostupné z: http://radio.feld.cvut.cz/personal/mikulak/MK/MK07_semestralky/prehled_zabezpeceni_GSM.pdf. Semestrální práce. ČVUT FEL.

[4] ŽALUD, Václav. *Softwarové a kognitivní rádio* [online]. Praha: ČVUT FEL [cit. 2012-12-11]. Dostupné z: http://radio.feld.cvut.cz/courses/X37ZRD/materialy.php - pod názvem SDR_a_CR.pdf

[5] SDR Radio - Software Defined Radio. *Co je vlastně SDR?* [online]. 2008 - 2012 [cit. 2012-12-11]. Dostupné z: http://sdr.ipip.cz/

[6] Universal Software Radio Peripheral. In: *Wikipedia: the free encyclopedia* [online]. 2001-2012, 2.11.2012 [cit. 2012-12-11]. Dostupné z: http://en.wikipedia.org/wiki/Universal_Software_Radio_Peripheral

[7] USRP: FAQ Intro. *GNU Radio* [online]. 2006-2011 [cit. 2012-12-11]. Dostupné z: http://gnuradio.org/redmine/projects/gnuradio/wiki/UsrpFAQIntro

[8] *GNU Radio: WikiStart - gnuradio.org* [online]. 2006-2012 [cit. 2012-12-11]. Celý web. Dostupné z: http://www.gnuradio.org

[9] RANGEPUBLIC. *OpenBTS: Public Release* [online]. Do 2012 [cit. 2012-12-11]. Celý web. Dostupné z: https://wush.net/trac/rangepublic/wiki

[10] OpenBTS. In: *Wikipedia: the free encyclopedia* [online]. 2001-2012, 7.11.2012 [cit. 2012-12-11]. Dostupné z: http://en.wikipedia.org/wiki/OpenBTS

[11] BACK, Andrew. Building a GSM network with open source. *The H Open: News and Features* [online]. 2012, 26.3.2012 [cit. 2012-12-11]. Dostupné z: http://www.h-online.com/open/features/Building-a-GSM-network-with-open-source-1476745.html

[12] RANGE NETWORKS. *OpenBTS P2.8: Users' Manual* [online]. 1. vyd. 2011 [cit. 2012-12-11]. Dostupné z: https://wush.net/trac/rangepublic/attachment/wiki/WikiStart/SoftwareP2.8Manual.pdf

[13] FÄHNLE, Matthias. *Software-Defined Radio with GNU Radio and USRP/2 Hardware Frontend: up and FM/GSM Applications* [online]. Ulm, Germany, 2010 [cit. 2012-12-11]. Dostupné z: http://www.hs-ulm.de/users/derr/_downloads/SDR_GNURadio_USRP_Feb2010.pdf. Bachelor Thesis. Hochschule Ulm, University of Applied Sciences; Institute of Communication Technology. Vedoucí práce Frowin Derr.

[14] WIJA, T., D. ZUKAL a M. VOZŇÁK. CESNET. *Asterisk a jeho použití: Technická zpráva* [online]. Praha, 2005 [cit. 2012-12-11]. Dostupné z: http://www.cesnet.cz/akce/20051115/pr/voz05_asterisk.pdf

[15] *GNURadio* [online]. do 2012, 14.8.2012 [cit. 2012-12-11]. Web a příslušná dokumentace v odkazech na spodní straně stránky. Dostupné z: http://opensource.telkomspeedy.com/wiki/index.php/GNURadio

[16] Using GNURadio and USRPs in a VM. In: KEMPKE, B. *ProjectWiki* [online]. 2011, 30.9.2011 [cit. 2012-12-11]. Dostupné z: http://novelflash.com/wiki/index.php?title=Using_GNURadio_and_USRPs_in_a_VM

[17] GNURadio fresh Linux installation. *Ruby Forum* [online]. 2010, 2011 [cit. 2012-12-11]. Dostupné z: http://www.ruby-forum.com/topic/204335

[18] *GSMdump* [online]. 2010 [cit. 2012-12-12]. Celý web. Dostupné z: http://www.gsmdump.de/

[19] *Osmocom projekt* [online]. - 2012 [cit. 2012-12-12]. Celý web, všechny projekty. Dostupné z: http://osmocom.org/

[20] *Airprobe* [online]. Do 2012 [cit. 2012-12-11]. Dostupné z: https://svn.berlin.ccc.de/projects/airprobe/

[21] GLENDRANGE M., HOVE K., HVIDEBERG E. *Decoding GSM* [online]. Norway, 2010 [cit. 2013-05-14]. Dostupné z: http://www.scribd.com/doc/105399614/50/AirProbe. Master of Science in Communication Technology. Norwegian University of Science and Technology: Department of Telematics.

[22] ČÍHAŘ, Michal. *[GW]ammu: Gammu* [online]. © 2003 - 2013 [cit. 2013-05-14]. Dostupné z: http://cs.wammu.eu

[23] SLADKÝ, Jan. Procházka logickými kanály GSM. *Idnes.cz* [online]. 2000 [cit. 2013-05-14]. Dostupné z: http://mobil.idnes.cz/prochazka-logickymi-kanaly-gsm-dm4-/mob_tech.aspx?c=A000120_0006348_mob_tech

[24] HRACH, Jan. Na GSM s OsmocomBB. *Abc Linuxu* [online]. 10. 7. 2012 [cit. 2013-05-17]. Dostupné z: http://www.abclinuxu.cz/clanky/na-gsm-s-osmocombb

[25] *Nokia NetMonitor Manual* [online]. 11.11.2002 [cit. 2013-05-18]. Dostupné z: http://www.nobbi.com/download/nmmanual.pdf

[26] Servisní menu pro telefony Nokia 3310 a jim podobné. *Pontiac_CZ - homepage* [online]. 2007 [cit. 2013-05-18]. Dostupné z: http://pilatus.unas.cz/index.php?sekce=gsm&stranka=netmonitor

SEZNAM ZKRATEK

ADC	Administrative Centre
AGCH	Access Grant Control Channel
AGPL	Affero General Public License
API	Application Programming Interface
ARFCN	Absolute Radio-Frequency Channel Number
AuC	Authentication Centre
BCCH	Broadcast Control Channel
BER	Bit Error Rate
BSC	Base Station Controller
BSIC	Base Station Identity Code
BSS	Base Station Subsystem
BTS	Base Transceiver Station
CCCH	Common Control Channel
CEPT	Conference of European Posts and Telecommunications
ČTÚ	Český telekomunikační úřad
DCCH	Dedicated Control Channel
DCS	Digital Cellular Service
DTX	Downlink Discountinuos Transmittion
DL	DownLink
EIR	Equipment Identity Register
EDGE	Enhanced Data rates for GSM Evolution
FACCH	Fast Associated Control Channel
FCCH	Frequency Correction Channel
FPGA	Field Programmable Gate Array
GMSK	Gaussian Minimum Shift Keying
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
GSMTAP	GSM Transferred Account Procedures
HLR	Home Location Register
ID	Identity Code
IP	Internet Protocol
ICMP	Internet Control Message Protocol
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
LAI	Location Area Identity
LAC	Location Area Code
LAP	Link Access Procedure
MCC	Mobile Country Code
MNC	Mobile Network Code

MS	Mobile Station
MSC	Mobile Switching Center
NMC	Network Management Centre
NSS	Network Switching Subsystem
OMC	Operational and Maintenance Centre
OSMOCOM BB	Open Source Mobile Communication Base Band
OSS	Operation Support Subsystem
PBX	Private Branch eXchange
PC	Personal Computer
PCH	Paging Channel
PID	Process Identifier
RACH	Random Access Channel
RR	Radio Resouces (Management Massages)
RSSI	Received Signal Strength Indication
Rx	Receive
SACCH	Slow Associated Control Channel
SCH	Synchronization Channel
SDCCH	Stand-alone Dedicated Channel
SDP	Session Description Protocol
SDR	Software Defined Radio
SI	System Information
SIM	Subscriber Identity Module
SMS	Short Message Service
SIP	Session Initiation Protocol
TA	Timing Advance
TCH/, /F, /H	Traffic Channel, Full Rate, Half Rate
TDMA	Time Division Multiple Access
TMSI	Temporary Mobile Subscriber Identity
TS	Time Slot
TSC	Tranining Sequence
Tx	Transmit
UDP	User Datagram Protocol
UHD	USRP Hardware Driver
UL	UpLink
UMTS	Universal Mobile Telecommunications System
USRP	Universal Software Radio Peripheral
VLR	Visitor Location Register
VM	Virtual Machine
VoIP	Voice over Internet Protocol

SEZNAM PŘÍLOH

A PRVNÍ PŘÍLOHA

A.1 Obsah přiloženého CD

Přiložené CD obsahuje elektronickou verzi této práce (Kilian_diplomova_prace.pdf).

Dále obsahuje v adresáři „*skripty/*“ upravený skript pro spuštění testovací GSM sítě (obts_startup.sh). Dále je v adresáři skript pro zjednodušení práce s aplikacemi OsmocomBB na Motorole C123 (run_osmo.sh). Třetím, zde nalézajícím se, je kript pro automatizaci měření a výpis kvalitativních parametrů pro měření na Nokii 3310 (n_gammu.sh).

V dalším adresáři „*osmocom_decode/*“ se pak nalézají data zachycená aplikací OsmocomBB na testovacím scénaři. Vyjímkou je však soubor „cell_log_vse.log“ s příslušným „cell_log_vse_wireshark“. Na těch bylo provedeno měření bez specifikace kanálu. Jak je již naznačeno ke každé aplikaci patří dva soubory, kdy v jednom je zachycen výstup programu (*.log) a příslušné data zachycené pro Wireshark (*_wireshark).

Posledním adresářem je „*gammu_decode/*“. V něm se nalézá klíčovací soubor pro nokii (nhm5_587.txt) a zachycený výstup (out.xml). Ten je dekódován pomocí gsmdecode do výstupů jako z testovacího skriptu. Takže se v této složce nalézá ještě další adresář „*out/*“ obsahující všechny dékódovaná zachycená data (all.out), imsi zachytávající stanice (imsi.out), naměřenou kvalitu BER (quality.xml) a úrovně signálu během hovoru (rxlev.out).

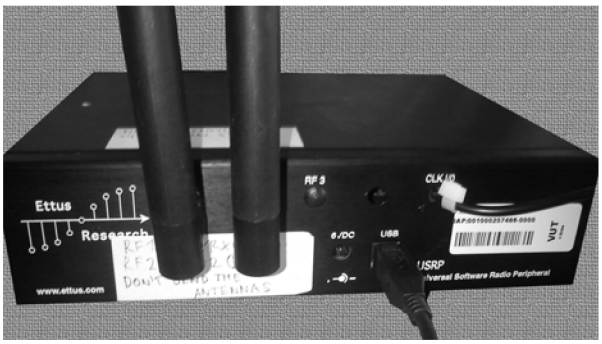
B DRUHÁ PŘÍLOHA – LABORATORNÍ ÚLOHA

A OPENBTS

V této úloze si vybudujete pomocí kitu softwarového rádia a open-source softwaru vlastní experimentální síť s jednou základnovou stanicí (BTS).

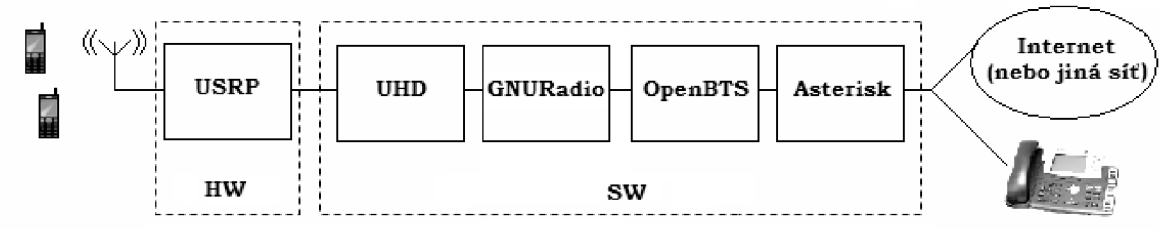
A1. Hardware

Softwarově definované rádio (SDR) je technologie, využívající digitální zpracování signálu. Hardwarově jsou v těchto zařízeních provedeny jednoduché vstupní obvody, hlavní funkce, rádiové přijímače a vysílače. Dále obsahuje programovatelné logické obvody a A/D převodníky, jimiž je signál řízen a zpracováván. Je tedy softwarem možné např. nastavovat frekvence, aktivovat (de)modulátory, filtry, komunikační protokoly atd.



Obrázek 1. Kit USRP1

Kit USRP1 od firmy Ettus Research je jedním ze zástupců SDR. Obsahuje 4 vysokorychlostní 12-ti bitové analogově-digitální konvertory s rychlostí 64MS/s (milionu vzorků za sekundu); dále 4 vysokorychlostní převodníky digitálně-analogové se 14 bity na vzorek dosahující rychlost 128MS/s. Veškeré vstupy a výstupy jsou připojeny k programovatelnému hradlovému poli (FPGA). Aby mohl celý kit komunikovat s počítačem, je zde provedeno připojení přes USB port (verze 2.0). USRP1 je doplněn u tzv. Daughterboard, která specifikuje rádiové vlastnosti přijímače. Zde je to deska WBX, která funuje na frekvencích od 50 MHz do 2,2 GHz s výstupním výkonem okolo 100 mW. Kit navíc obsahuje externí generátor hodinového impulsu pro zajištění dobré synchronizace s mobilními stanicemi (MS).



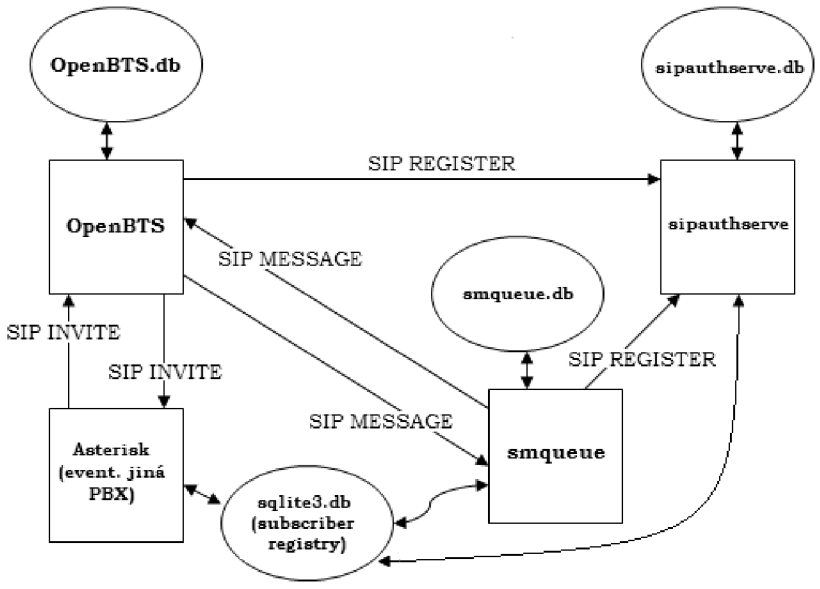
Obrázek 2. Koncept funkce SDR s OpenBTS

A2. Software realizující GSM síť

K primárnímu zacházení s kitem USRP slouží **UHD** (USRP Hardware Driver), který poskytuje řadu ovladačů a také aplikační programovací rozhraní. Na toto rozhraní je napojen toolkit **GNU Radio**. Ten poskytuje signálové bloky pro softwarové řešení filtrů, kanálového kódování, synchronizační prvky, ekvalizéry, de/modulátory, vocodéry, de/kodéry a taky spoustu dalších prvků, které se v rádiových systémech vyskytují. Je i možné vytvoření specifického bloku. Mezi bloky se pak mohou předávat data ať už v bitech, bajtech, vektorech, či složitějších datových typech.

OpenBTS (Open Base Transceiver Station) je Unixovou aplikací užívající softwarové rádio pro vytváření GSM bezdrátové sítě, respektive jejího přístupového bodu. Softwarové řešení OpenBTS zastupuje funkce rozhraní U_m v GSM architektuře. Ta zhruba odpovídá vrstvám OSI modelu, takže rozhraní U_m (a tím pádem i SW OpenBTS) tvoří tři jeho nejnižší vrstvy: nejnižší je fyzická, pak linková (spojová) a síťová. Na fyzické vrstvě U_m rozhraní GSM standardu se řeší problematika ve třech podvrstvách: radiomodemu, mutiplexování a časování a také kódování. OpenBTS P2.8 podporuje na úrovni radiomodemu standardně GMSK modulaci se šířkou kanálu 200 MHz. Součástí je také podpora pásem GSM850, PGSM900 (EGSM900), DCS1800, PCS1900.

Součástí balíku s OpenBTS je řada podpůrných prostředků (dalšího SW) pro spojování mezi softwarovou PBX a OpenBTS. První komponentou je *sipauthserve* pro registraci SIP. Druhým je *subscriberRegistry* pro registraci účastníků. Posledním je *smqueue* pro přenos SMS v síti. Všechny prostředky projektu jsou nastavovány pomocí databází. Fungování a propojení použitých aplikací naznačuje Obrázek 3.



Obrázek 3. Propojení OpenBTS, Asterisku a dalších komponent v projektu

K přepojování hovorů slouží v tomto případě **Asterisk**, který poskytuje kompletní řešení softwarové PBX ve formě open source projektu, který běží na platformách Linux a Unix. Zde se pro tvorbu GSM využívá služby SIP. Každá mobilní stanice je tak registrována

jako SIP uživatel, který se identifikuje pomocí IMSI SIM karty. Každé IMSI pak musí být přidáno do konfiguračních souborů Asterisku.

ÚKOL A1: Seznamte se s pracovištěm a zapojte jej.

Zkontrolujte připojení kitu USRP1 do USB portu PC. Zapojte obě napájení – pro kit i externí generátor hodinového signálu. Dále spusťte terminál. Do něj zadejte příkaz: *uhd_find_device*. Měl by se vypsat typ zařízení a sériové číslo.

ÚKOL A2: Zvolte číslo kanálu a ověřte jeho volnost.

Podle webu gsmweb.cz zjistěte frekvenční příděl GSM v ČR. Zaměřte se na pásmo DCS (1800 MHz) a zvolte ARFCN kanál tak, aby nebyl přidělen žádnému operátorovi. Ideálně by mohly být dva sousedící ARFCN (z obou stran) taktéž volné. U zvolené ARFCN si zjistěte frekvence, na kterých se kanál nachází (např. pomocí arfcncalc na internetu).

Spusťte si v terminálu příkaz *usrp_fft.py*, pro spuštění spektrálního analyzátoru, využívající SDR. Do kolonky „Center freq" zadejte čísla zjištěných frekvencí Vámi vybraného kanálu. (Pozn. Zadávejte i hodnoty blízké dané frekvenci. Kit totiž může na zadané frekvenci vytvořit drobnou špičku). Je doporučeno zaškrtnout kolonku průměrování (Average). Pokud by se na zvolených frekvencích vyskytoval silnější signál, vyberte jiné ARFCN.

ÚKOL A3: Nastavení databáze OpenBTS

Zde se nastavují parametry sítě. V terminálu zadejte:

```
~$      sudo sqlitebrowser /etc/OpenBTS/OpenBTS.db
```

Otevře se databázový prohlížeč. Parametry vybudovávané sítě se nacházejí v kartě „Browse Data." Projděte si nastavené parametry a jejich hodnoty, ale žádné zatím neměňte. Na základě zjištěných parametrů zkuste například říci, jaký bude mít síť název a jak bude identifikována.

Aby se přepojování hovorů řídilo pouze hodnotami Asterisku, musí být v databázi nastavena otevřená registrace. Zkontrolujte tedy zda je v položce „**Control.LUR.OpenRegistration**" hodnota „ **.*** " (tečka a hvězdička).

Dále nastavte číslo Vámi zvoleného ARFCN. To specifikuje položka „**GSM.Radio.C0**". Nezapomeňte, zkontrolovat správné nastavení pásma GSM (položka „**GSM.Radio.Band**").

ÚKOL A4: Spusťte testovací síť a registrujte do ní MS.

Testovací síť se zprovozní spuštěním všech pěti jejich částí (asterisk, smqueue, sipauthserve, openBTS a konzole OpenBTSCLI). Aby jste nemuseli spouštět každou aplikaci zvlášť, použijte k této akci vytvořený skript (může dojít k výzvě na zadání administrátorského hesla):

```
~$ ./obts_startup.sh
```

Tento skript za Vás také kontroluje, zda nedochází k duplicitě spuštěných programů, což by vedlo k chybě. Po zadání příkazu se spustí další tři terminály (přičemž v jednom jsou dvě aplikace). Počkejte než se v původním terminálu (kde je spuštěn OpenBTS) vypíše hláška: „I'm ready. Use OpenBTSCLI utility to access CLI"

Pomocí svých (či testovacích) mobilních telefonů se připojte k testovací síti. Testovací síť je nestabilní a tak připojení budete možná muset zkusit několikrát za sebou (v závislosti na kvalitě přijímače telefonů). Připojená síť se bude hlásit jako „01 001" nebo „Range" (někdy i Test). Až se Vám podaří přihlásit do testovací sítě (změní se její název), je možné, že Vám přijde SMS o úspěšné registraci s Vaším IMSI.

ÚKOL A5: Vytvořte v Asterisku své SIP účty a propojení mezi nimi.

Editujte soubor se SIP účty – *sip.conf*. Ten se nachází v adresáři /etc/asterisk/. Podle vytvořených příkladů, nacházejících se v souboru, vytvořte své účty. Pokud neznáte své IMSI (nepřišla Vám registrační SMS) a jste registrováni do sítě, můžete je vypsat v konzole OpenBTSCLI zadánim:

```
OpenBTS> tmsis
```

IMSI, který jste identifikován zadávejte k účtům ve formátu např.: IMSI012345678901234.

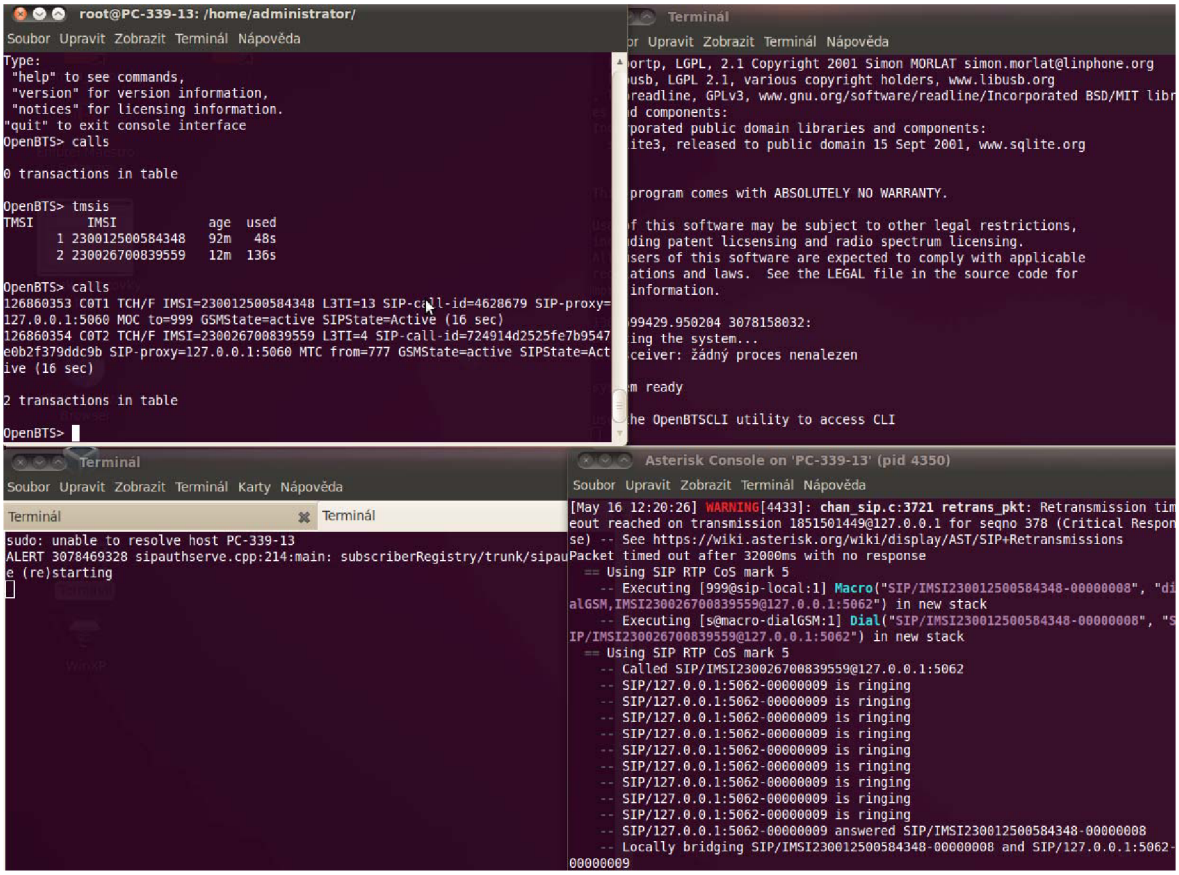
Dále editujte soubor *extensions.conf* (také v /etc/asterisk/) sloužící k propojování hovorů. Za jednotlivé IMSI je ještě nutné, z důvodu zajištění správné funkce, přidána adresa a port spojovací smyčky (tedy *@127.0.0.1:5062*). Jakým způsobem toto propojování mezi Vámi vytvořit, můžete odhadnout dle příkladu v otevřeném souboru. Číslo, pod jakým budete v této síti vystupovat si samozřejmě volte libovolně, dle Vašeho uvážení. Soubory pak uložte. Nakonec musíte **restartovat Asterisk** tak, že do jeho konzole zadáte: „***core restart now***“.

ÚKOL A6: Realizujte hovor mezi Vámi.

Pokud jste dodrželi následující postup, měli by jste být schopni realizovat hovor, vytočením Vámi zvoleného čísla kolegy. Sledujte přitom děje v síti (především v Asterisku). Můžete také v konzole OpenBTSCLI používat systémové nástroje. Jejich přehled vyvoláte zadáním:

```
OpenBTS> help
```

Realizace by pak mohla vypadat tak, jak ji ukazuje Obrázek 4. V tuto chvíli nechejte celou realizace spuštěnou a v Ubuntu zvolte jinou plochu.



Obrázek 4: Realizace hovoru na spuštěné testovací síti

B Analýza sítě s OsmocomBB

V této části budete analyzovat jak vytvořenou síť, tak i ostatní pomocí projektu OsmocomBB.

Popis softwaru

Základní projekt Osmocom (Open Source Mobile Communication) je soubor open-source softwaru pro oblast mobilních komunikací. Realizuje celou řadu projektů mezi nimiž je jak GSM/GPRS, tak bezšňůrová (DECT) či satelitní telefonie. Pro práci s GSM sítěmi je určen dílčí projekt s názvem OsmocomBB (Osmocom Base Band). Jeho cílem je vytvořit plnohodnotné open-source GSM zařízení (stack) 1 až 3 vrstvy. Za celou dobu své existence se stal nejspíše nejvíce užívaným open-source projektem pro implementace, vytváření a testování GSM sítí. Projekt vznikl k účelům zvýšení bezpečnosti u zařízení na veřejné síti, dále pro vzdělávání a také výzkum v oblasti GSM.

OsmocomBB používá pro své aplikace mobilní telefony, které jsou postaveny na čipech Calypso. U těchto telefonů se totiž podařilo zjistit, jakým způsobem pracují a jak v něm spustit vlastní kód. Mezi takové patří v hojném zástupu telefony Motorola (zde C123). Do telefonu je možné nahrávat svůj firmware, takže tento telefon využívá svůj hardware tak, jak je zrovna potřeba.

Aplikace, které OsmocomBB poskytuje, je možné rozdělit do skupin:
- aplikace běžící na čipu telefonu
- aplikace běžící v počítači; komunikující přes sériové rozhraní

Druhé jmenované je možné dále rozdělit na aplikace pro správu firmwaru (nahrávání..) a aplikace GSM vrstvy L2/3.

K aplikacím běžícím na čipu telefonu patří:
- *layer1* – jednoduchý představitel první GSM vrstvy; sama o sobě tato aplikace nic nedělá – čeká na příkazy z vyšších vrstev
- *loader.bin* – zavaděč pro flash paměti
- *l1test* – testovací verze vylepšené aplikace layer1
- *bootloader* – univerzální zavaděč pro telefony s Calypso čipy.
- *hello_world* – výpis klasického „Hello world“
- *rssi* – aplikace určená ke sledování přijímaného signálu; dokáže měřit jednotlivé kanály i celé spektrum.

Aplikacemi pro správu jsou:
- *osmocon* – konzolový nástroj propojující firmware v telefonu s aplikacemi na počítači.
- *osmoload* – slouží k zápisu, mazání a prohlížení flash paměti
- *calypso_pll* – zobrazení možných kombinací násobičky/děličky pro výstupní frekvence
- *rita_pll* – zobrazení možných kombinací násobičky/děličky pro výstupní frekvence

Mezi aplikace pro GSM vrstvy L2/3 patří:

- *mobile* – aplikace vytvářející chování standardního mobilního telefonu rozšířené o zajímavé funkce
- *cell_log* – skenuje dostupné frekvence a sbírá informace z logického kanálu BCCH (Broadcast Control CHannel). Díky tomu je možné vytvořit seznam použitých ARFCN s údaji o úrovni, MNC, MCC a informacemi o systému
- *ccch_scan* – aplikace se synchronizuje s ARFCN, zaznamenává měření výkonu a údaje z logického kanálu CCCH (Common Control Channel)
- *bcch_scan* – předchůdce aplikace cell_log
- *cbch_sniff* – zaznamenává informační zprávy vysílané o dané buňce

Přehrávání SW v telefonu se provádí příkazy, které specifikují aplikaci, port, zařízení a typ telefonu. Aplikace jako *rssi.bin* jsou náročnější na paměť, proto potřebují nejprve upravit ovládání paměti telefonu přes tzv. „chainload," a příkaz je delší. V případě aplikací L2/3 vrstvy se zase spouštějí dva terminály – jeden pro aplikaci ovládající první vrstvu, druhý pro specifikaci nástroje L2/3. Proto byl vytvořen skript pro jednoduší ovládání.

Nahrávání aplikace se provádí stisknutím „enter" a také krátkým stiskem červeného tlačítka na MS (aby se nezapnul). Nahrání provází problémy, proto je třeba (všimnete-li si v terminálu vertikálního výpisu error) červené tlačítko zmáčknout vícekrát. Pokud to nefunguje nebo program zůstal „viset," nezbývá než vše zrušit, vytáhnout a vložit baterii telefonu a zkusit to znova. Vytažením baterie se vymaže veškerý nahraný SW a MS se vrací do původního stavu. Měření se ukončuje stiskem ctrl+c v terminálu.

ÚKOL B1: Zkontrolujte zapojení MS Motorola C123 (do sluchátkového jacku) k PC (do USB) a ujistěte se, že je MS vypnutá.

ÚKOL B2: Na PC spusťte terminál a přesuňte se do složky se skriptem *run_osmo.sh*.

ÚKOL B3: Vyzkoušejte si nahrávání SW do telefonu pomocí „Hello world."
Spusťte skript zadáním *./run_osmo.sh* do terminálu. Volbou *„h"* a stiskem enter bude aplikace čekat na povolení nahrávání uživatelem na MS. Stiskněte krátce červené tlačítko a počkejte, než se aplikace nahraje. Správné nahrání signalizuje nápis „Hello world" na obrazovce telefonu. Povedlo-li se, můžete telefon červeným tlačítkem vypnout a v terminálu stisknout ctrl+c pro zastavení běhu programu.

ÚKOL B4: Pomocí aplikace *rssi.bin* proměřte testovací síť. Dále pomocí této aplikace zjistěte, jaké jsou v okolí aktivní ARFCN.
Postupujte při nahrání aplikace do telefonu jako v předchozím úkolu. Zvolte ale možnost *„r"* pro aplikaci *rssi.bin*.

Rssi.bin se plně ovládá na klávesnici telefonu a je celkem intuitivní. U základní obrazovky zobrazující číslo kanálu, frekvenci kanálu a sílu signálu, lze měnit jak normy (DCS, PCS), tak i směr analýzy (downlink, uplink). To se provádí přidruženými tlačítky pod displejem. Volba frekvence je možná buď to po každém kanálu – tlačítky vlevo, vpravo – nebo přímým zadáním ARFCN na číselníku. Tlačítka „nahoru, dolů" ovládají hlasitost (resp. i zapínání) akustického oznamování síly signálu. Pro zadržení maximální hodnoty naměřené síly signálu slouží prostřední tlačítko.

Další obrazovkou aplikace rssi.bin je „**zobrazení spektra**“ do níž se přechází stisknutím **hvězdičky**. Obrazovka v tomto módu zobrazuje spektrum - tedy sílu signálu na jednotlivých GSM kanálech. Celým spektrem se dá pohybovat (vlevo, vpravo) po jednotlivých kanálech, šipkami nahoru a dolů se celé spektrum zvětšuje a zmenšuje (volba měřítka).

Zelené tlačítko (přijetí hovoru) se používá k přechodu **na obrazovku synchronizace**. Původní obrazovka zobrazuje informace o základnové stanici (BSIC, cell id), síti (MCC, MNC, LAC) s doplňkem o aktuální sílu signálu. Při stisku tlačítka „nahoru“ se přechází do zobrazení, které zobrazuje úroveň signálu každého timeslotu. Pokud na původní obrazovce dojde ke stisku tlačítka „dolů“ zobrazí se kanály současné i sousedních buněk. Opětovným stiskem zeleného tlačítka se telefon pokusí o měření vzdálenosti a zpoždění od dané BTS. Tato volba však nefunguje zcela korektně u neupraveného filtru v telefonu.

Pomocí aplikace rssi.bin zjistěte a poznamenejte si všechny možné informace o testovací síti.

Poté zjistěte co nejvíce obsazených ARFCN od komerčních poskytovatelů. S vysílajícími BTS musí být schopna se Motorola synchronizovat. Využijte obě pásma GSM (900 i 1800). Zjištěné hodnoty si poznamenejte.

Až budete hotový, Motorolu C123, stejně jako terminál, opět vypněte.

ÚKOL B5: Aplikací cell_log prověřte výsledky měření předchozího bodu.

Protože se jedná o aplikaci vyšší vrstvy, spouští startovací skript *run_osmo.sh* další terminál na víc, ve kterém běží aplikace ovládající první vrstvu GSM stacku. O všech aplikací Vás bude ale zajímat výstup do původního terminálu a Wiresharku, který skript také automaticky spouští. Aplikace *cell_log* taktéž vytváří ze zachycených dat log s naměřenými daty. Ten Vás zde bude zajímat nejvíc.

Předtím, než spustíte nahrávání aplikace do zařízení, přečtěte si následující postup. Na začátku zvolíte *„a“* pro cell_log na všech kanálech. Následuje spuštění Wiresharku, který rovnou začne zachytávat na loopbacku. V nastartovaném Wiresharku zadejte do filtru (kolonka Filter) „gsmtap“. Tím je odfiltrována jiná komunikace na smyčce (loopback). Na zadání máte méně než 15 sekund. Poté se vraťte zpět do terminálu. Po zmíněných 15 s. se otevře další okno terminálu. Jakmile jej uvidíte, spusťte nahrávání aplikace krátkým stiskem červeného tlačítka na telefonu. Možné je stisk opakovat, objeví-li se vertikální výpis error. Správné nahrávání identifikuje průběh nahrávání dat do telefonu. Po nahrání aplikace by se na obrazovce telefonu měla objevit hláška „Layer1.“ V původním terminálu se začnou vypisovat rychle za sebou hodnoty naměřené na kanálech. Situaci můžete sledovat ve Wiresharku, ve kterém zjistíte jak se data vypisují. Celou aplikaci nechejte klidně běžet 1-2 minuty. Po té obě aplikace v terminálech vypněte (ctrl+c) a zachytávání ve Wiresharku taktéž.

Po provedeném zachytávání si v terminálu otevřete zachycený log, v němž jsou vypsány hodnoty měření:

```
~$  gedit cell_log_analyza.log
```

V úvodu souboru se nalézají hodnoty *power*, které nejsou moc důležité. Hodnoty ***sysinfo*** zobrazují kanály, na kterých se povedla synchronizace zařízení s BTS. Tyto hodnoty si projděte a porovnejte je s hodnotami získanými z předchozího úkolu.

ÚKOL B6: Zachyťte provoz testovací sítě při hovoru pomocí aplikace ccch_scan.

U aplikace *ccch_scan* (stejně jako u *cbch_sniff* a *cel_log* pro jeden kanál) se postupuje téměř stejně jako v předchozím úkolu. Rozdíl je ve volbě (*c, l, s*), avšak navíc je přidán mezikrok – po volbě aplikace, kde je zadáváno ARFCN skenovaného kanálu. S vědomím těchto rozdílu proveďte nahrání *ccch_scan* dle postupu z předchozího úkolu. Při zachytávání se může objevit velké množství chyb (označené červeně). Pokud by to tak bylo vyjměte a vložte baterii a spusťte analýzu znova.

Během spuštěného zachytávání proveďte mezi sebou testovací hovor. Sledujte přitom terminál, ve kterém je spuštěn program *ccch_scan*. Pokuste se vysvětlit, co do něj aplikace vypisuje.

Po dokončení analýzy projděte zachycená data Wiresharkem. Prohlédněte si hodnoty v jednotlivých zprávách. Pokuste se určit, kde začal, či probíhal testovací hovor. Pokud se Vám to nepodaří, zkuste do filtru Wiresharku připsat ke „gsmtap or sip“, čímž se vypíše aktivita SIP ze strany Asterisku. Poté můžete Wireshark zavřít.

ÚKOL B7: Použijte aplikaci cbch_sniff k porovnání výpisu na testovací síti a komerční síti v okolí.

Zvolte si jeden kanál ARFCN z komerční sítě, na kterém jste v předchozích úkolech změřil silný signál. Je jedno, která síť eventuelně BTS to bude – záleží na Vás.

Nejprve, podobně jako v předchozích dvou úkolech spusťte startovacím skriptem aplikaci *cbch_sniff* na **ARFCN testovací sítě**. Během zachytávání opět realizujte hovor. Poté ve Wiresharku vypněte zachytávání a projděte si zachycená data. Poté ale Wireshark NEVYPÍNEJTE, pouze minimalizujte.

Nyní spusťte opět *cbch_sniff*, ale teď na kanále Vámi zvolené komerční sítě. V nově spuštěném Wiresharku zadejte „gsmtap“ a testování této sítě nechejte běžet cca minutu. Po té zachytávání stopněte (v terminálu i Wiresharku).

Data v obou Wiresharkcích porovnejte. Pokuste se vysvětlit rozdíl zachycených dat.

ÚKOL B8: Získané výsledky prezentujte vyučujícímu.

ÚKOL B9: Uveďte pracoviště do původního stavu.

Vypněte terminály, odpojte kit USRP ze zásuvky. MS Motorola vypněte a odpojte od PC.

V souborech Asterisku *sip.conf* a *extensions.conf*, smažte Vámi vytvořené účty. Původní ukázkovou konfiguraci v souborech **ponechejte!**

C TŘETÍ PŘÍLOHA – ZDROJOVÉ VÝPISY SKRIPTŮ

C.1 Skript obts_startup.sh

```
#!/bin/bash

echo Runs OpenBTS, Asterisk, smqueue, sipauthserve, and finally OpenBTSCLI
echo Make sure that the OBTS_ROOT variables in this script are set
      correctly and that your radio is attached via USB!

#get sudo privileges
sudo echo

#store Asterisk and OpenBTS root locations
OBTS_ROOT="openbts/"
#FS_ROOT="/home/administrator/asterisk/"

#start Asterisk
#cd $FS_ROOT/bin
sudo killall asterisk &> /tmp/tmp
ulimit -s 240
sleep 2s
sudo gnome-terminal -x sh -c "sudo asterisk -vvvvvvvvvvc" &


#start smqueue and sipauthserve
cd $OBTS_ROOT
sudo gnome-terminal --tab -e "sudo smqueue/trunk/smqueue/smqueue" --tab -e
      "sudo subscriberRegistry/trunk/sipauthserve" &

#start OpenBTS
cd $OBTS_ROOT/openbts/trunk/apps
sudo killall transceiver &> /tmp/tmp #sometimes necessary
sudo gnome-terminal -x sh -c "sudo ./OpenBTS" &


#finally, start the CLI
echo ---All other processes initialized. Please wait a few seconds for
      Asterisk to initialize---
echo
sudo ./OpenBTSCLI
```

C.2 Skript run_osmo.sh

```
#!/bin/bash
#Volba programu pro osmocom

#Cesta k projektu - kdyztak menit
OSMO_ROOT="osmocom-bb"

#Cesta k apl osmocon
CON=src/host/osmocon

#Cesta k L2/3 aplikacim
MISC=src/host/layer23/src/misc

sudo echo

echo Aplikace pro usnadneni prace se softwarem OsmocomBB na Motorole C123.
echo

echo Zadejte co chcete pouzit", pro konec muzes kdykoliv dat ctrl+c:"
echo "'h'   -spusti hello_world"
echo "'r'   -spusti rssi.bin"
echo "'c'   -spusti ccch_scan"
echo "'a'   -spusti cell_log se skenovanim vsech kanalu"
echo "'l'   -spusti cell_log na jednom kanalu"
echo "'s'   -spusti cbch_sniff"

read parametr

case $parametr in
      h)
      echo Spousti se HELLO WORLD
      echo Stiskni KRATCE cervene tlacitko a pockej nez se aplikace
      nahraje "pokud to trva dele jak pul minuty - tak stiskni ctrl+c"
      sudo ./$OSMO_ROOT/$CON/osmocon -p /dev/ttyUSB0 -m c123xor
      $OSMO_ROOT/$CON/../../target/firmware/board/compal_e88/hello_world.c
      ompalram.bin || exit ;;
      r)
      echo Spousti se rssi.bin
      echo Stiskni KRATCE cervene tlacitko a pockej nez se aplikace
      nahraje "pokud to trva dele jak pul minuty - tak stiskni ctrl+c"
      ./$OSMO_ROOT/$CON/osmocon -p /dev/ttyUSB0 -m c123xor -c
      $OSMO_ROOT/$CON/../../target/firmware/board/compal_e88/rssi.highram.
      bin
      $OSMO_ROOT/$CON/../../target/firmware/board/compal_e88/chainload.com
      palram.bin || exit ;;
      c)
      echo Spousti se ccch_scan, ale je ted treba zadat ARFCN kanalu":"
      read kanal
      echo Nejprve zapnu Wireshark nastav "do filtru gsmtap a vrat se do
      konzole"
      sleep 3s
      sudo wireshark -i lo -k &
      sleep 15s
```

```
echo Stiskni kratce cervene tlacitko a pockej nez se aplikace
nahraje "pokud to trva dele jak pul minuty - tak stiskni ctrl+c"
sudo gnome-terminal -x sh -c "./$OSMO_ROOT/$CON/osmocon -p
/dev/ttyUSB0 -m c123xor
$OSMO_ROOT/$CON/../../target/firmware/board/compal_e88/layer1.compal
ram.bin" &
sleep 10s
./$OSMO_ROOT/$MISC/ccch_scan -s /tmp/osmocom_l2 -a $kanal -i
127.0.0.1 || exit;;
a)
sudo rm cell_log_analyza.log
echo Spousti se cell_log a skenuje vse
echo Nejprve zapnu Wireshark nastav "do filtru gsmtap a vrat se do
konzole"
sleep 3s
sudo wireshark -i lo -k &
sleep 15s
echo Stiskni kratce cervene tlacitko a pockej nez se aplikace
nahraje "pokud to trva dele jak pul minuty - tak stiskni ctrl+c"
sudo gnome-terminal -x sh -c "./$OSMO_ROOT/$CON/osmocon -p
/dev/ttyUSB0 -m c123xor
$OSMO_ROOT/$CON/../../target/firmware/board/compal_e88/layer1.compal
ram.bin" &
sleep 10s
./$OSMO_ROOT/$MISC/cell_log -s /tmp/osmocom_l2 -l
cell_log_analyza.log -i 127.0.0.1 || exit;;
l)
sudo rm cell_log_kanal.log
echo Spousti se cell_log na jednom kanalu, jehoz ARFCN je treba ted
zadat":"
read kanal
echo Nejprve zapnu Wireshark nastav "do filtru gsmtap a vrat se do
konzole"
sleep 3s
sudo wireshark -i lo -k &
sleep 15s
echo Stiskni kratce cervene tlacitko a pockej nez se aplikace
nahraje "pokud to trva dele jak pul minuty - tak stiskni ctrl+c"
sudo gnome-terminal -x sh -c "./$OSMO_ROOT/$CON/osmocon -p
/dev/ttyUSB0 -m c123xor
$OSMO_ROOT/$CON/../../target/firmware/board/compal_e88/layer1.compal
ram.bin" &
sleep 10s
./$OSMO_ROOT/$MISC/cell_log -s /tmp/osmocom_l2 -l cell_log_kanal.log
-i 127.0.0.1 -A $kanal || exit;;
s)
echo Spousti se cbch_sniff, ale je ted treba zadat ARFCN kanalu":"
read kanal
echo Nejprve zapnu Wireshark nastav "do filtru gsmtap a vrat se do
konzole"
sleep 3s
sudo wireshark -i lo -k &
sleep 15s
echo Stiskni kratce cervene tlacitko a pockej nez se aplikace
nahraje "pokud to trva dele jak pul minuty - tak stiskni ctrl+c"
```

```
        sudo gnome-terminal -x sh -c "./$OSMO_ROOT/$CON/osmocon -p
        /dev/ttyUSB0 -m c123xor
        $OSMO_ROOT/$CON/../../target/firmware/board/compal_e88/layer1.compal
        ram.bin" &
        sleep 10s
        ./$OSMO_ROOT/$MISC/cbch_sniff -s /tmp/osmocom_l2 -a $kanal -i
        127.0.0.1 || exit;;
        *)
        echo CHYBA zadani"!!" Spust skript znova..
        exit
esac
```

C.3 Skript n_gammu.sh

```
#!/bin/bash
#Pouziti gammu pro Nokii.

sudo echo

cesta=gnumonks_airprobe/airprobe/gsmdecode/src

if [ -d out ]; then
      rm -rf out
fi

mkdir out

if [ ! -s nhm5_587.txt ]; then
      echo CHYBA"!!!!!!!!!!!!"
      echo Zkontruljte umisteni. Nenalezen klicovaci soubor nhm5_587.txt
      exit
fi

sudo gammu-comfig

echo Pouzivani gammu pro Nokii 3310 s vypisem zachycenych dat.

gammu --nokiadebug nhm5_587.txt v20-25,v18-19 &
sleep 2s

echo Je Nokie pripojena? Spoustim zachytavani "do" tohoto adresare..

echo Probiha zachytavani... Po testovani stisknete Enter
read

if [ -s out.xml ]; then
sudo ../$cesta/gsmdecode -x <out.xml | uniq > out/all.out
sudo ../$cesta/gsmdecode -x <out.xml | grep "7/odd" > out/imsi.out
sudo ../$cesta/gsmdecode -x <out.xml | grep "Quality Full" | awk '{print
      $9}' | sort | uniq -c > out/quality.out
sudo ../$cesta/gsmdecode -x <out.xml | grep "RxLev Full Serving Cell" |
      awk '{print $8}' | sort | uniq -c > out/rxlev.out
else echo Chyba. Nebyla zachycena zadna data..
fi

sudo killall gammu

if [ -s out/all.out ]; then
echo Vsechna zachycena a dekodovana data se nalezaji ve slozce out"/"
echo Otviram soubory se zachycenymi udaji o IMSI, kvalite a urovne
      prijimaneho signalu
sleep 2s
gedit out/imsi.out
gedit out/quality.out
gedit out/rxlev.out
fi
```