OsmocomBB, der Open Source GSM Stack

Open HW SW Event, München, 4. Dezember 2010

Dieter Spaar
spaar@mirider.augusta.de

Agenda

- Die Motivation sich mit GSM zu beschäftigen
- Kurze Vorstellung der wichtigsten Open Source GSM Projekte
- Welche Möglichkeiten gibt es die Telefonseite zu kontrollieren
- OsmocomBB im Detail

Motivation

- GSM/3G im Vergleich zu TCP/IP kaum untersucht
  (obwohl es GSM in Deutschland seit 1992 gibt)

- „Geschlossene“ GSM Industrie, kaum Hardware verfügbar

- Situation hat sich geändert:

  * Gebrauchte Hardware
  * OpenBTS (seit 2007/8)
  * OpenBSC (seit Ende 2008)
  * Airprobe (seit 2007/8)
  * OsmocomBB (seit Anfang 2010)

- Spezifikation bei der 3GPP oder ETSI verfügbar
  (umfangreich, ca. 2000 Dokumente)

Vereinfachte Struktur des GSM Netzwerk

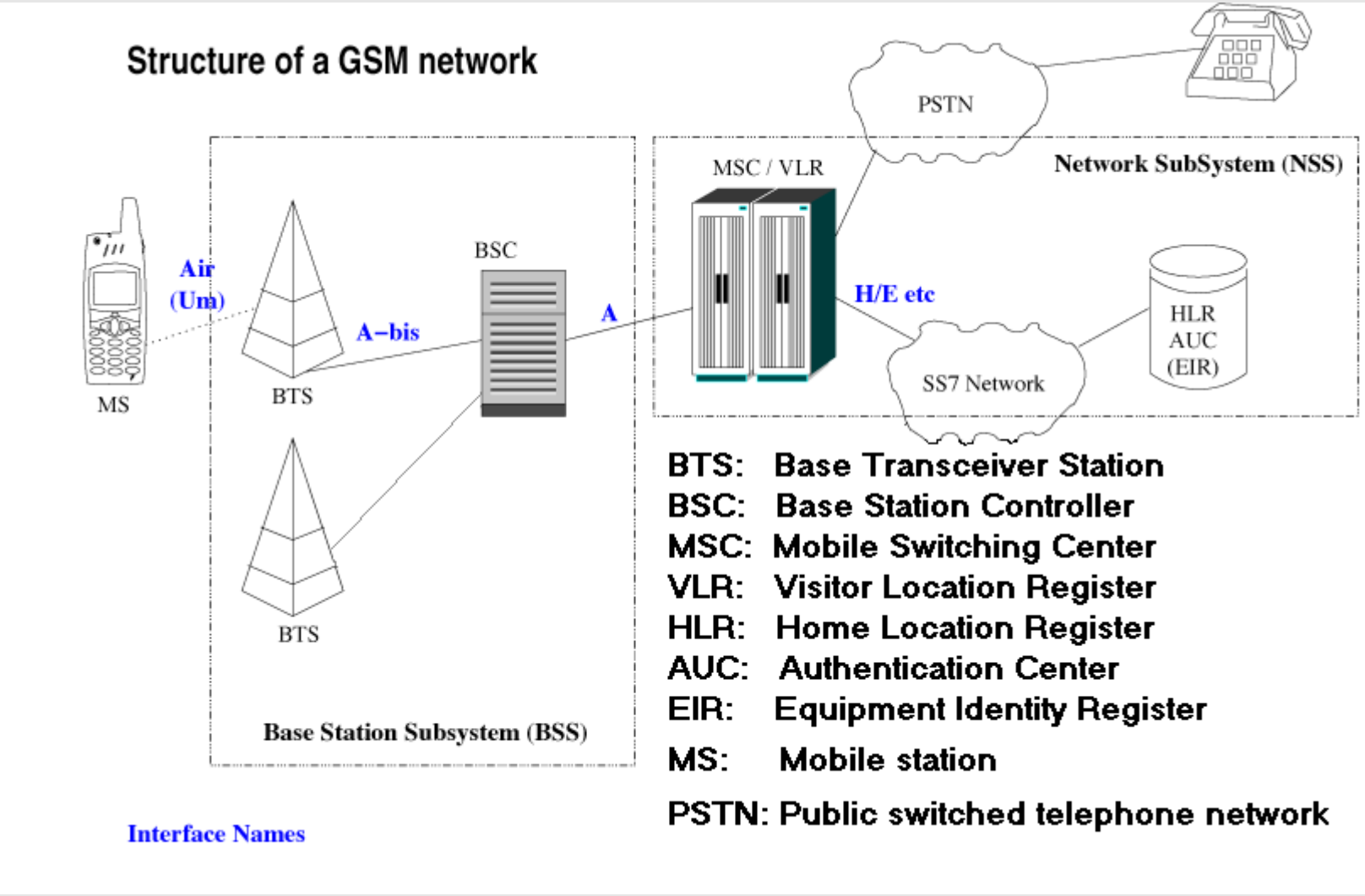


Diagram derived from Wikipedia (http://en.wikipedia.org/wiki/File:Gsm_network.png), GNU Free Documentation License applies to this diagram

OpenBTS

- Hardware basiert auf dem USRP (Universal Software Radio Peripheral)

- Luft-Schnittstelle (Um) basiert auf Software Defined Radio

- Keine klassische GSM Architektur (BSC, MSC, ...)
  Ansatz: direkt von „Um“ nach „SIP“

- Asterisk wird eingesetzt, SIP (Session Initiation Protocol) und Voice-over-IP

OpenBSC

- Abis Protokoll Implementierung plus BSC/MSC/HLR

- Unterstützt die Siemens BS11 microBTS

  * GSM 900, 2 W (kann bis auf 30 mW reduziert werden)
  * ca. 11 Jahre alt
  * günstig
  * schwer (30 bis 40 Kg, abhängig von der Konfiguration)
  * E1 Schnittstelle für Abis

- Unterstützt die ip.access nanoBTS

  * verschiedene RF Bänder, 200 mW bei DCS 1800
  * klein
  * GPRS/EDGE
  * Abis über IP

- 26C3: GSM Netzwerk mit fünf nanoBTS

nanoBTS im Vergleich zur BS11

Airprobe

- Passives Abhören der GSM Luft-Schnittstelle
- Basiert auf dem USRP und GNU Radio
- Verwendet WireShark für die Protokoll-Analyse
- Kann auch für das Projekt zum Brechen der A5/1 GSM Verschlüsselung verwendet werden

Kontrolle über das Mobiltelefon

- Ein Ansatz: Irgendetwas aus allgemein verfügbaren Komponenten bauen (DSP, CPU, ADC, FPGA):

  * Kein Reverse Engineering nötig
  * Viel Aufwand fürs Hardware Design + Debugging
  * Nur kleine Stückzahlen, daher teuer

- Ein andere Ansatz: Irgendetwas mit Baseband Chipsets bauen

  * Reverse Engineering oder Zugriff auf geleakte Doku
  * Weniger Aufwand für ’Layer 0’
  * Nach wie vor kleine Stückzahlen, teuer

Kontrolle über das Mobiltelefon

- Alternativer Ansatz: Vorhandene Telefone „zweckentfremden“

  * Hardware funktioniert bereits
  * Keine Prototypen oder Hardware Revisionen
  * Reverse Engineering ist erforderlich
  * Hardware Treiber müssen entwickelt werden
  * Aber: Mehr Zeit für die eigentliche Aufgabe: Protokoll Stack

- Suche nach geeigneten Telefonen

  * So billig wie möglich
  * Verfügbar: Jeder Interessent kann die Hardware kaufen
  * So alt/einfach wie möglich -> niedrige Komplexität
  * Baseband Chipset mit viel geleakter Dokumentation

Baseband Chips mit geleakter Dokumentation

- Texas Instruments Calypso

  * DBB Dokumentation auf cryptome.org
  * ABB Dokumentation auf Webseiten chinesischer Telefon Entwickler
  * Source Code eines GSM Stack war auf sf.net (TSM30)
  * „End of life“, keine neuen Telefone seit ca. 2008
  * Kein kryptographisch gesicherter Bootloader

- Mediatek MT622x Chipsets

  * Viel Dokumentation auf chinesischen Webseiten
  * SDK mit „binary-only“ GSM Stack Libraries auf chinesischen Webseiten
  * 95 Mio. produziert/verkauft im Q1/2010

=> Ausgangsbasis: TI Calypso (GSM Stack Source verfügbar)

OsmocomBB

- Projekt Start Januar 2010

- Implementierung der GSM Baseband Software:

  * GSM MS-Seite des Protokoll Stack von Layer 1 bis Layer 3
  * Hardware Treiber für den GSM Baseband Chipset
  * Einfaches User Interface auf dem Telefon
  * Umfangreiches User Interface auf dem PC

- Zum Projekt Namen:

  * Osmocom = **O**pen **S**ource **MO**bile **COM**munication
  * BB = **B**ase **B**and

OsmocomBB Software Architektur

- Code aus OpenBSC wird wieder verwendet (libosmocore)

- Möglichst wenig Code läuft auf dem Telefon (RAM !)

  * Debugging auf dem PC ist viel einfacher
  * Viel mehr Platz auf dem Bildschirm
  * Hardware Treiber und Layer1 läuft auf dem Telefon
  * Layer2/3 und das eigentliche „Telefon“ läuft auf dem PC
  * Layer2/3 kann später auf das Telefon übertragen werden

- Schnittstelle zum Layer1/2 kann von anderen Anwendungen verwendet werden

- Protokoll Tracing des Layer2/3 mit WireShark und GSMTAP

OsmocomBB Hardware Unterstützung

- Baseband Chipsets

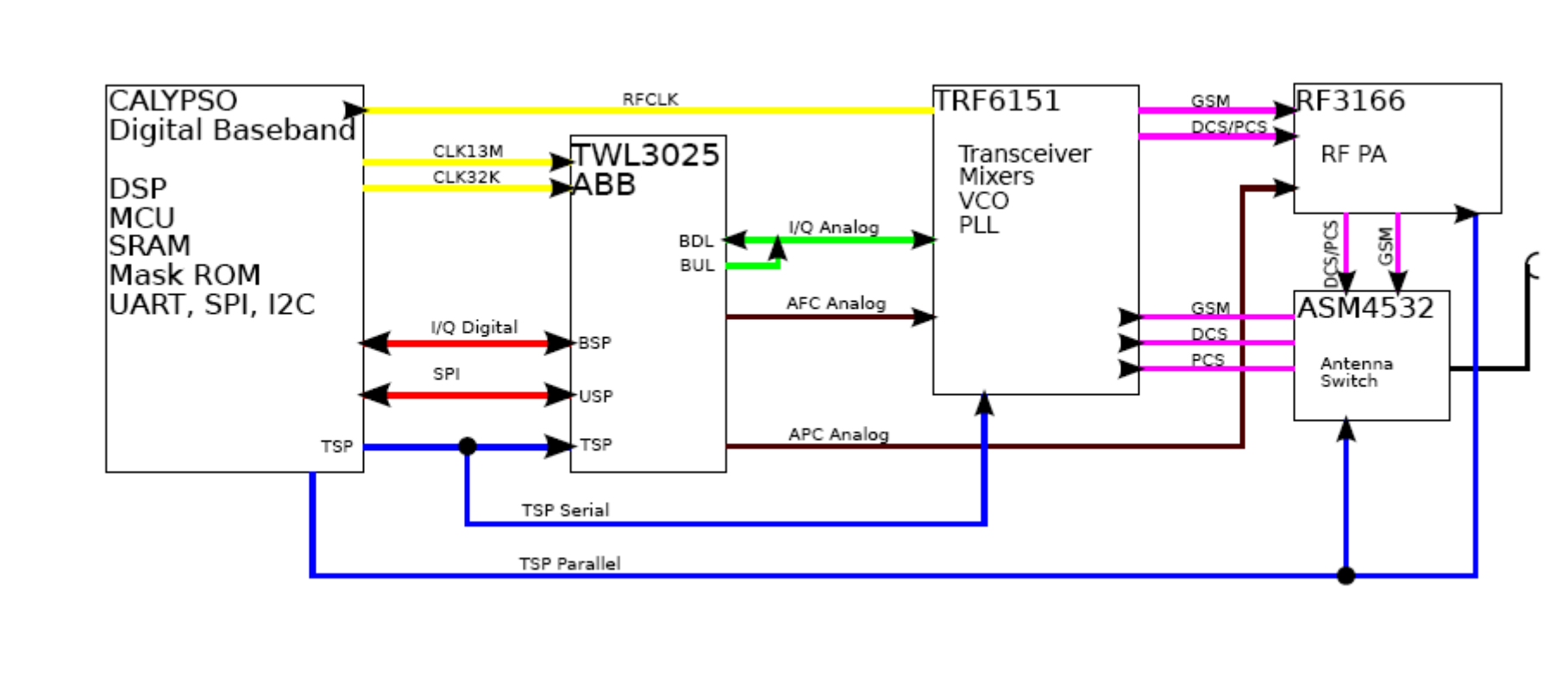
  * TI Calypso/Iota/Rita
  * Erste Experimente mit dem Mediatek MT622x
    (MT6235: ARM9 -> Linux + U-Boot)

- Telefone

  * Compal/Motorola C11x, C12x, C13x, C14x und C15x
  * Haupt-Entwicklungsplatform: C123 und C155
  * GSM Modem des Openmoko Neo1973 und Freerunner

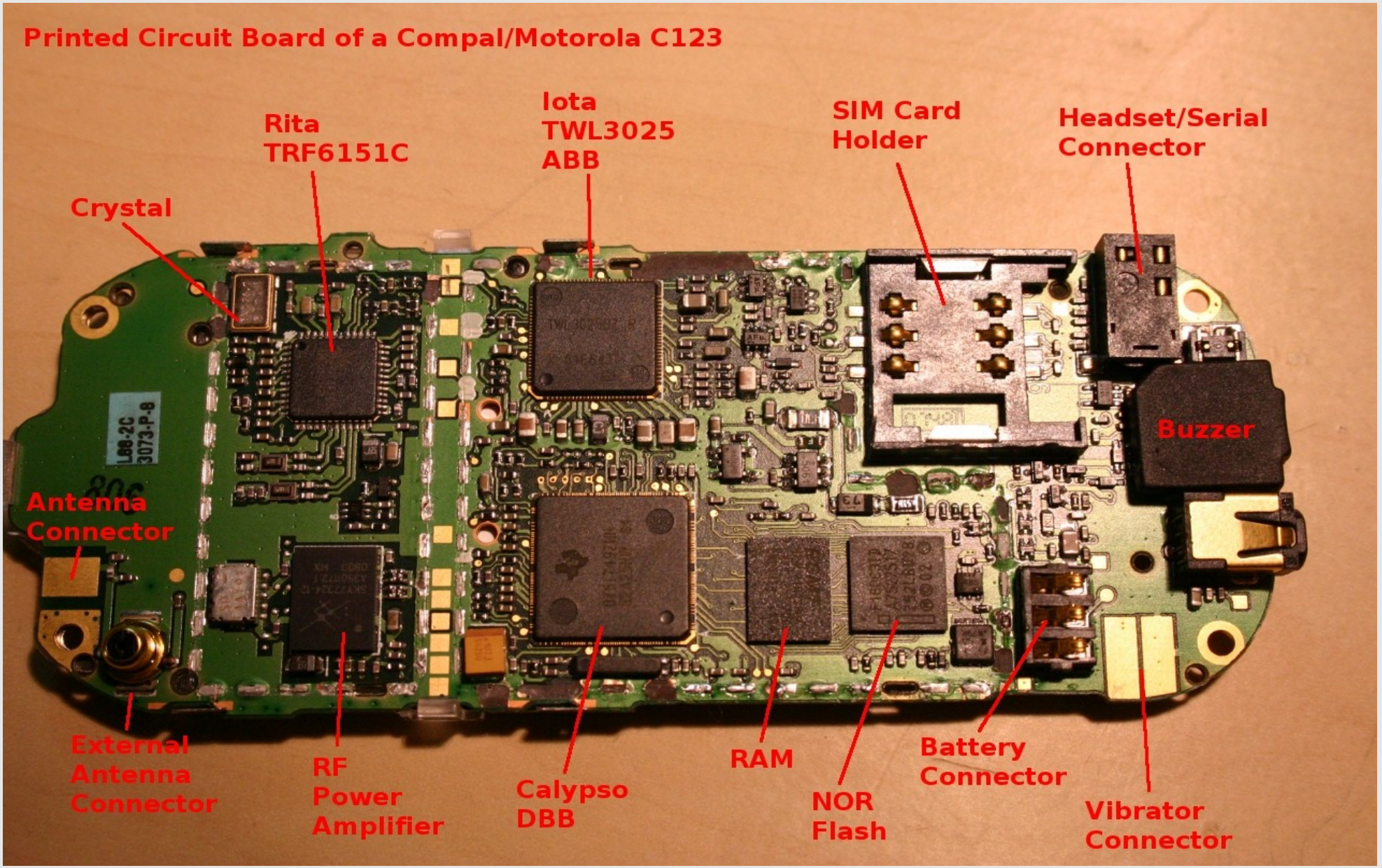
- Einfache „Feature Phones“ basierend auf einem ARM7TDMI im DBB

Der GSM Baseband Chipset



http://laforge.gnumonks.org/papers/gsm_phone-anatomy-latest.pdf

Das Motorola/Compal C123



OsmocomBB Projekt Status: Was funktioniert (1/2)

- Hardware Treiber für Calypso/Iota/Rita fast vollständig

- Treiber für Audio/Sprache

- Layer1:

  * Messung Empfangsfeldstärke
  * Träger/Bit/TDMA Synchronisation
  * Empfangen/Senden von „Normal Bursts“ auf dem SDCCH
  * Senden von „RACH Bursts“
  * Automatische Rx Verstärkungsregelung (AGC)
  * „Frequency Hopping“
  * Automatische Tx Leistungsregelung (APC)

- Layer2 UI/SABM/UA Frames und ABM (Asynchronous Balanced Mode) Modus -> LAPDm

OsmocomBB Projekt Status: Was funktioniert (2/2)

- Layer3 Nachrichten für RR / MM / CC

- Zellen (Re-)Selektion entsprechend GSM 03.22

- OsmocomBB kann Sprach-Verbindungen (seit 08/2010)

  * „Very Early Assignment“ + „Late Assignment“
  * A3/A8 Authentisierung der SIM
  * A5/1 + A5/2 Verschlüsselung
  * „Full Rate“ (FR), „Half Rate“ (HR) und „Enhanced Full Rate“ (EFR) Codec

OsmocomBB Projekt Status: Was funktioniert (noch) nicht:

- Vollständige Unterstützung des SIM Kartenleser im Telefon

- Layer1

    * Nachbarzellen Messungen
    * Handover innerhalb einer Verbindung

- User Interface auf dem Telefon

- Circuit Switched Data (CSD)

- GPRS (Packet Data)

- kein „Type Approval“ !

OsmocomBB Projekt Status: Zusammenfassung

- Kontroll- und Sprachkanäle können aufgebaut werden

- Kontrolle über den Synthesizer -> GSM-R

- Beliebige Nachrichten auf den Kontrollkanälen

  * RR Nachrichten an den BSC
  * MM/CC Nachrichten ans MSC
  * SMS Nachrichten ans MSC/SMSC (Short Message Service Center)

- TCH (Traffic Channel) für Sprachverbindungen

  * Mehrere 20+ Minuten Verbindungen in Test-Netze
  * 30+ Minuten Verbindungen ins „echte“ Netz (zumindest wird das behauptet... ;-)

Links

- OsmocomBB http://bb.osmocom.org
- OpenBTS: http://openbts.sourceforge.net/
- OpenBSC: http://openbsc.gnumonks.org/
- Airprobe: http://airprobe.org/
- 3GPP: http://www.3gpp.org/
- ETSI: http://www.etsi.org/
- http://laforge.gnumonks.org/papers/gsm_phone-anatomy-latest.pdf

Danke an

Harald Welte, Andreas Eversberg, Sylvain Munaut

Fragen